ALIBABA CLOUD

Alibaba Cloud

ApsaraDB for RDS

RDS MySQL Database

Document Version: 20220712

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	Danger: Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	Warning: Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	Notice: If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	Note: You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.Overview of ApsaraDB RDS for MySQL

This topic provides an overview of ApsaraDB RDS for MySQL and describes the related terms.

ApsaraDB for RDS is a stable, reliable, and scalable online database service. It is designed based on the Apsara Distributed File System and high-performance SSD storage media of Alibaba Cloud. It supports five database engines: MySQL, SQL Server, PostgreSQL, and MariaDB. It also provides a complete suite of solutions for various scenarios, such as disaster recovery, backup, restoration, monitoring, and migration. These solutions facilitate database operation and maintenance (O&M). For more information about the benefits of ApsaraDB for RDS, see Competitive advantages of ApsaraDB RDS instances over self-managed databases.

You can submit a if you require technical support. If your workloads are complex, you can purchase a support plan on the Alibaba Cloud After-Sales Support page. This allows you to seek advice from instant messaging (IM) enterprise groups, technical account managers (TAMs), and service managers.

For more information about ApsaraDB for RDS, visit the ApsaraDB RDS for MySQL product page.

Disclaimer

Some features or functions that are described in this document may be unavailable. For more information, see the specific terms and conditions in your commercial contract. This document serves as a user guide that is for reference only. No content in this document can constitute any expressed or implied warranty.

ApsaraDB RDS for MySQL

ApsaraDB RDS for MySQL is developed based on a branch of the MySQL source code. Its excellent performance has been proven over years of Double 11, during which it needs to handle large volumes of concurrent traffic. ApsaraDB RDS for MySQL provides basic features, such as instance management, account management, database management, backup and restoration, control access, Transparent Data Encryption (TDE), and data migration. ApsaraDB RDS for MySQL also provides the following advanced features and functions:

- ApsaraDB MyBase dedicated clusters: An ApsaraDB MyBase dedicated cluster consists of multiple hosts, such as ECS instances of the ecs.i2.xlarge instance type and ECS Bare Metal instances. You can run instances on these hosts. For more information, see What is ApsaraDB for MyBase?
- Read-only RDS instances: If the primary RDS instance is overwhelmed by a large number of read requests, your workloads may be interrupted. In this case, you can create one or more read-only RDS instances to offload read requests from the primary RDS instance. For more information, see Overview of ApsaraDB RDS for MySQL read-only instances. This scales up the read capability of your database system and increases the throughput of your application.
- Read/write splitting: The read/write splitting function provides a read/write splitting endpoint. This endpoint connects to the primary RDS instance and all of the read-only RDS instances to establish an automatic read/write splitting link. For more information, see Read/write splitting. Your application can read and write data into your database system after it connects to this endpoint. ApsaraDB for RDS distributes write requests to the primary RDS instance and read requests to the read-only RDS instances based on the specified read weights. You can create more read-only RDS instances to scale up the read capability of your database system. In addition, you do not need to modify your application.
- Dedicated proxy: A dedicated proxy uses dedicated computing resources. It provides more advanced

functions, such as read/write splitting, short-lived connection optimization, and transaction splitting. For more information, see What are database proxies?.

- Database Autonomy Service (DAS): DAS supports intelligent diagnostics and optimization at the instance level based on various metrics. These metrics include SQL execution performance, CPU utilization, input/output operations per second (IOPS) utilization, memory usage, disk usage, number of connections, locks, and hotspot tables. For more information, see DAS overview. DAS allows you to identify existing and potential issues that may compromise the health of your database system. In addition, DAS provides details and solutions for the identified issues. This facilitates database maintenance.

ApsaraDB RDS for MySQL supports only two storage engines: InnoDB and X-Engine. For more information, see Features of ApsaraDB RDS for MySQL instances.

Basic terms

- Instance: An RDS instance is a database process that consumes independent physical memory resources. You can specify a specific memory size, disk capacity, and database type for an RDS instance. The performance of an RDS instance varies based on the specified memory size. After an RDS instance is created, you can change its specifications or delete the instance.
- Database: A database is a logical unit that is created on an RDS instance. One RDS instance can have multiple databases. Each database must have a unique name on the RDS instance where it is created.
- Region and zone: Each region is a physical data center. Each region contains a number of isolated locations that are known as zones. Each zone has an independent power supply and network. For more information, visit the Alibaba Cloud's Global Infrastructure page.

General terms

Term	Description
On-premises database	A database that is deployed in an on-premises data center or a database that is not deployed on an ApsaraDB for RDS instance.
ApsaraDB RDS for XX (XX represents one of the following database engines: MySQL, SQL Server, PostgreSQL, and MariaDB.)	ApsaraDB for RDS with a specific database engine. For example, ApsaraDB RDS for MySQL indicates an ApsaraDB for RDS instance that runs MySQL.

2.Limits

This topic describes the limits of ApsaraDB RDS for MySQL. Before you use ApsaraDB RDS for MySQL, we recommend that you take note of these limits to ensure the stability and security of your database system.

For more information about the limits of ApsaraDB RDS that is used together with a different database engine, see the following topics:

- Limits of ApsaraDB RDS for SQL Server
- Limits of ApsaraDB RDS for PostgreSQL
- Limits of ApsaraDB RDS for MariaDB TX

Limits on specifications and performance

Item	Specification	Description
Storage capacity	• RDS instances that use local SSDs: up to 6,000 GB. • RDS instances that use standard SSDs: up to 6,000 GB. • RDS instances that use enhanced SSDs (ESSDs): up to 32,000 GB.	The maximum storage capacity that is allowed for an RDS instance varies based on the instance type. For more information, see Primary ApsaraDB RDS instance types.
Temporary table size	• RDS instances that use local SSDs: up to 300 GB. • RDS instances that use standard SSDs or ESSDs: unlimited.	If the size of temporary tables in an RDS instance is larger than 300 GB, we recommend that you change the storage type to ESSDs. The maximum size of temporary tables in an RDS instance that uses local SSDs is 300 GB. If the size of temporary tables exceeds 300 GB, the RDS instance may encounter frequent high-availability switchovers and downtime issues.
Number of connections	Up to 100,000.	The maximum number of connections that are allowed for an RDS instance varies based on the instance type. For more information, see Primary ApsaraDB RDS instance types.
IOPS	• RDS instances that use local SSDs: up to 144,000. • RDS instances that use standard SSDs or ESSDs: For more information, see Maximum IOPS for standard SSDs and ESSDs.	None.

Item	Specification	Description
Memory capacity	• RDS instances that use local SSDs: up to 720 GB. • RDS instances that use standard SSDs or ESSDs: up to 768 GB.	For RDS instances that use standard SSDs or ESSDs, the memory includes the memory that is occupied by the RDS-related management services and the underlying operating system. Therefore, the available memory of an RDS instance may be less than the memory capacity that is supported by the instance type. The following list provides the amount of memory that is occupied by different components: • The underlying operating system occupies 500 MB to 700 MB of memory. • The RDS-related management services occupy approximately 500 MB of memory.

Limits on quotas

Item	Description
Read-only RDS instances	• If the memory capacity of a primary RDS instance is greater than or equal to 64 GB, up to 10 read-only RDS instances can be created and attached to the primary RDS instance. • If the memory capacity of a primary RDS instance is less than 64 GB, up to 5 read-only RDS instances can be created and attached to the primary RDS instance. For more information about read-only RDS instances, see Overview of read-only ApsaraDB RDS for MySQL instances.
Primary RDS instances	Up to 30 pay-as-you-go primary RDS instances can be created within each Alibaba Cloud account. You can go to the Quota Center to apply for a quota increase for your Alibaba Cloud account.
Tags	The key of a tag must be unique. You can add up to 20 tags to an RDS instance. You can add tags to up to 50 RDS instances at a time. For more information, see Add tags to ApsaraDB RDS instances.
Free quota for backup storage	• RDS instances that use local SSDs: Free quota for backup storage = 50% × Purchased storage capacity • RDS instances that use standard SSDs or ESSDs: Free quota for backup storage = 200% × Purchased storage capacity If your backup storage exceeds the free quota, you are charged for the excess backup storage that you use. You can calculate your excess backup storage by using the following formula: Excess backup storage = Size of data backup files + Size of log backup files - Free quota. Unit: GB. You can round the obtained result only up to the next integer. For more information, see Backup storage pricing of an ApsaraDB RDS for MySQL instance.

Item	Description
Backup retention period	The default retention period is 7 days, and the maximum retention period is 730 days. Data backup files that are retained for more than 730 days are archived. You are charged less for archived backup files than for regular backup files. For more information, see Backup storage pricing of an ApsaraDB RDS for MySQL instance.
Log retention period	• Error logs are retained for 30 days. • The details about slow query logs are retained for seven days. • The summary of slow query logs is retained for seven days. • Primary/secondary switchover logs are retained for 30 days. For more information, see View the logs of an ApsaraDB RDS for MySQL instance.

Limits on names

Item	Description
Instance names	• The name of an RDS instance must be 2 to 255 characters in length. • The name of an RDS instance can contain letters, digits, underscores (_), and hyphens (-). • The name of an RDS instance must start with a letter.
Usernames	• For RDS instances that run MySQL 8.0 or MySQL 5.7, the username of an account must be 2 to 32 characters in length. For RDS instances that run MySQL 5.6, the username of an account must be 2 to 16 characters in length. • The username of an account can contain lowercase letters, digits, and underscores (_). • The username of an account must start with a lowercase letter and end with a lowercase letter or a digit. • The username of an account must be unique. • The username of an account cannot contain SQL keywords. For more information, see SQL keywords.
Database names	• The name of a database can contain up to 64 characters in length. • The name of a database can contain lowercase letters, digits, underscores (_), and hyphens (-). • The name of a database must start with a lowercase letter and end with a lowercase letter or a digit. • The name of a database must be unique. • The username of an account cannot contain SQL keywords. For more information, see SQL keywords.
User-defined function (UDF) name	Do not use the reserved keywords in the name of a UDF. For more information, see Reserved keywords of an ApsaraDB RDS for MySQL instance.

Limits on security

Item	Description
Passwords	The password of an account must meet the following requirements: • The password of the account must be 8 to 32 characters in length. • The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. The password can contain the following special characters: `! @ # $ % ^ & * () _ + - =`
Ports	By default, an RDS instance is connected over port 3306. You can change the port number based on your business requirements. For more information, see View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance.
Disk encryption	You can enable disk encryption for an RDS instance only when you purchase the instance. Disk encryption cannot be disabled after it is enabled. For more information, see Configure the disk encryption feature for an ApsaraDB RDS for MySQL instance.
Number of security groups	You can configure up to 10 security groups for an RDS instance. • After you configure security groups for an RDS instance, the Elastic Compute Service (ECS) instances in the configured security groups can communicate with the RDS instance. • The security groups that you configured for an RDS instance must have the same network type as the RDS instance. This means that the network types of the RDS instance and the security groups that you want to configure must both be Virtual Private Cloud (VPC) or classic network. For more information, see Configure a security group for an ApsaraDB RDS for MySQL instance.
Number of IP address whitelists	You can configure up to 50 IP address whitelists for an RDS instance and add up to 1,000 IP addresses and CIDR blocks to an IP address whitelist. For more information, see Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance.

Account permissions	• The permissions of the root account or the system administrator account are not provided.
	• The XA_RECOVER_ADMIN permission is subject to the following limits:
	  ◦ The permission is supported only for RDS instances that run MySQL 8.0.
	  ◦ Only the accounts that you create in the ApsaraDB RDS console have the permission. The accounts that you create by using the CREATE USER statement in the Data Management (DMS) console or on a database client do not have the permission and cannot be granted the permission.

Privileged accounts	You can create and manage privileged accounts in the ApsaraDB RDS console or by using the ApsaraDB RDS API. The privileged account of an RDS instance has permissions to disconnect the database connections that are established by using standard accounts. Only one privileged account can be created for each RDS instance. For more information, see Create an account on an ApsaraDB RDS for MySQL instance.

Item	Description
Standard accounts	• You can create and manage standard accounts in the ApsaraDB RDS console. You can also use the ApsaraDB RDS API or execute SQL statements to create and manage standard accounts. • You cannot use a standard account to create and manage accounts, or disconnect the database connections that are established by using other accounts. • By default, a standard account has the permissions only on the database to which the standard account is connected. You must manually grant the permissions on specified databases to each standard account. For more information, see Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance. You can also execute the GRANT statement to grant the specified permissions to a standard account. For more information, see Account permissions.
Instance parameters	You can modify most of the instance parameters in the ApsaraDB RDS console or by using the ApsaraDB RDS API. For security and stability purposes, some parameters cannot be modified. For more information, see Modify the parameters of an ApsaraDB RDS for MySQL instance.

Limits on operations

Item	Description

Backup and restoration	• Limits on backups:
	  ◦ You can perform logical backups by using the CLI or the GUI.
	  ◦ You can perform physical backups in the ApsaraDB RDS console or by using the ApsaraDB RDS API.
	• Limits on restoration:
	  ◦ You can restore data from logical backup files by using the CLI or the GUI.
	  ◦ You can restore data from physical backup files in the ApsaraDB RDS console or by using the ApsaraDB RDS API.

Storage engines	Only the InnoDB storage engine and the X-Engine storage engine are supported. For more information about X-Engine, see Introduction to X-Engine. • The TokuDB storage engine is not supported. Percona no longer provides support for TokuDB. A large number of known issues in TokuDB cannot be fixed. In extreme cases, these issues may cause business loss. Therefore, ApsaraDB RDS for MySQL no longer supports TokuDB as of August 1, 2019. For more information about how to switch an RDS instance from TokuDB to a different storage engine, see [Notice] The storage engine was switched from TokuDB to InnoDB. • The MyISAM storage engine is not supported due to its inherent defects that may cause data loss. If you create a MyISAM table, ApsaraDB RDS converts the MyISAM table to an InnoDB table. For more information, see Why does ApsaraDB RDS for MySQL not support the MyISAM storage engine? • The MEMORY storage engine is not supported. If you create a Memory table, ApsaraDB RDS converts the MEMORY table to an InnoDB table.

Item	Description
Binlog	• By default, the binary logging feature is enabled. The binary logging feature cannot be disabled after it is enabled. • The value of the binlog_format parameter is fixed as ROW. This value cannot be changed.
Primary/secondary replication	ApsaraDB RDS for MySQL provides a primary/secondary replication architecture. In this architecture, a secondary RDS instance is provided as a hot standby for the primary RDS instance that you create. The secondary RDS instance is hidden and inaccessible. This architecture is not supported for RDS Basic Edition.
Instance restart	You can restart an RDS instance in the ApsaraDB RDS console or by using the ApsaraDB RDS API.
Network settings	If an RDS instance runs MySQL 5.5 or MySQL 5.6 in the classic network and the database proxy feature is enabled for the RDS instance, you cannot enable timestamps in SNAT mode by configuring the net.ipv4.tcp_timestamps parameter.
Storage	If the storage usage of an RDS instance is excessively high, ApsaraDB RDS locks the RDS instance to prevent data loss that may be caused by accidental operations. You can upgrade the instance type to unlock the RDS instance.
Size of a single table	The maximum size of a single table is 2 TB due to the maximum file size that is allowed by the operating system. For more information, see Limits on the size of a single table in ApsaraDB RDS for MySQL. **Note** We recommend that you make sure all tables in your RDS instance meet the following requirements to ensure optimal performance: • The number of data records in each table does not exceed 20 million. • The size of each table does not exceed 10 GB.
Help information	If an RDS instance runs MySQL 8.0, you can query detailed help information from the mysql.help_topic table. If an RDS instance does not run MySQL 8.0, the mysql.help_topic table is empty.
Public endpoints	If you want to connect to an RDS instance by using a public endpoint, you must manually apply for a public endpoint. For more information, see Apply for or release a public endpoint for an ApsaraDB RDS for MySQL instance.
Instance restart	You can restart an RDS instance in the ApsaraDB RDS console or by using the ApsaraDB RDS API.

3.Features

3.1. MySQL 8.0

This topic provides an overview of the features supported by ApsaraDB RDS instances that run MySQL 8.0. In the following table, ticks (√) indicate that a feature is supported, and crosses (×) indicate that a feature is not supported.

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Data migration	Overview of data migration methods	✔	✔	✔	✔	✔	✔
Data synchronization	Overview of data synchronization	✔	✔	✔	✔	✔	✔
	Create an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Change the specifications of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance	□	□	✔	✔	□	□

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Instance management	Enable the automatic scale-up feature for an ApsaraDB RDS for MySQL instance	□	✔☺	✔☺	✔☺	□	□
	Migrate an ApsaraDB RDS for MySQL instance across zones in the same region	□	✔☺	✔☺	✔☺	✔☺	✔☺
	Switch workloads over between a primary RDS instance and its secondary ApsaraDB RDS for MySQL instance	✔☺	✔☺	✔☺	✔☺	□	□
	Change the data replication mode of an ApsaraDB RDS for MySQL instance	□	✔☺	□	□	□	□

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Use a parameter template to manage the parameters of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Create a disaster recovery ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	□	□
	Restart an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Set the maintenance window of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Release an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Manage ApsaraDB RDS for MySQL instances in the recycle bin	✔	✔	✔	✔	✔	✔
Instance upgrade	Update the minor engine version of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Upgrade the major engine version of an ApsaraDB RDS for MySQL instance	□	□	□	□	□	□
	Upgrade an ApsaraDB RDS for MySQL instance from Basic Edition to High-availability Edition	□	□	□	□	✔	✔

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Account management	Create an account on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure a custom password policy for an ApsaraDB RDS for MySQL instance	□	□	□	□	□	□
	Reset the password of an account of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Modify the permissions of an account of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Grant permissions to the service account of an ApsaraDB RDS for MySQL instance	✔	✔	□	□	□	□
	Delete an account from an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Reset the permissions of the privileged account for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
Database management	Create a database on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Delete a database from an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Database connection	Connect to an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure endpoints for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Apply for a public endpoint for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Monitoring and alerting	View the resource metrics, engine metrics, and deployment metrics of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Set the monitoring frequency of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure an alert rule for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Change the network type of an ApsaraDB RDS for MySQL instance	✔	✔	□	□	□	□

Network management Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Switch an ApsaraDB RDS for MySQL instance to a new VPC and a new vSwitch	✔	✔	✔	✔	✔	✔
Read-only instance and read/write splitting	Create a read-only ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘
	Enable the read/write splitting feature for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘
	Change the network type of the read/write splitting endpoint of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Security management	Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Switch an ApsaraDB RDS for MySQL instance to the enhanced whitelist mode	□	□	□	□	□	□
	Configure SSL encryption for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	□	□
	Configure TDE for an ApsaraDB RDS for MySQL instance	□	✔	□	□	□	□

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Enable or disable the release protection feature for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure disk encryption for an ApsaraDB RDS for MySQL instance	✘	✘	✔	✔	✘	✘
Audit	Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘
	Manage the logs of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	View the event history of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
Backup	Back up an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Free quota for backup storage for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Download the backup files of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Enable cross-region backups for an ApsaraDB RDS for MySQL instance	□	✔	□	□	□	□

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Restoration	Restore the data of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Restore individual databases and tables of an ApsaraDB RDS for MySQL instance	□	✔	□	□	□	□
	Restore the data of an ApsaraDB RDS for MySQL instance across regions	□	✔	□	□	□	□
Dedicated proxy	Use the dedicated proxy feature on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	□	□
Diagnosis and optimization	DAS overview	✔	✔	✔	✔	□	□
AliSQL	AliSQL	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 8.0					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Tag management	Add tags to ApsaraDB RDS instances	✔	✔	✔	✔	✔	✔
	Remove tags from an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Use tags to filter ApsaraDB RDS for MySQL instances	✔	✔	✔	✔	✔	✔

3.2. MySQL 5.7

This topic provides an overview of the features supported by ApsaraDB RDS instances that run MySQL 5.7. In the following table, ticks (√) indicate that a feature is supported, and crosses (×) indicate that a feature is not supported.

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Data migration	Overview of data migration methods	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Data synchronization	Overview of data synchronization	✔	✔	✔	✔	✔	✔
	Create an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Change the specifications of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance	□	□	✔	✔	□	□
	Enable the automatic scale-up feature for an ApsaraDB RDS for MySQL instance	□	✔	✔	✔	□	□

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Instance management	Migrate an ApsaraDB RDS for MySQL instance across zones in the same region	✗	✔	✔	✔	✔	✔
	Switch workloads over between a primary RDS instance and its secondary ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✗	✗
	Change the data replication mode of an ApsaraDB RDS for MySQL instance	✗	✔	✗	✗	✗	✗

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Use a parameter template to manage the parameters of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Create a disaster recovery ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	□	□
	Restart an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Set the maintenance window of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Release an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Manage ApsaraDB RDS for MySQL instances in the recycle bin	✔	✔	✔	✔	✔	✔
Instance upgrade	Update the minor engine version of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Upgrade the major engine version of an ApsaraDB RDS for MySQL instance	□	□	□	□	□	□
	Upgrade an ApsaraDB RDS for MySQL instance from Basic Edition to High-availability Edition	□	□	□	□	✔	✔

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Account management	Create an account on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure a custom password policy for an ApsaraDB RDS for MySQL instance	□	✔	✔	✔	✔	✔
	Reset the password of an account of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Modify the permissions of an account of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Grant permissions to the service account of an ApsaraDB RDS for MySQL instance	✔	✔	□	□	□	□
	Delete an account from an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Reset the permissions of the privileged account for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
Database management	Create a database on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Delete a database from an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Database connection	Connect to an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure endpoints for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Apply for a public endpoint for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Monitoring and alerting	View the resource metrics, engine metrics, and deployment metrics of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Set the monitoring frequency of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure an alert rule for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Change the network type of an ApsaraDB RDS for MySQL instance	✔	✔	□	□	✔	✔

Network management Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Switch an ApsaraDB RDS for MySQL instance to a new VPC and a new vSwitch	✔	✔	✔	✔	✔	✔
Read-only instance and read/write splitting	Create a read-only ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	-	-
	Enable the read/write splitting feature for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	-	-
	Change the network type of the read/write splitting endpoint of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	-	-

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Security management	Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Switch an ApsaraDB RDS for MySQL instance to the enhanced whitelist mode	✔	✔	✔	✔	✘	✘
	Configure SSL encryption for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘
	Configure TDE for an ApsaraDB RDS for MySQL instance	✔	✔	✘	✘	✘	✘

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	Enable or disable the release protection feature for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Configure disk encryption for an ApsaraDB RDS for MySQL instance	✘	✘	✔	✔	✘	✘
Audit	Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘
	Manage the logs of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
	View the event history of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
Backup	Back up an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Free quota for backup storage for an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Download the backup files of an ApsaraDB RDS for MySQL instance	✔	✔	□	□	□	□
	Enable cross-region backups for an ApsaraDB RDS for MySQL instance	□	✔	□	□	□	□

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Restoration	Restore the data of an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Restore individual databases and tables of an ApsaraDB RDS for MySQL instance	✔	✔	✘	✘	✘	✘
	Restore the data of an ApsaraDB RDS for MySQL instance across regions	✘	✔	✘	✘	✘	✘
Dedicated proxy	Use the dedicated proxy feature on an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✘	✘
Diagnosis and optimization	DAS overview	✔	✔	✔	✔	✘	✘
AliSQL	AliSQL	✔	✔	✔	✔	✔	✔

Category	Feature	MySQL 5.7					
		RDS Enterprise Edition	RDS High-availability Edition			RDS Basic Edition	
		Local SSD	Local SSD	Standard SSD	ESSD	Standard SSD	ESSD
Tag management	Add tags to ApsaraDB RDS instances	✔	✔	✔	✔	✔	✔
	Remove tags from an ApsaraDB RDS for MySQL instance	✔	✔	✔	✔	✔	✔
	Use tags to filter ApsaraDB RDS for MySQL instances	✔	✔	✔	✔	✔	✔

3.3. MySQL 5.6

This topic provides an overview of the features that are supported by ApsaraDB RDS instances that run MySQL 5.6. In the following tables, ticks (√) indicate that a feature is supported and crosses (×) indicate that a feature is not supported.

Category	Feature	Features of ApsaraDB RDS for MySQL
		RDS High-availability Edition
		Local SSD
Data migration	Overview of data migration methods	✔
Data synchronization	Overview of data synchronization	✔
	Create an ApsaraDB RDS for MySQL instance	✔
	Change the specifications of an ApsaraDB RDS for MySQL instance	✔

Category	Feature	Features of ApsaraDB RDS for MySQL	RDS High-availability Edition	Local SSD
Instance management	Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance	□		
	Enable the automatic scale-up feature for an ApsaraDB RDS for MySQL instance	✔		
	Migrate an ApsaraDB RDS for MySQL instance across zones in the same region	✔		
	Switch workloads over between primary and secondary ApsaraDB RDS for MySQL instances	✔		
	Change the data replication mode of an ApsaraDB RDS for MySQL instance	✔		
	Use a parameter template to manage the parameters of an ApsaraDB RDS for MySQL instance	✔		
	Restart an ApsaraDB RDS for MySQL instance	✔		
	Set the maintenance window of an ApsaraDB RDS for MySQL instance	✔		
	Release or unsubscribe from an ApsaraDB RDS for MySQL instance	✔		
	Manage ApsaraDB RDS for MySQL instances in the recycle bin	✔		
Instance upgrade	Update the minor engine version of an ApsaraDB RDS for MySQL instance	✔		
	Upgrade the major engine version of an ApsaraDB RDS for MySQL instance	✔		
	Create an account on an ApsaraDB RDS for MySQL instance	✔		

Category	Feature	Features of ApsaraDB RDS for MySQL RDS High-availability Edition Local SSD
Account management	Reset the password of an account of an ApsaraDB RDS for MySQL instance	✔
	Modify the permissions of a standard account of an ApsaraDB RDS for MySQL instance	✔
	Authorize the service account of an ApsaraDB RDS for MySQL instance	✔
	Delete a standard account from an ApsaraDB RDS for MySQL instance	✔
	Reset the permissions of the privileged account of an ApsaraDB RDS for MySQL instance	✔
	Configure a custom password policy for an ApsaraDB RDS for MySQL instance	□
Database management	Create a database on an ApsaraDB RDS for MySQL instance	✔
	Delete a database from an ApsaraDB RDS for MySQL instance	✔
Database connection	Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance	✔
	Configure an endpoint for an ApsaraDB RDS for MySQL instance	✔
	View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance	✔
	Apply for or release a public endpoint for an ApsaraDB RDS for MySQL instance	✔

Category	Feature	Features of ApsaraDB RDS for MySQL RDS High-availability Edition Local SSD
Monitoring and altering	View the resource, engine, and deployment metrics of an ApsaraDB RDS for MySQL instance	✔
	Set the monitoring frequency of an ApsaraDB RDS for MySQL instance	✔
	Configure an alert rule for an ApsaraDB RDS for MySQL instance	✔
Network management	Change the network type of an ApsaraDB RDS for MySQL instance	✔
	Migrate an ApsaraDB RDS for MySQL instance to a new VPC and a new vSwitch	✔
Read-only instance and read/write splitting	Create a read-only ApsaraDB RDS for MySQL instance	✔
	Enable the read/write splitting feature for an ApsaraDB RDS for MySQL instance (shared proxy)	✔
	Change the network type of the read/write splitting endpoint of an ApsaraDB RDS for MySQL instance	✔
Security management	Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance	✔
	Switch the network isolation mode of an ApsaraDB RDS for MySQL instance to the enhanced whitelist mode	✔
	Configure SSL encryption on an ApsaraDB RDS for MySQL instance	✔
	Configure TDE for an ApsaraDB RDS for MySQL instance	✔
	Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance	✔

Audit Category	Feature	Features of ApsaraDB RDS for MySQL	RDS High-availability Edition	Local SSD
	View the logs of an ApsaraDB RDS for MySQL instance	✔		
Backup	Enable automatic backups for an ApsaraDB RDS for MySQL instance	✔		
	Backup storage pricing for an ApsaraDB RDS for MySQL instance	✔		
	Download the backup files of an ApsaraDB RDS for MySQL instance	✔		
	Enable cross-region backups for an ApsaraDB RDS for MySQL instance	✔		
Restoration	Restore the data of an ApsaraDB RDS for MySQL instance	✔		
	Restore specific individual databases or tables of an RDS instance	✔		
	Restore the individual databases and tables of an ApsaraDB RDS for MySQL instance	✔		
Dedicated proxy	Introduction to database proxies	□		
AliSQL	Overview of AliSQL features	✔		
Tag management	Add tags to ApsaraDB RDS instances	✔		
	Remove tags from an ApsaraDB RDS for MySQL instance	✔		
	Use tags to filter ApsaraDB RDS for MySQL instances	✔		

3.4. MySQL 5.5

This topic provides an overview of the features supported by ApsaraDB RDS instances that run MySQL 5.5. In the following table, ticks (√) indicate that a feature is supported, and crosses (×) indicate that a feature is not supported.

Category	Feature	MySQL 5.5	RDS High-availability Edition	Local SSD
Data migration	Overview of data migration methods	✔	✔	✔
Data synchronization	Overview of data synchronization	✔	✔	✔
Instance management	Create an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Change the specifications of an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Enable the automatic scale-up feature for an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Migrate an ApsaraDB RDS for MySQL instance across zones within the same region	✔	✔	✔
	Switch workloads over between primary and secondary ApsaraDB RDS for MySQL instances	✔	✔	✔
	Change the data replication mode of an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Use a parameter template to manage the parameters of an ApsaraDB RDS for MySQL instance	□	✔	✔
	Create a disaster recovery ApsaraDB RDS for MySQL instance	□	✔	✔
	Restart an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Set the maintenance window of an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Release an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Manage ApsaraDB RDS for MySQL instances in the recycle bin	✔	✔	✔

Category	Feature	MySQL 5.5 RDS High-availability Edition Local SSD
Instance upgrade	Update the minor engine version of an ApsaraDB RDS for MySQL instance	✔
	Upgrade the major engine version of an ApsaraDB RDS for MySQL instance	✔
Account management	Create an account on an ApsaraDB RDS for MySQL instance	✔
	Reset the password of an account of an ApsaraDB RDS for MySQL instance	✔
	Modify the permissions of an account of an ApsaraDB RDS for MySQL instance	✔
	Grant permissions to the service account of an ApsaraDB RDS for MySQL instance	✔
	Delete an account from an ApsaraDB RDS for MySQL instance	✔
	Reset the permissions of the privileged account of an ApsaraDB RDS for MySQL instance	✔
	Configure a custom password policy for an ApsaraDB RDS for MySQL instance	□
Database management	Create a database on an ApsaraDB RDS for MySQL instance	✔
	Delete a database from an ApsaraDB RDS for MySQL instance	✔
	Connect to an ApsaraDB RDS for MySQL instance	✔
	Configure endpoints for an ApsaraDB RDS for MySQL instance	✔

Database connection Category	Feature	MySQL 5.5	RDS High-availability Edition	Local SSD
	View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Apply for a public endpoint for an ApsaraDB RDS for MySQL instance	✔	✔	✔
Monitoring and alerting	View the resource metrics, engine metrics, and deployment metrics of an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Set the monitoring frequency of an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Configure an alert rule for an ApsaraDB RDS for MySQL instance	✔	✔	✔
Network management	Change the network type of an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Switch an ApsaraDB RDS for MySQL instance to a new VPC and a new vSwitch	✔	✔	✔
Read-only instances and read/write splitting	Create a read-only ApsaraDB RDS for MySQL instance	□	□	□
	Enable the read/write splitting feature for an ApsaraDB RDS for MySQL instance	□	□	□
	Change the network type of the read/write splitting endpoint for an ApsaraDB RDS for MySQL instance	□	□	□
	Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance	✔	✔	✔
	Switch an ApsaraDB RDS for MySQL instance to the enhanced whitelist mode	✔	✔	✔

Security management Category	Feature	MySQL 5.5 RDS High-availability Edition Local SSD
	Configure SSL encryption for an ApsaraDB RDS for MySQL instance	□
	Configure TDE for an ApsaraDB RDS for MySQL instance	□
Audit	Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance	✔
	Manage the logs of an ApsaraDB RDS for MySQL instance	✔
Backup	Back up an ApsaraDB RDS for MySQL instance	✔
	Free quota for backup storage of an ApsaraDB RDS for MySQL instance	✔
	Download the backup files of an ApsaraDB RDS for MySQL instance	✔
	Enable cross-region backups for an ApsaraDB RDS for MySQL instance	□
Restoration	Restore the data of an ApsaraDB RDS for MySQL instance	✔
	Restore the individual databases and tables of an ApsaraDB RDS for MySQL instance	□
	Restore the data of an ApsaraDB RDS for MySQL instance across regions	□
Dedicated proxy	Use the dedicated proxy feature on an ApsaraDB RDS for MySQL instance	□
AliSQL	AliSQL	□
	Add tags to ApsaraDB RDS instances	✔
	Remove tags from an ApsaraDB RDS for MySQL instance	✔

Tag management	Feature	MySQL 5.5
		RDS High-availability Edition
		Local SSD
	Use tags to filter ApsaraDB RDS for MySQL instances	✔

4.Specifications

4.1. Primary ApsaraDB RDS for MySQL instance types

This topic provides an overview of primary ApsaraDB RDS for MySQL instance types, which include the most recent and earlier instance types. The overview includes the specifications for each instance type.

> **Note** Some instances may no longer be available. You can select only the instance types that are available on the ApsaraDB RDS buy page.

4.2. Read-only ApsaraDB RDS for MySQL instance types

This topic provides an overview of read-only ApsaraDB RDS for MySQL instance types. This overview includes the most recent instance types and the specifications for each instance type.

For more information about how to create a read-only ApsaraDB RDS for MySQL instance, see Create a read-only ApsaraDB RDS for MySQL instance.

> **Note** The subscription and pay-as-you-go billing methods are supported for read-only ApsaraDB RDS for MySQL instances. For more information, see Pricing. For more information about the prices, go to the ApsaraDB RDS buy page.

Read-only ApsaraDB RDS for MySQL instances with local SSDs

Role	MySQL version	Instance family	Instance type	CPU and memory specifications	Maximum number of connections	Maximum IOPS	Storage capacity
			rds.mysql.t1.small	1 core, 1 GB	300	600	
			rds.mysql.s1.small	1 core, 2 GB	600	1,000	
			rds.mysql.s2.large	2 cores, 4 GB	1,200	2,000	
			rds.mysql.s2.xlarge	2 cores, 8 GB	2,000	4,000	
			rds.mysql.s3.large	4 cores, 8 GB	2,000	5,000	
							5 GB to

Role	MySQL version	General-purpose instance family	Instance type	CPU and memory specifications	Maximum number of connections	Maximum IOPS	2,000 GB Storage capacity
Read-only instance	MySQL 8.0, MySQL 5.7, and MySQL 5.6		rds.mysql.m1.medium	4 cores, 16 GB	4,000	7,000	
			rds.mysql.c1.large	8 cores, 16 GB	4,000	8,000	
			rds.mysql.c1.xlarge	8 cores, 32 GB	8,000	12,000	
			rds.mysql.c2.xlarge	16 cores, 64 GB	16,000	14,000	5 GB to 3,000 GB
			rds.mysql.c2.xlp2	16 cores, 96 GB	24,000	16,000	
		Dedicated instance (with a large memory capacity)	mysqlro.x8.medium.1	2 cores, 16 GB	2,500	4,500	50 GB to 2,000 GB
			mysqlro.x8.large.1	4 cores, 32 GB	5,000	9,000	50 GB to 2,000 GB
			mysqlro.x8.xlarge.1	8 cores, 64 GB	10,000	18,000	500 GB to 3,000 GB
			mysqlro.x8.2xlarge.1	16 cores, 128 GB	20,000	36,000	500 GB to 3,000,GB
			mysqlro.x8.4xlarge.1	32 cores, 256 GB	40,000	72,000	1,000 GB to 6,000 GB
			mysqlro.x8.8xlarge.1	64 cores, 512 GB	80,000	144,000	1,000 GB to 6,000 GB
		Dedicated instance family (with a large	mysqlro.x4.large.1	4 cores, 16 GB	2,500	4,500	50 GB to 2,000GB
			mysqlro.x4.xlarge.1	8 cores, 32 GB	5,000	9,000	500 GB to 3,000 GB
			mysqlro.x4.2xlarge.1	16 cores, 64 GB	10,000	18,000	500 GB to 3,000 GB

Role	MySQL version	large number of cores) Instance family	Instance type	CPU and memory specifications	Maximum number of connections	Maximum IOPS	Storage capacity
			mysqlro.x4.4xlarge.1	32 cores, 128 GB	20,000	36,000	1,000 GB to 6,000 GB
		Dedicated host instance family	rds.mysql.st.h43	60 cores, 470 GB	100,000	120,000	3,000 GB or 6,000 GB
			rds.mysql.st.v52	90 cores, 720 GB	150,000	140,000	1,000 GB to 6,000 GB

Read-only ApsaraDB RDS for MySQL instances with standard SSDs or enhanced SSDs (ESSDs)

Role	MySQL version	Instance family	Instance type	CPU and memory specifications	Maximum number of connections	Storage		Storage capacity
						Maximum IOPS	Maximum I/O bandwidth (Mbit/s)	
		General-purpose instance family	mysqlro.n2.small.1c	1 core, 2 GB	2,000	10,000	1,024	
			mysqlro.x2.medium.1c	2 cores, 4 GB	4,000	10,000	1,024	
			mysqlro.x2.large.1c	4 cores, 8 GB	6,000	20,000	1,536	
			mysqlro.x2.xlarge.1c	8 cores, 16 GB	8,000	25,000	2,048	
			mysqlro.x2.3large.1c	12 cores, 24 GB	12,000	30,000	2,560	
			mysqlro.x2.2xlarge.1c	16 cores, 32 GB	16,000	40,000	3,072	

Role	MySQL version	Instance family	Instance type	CPU and memory specifications	Maximum number of connections	Storage		Storage capacity
						Maximum IOPS	Maximum I/O bandwidth (Mbit/s)	
Read-only instance	MySQL 8.0, MySQL 5.7, and MySQL 5.6	Dedicated instance family	mysqlro.x2.3xlarge.1c	24 cores, 48 GB	24,000	50,000	4,096	Read-only ApsaraDB RDS for MySQL instances with standard SSDs: 20 GB to 6,000 GB
			mysqlro.x2.4xlarge.1c	32 cores, 64 GB	32,000	60,000	5,120	
			mysqlro.x2.13large.1c	52 cores, 96 GB	52,000	100,000	8,192	
			mysqlro.x2.13xlarge.1c	104 cores, 192 GB	104,000	200,000	16,384	
			mysqlro.x4.medium.1c	2 cores, 8 GB	6,000	10,000	1,024	
			mysqlro.x4.large.1c	4 cores, 16 GB	8,000	20,000	1,536	
			mysqlro.x4.xlarge.1c	8 cores, 32 GB	10,000	25,000	2,048	
			mysqlro.x4.3large.1c	12 cores, 48 GB	15,000	50,000	4,096	
			mysqlro.x4.2xlarge.1c	16 cores, 64 GB	20,000	40,000	3,072	
			mysqlro.x4.3xlarge.1c	24 cores, 96 GB	30,000	50,000	4,096	
			mysqlro.x4.4xlarge.1c	32 cores, 128 GB	40,000	60,000	5,120	
			mysqlro.x4.13large.1c	52 cores, 192 GB	65,000	100,000	8,192	
			mysqlro.x4.13xlarge.1c	104 cores, 384 GB	130,000	200,000	16,384	
			mysqlro.x8.medium.1c	2 cores, 16 GB	8,000	10,000	1,024	
			mysqlro.x8.large.1c	4 cores, 32 GB	12,000	20,000	1,536	

Role	MySQL version	Instance family	Instance type	CPU and memory specifications	Maximum number of connections	Storage		Storage capacity
						Maximum IOPS	Maximum I/O bandwidth (Mbit/s)	
			mysqlro.x8.xlarge.1c	8 cores, 64 GB	16,000	25,000	2,048	
			mysqlro.x8.3large.1c	12 cores, 96 GB	24,000	30,000	2,560	
			mysqlro.x8.2xlarge.1c	16 cores, 128 GB	32,000	40,000	3,072	
			mysqlro.x8.3xlarge.1c	24 cores, 192 GB	48,000	50,000	4,096	
			mysqlro.x8.4xlarge.1c	32 cores, 256 GB	64,000	60,000	5,120	
			mysqlro.x8.13large.1c	52 cores, 384 GB	104,000	100,000	8,192	
			mysqlro.x8.8xlarge.1c	64 cores, 512 GB	128,000	120,000	10,240	
			mysqlro.x8.13xlarge.1c	104 cores, 768 GB	208,000	200,000	16,384	

5.Quick start

5.1. General workflow to use ApsaraDB RDS for MySQL

This topic walks you through the general workflow of creating an ApsaraDB RDS for MySQL instance, configuring the basic parameters of the RDS instance, and connecting to the RDS instance.

General workflow

To use ApsaraDB RDS for MySQL, you must perform three steps. For more information, see the following topics:

1. Create an ApsaraDB RDS for MySQL instance
2. Create databases and accounts for an ApsaraDB RDS for MySQL instance
3. Use DMS to log on to an ApsaraDB RDS for MySQL instance

More operations

- Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance
- Migrate data from a self-managed MySQL database to an ApsaraDB RDS for MySQL instance

> **Note**
>
> For more information about the general workflow to use an RDS instance that runs a different database engine, see the following topics:
>
> - General workflow to use ApsaraDB RDS for SQL Server
> - General workflow to use ApsaraDB RDS for PostgreSQL
> - General workflow to use ApsaraDB RDS for MariaDB TX

5.2. Get ready to use ApsaraDB RDS for MySQL

This topic describes how to get ready before you use ApsaraDB RDS for MySQL. An RDS instance is a virtual database server. can communicate with RDS instances, and the data of application servers can be stored in RDS instances. Before you create an RDS instance or connect an application server to an RDS instance, you must obtain the information about the application server. Application servers

Procedure

1. Check whether your application has been deployed or will be deployed on an Alibaba Cloud service such as an .
   - If your application has been deployed or will be deployed on an ECS instance, go to Step 2.
   - If your application has not been deployed or will not be deployed on an ECS instance and you do not want to migrate data to an RDS instance, skip the following operations. In this case, create

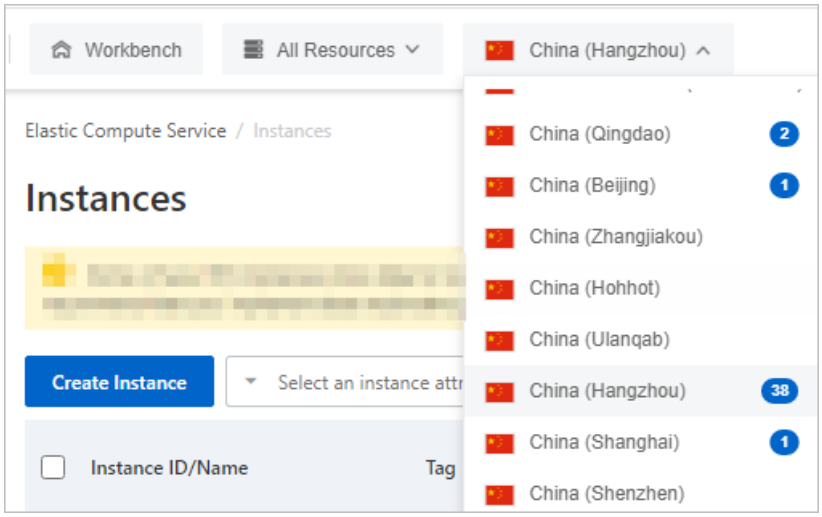
an RDS instance and connect to the RDS instance over the Internet.

> **Note** Communication over an internal network provides higher security, performance, and stability than communication over the Internet.

Elastic Compute Service (ECS) instance

2. **Confirm the ECS instance.**
   i. Log on to the ECS console. In the left-side navigation pane, click Instances. In the top navigation bar, select the region where the ECS instance resides. The number in a blue circle indicates the number of ECS instances that reside in the region.

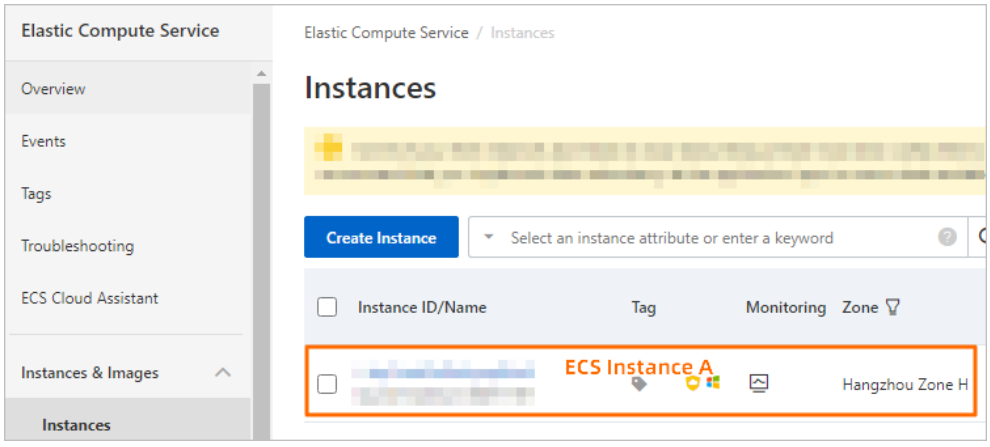
      Select a region

   > **Note** If no ECS instance is created, you must create an ECS instance. For more information, see Create and manage an ECS instance by using the ECS console.

   ii. Find the ECS instance on which your application has been deployed or will be deployed. This instance is **the ECS instance that you want to connect to the RDS instance**. The following figure shows a sample ECS instance.

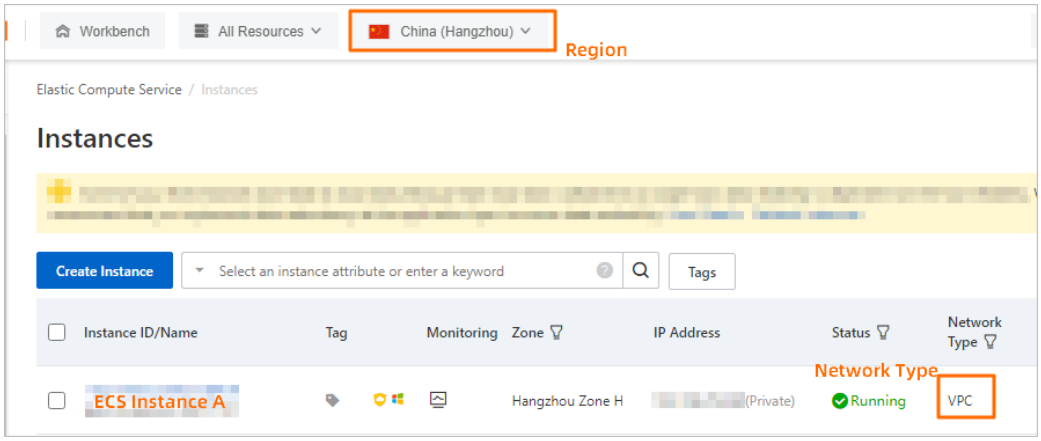
      Confirm the ECS instance

3. **View the region and**

> **Note** If the ECS instance resides in the classic network, we recommend that you migrate the ECS instance to a VPC. For more information, see Migrate ECS instances from the classic network to a VPC.

View the region and network type of the ECS instance

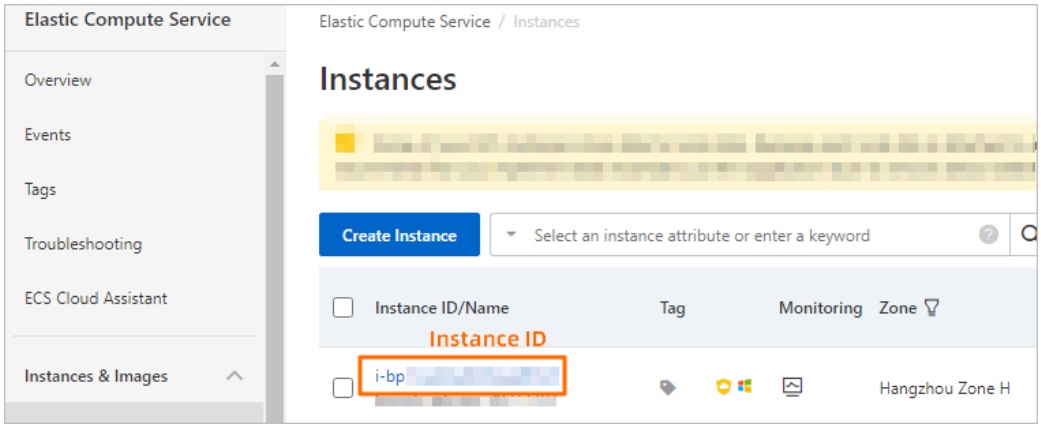


network type

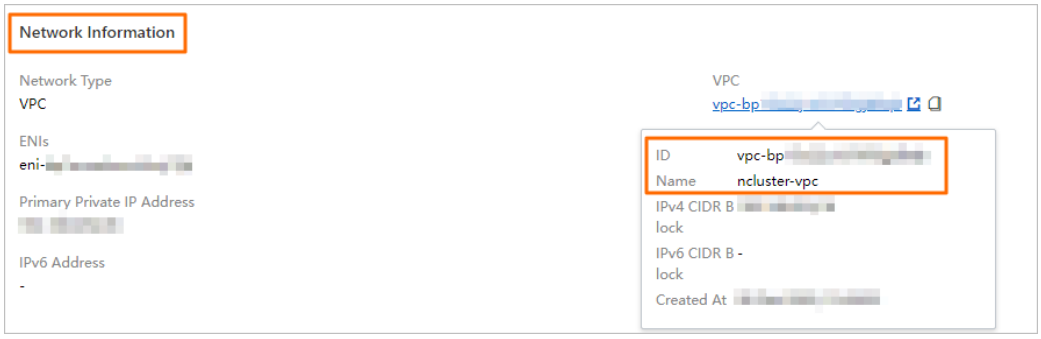
4. **View the VPC information of the ECS instance.**

   If the ECS instance resides in a VPC, click the Instance ID and go to the Instance Details tab. You can view the ID and name of the VPC in the **Network Information** section of the tab.

   Click the Instance ID



View the ID and name of the VPC



What to do next

Create an ApsaraDB RDS for MySQL instance

5.3. Create an ApsaraDB RDS for MySQL instance

This topic describes how to create an ApsaraDB RDS for MySQL instance.

> **Note** You are offered a reduced price on your first purchase of an RDS instance. For more information, visit the ApsaraDB RDS promotion page.

Prerequisites

The AliyunRDSFullAccess policy is attached to the RAM user that you used to create the RDS instance. For more information, see Use RAM for resource authorization.

Procedure

1. Go to the ApsaraDB RDS buy page.
2. Configure the **Billing Method** parameter.

Billing method	Description	Benefit
Subscription	A subscription instance is an instance for which you pay an upfront fee. If you want to use an instance for a long period of time, we recommend that you select the **Subscription** billing method. If you select the subscription billing method, configure the **Duration** parameter in the lower part of the page.	In most cases, the subscription billing method is more cost-effective than the pay-as-you-go billing method for long-term usage. Alibaba Cloud provides lower prices for longer subscription periods.
Pay-As-You-Go	You are charged on an hourly basis for a pay-as-you-go instance based on your actual resource usage. If you want to use an instance for a short period of time, we recommend that you select the **Pay-As-You-Go** billing method. You can create a **pay-as-you-go** RDS instance. After you confirm that the new RDS instance meets your business requirements, you can change the billing method of the RDS instance from pay-as-you-go to **subscription**.	You can release a pay-as-you-go RDS instance based on your business requirements. The billing cycle of a pay-as-you-go RDS instance immediately stops after you release the instance.

> **Note** You can view the price in the lower-right corner of the page. The price is displayed only after you configure all required parameters.

3. Configure the **Region** parameter.

   We recommend that you use an RDS instance that resides in the same region as on which your application is deployed. If the RDS instance and the ECS instance reside in different regions, you cannot connect these instances over an internal network. In this case, these instances cannot

deliver the optimal performance.

> **Note**
>
> - After an RDS instance is created, you cannot change the region of the RDS instance. If you want to connect an ECS instance and an RDS instance over an internal network, make sure that the RDS instance and the ECS instance reside in the same region.
> - For more information about how to view the region in which an ECS instance resides, see Get ready to use ApsaraDB RDS for MySQL.
> - If your application is deployed on an on-premises server or on-premises computer, we recommend that you select a region that is near your on-premises server or on-premises computer. This way, you can use the public endpoint of the RDS instance to connect to the RDS instance from your application.

the Elastic Compute Service (ECS) instance

4. Configure the **Database Engine** parameter.

   In this example, select **MySQL**.

   We recommend that you select MySQL 8.0 or MySQL 5.7 or select the database engine version that your self-managed MySQL instance runs. The default value of this parameter is 8.0.

5. Configure the **Edition** parameter. The default value of this parameter is **High-availability**.

Edition	Description	Benefit
Basic	In RDS Basic Edition, the database system consists of only a primary RDS instance.	RDS Basic Edition is cost-effective and is suitable for learning and testing scenarios. RDS instances that run RDS Basic Edition require a long period of time to restart or recover from faults.
High-availability	This is the recommended RDS edition. In RDS High-availability Edition, the database system consists of a primary RDS instance and a secondary RDS instance. You can create read-only RDS instances and attach the read-only RDS instances to the primary RDS instance.	RDS High-availability Edition is suitable for more than 80% of business scenarios that require production environments.
Enterprise	In RDS Enterprise Edition, the database system consists of a primary RDS instance, a secondary RDS instance, and a logger RDS instance. You can create read-only RDS instances and attach the read-only RDS instances to the primary RDS instance.	RDS Enterprise Edition is suitable for financial institutions that have high requirements for reliability.

> **Note**
>
> - The available RDS editions vary based on the region and database engine version that you select. If you select MySQL 5.6, Basic is not displayed.
> - For more information, see Overview of ApsaraDB RDS editions.

6. Configure the **Storage Type** parameter.

Comparison item	**ESSD** (recommended)	**Local SSD**
Scalability	★★★★★ ○ You can increase the storage capacity up to 32 TB. ○ No transient connections occur during storage expansion. ○ You can upgrade, downgrade, create, or release the RDS instance in minutes. ○ Automatic storage expansion is supported.	★★ ○ You can increase the storage capacity up to 6 TB. ○ Transient connections occur during storage expansion. ○ A few hours may be required to upgrade, downgrade, create, or release the RDS instance. ○ Automatic storage expansion is not supported.
Performance	★★★★★ ○ PL1<PL2<PL3 ○ An enhanced SSD (ESSD) of performance level 2 (PL2) provides twice the IOPS and throughput of an ESSD of performance level 1 (PL1). ○ An ESSD of PL3 provides 20 times the IOPS and 11 times the throughput of an ESSD of PL1.	★★★★★
Backup	★★★★★ ○ A backup can be completed in minutes or seconds. ○ The highest backup frequency is one backup every 15 minutes.	★★★ ○ A backup requires a long period of time to complete. ○ The highest backup frequency is one backup every day.

> **Note**
>
> - RDS instances that run MySQL 5.7 or MySQL 8.0 on RDS Basic Edition support only standard SSDs and ESSDs. RDS instances that run MySQL 8.0 or MySQL 5.7 on RDS Enterprise Edition support only local SSDs. RDS instances that run MySQL 5.6 or MySQL 5.5 on RDS High-availability Edition support only local SSDs. Serverless RDS instances support only standard SSDs and ESSDs.
> - The available features vary based on the storage type that you select. For more information, see MySQL 8.0.
> - For more information about different types of storage media, see Features.

7. Select the . zone
   - Select a zone.
     - No significant differences exist between the zones in the same region.
     - If the RDS instance resides in the same zone as the ECS instance on which your application is deployed, these instances can provide the optimal performance. If the RDS instance and the ECS instance reside in different zones in the same region, the performance of the RDS instance and the ECS instance is slightly lower than the performance of the RDS instance and the ECS instance that reside in the same zone.
   - Select a deployment method.
     - **Multi-zone Deployment**: The RDS instance and its secondary RDS instance reside in different zones to allow users to perform zone-disaster recovery. This is the recommended deployment method.
     - **Single-zone Deployment**: The RDS instance and its secondary RDS instance reside in the same zone.

     > **Note** If you select **Basic** for the Edition parameter, only the **Single-zone Deployment** method is supported.

8. Configure the **Instance Type** parameter.

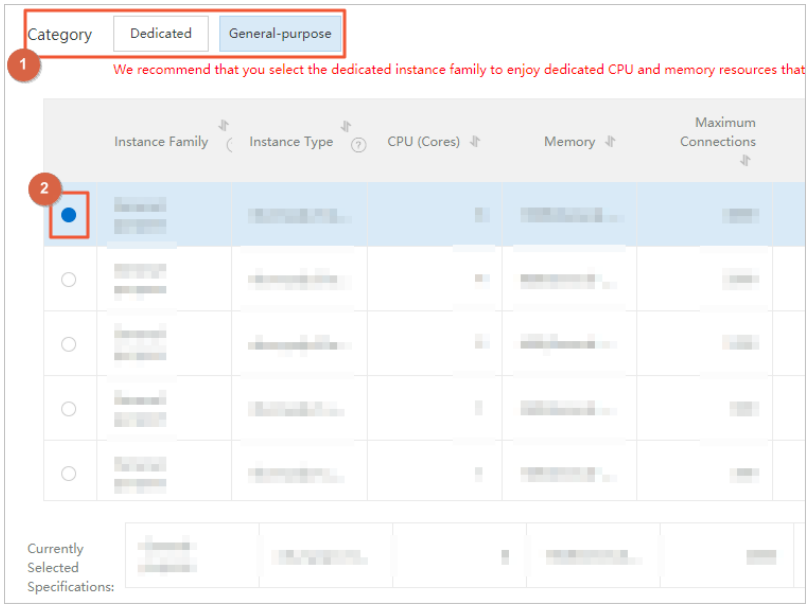
i. Select an **instance family**. You can select General-purpose or Dedicated.

Instance family	Description	Benefit
General-purpose	A general-purpose RDS instance occupies all the allocated memory and I/O resources. A general-purpose RDS instance shares CPU and storage resources with other general-purpose RDS instances that are deployed on the same host.	General-purpose RDS instances are cost-effective.
Dedicated	A dedicated RDS instance occupies all the allocated CPU, memory, storage, and I/O resources. **Note** The dedicated host instance family is the highest configuration of the dedicated instance family. A dedicated host RDS instance occupies all CPU, memory, storage, and I/O resources of the host on which the RDS instance is deployed.	A dedicated RDS instance provides higher performance and higher stability. **Note** An RDS instance that runs RDS Basic Edition does not support the dedicated instance family.

ii. Select an instance type.

- In a test environment, select an instance type that provides one or more CPU cores.
- In a production environment, select an instance type that provides four or more CPU cores.

Note For more information, see Primary ApsaraDB RDS for MySQL instance types.



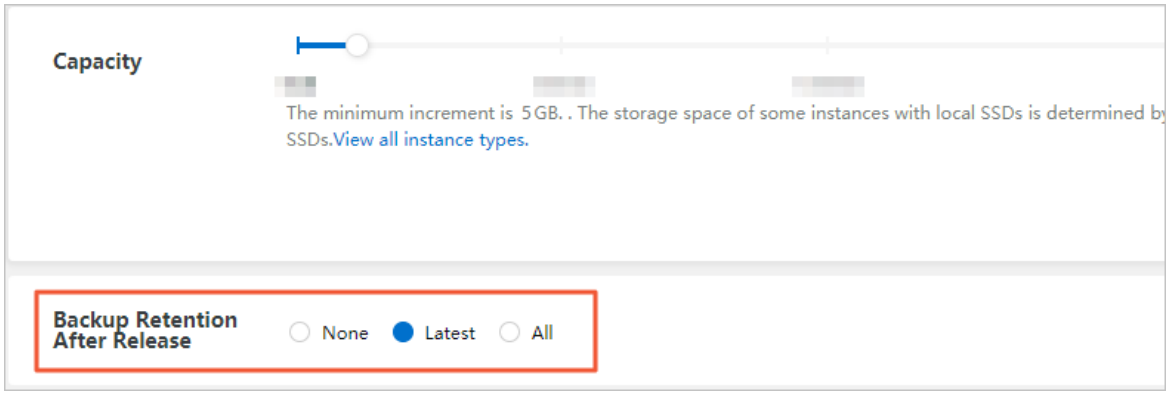
9. Configure the **Capacity** parameter.

The value range of the storage capacity varies based on the instance type and storage type that you select.

You can change the storage capacity at a step size of 5 GB.

10. Configure the following parameters. This step is required only if you select the **Subscription** billing method and the **Local SSD** storage type.

    We recommend that you set the **Backup Retention After Release** parameter to **Latest** or **All**. This way, you can retrieve the data of the RDS instance if the RDS instance is released due to overdue payments and data is lost.



11. In the lower-right corner of the page, click **Next: Instance Configuration**.

12. Configure the . Network Type

    i. **Network Type**: If your application is deployed on an ECS instance, the network type of the ECS instance and the RDS instance must be the same. Otherwise, you cannot connect the ECS instance and the RDS instance over an internal network.

> **Note**
>
> - For more information about how to view the network type of the ECS instance, see Get ready to use ApsaraDB RDS for MySQL.
> - If you do not want to connect the ECS instance and the RDS instance over an internal network, you can select the classic network type or the VPC network type.
> - Specific RDS instances do not support the classic network type.

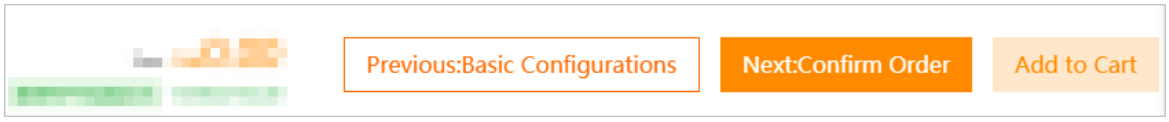
ii. **VPC** and **vSwitch**: If you select the VPC network type, you must also select a VPC and a vSwitch. We recommend that you select the VPC of the ECS instance on which your application is deployed. If the RDS instance and the ECS instance reside in different VPCs, you cannot connect these instances over an internal network.

> **Note**
> - For more information about how to view the VPC in which your ECS instance resides, see Get ready to use ApsaraDB RDS for MySQL.
> - You can connect the RDS instance and the ECS instance over an internal network even if the instances use different vSwitches in the same VPC.

13. Configure other custom parameters. **If you do not have special business requirements, you can use the default values of these parameters.**

Parameter	Description
Release Protection	Specifies whether to enable the release protection feature. The release protection feature is used to prevent a pay-as-you-go RDS instance from being released due to incorrect operations. For more information, see Enable or disable the release protection feature for an ApsaraDB RDS for MySQL instance.
Minor Version Upgrade Policy	The policy based on which the minor engine version of the RDS instance is updated. ○ **Automatic Upgrade**: ApsaraDB RDS automatically updates the minor engine version of the RDS instance to the most recent version during the scheduled maintenance window. For more information about how to change the maintenance window, see Set the maintenance window of an ApsaraDB RDS for MySQL instance. For more information about how to change the upgrade time, see Manage scheduled events. ○ **Manual Upgrade**: You must manually update the minor engine version of the RDS instance on the Basic Information page. If you do not want to use the latest minor engine version, select **Select Minor Version**. Then, you can select a minor engine version from the drop-down list that is displayed.
Resource Group	The resource group to which the RDS instance belongs. You can use the default resource group or select a custom resource group based on your business requirements.

14. In the lower-right corner of the page, click Next: Confirm Order.



15. Confirm the configuration of the RDS instance in the Parameters section, configure the **Purchase Plan** and **Duration** parameters, read and select **Terms of Service**, and then click **Pay Now**. You must configure the Duration parameter only if you select the subscription billing method for the RDS instance.

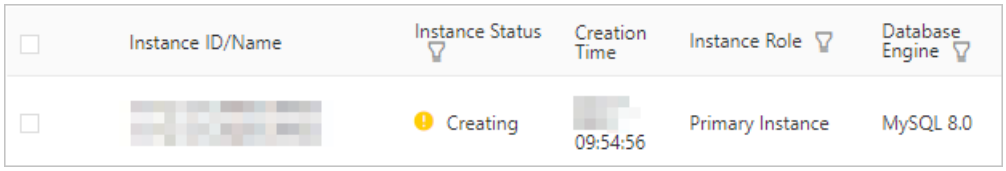
> **Note** If you select the subscription billing method for the RDS instance, we recommend that you select **Auto-Renew Enabled**. This way, you can prevent interruptions on your application even if you forget to renew the RDS instance.

The "**Congratulations.**" or "**The service is activated**" message is displayed in the ApsaraDB RDS console.

16. View the RDS instance.

    Go to the Instances page. In the top navigation bar, select the region where the RDS instance resides. Then, find the RDS instance based on the **Creation Time** parameter.

    ApsaraDB RDS requires 1 to 10 minutes to create an RDS instance. You can refresh the page to view the RDS instance that you created.



What to do next

Create databases and accounts for an ApsaraDB RDS for MySQL instance

FAQ

Why am I unable to find the RDS instance that I created?

Possible cause	Description	Suggestion
Incorrect region	The RDS instance does not reside in the region that you selected.	In the top navigation bar, select the region where the RDS instance resides. Then, you can find the RDS instance.
Insufficient resources	The zone that you selected cannot provide sufficient resources. If the RDS instance cannot be created, you can go to the Orders page in the Billing Management console to view the refunded fee.	We recommend that you select a different zone and try again.
RAM policies that do not allow users to create unencrypted RDS instances	RAM policies that do not allow users to create unencrypted RDS instances are attached to RAM users. • If you use the credentials of a RAM user to create an RDS instance that uses local SSDs, the RDS instance cannot be created. When you create an RDS instance that uses local SSDs, you cannot enable disk encryption. • If you use the credentials of a RAM user to create an RDS instance that uses standard SSDs or ESSDs and you do not enable disk encryption for the RDS instance, the RDS instance cannot be created. For more information, see Use RAM policies to manage the permissions of RAM users on ApsaraDB RDS instances.	When you create an RDS instance, select the standard SSD or ESSD storage type, select Disk Encryption, set the Key parameter, and then try again.

References

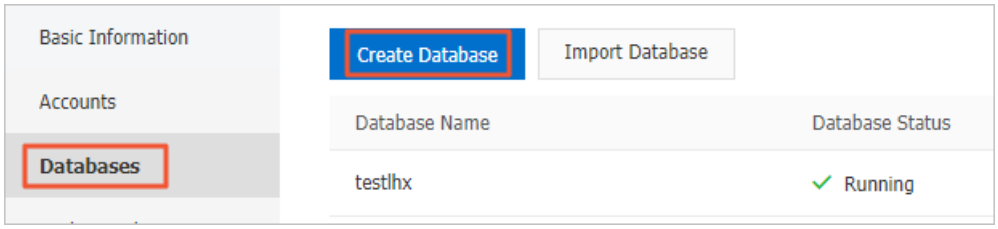
- For more information about how to create an RDS instance by calling an API operation, see Create an instance.
- For more information about how to create an RDS instance that runs a different database engine, see the following topics:
  - Create an ApsaraDB RDS for SQL Server instance
  - Create an ApsaraDB RDS for PostgreSQL instance
  - Create an ApsaraDB RDS for MariaDB TX instance

5.4. Create databases and accounts for an ApsaraDB RDS for MySQL instance

This topic describes how to create databases and accounts for an ApsaraDB RDS for MySQL instance.

Create a database

1.
2. In the left-side navigation pane, click **Databases**.
3. Click **Create Database**.



4. Configure the following parameters.

Parameter	Description
Database Name	○ The name must be 2 to 64 characters in length. ○ The name must start with a lowercase letter and end with a lowercase letter or a digit. ○ The name can contain lowercase letters, digits, underscores (_), and hyphens (-). ○ The name must be unique within the RDS instance.
Supported Character Set	Select the character set that is supported by the database.

5. Click **Create**.

Create an account

1.
2. In the left-side navigation pane, click **Accounts**.

3. Click **Create Account**.
4. In the **Database Account** field, enter the username of the account.
   - The name must start with a lowercase letter and end with a lowercase letter or a digit.
   - The name can contain lowercase letters, digits, and underscores (_).
5. Specify the **Account Type** parameter.
   - **Standard Account**: Select databases from the **Unauthorized Databases** section, click the right arrow to move the selected databases to the Authorized Databases section, and then grant the **Read /Write (DDL + DML)**, **Read-only**, **DDL Only**, or **DML Only** permissions to the account.
   - **Privileged Account**: The privileged account has permissions on all databases that are created on the RDS instance. You do not need to grant permissions on specific databases to the privileged account.

     > Note
     > - For more information about the differences between privileged accounts and standard accounts, see Account types.
     > - If the **Privileged Account** option is dimmed, a privileged account has been created on the RDS instance.
6. In the **Password** field, enter the password of the account.
   - The password must be 8 to 32 characters in length.
   - The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. The password can contain the following special characters: !@ # $ % ^ & * ( ) _ + - =
7. Click **OK**.

FAQ

- Can I manage accounts at fine-grained levels, such as the source IP address level and the table level?

  For more information, see Authorize an account to access its authorized databases from specified IP addresses in an ApsaraDB RDS for MySQL instance and Authorize accounts to manage tables, views, and fields.

- Does ApsaraDB RDS provide accounts that are equivalent to root or superuser accounts?

  No, ApsaraDB RDS does not provide accounts that are equivalent to root or superuser accounts. This way, your RDS instance can be protected from data loss and leaks that are caused by accidental operations.

References

- For more information about how to create an account for an RDS instance by calling an API operation, see Create an account.
- For more information about how to create a database for an RDS instance by calling an API operation, see Create database.
- For more information about how to create databases and accounts for an RDS instance that runs a different database engine, see the following topics:
  - Create an account and a database for an ApsaraDB RDS instance that runs SQL Server 2012, 2014、

2016, 2017 SE, or 2019 SE
- Create an account and a database for an ApsaraDB RDS instance that runs SQL Server 2008 R2
- Create databases and accounts for an ApsaraDB RDS for PostgreSQL instance
- Create databases and accounts for an ApsaraDB RDS for MariaDB TX instance

5.5. Use DMS to log on to an ApsaraDB RDS for MySQL instance

This topic describes how to log on to an ApsaraDB RDS for MySQL instance by using Data Management (DMS). DMS supports features such as data management, user authorization, security audit, lock-free changes, data tracking, and data visualization.

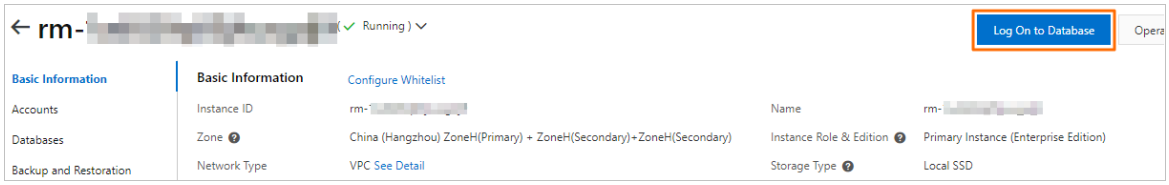
Prerequisites

The following operations are complete:

1. Create an ApsaraDB RDS for MySQL instance
2. Create databases and accounts for an ApsaraDB RDS for MySQL instance

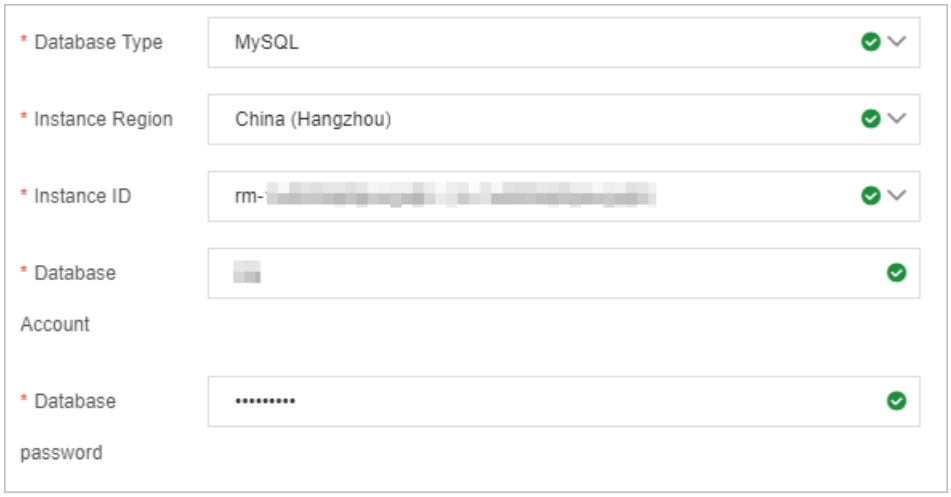
Procedure

1.
2. In the upper-right corner of the Basic Information page, click **Log On to Database**.

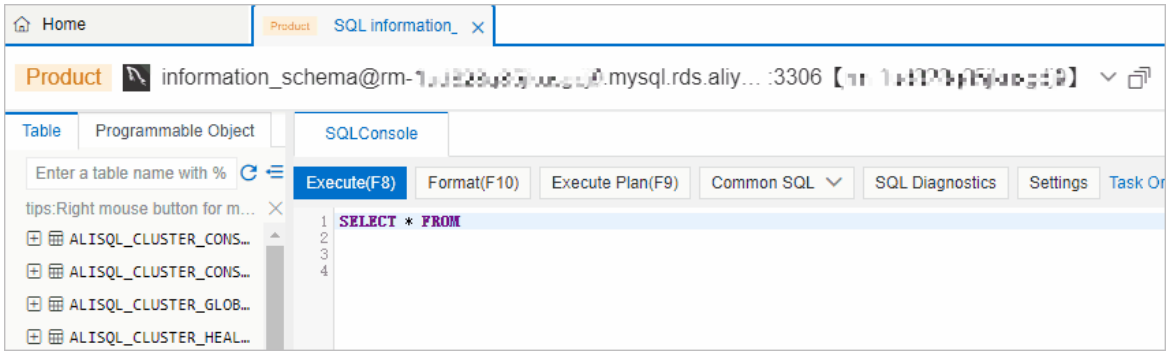


3. In the Login instance dialog box, enter the username and password of the account that is used to log on to the RDS instance. Then, click **Login**.

> **Note** To obtain the username and password of an account, you can perform the following operations:
>
> i.
>
> ii. In the left-side navigation pane, click **Accounts**.



4. After you log on to the RDS instance, manage the databases on which the account has permissions based on your business requirements. You can click the name of a database to switch to that database. In the example shown in the following figure, you switch from the information_schema database to a different database.



Note The information_schema database and the mysql database are system databases.

5.6. Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance

This topic describes how to configure IP address whitelists and use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.

Prerequisites

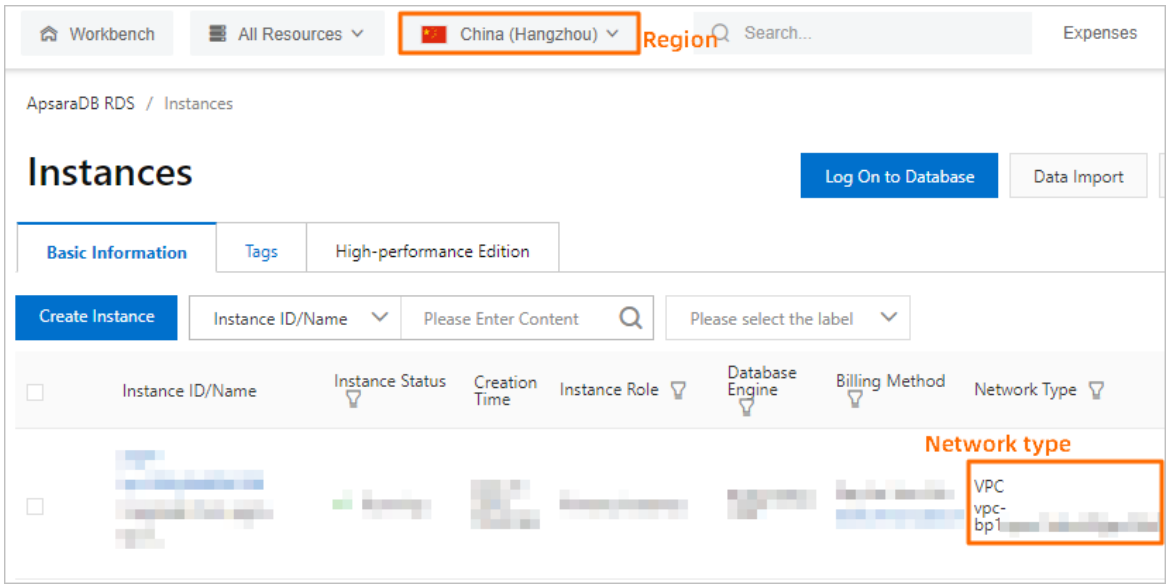
The operations that are described in the following topics are complete:

- Create an ApsaraDB RDS for MySQL instance
- Create databases and accounts for an ApsaraDB RDS for MySQL instance

Step 1: Check whether your application can connect to the RDS instance over an internal network

1. View the region of the instance on which your application is deployed. For more information, see Get ready to use ApsaraDB RDS for MySQL. Elastic Compute Service (ECS)network type
2. View the region and network type of the RDS instance.

   Log on to the ApsaraDB RDS console and go to the Instances page. In the top navigation bar, select the region where the RDS instance resides. Then, find the RDS instance and click the instance ID. On the page that appears, you can view the region, network type, and virtual private cloud (VPC) ID of the RDS instance.



3. Check whether the ECS instance and the RDS instance meet the following conditions for communication over an internal network:
   i. The ECS instance and the RDS instance reside in the same region.
   ii. The ECS instance and the RDS instance reside in the same type of network. If the ECS instance and the RDS instance both reside in VPCs, these instances must reside in the same VPC.

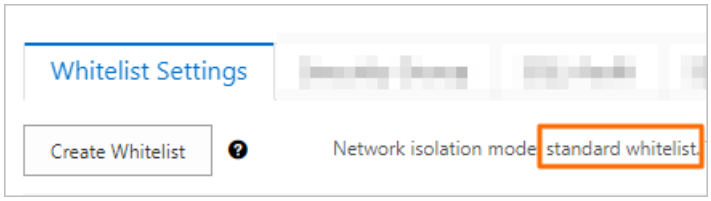
> **Note** If one of the preceding conditions is not met, the ECS instance cannot communicate with the RDS instance over an internal network.

Step 2: Configure IP address whitelists for the RDS instance

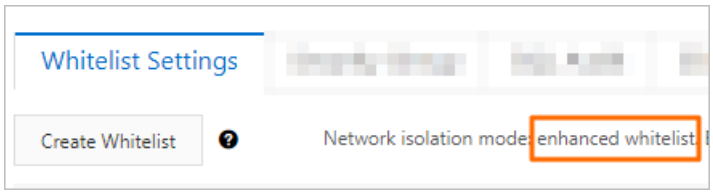
1.
2. In the left-side navigation pane, click **Data Security**.
3. View the network isolation mode of the RDS instance.

> **Note** Existing RDS instances may run in enhanced whitelist mode. New RDS instances run in standard whitelist mode.

Standard whitelist mode

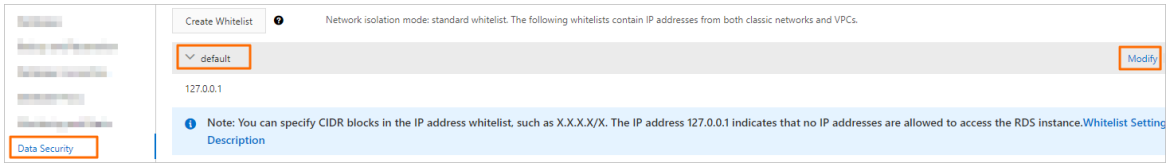


Enhanced whitelist mode



4. Click **Modify** to the right of the IP address whitelist named **default**.

> **Note** You can also click **Create Whitelist** to create an IP address whitelist.

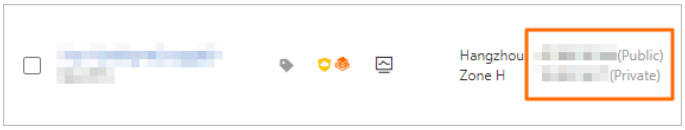


5. Add the IP address of the server on which your application is deployed to the default IP address whitelist.

   The server can communicate with the RDS instance only after you add the IP address of the server to the default IP address whitelist.

   The following table describes various connection scenarios. You can obtain the required IP address based on your connection scenario and add the IP address to an IP address whitelist of the RDS instance. Obtain IP addresses

Connection scenario	IP address to be obtained	How to obtain the IP address
the conditions for communication over an internal network	The private IP address of the ECS instance	i. Log on to the ECS console and go to the Instances page.

Connection scenario	IP address to be obtained	How to obtain the IP address
You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance do not meet the conditions for communication over an internal network.	The public IP address of the ECS instance	ii. In the top navigation bar, select the region where the ECS instance resides. iii. View the public IP address and private IP address of the ECS instance. 
You want to connect to the RDS instance from an on-premises device.	The public IP address of the on-premises device	On the on-premises device, use a search engine such as Google to search for **IP**. **Note** The IP address that you obtain by using this method may be inaccurate. For more information about how to obtain the accurate IP address of an on-premises device, see Why am I unable to connect to my ApsaraDB RDS for MySQL or ApsaraDB RDS for MariaDB instance from a local server over the Internet?

Note

- If you add multiple IP addresses and CIDR blocks to an IP address whitelist, you must separate the IP addresses and CIDR blocks with commas (,) and leave no spaces before and after each comma.
- You can add a maximum of 1,000 IP addresses and CIDR blocks in total for each RDS instance. If you want to add a large number of IP addresses, we recommend that you merge the IP addresses into CIDR blocks, such as 10.10.10.0/24.
- If an RDS instance runs in standard whitelist mode, you do not need to take note of special considerations when you configure IP address whitelists for the RDS instance. **If an RDS instance runs in enhanced whitelist mode, you must take note of the following considerations when you configure IP address whitelists for the RDS instance:**
  - Add **public IP addresses** or the private IP addresses of -hosted ECS instances to IP address whitelists of the **classic network type**. classic network
  - Add the private IP addresses of VPC-hosted ECS instances to IP address whitelists of the **VPC network type**.

6. Click **OK**.

Step 3: Connect to the RDS instance

To connect to the RDS instance by using the CLI, perform the following steps:

1. Log on to the server from which you want to connect to the RDS instance. For example, the server can be an ECS instance or an on-premises device.

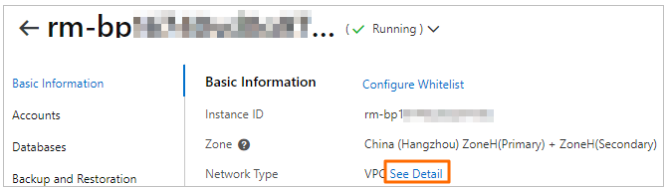
   > **Note** For more information about how to log on to an ECS instance, see the "**Connect to an instance**" section in Create and manage an ECS instance by using the ECS console (express version).

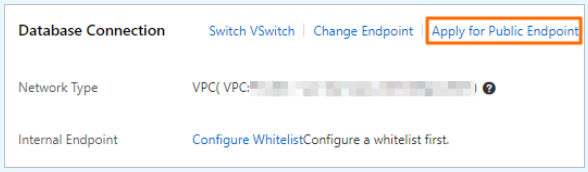
2. Run the following command:

   ```
   mysql -hEndpoint -PPort number -uUsername -p      //Take note that the uppercase letter P precedes the lowercase letter p.
   ```

   - Endpoint and port number: Enter the endpoint and port number that are used to connect to the RDS instance.

Connection scenario	Endpoint to be obtained	How to obtain the endpoint

<table>
<tr><th>Connection scenario</th><th>Endpoint to be obtained</th><th>How to obtain the endpoint</th></tr>
<tr><td>You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance meet the conditions for communication over an internal network. For more information, see the "Step 1: Check whether your application can connect to the RDS instance over an internal network" section of this topic.</td><td>The internal endpoint of the RDS instance</td><td rowspan="2">a.
b. In the Basic Information section of the page that appears, click See Details to the right of the Network Type parameter to view the endpoint and port number that are used to connect to the RDS instance.
 

Note
■ Before you can view the endpoint and port number that are used to connect to the RDS instance, you must configure IP address whitelists for the RDS instance.
■ A public endpoint is displayed only after you click Apply for Public Endpoint to apply for a public endpoint for the RDS instance.
 
</td></tr>
<tr><td>You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance do not meet the conditions for communication over an internal network.</td><td>The public endpoint of the RDS instance</td></tr>
</table>

Connection scenario	Endpoint to be obtained	How to obtain the endpoint
Connect to the RDS instance from an on-premises device.		

- Username and password: Obtain the username and password of the account that is used to connect to the RDS instance from the page. Accounts

  Example

  ```
  root@ -> mysql -h .mysql.rds.aliyuncs.com -P3306 -u -p
  Enter password:
  ```

  Successful connection

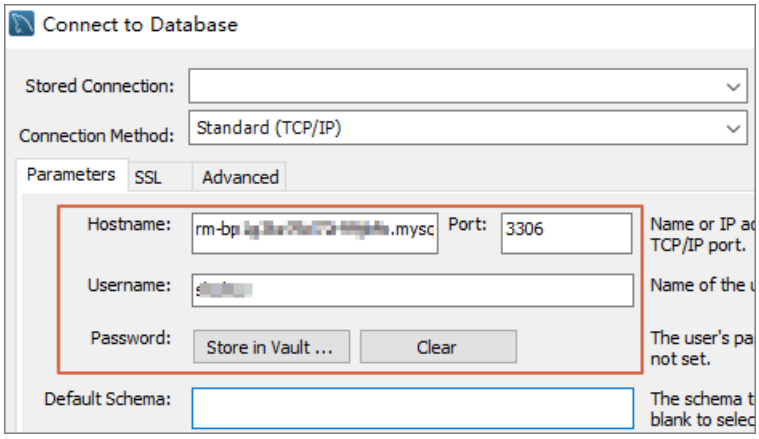
  ```
  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 51325
  Server version: 8.0.18 Source distribution
  ```

  > **Note** If connection errors occur, you can troubleshoot the errors by following the instructions provided in Common connection errors.

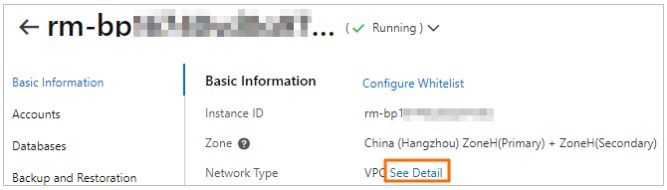
To connect to the RDS instance by using a database client, perform the following steps:

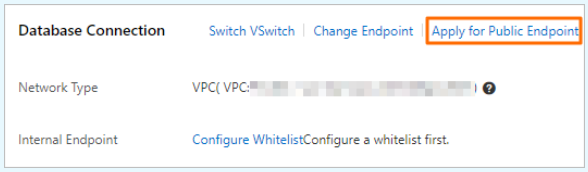
You can use a general-purpose MySQL client to connect to the RDS instance. In this example, MySQL Workbench is used. The methods of using other database clients to connect to the RDS instance are similar.

1. Go to the MySQL Community Downloads page, select the MySQL Workbench software package that can be used with your operating system, and then click **Download**.
2. Install MySQL Workbench.
3. Start MySQL Workbench and choose **Database > Connect to Database**.
4. Enter the information that is used to connect to the RDS instance.



- **Hostname** and **Port**: Enter the endpoint and port number that are used to connect to the RDS instance.

<table>
<tr><th>Connection scenario</th><th>Endpoint to be obtained</th><th>How to obtain the endpoint</th></tr>
<tr><td>You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance meet the conditions for communication over an internal network. For more information, see the "Step 1: Check whether your application can connect to the RDS instance over an internal network" section of this topic.</td><td>The internal endpoint of the RDS instance</td><td rowspan="2">a.
b. In the Basic Information section of the page that appears, click See Details to the right of the Network Type parameter to view the endpoint and port number that are used to connect to the RDS instance.
 

Note
■ Before you can view the endpoint and port number that are used to connect to the RDS instance, you must configure IP address whitelists for the RDS instance.
■ A public endpoint is displayed only after you click Apply for Public Endpoint to apply for a public endpoint for the RDS instance.
 
</td></tr>
<tr><td>You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance do not meet the conditions for communication over an internal network.</td><td>The public endpoint of the RDS instance</td></tr>
</table>

Connection scenario	Endpoint to be obtained	How to obtain the endpoint
Connect to the RDS instance from an on-premises device.		

- **Username** and **Password**: Obtain the username and password of the account that is used to connect to the RDS instance from the page. Accounts

Common connection errors

Error message	Cause and solution
mysql command not found	MySQL is not installed. Run the following commands to install MySQL: • If you use a CentOS operating system, run the `yum install mysql` command. • If you use an Ubuntu operating system, run the `apt-get update` command and then the `apt install mysql-server` command.
SSL connection error: SSL is required but the server doesn't support it	You are using the latest version of MySQL Workbench. In this version, standard TCP/IP connections require SSL encryption. However, the connected server does not support SSL encryption. In this case, you can download an earlier version of MySQL Workbench to establish regular connections.
Can't connect to MySQL server on 'rm-bp1xxxxxxxxxxxxx.mysql.rds.aliyuncs.com'(10060) Cannot Connect to Database Server Your connection attempt failed for user 'xx" to the MySQL server	• In most cases, this error occurs because the IP address whitelists that you configured are inappropriate. For more information, see the "**Step 2: Configure IP address whitelists for the RDS instance** section of this topic. • In a few cases, this error occurs because the RDS instance and the ECS instance do not meet the but you attempt to connect to the internal endpoint of the RDS instance. conditions for communication over an internal network
Access denied for user 'xxxxx'@'xxxxx'(using password:YES)	This error occurs because the username and password that you entered are incorrect. You can obtain the correct username and password from the page. Accounts
Unknown MySQL server host 'xxxxxxxxx'(11001)	This error occurs because the endpoint that you entered is invalid. Valid endpoints are in the rm-xxxxxx.mysql.rds.aliyuncs.com format.

References

- For more information about how to troubleshoot connection errors, see What do I do if I cannot connect an ECS instance to an ApsaraDB for RDS instance?
- For more information about how to connect to an RDS instance in a more convenient and efficient manner, see Use DMS to log on to an ApsaraDB RDS for MySQL instance.
- For more information about how to connect to an RDS instance that runs a different database engine, see the following topics:
  - Connect to an ApsaraDB RDS for SQL Server instance
  - Connect to an ApsaraDB RDS for PostgreSQL instance
  - Connect to an ApsaraDB RDS for MariaDB TX instance

6.Data migration

6.1. Overview of data migration methods

This topic describes the methods that you can use to migrate data among self-managed data centers, third-party clouds, and ApsaraDB RDS with no downtime.

Scenario	References
Migrate data from a MySQL database in a self-managed data center to an ApsaraDB RDS for MySQL instance	• Migrate the data of a self-managed MySQL 5.7 or MySQL 8.0 instance to an ApsaraDB RDS for MySQL instance • Migrate data from a self-managed MySQL database to an ApsaraDB RDS for MySQL instance • Migrate data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL instance • Migrate data from a self-managed Db2 database to an ApsaraDB RDS for MySQL instance • Use mysqldump to migrate data from a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance • Migrate data from a self-managed Oracle database to an ApsaraDB RDS for MySQL instance
Migrate data from a MySQL database on a third-party cloud to an ApsaraDB RDS for MySQL instance	• Migrate data from an Amazon RDS for MySQL instance to an ApsaraDB RDS for MySQL instance • Migrate data from an Amazon RDS for Oracle instance to an ApsaraDB RDS for MySQL instance • Migrate data from an Amazon Aurora MySQL cluster to an ApsaraDB RDS for MySQL instance • Migrate a MySQL database from Google Cloud to Alibaba Cloud
Migrate data between ApsaraDB RDS for MySQL instances	• Migrate data between RDS instances • Migrate data between databases that have different names • Migrate data between RDS instances of different Alibaba Cloud accounts • Database clone

6.2. Data Migration from a User-created Database to an ApsaraDB RDS MySQL Instance

6.2.1. Migrate data from a self-managed MySQL database to an ApsaraDB RDS for MySQL instance

This topic describes how to migrate data from a self-managed MySQL database to an ApsaraDB RDS for MySQL instance by using Data Transmission Service (DTS). DTS supports schema migration, full data migration, and incremental data migration. When you migrate data from a self-managed MySQL database to Alibaba Cloud, you can use all of the supported migration types to ensure service continuity.

Prerequisites

- An ApsaraDB RDS for MySQL instance is created. For more information, see Create an ApsaraDB RDS for MySQL instance.
- The version of the self-managed MySQL database is 5.1, 5.5, 5.6, 5.7, or 8.0.
- The available storage space of the destination ApsaraDB RDS for MySQL instance is larger than the total size of the data in the self-managed MySQL database.

Precautions

- DTS uses read and write resources of the source and destination databases during full data migration. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you migrate data, evaluate the impact of data migration on the performance of the source and destination databases. We recommend that you migrate data during off-peak hours. For example, you can migrate data when the CPU utilization of the source and destination databases is less than 30%.
- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- DTS uses the `ROUND(COLUMN,PRECISION)` function to retrieve values from columns of the FLOAT or DOUBLE data type. If you do not specify a precision, DTS sets the precision for the FLOAT data type to 38 digits and the precision for the DOUBLE data type to 308 digits. You must check whether the precision settings meet your business requirements.
- DTS automatically creates a destination database in the ApsaraDB RDS for MySQL instance. However, if the name of the source database is invalid, you must manually create a database in the ApsaraDB RDS for MySQL instance before you configure the data migration task.

  > **Note** For more information about the naming conventions of ApsaraDB RDS for MySQL databases and how to create a database, see Create a database on an ApsaraDB RDS for MySQL instance.

- If a data migration task fails, DTS automatically resumes the task. Before you switch your workloads to the destination instance, stop or release the data migration task. Otherwise, the data in the source database will overwrite the data in the destination instance after the task is resumed.

Billing

Migration type	Task configuration fee	Internet traffic fee
Schema migration and full data migration	Free of charge.	Charged only when data is migrated from Alibaba Cloud over the Internet. For more information, see Pricing.
Incremental data migration	Charged. For more information, see Pricing.	

Migration types

- Schema migration

  DTS migrates the schemas of the required objects to the destination instance. DTS supports schema migration for the following types of objects: table, view, trigger, stored procedure, and function.

  > Note
  > - During schema migration, DTS changes the value of the SECURITY attribute from DEFINER to INVOKER for views, stored procedures, and functions.
  > - DTS does not migrate user information. Before a user can call views, stored procedures, and functions of the destination database, you must grant the read and write permissions to the user.

- Full data migration

  DTS migrates historical data of the required objects from the self-managed MySQL database to the destination database in the ApsaraDB RDS for MySQL instance.

  > Note During full data migration, concurrent INSERT operations cause fragmentation in the tables of the destination database. After full data migration is complete, the tablespace of the destination database is larger than that of the source database.

- Incremental data migration

  After full data migration is complete, DTS retrieves binary log files from the self-managed MySQL database. Then, DTS synchronizes incremental data from the self-managed MySQL database to the destination ApsaraDB RDS for MySQL instance. Incremental data migration allows you to ensure service continuity when you migrate data from a self-managed MySQL database to Alibaba Cloud.

SQL operations that can be synchronized during incremental data migration

Operation type	SQL statements
DML	INSERT, UPDATE, DELETE, and REPLACE

Operation type	SQL statements
DDL	• ALTER TABLE and ALTER VIEW • CREATE FUNCTION, CREATE INDEX, CREATE PROCEDURE, CREATE TABLE, and CREATE VIEW • DROP INDEX and DROP TABLE • RENAME TABLE • TRUNCATE TABLE

Permissions required for database accounts

Database	Schema migration	Full data migration	Incremental data migration
Self-managed MySQL database	The SELECT permission	The SELECT permission	The REPLICATION SLAVE, REPLICATION CLIENT, SHOW VIEW, and SELECT permissions
ApsaraDB RDS for MySQL instance	The read and write permissions	The read and write permissions	The read and write permissions

For more information about how to create and authorize a database account, see the following topics:

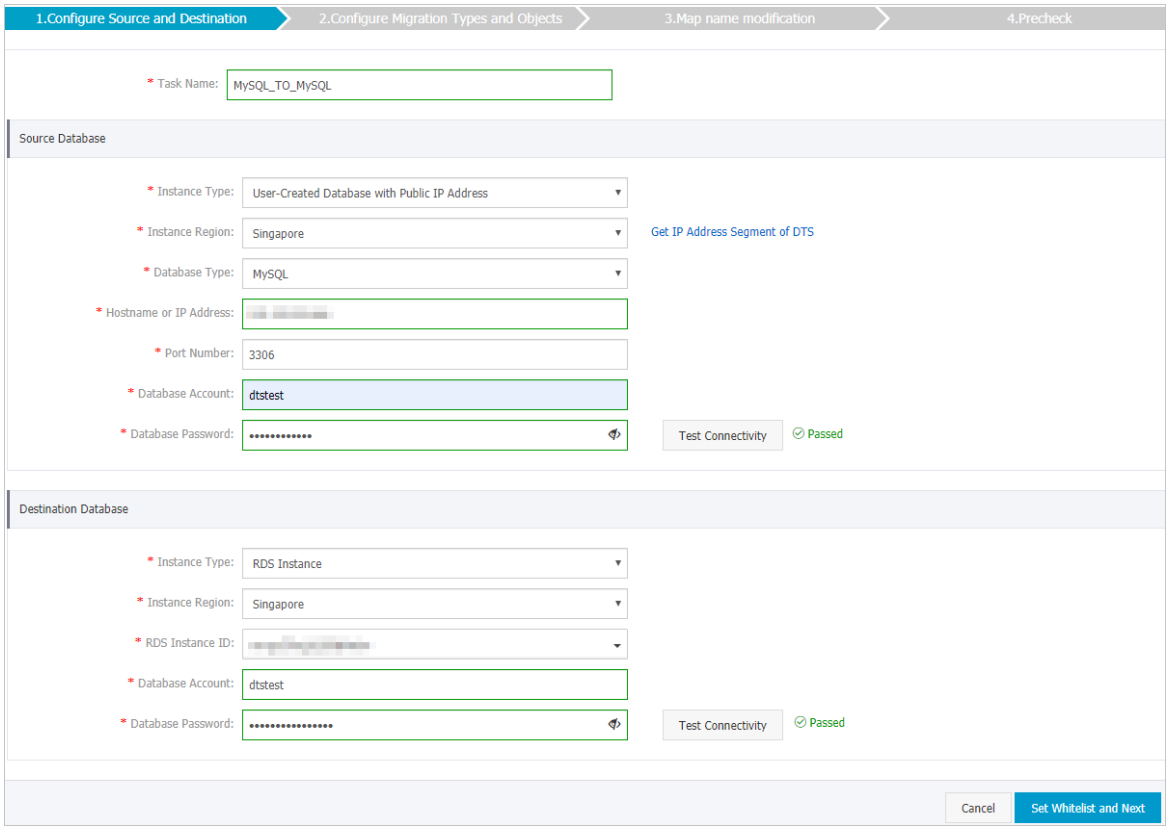
- Self-managed MySQL database: Create an account for a user-created MySQL database and configure binary logging
- ApsaraDB RDS for MySQL instance: Create an account on an ApsaraDB RDS for MySQL instance and Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance.

Before you begin

Create an account for a user-created MySQL database and configure binary logging

Procedure

1. Log on to the DTS console.
2. In the left-side navigation pane, click **Data Migration**.
3. In the upper part of the **Migration Tasks** page, select the region where the RDS instance resides.
4. In the upper-right corner of the page, click **Create Migration Task**.
5. Configure the source and destination databases.



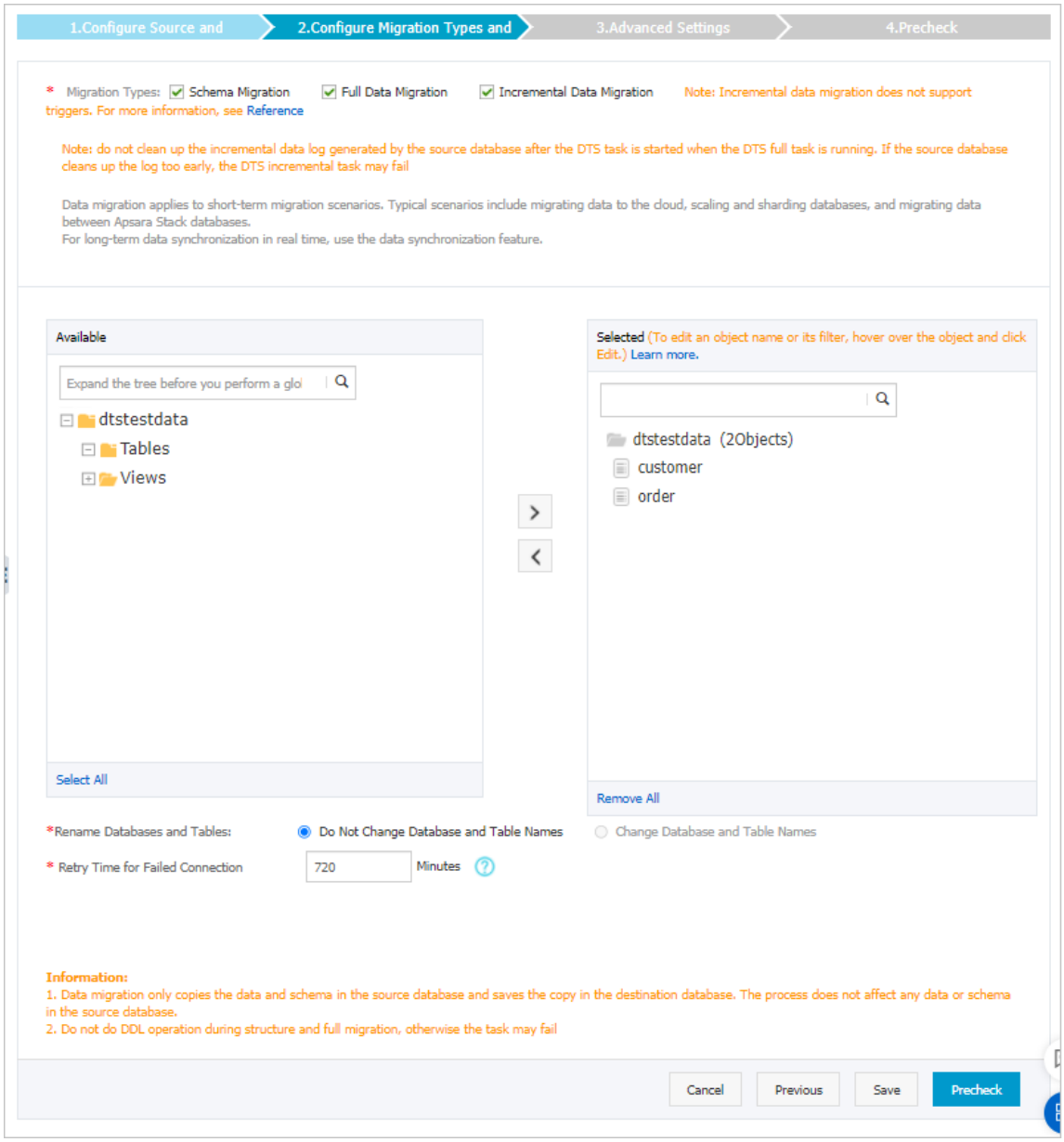
Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Database	Instance Type	Select an instance type based on the deployment of the source database. In this example, select **User-Created Database with Public IP Address**. **Note** If you select other instance types, you must deploy the network environment for the source database. For more information, see Preparation overview.
	Instance Region	If the instance type is set to **User-Created Database with Public IP Address**, you do not need to specify the **instance region**. **Note** If a whitelist is configured for the self-managed MySQL database, you must manually add the CIDR blocks of DTS servers to the whitelist of the database. You can click **Get IP Address Segment of DTS** next to **Instance Region** to obtain the CIDR blocks of DTS servers.
	Database Type	Select **MySQL**.

Database Section	Parameter	Description
	Hostname or IP Address	Enter the endpoint that is used to connect to the self-managed MySQL database. In this example, enter the public IP address.
	Port Number	Enter the service port number of the self-managed MySQL database. The port must be accessible over the Internet. The default port number is **3306**.
	Database Account	Enter the account of the self-managed MySQL database. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the information about the self-managed Oracle database, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
Destination Database	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the destination RDS instance resides.
	RDS Instance ID	Select the ID of the destination RDS instance.
	Database Account	Enter the database account of the destination RDS instance. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the information about the RDS instance, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data migration task. For more information, see Configure SSL encryption on an ApsaraDB RDS for MySQL instance. **Note** The **Encryption** parameter is available only for regions in mainland China and the China (Hong Kong) region.

6. In the lower-right corner of the page, click **Set Whitelist and Next**.

> **Note** DTS adds the CIDR blocks of DTS servers to the whitelist of the destination RDS instance. This ensures that DTS servers can connect to the destination RDS instance.

7. Select the migration types and the objects to be migrated.



Setting	Description

Setting	Description
Select the migration types	◦ To perform only full data migration, select **Schema Migration** and **Full Data Migration**. ◦ To migrate data with minimal downtime, select **Schema Migration**, **Full Data Migration**, and **Incremental Data Migration**. **Note** If **Incremental Data Migration** is not selected, do not write data to the source database during full data migration. This ensures data consistency between the source and destination databases.
Select the objects to be migrated	Select one or more objects from the **Available** section and click the › icon to move the objects to the **Selected** section. **Note** ◦ You can select columns, tables, or databases as the objects to be migrated. If you select tables or columns as the objects to be migrated, DTS does not migrate other objects such as views, triggers, and stored procedures to the destination database. ◦ By default, after an object is migrated to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to change the names of the objects that are migrated to the destination database. For more information, see Object name mapping. ◦ If you use the object name mapping feature on an object, other objects that are dependent on the object may fail to be migrated.
Specify whether to rename object names	You can use the object name mapping feature to change the names of the objects that are migrated to the destination instance. For more information, see Object name mapping.
Specify the retry time for failed connections to the source or destination database	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 12 hours. You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data migration task. Otherwise, the data migration task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

Setting	Description
Specify whether to copy temporary tables to the destination database when DMS performs online DDL operations on the source table	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to migrate temporary tables generated by online DDL operations. ○ **Yes**: DTS migrates the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the migration task will be delayed. ○ **No**: DTS does not migrate the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is migrated. **Note** If you select No, the tables in the destination database may be locked.

8. Click **Precheck**.

   Note
   - A precheck is performed before the migration task starts. The migration task only starts after the precheck succeeds.
   - If the precheck fails, click the icon next to each failed check item to view the related details. Fix the issues as instructed and run the precheck again.

9. After the data migration task passes the precheck, click **Next**.
10. In the **Confirm Settings** dialog box, configure the **Channel Specification** parameter. Then, read and select **Data Transmission Service (Pay-as-you-go) Service Terms**.
11. Click **Buy and Start** to start the data migration task.

Stop the migration task

Warning We recommend that you prepare a rollback solution to migrate incremental data from the destination database to the source database in real time. This allows you to minimize the negative impact of switching your workloads to the destination database. For more information, see Switch workloads to the destination database. If you do not need to switch your workloads, you can perform the following steps to stop the migration task.

- Full data migration

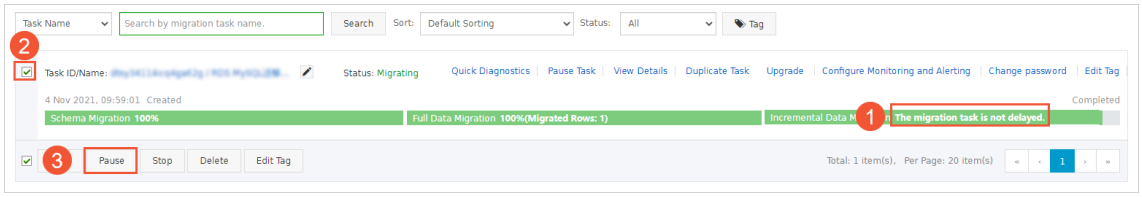
  Do not manually stop a task during full data migration. Otherwise, the system may fail to migrate all data. Wait until the migration task automatically ends.

- Incremental data migration

  The task does not automatically end during incremental data migration. You must manually stop the migration task.

  i. Wait until the task progress bar shows **Incremental Data Migration** and **The migration task is not delayed**. Then, stop writing data to the source database for a few minutes. In some cases, the progress bar shows the delay time of **incremental data migration**.

  ii. After the status of **incremental data migration** changes to **The migration task is not delayed**, manually stop the migration task.



What to do next

The database accounts that are used for data migration have the read and write permissions. After you migrate data, you must delete the database accounts to ensure security.

FAQ

- Q: What can I do if a migration task fails to pass the precheck?

  A: For more information, see Source database connectivity.

- Q: How can I troubleshoot a failed migration task?

  A: For more information, see Fix a failed data migration task.

6.2.2. Migrate the data of a self-managed MySQL 5.7 or MySQL 8.0 instance to an ApsaraDB RDS for MySQL instance

This topic describes how to migrate the data of a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance that runs the same MySQL version as the self-managed MySQL instance. You can perform a full backup on the self-managed MySQL instance, upload the full backup file to an Object Storage Service (OSS) bucket, import the full backup file from the OSS bucket into a temporary RDS instance, and then restore the data from the full backup file to the destination RDS instance.

Prerequisites

- An Alibaba Cloud account is created.
- The self-managed instance meets the migration conditions. For more information, see Appendix 5: Limits.
- An OSS bucket is created in the region where the destination RDS instance resides. For more information, see Create buckets.

> **Note** The OSS bucket that you create must reside in the same region as the destination RDS instance.

Migration process

The migration process consists of the following steps:

Step 1: Install Percona XtraBackup

Step 2: Install MySQL Backup Helper

Step 3: Back up the self-managed MySQL instance and migrate the backup data to the RDS instance

> **Note** For more information, seeVideo tutorial.

Environment

In this topic, the self-managed MySQL instance is deployed on an Elastic Compute Service (ECS) instance. The image that is used to create the self-managed MySQL instance runs CentOS Linux V8.3.2011. For more information, see Create an instance by using the wizard.

Step 1: Install Percona XtraBackup

Percona XtraBackup is developed by Percona to help you back up MySQL databases without impacting uptime. Percona XtraBackup is compatible with various storage engines. If you use an Ubuntu operating system, see Appendix 1: Install Percona XtraBackup in an Ubuntu operating system.

1. Install the Percona repository.

```
yum install -y https://repo.percona.com/yum/percona-release-latest.noarch.rpm
```

2. Enable the Percona repository.

```
percona-release enable-only tools release
```

3. Install Percona XtraBackup 2.4 or Percona XtraBackup 8.0.

```
yum install -y percona-xtrabackup-24  # Install Percona XtraBackup 2.4.
yum install -y percona-xtrabackup-80 # Install Percona XtraBackup 8.0.
```

> **Note** You must select the version of Percona XtraBackup based on the MySQL version.
>
> - MySQL 5.7: Install Percona XtraBackup 2.4.
> - MySQL 8.0: Install Percona XtraBackup 8.0.

Step 2: Install MySQL Backup Helper

Prerequisites

- The Go programming language is installed. If Go is not installed, run the following command in the CLI to install Go:

```
yum install -y go
```

- The UnZip utility is installed. If UnZip is not installed, run the following command in the CLI to install

UnZip:

```
yum install -y unzip
```

Note The preceding commands are supported only for CentOS. If you use an Ubuntu operating system, see Appendix 2: Install Go and UnZip in an Ubuntu operating system.

1. Download the MySQL Backup Helper source package.

   ```
   wget https://github.com/aliyun/mysql-backup-helper/archive/refs/heads/master.zip
   ```

   Note You can download the MySQL Backup Helper source package from the mysql-backup-helper page.

2. Decompress the MySQL Backup Helper source package.

   ```
   unzip master.zip
   ```

3. Go to the *mysql-backup-helper-master* folder and compile the main.go file into an executable file named backup_helper.

   ```
   cd mysql-backup-helper-master
   go build -a -o backup-helper main.go
   ```

4. Go to the *oss_stream* folder and compile the oss_stream.go file into an executable file named oss_stream.

   ```
   cd oss_stream
   go build -a -o oss_stream oss_stream.go
   ```

Step 3: Back up the self-managed MySQL instance and migrate the backup data to the RDS instance

1. Use MySQL Backup Helper to check whether the self-managed MySQL instance supports backups.

   ```
   cd ~/mysql-backup-helper-master && ./backup-helper -host <The IP address of the host in which the self-managed MySQL instance resides> -port <The port number that is used to connect to the self-managed MySQL instance> -user <The username of the root account that is used to connect to the self-managed MySQL instance> --password <The password of the root account that is used to connect to the self-managed MySQL instance>
   ```

2. After the self-managed MySQL instance passes the check, perform a full backup on the instance and upload the full backup file to your OSS bucket. If no OSS bucket is created, create one. For more information, see the "Prerequisites" section of this topic.

```
innobackupex --backup --host=<The IP address of the host in which the self-managed MySQ
L instance resides> --port=<The port number that is used to connect to the self-managed
MySQL instance> --user=<The username of the root account that is used to connect to the
self-managed MySQL instance> --password=<The password of the root account that is used
to connect to the self-managed MySQL instance> --stream=xbstream --compres <The tempora
ry directory that is used to store the full backup file> | ./mysql-backup-helper-master
/oss_stream/oss_stream -accessKeyId <The AccessKey ID of your Alibaba Cloud account> -a
ccessKeySecret <The AccessKey secret of your Alibaba Cloud account> -bucketName <The na
me of your OSS bucket> -endpoint <The endpoint that is used to connect to your OSS buck
et> -objectName <The name of the full backup file after the file is uploaded as an obje
ct to your OSS bucket>
```

Examples:

```
innobackupex --backup --host=127.0.0.1 --port=3306 --user=root --password=Aa123456@ --s
tream=xbstream --compress /root/mysql/data | ./mysql-backup-helper-master/oss_stream/os
s_stream -accessKeyId LTAI5tCqY18jvvKk******** -accessKeySecret 4A5Q7ZVzcYnWMQPysXFxld*
******* -bucketName test -endpoint oss-********.aliyuncs.com -objectName backup_qp.xb
```

Note

- The status of the self-managed MySQL instance during the full backup affects the time that is required to complete the full backup. For example, if a large number of redo log records are generated from a large number of write operations or large transactions are run during the full backup, the time that is required increases. When the full backup is complete, the system displays the "`completed OK !`" message.
- The amount of data in the self-managed MySQL instance also affects the time that is required to complete the full backup. A larger amount of data requires a longer period of time. If the self-managed MySQL instance has a large amount of data, we recommend that you run the nohup command to perform the full backup in the background. This way, you can prevent interruptions to the full backup in the event of unexpected logoffs. Sample statement:

  ```
  nohup sh -c 'innobackupex --backup --host=127.0.0.1 --port=3306 --user=root -
  -password=Aa123456@ --stream=xbstream --compress /root/mysql/data | ./mysql-b
  ackup-helper-master/oss_stream/oss_stream -accessKeyId LTAI5tCqY18jvvKk******
  ** -accessKeySecret 4A5Q7ZVzcYnWMQPysXFxld******** -bucketName test -endpoint
  oss-ap-southeast-1.aliyuncs.com -objectName backup_qp.xb' &
  ```

- If your OSS bucket is temporarily inaccessible, we recommend that you save the full backup file to your computer. When your OSS bucket is restored to normal, you can upload the full backup file to your OSS bucket. For more information, see Appendix 3: Perform a full backup, save the full backup file to your computer, and then upload the full backup file to your OSS bucket.
- After you upload the full backup file to your OSS bucket, you can log on to the OSS console to check whether the upload is successful. If the upload failed, you can repeat this step.

3. Log on to the ApsaraDB RDS console. In the top navigation bar, select the region of the RDS instance to which you want to restore the data of the self-managed MySQL instance.
4. In the left-side navigation pane, click **Backups**.

5. On the tab that appears, click **Import Backup**. In the wizard that appears, read the messages that are displayed and click **Next** until you enter the **3. Import Data** step.
6. Select the name of your OSS bucket from the OSS Bucket drop-down list. In the **File Name** section, select the full backup file that you want to import. Select the zone to which you want to import the full backup file from the **Zone** drop-down list. Then, click **OK**.

   > **Note**
   > - If ApsaraDB RDS is not authorized to access OSS resources, click **Authorize**. In the lower-left corner of the page that appears, click **Confirm Authorization Policy**.
   > - ApsaraDB RDS creates a task to check the backup file. You can view the status of the task on the **User Backups** page. When **Status** of the task changes from **Verifying** to **Completed**, the task is complete. The time that is required to complete the task varies based on the status of the self-managed MySQL instance during the full backup. For example, if a large number of redo log records are generated from a large number of write operations or large transactions are run during the full backup, the time that is required to complete the task increases.

7.
8. Configure the following parameters and click **Next: Instance Configuration.**

Parameter	Description
Zone of Primary Node	The zone to which the primary RDS instance belongs. **Note** If you did not select a zone for the OSS bucket that stores the full backup file when you import the file, this parameter is displayed. If you selected a zone for the OSS bucket that stores the full backup file when you import the file, this parameter is not displayed.
Storage type	- **ESSD PL1**: An enhanced SSD (ESSD) of performance level 1 (PL1) is a regular ESSD. - **Standard SSD**: A standard SSD is an elastic block storage device that is designed based on the distributed storage architecture of Alibaba Cloud. You can store data on standard SSDs to separate computing from storage. **Note** For more information about storage types, see Storage types.
Instance Type	**General-purpose**: specifies the general-purpose instance family. A general-purpose instance exclusively occupies the allocated memory and I/O resources. However, it shares CPU and storage resources with the other general-purpose instances that are deployed on the same server. **Note** Each instance type supports a specific number of cores, memory capacity, maximum number of connections, and maximum IOPS. For more information, see Primary ApsaraDB RDS instance types.

Parameter	Description
Capacity	The storage capacity that is used to store data files, system files, binary log files, and transaction files in the RDS instance. You can adjust the storage capacity at a step size of 5 GB.

9.
10.

Appendix 1: Install Percona XtraBackup in an Ubuntu operating system

1. Download the latest Percona package.

```
wget https://repo.percona.com/apt/percona-release_latest.$(lsb_release -sc)_all.deb
```

2. Install the downloaded Percona package.

```
sudo dpkg -i percona-release_latest.$(lsb_release -sc)_all.deb
```

3. Enable the Percona repository.

```
percona-release enable-only tools release
```

4. Update the list of software applications in the local repository.

```
apt-get update
```

5. Install Percona XtraBackup 2.4 or Percona XtraBackup 8.0.

```
sudo apt-get install -y percona-xtrabackup-24 //Install Percona XtraBackup 2.4.
sudo apt-get install percona-xtrabackup-80 //Install Percona XtraBackup 8.0.
```

> **Note** You must select the version of Percona XtraBackup based on the MySQL version.
>
> - MySQL 5.7: Install Percona XtraBackup 2.4.
> - MySQL 8.0: Install Percona XtraBackup 8.0.

6. Install the qpress tool.

```
sudo apt-get install -y qpress
```

> **Note** The qpress tool is used to unzip the backup files that are generated by Percona XtraBackup. If you are using an Ubuntu operating system, Percona XtraBackup is not integrated with the qpress tool and you must install this tool.

> **Note** If a message similar to "`The following packages have unmet dependencies`" is displayed when you perform any of the preceding steps, run the `apt-get -f install` command to install the required dependency packages. Then, perform the step again.

Appendix 2: Install Go and UnZip in an Ubuntu operating system

- Install the Go programming language.

```
sudo apt-get install -y software-properties-common
sudo add-apt-repository ppa:longsleep/golang-backports
sudo apt-get update
sudo apt-get install -y golang-go
```

- Install the UnZip utility.

```
sudo apt-get -y install unzip
```

Appendix 3: Perform a full backup, save the full backup file to your computer, and then upload the full backup file to your OSS bucket

1. Perform a full backup on the self-managed MySQL instance and save the full backup file to your computer.

```
innobackupex --backup --host=<The IP address of the host in which the self-managed MySQ
L instance resides> --port=<The port number that is used to connect to the self-managed
MySQL instance> --user=<The username of the root account that is used to connect to the
self-managed MySQL instance> --password=<The password of the root account that is used
to connect to the self-managed MySQL instance> --stream=xbstream --compress <The tempor
ary directory that is used to store the full backup file> > /<The directory that is use
d to store the full backup file>/<The name of the full backup file>_qp.xb
```

   Examples:

```
innobackupex --backup --host=127.0.0.1 --port=3306 --user=root --password=Aa123456@ --s
tream=xbstream --compress /root/mysql/data > /root/backup_qp.xb
```

2. Upload the full backup file to your OSS bucket by using OSS_Stream.

```
cat /<The directory that is used to store the full backup file>/<The name of the full b
ackup file>_qp.xb | ./mysql-backup-helper-master/oss_stream/oss_stream -accessKeyId LTA
I5tCqY18jvvKk******** -accessKeySecret 4A5Q7ZVzcYnWMQPysXFxld******** -bucketName test
-endpoint oss-********.aliyuncs.com -objectName backup_qp.xb
```

   Examples:

```
cat /root/backup_qp.xb | ./mysql-backup-helper-master/oss_stream/oss_stream -accessKeyI
d LTAI5tCqY18jvvKk******** -accessKeySecret 4A5Q7ZVzcYnWMQPysXFxld******** -bucketName
test -endpoint oss-********.aliyuncs.com -objectName backup_qp.xb
```

Appendix 5: Limits

Item	Description

Item	Description
MySQL version	The self-managed MySQL instance must run one of the following MySQL versions: • MySQL 5.7.32 or earlier • MySQL 8.0.18 or earlier **Note** The self-managed MySQL instance must run the same MySQL version as the destination RDS instance. For example, the backup data of a self-managed MySQL instance that runs MySQL 5.7 can be restored only to an RDS instance that runs MySQL 5.7.
Self-managed MySQL instance	• The data of the self-managed MySQL instance must be stored in the datadir directory on the host in which the instance resides. You can run the following command in the CLI to access the datadir directory: `mysqladmin -u<The user name of the root account that is used to connect to the self-managed MySQL instance> -p<The password of the root account that is used to connect to the self-managed MySQL instance> variables | grep datadir`. • You must set the `innodb_data_file_path` parameter to the default value `ibdata1`.
Backup	• After the full backup of the self-managed MySQL instance is complete, the incremental data that is generated in the self-managed instance is not included in the full backup file. • If the self-managed MySQL instance runs MySQL 5.7, you must use Percona XtraBackup 2.4 to back up the instance. • When you use Percona XtraBackup to back up the self-managed MySQL instance, you cannot configure the `--tables`, `--tables-exclude`, `--tables-file`, `--databases`, or `--databases-file` option. • You cannot read encrypted objects from OSS buckets. Therefore, you must set the **Encryption Method** parameter to **None** when you create an OSS bucket. • Differential backup files and log backup files are not supported. • The name of the full backup file cannot contain special characters. If the name of the full backup file contains special characters, the file cannot be imported into the destination RDS instance. • After you authorize the service account of ApsaraDB RDS to access OSS buckets, a RAM role named AliyunRDSImportRole is created in Resource Access Management (RAM). Do not modify or delete this RAM role. If you modify or delete this RAM role, ApsaraDB RDS cannot download objects from OSS buckets. • Before the migration is complete, do not delete the full backup file from the OSS bucket. If you delete the full backup file before the migration is complete, the migration fails. • The name of the full backup file that is stored as an object in your OSS bucket must be suffixed by _QP.XB or TAR.GZ.

Item	Description
OSS	• If you use the CLI to upload the full backup file as a set of parts to your OSS bucket, you must make sure that the size of the file does not exceed 16 TB. For more information, see Limits. • Your OSS bucket must reside in the same region as the destination RDS instance.
Restoration	• You can migrate the data of the self-managed MySQL instance only to a new RDS instance. This way, you can prevent data overwrites in an existing RDS instance due to unintended operations. • You cannot migrate the data of the self-managed MySQL instance to an RDS instance whose storage capacity is less than the amount of data in the self-managed MySQL instance. For more information, see Primary ApsaraDB RDS instance types. • When you import the full backup file from your OSS bucket into ApsaraDB RDS, ApsaraDB RDS creates a temporary RDS instance, imports the full backup file into the temporary RDS instance, and then restores the data from the temporary RDS instance to the destination RDS instance. The default storage capacity of the temporary RDS instance is five times the size of the full backup file. If the available storage on the temporary RDS instance is insufficient after the full backup file is imported, you can increase the storage capacity of the temporary RDS instance. • The accounts, custom functions, and stored procedures of the self-managed MySQL instance cannot be migrated to the destination RDS instance. You must record the accounts, custom functions, and stored procedures. After the migration is complete, you must manually add the accounts, custom functions, and stored procedures to the destination RDS instance. • The time zone information of the self-managed MySQL instance cannot be migrated to the destination RDS instance. You must record the time zone information. After the migration is complete, you must manually configure the time zone of the destination RDS instance. • The destination RDS instance must be a pay-as-you-go instance that runs MySQL 5.7 or MySQL 8.0 on RDS Basic Edition with standard SSDs. **Note** After the migration is complete, you can perform the following operations: ◦ Upgrade the major engine version of an ApsaraDB RDS for MySQL instance ◦ Upgrade an ApsaraDB RDS for MySQL instance from Basic Edition to High-availability Edition ◦ Change the specifications of an ApsaraDB RDS for MySQL instance ◦ Change the billing method of an ApsaraDB RDS for MySQL instance from pay-as-you-go to subscription

Item	Description
Replication	• Data can be replicated based only on global transaction identifiers (GTIDs). Therefore, you must enable GTID-based replication and set the gtid_mode parameter and the enforce_gtid_consistency parameter to `ON` in the self-managed MySQL instance. • The default retention period of backup validation records is seven days. ApsaraDB RDS automatically deletes the backup validation records that are generated seven days ago and the snapshots of these records. Therefore, after the migration is complete, we recommend that you replicate the incremental data of the self-managed MySQL instance to the destination RDS instance at your earliest opportunity.

6.2.3. Migrate data from a self-managed Oracle database to an ApsaraDB RDS for MySQL instance

This topic describes how to migrate data from a self-managed Oracle database to an ApsaraDB RDS for MySQL instance by using Data Transmission Service (DTS). DTS supports schema migration, full data migration, and incremental data migration. When you migrate data from a self-managed Oracle database, you can select all of the supported migration types to ensure service continuity.

Prerequisites

- The version of the self-managed Oracle database is 9i, 10g, 11g, 12c, 18c, or 19c.
- Supplemental logging, SUPPLEMENTAL_LOG_DATA_PK, and SUPPLEMENTAL_LOG_DATA_UI are enabled for the self-managed Oracle database. For more information, see Supplemental Logging.
- The self-managed Oracle database runs in ARCHIVELOG mode. Archived log files of the Oracle database are accessible, and an appropriate retention period is specified for the archived log files. For more information, see Managing Archived Redo Log Files.
- The size of available storage in the RDS instance is larger than the size of data that you want to migrate from the self-managed Oracle database.

Precautions

- DTS uses read and write resources of the source and destination databases during full data migration. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you migrate data, evaluate the impact of data migration on the performance of the source and destination databases. We recommend that you migrate data during off-peak hours. For example, you can migrate data when the CPU utilization of the source and destination databases is less than 30%.
- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- If the self-managed Oracle database is deployed in a Real Application Cluster (RAC) architecture and is connected to DTS over an Alibaba Cloud virtual private cloud (VPC), you must connect the Single Client Access Name (SCAN) IP address of the Oracle RAC and the virtual IP address (VIP) of each node

to the VPC and configure routes. The settings ensure that your DTS task can run as expected. For more information, see Configure a route between DTS and Express Connect, VPN Gateway, or Smart Access Gateway.

> **Notice** When you configure the source Oracle database in the DTS console, you can specify the SCAN IP address of the Oracle RAC as the database endpoint or IP address.

- Table names in the RDS instance are not case-sensitive. If a table name in the self-managed Oracle database contains uppercase letters, ApsaraDB RDS for MySQL converts all uppercase letters to lowercase letters and then creates the table.

  If the self-managed Oracle database contains identical table names that differ only in capitalization, the table names are identified as duplicates. As a result, the "The object already exists" message may be displayed during schema migration. To prevent table name conflicts in the RDS instance, you can rename the migrated objects by using the object name mapping feature of DTS. For more information, see Object name mapping.

- DTS automatically creates a destination database in the ApsaraDB RDS for MySQL instance. If the name of the source database is invalid, you must manually create a destination database in the RDS instance before you configure a data migration task. For more information about the naming conventions of ApsaraDB RDS for MySQL databases and how to create a database, see Create a database.

Billing

Migration type	Task configuration fee	Internet traffic fee
Schema migration and full data migration	Free of charge.	Charged only when data is migrated from Alibaba Cloud over the Internet. For more information, see Pricing.
Incremental data migration	Charged. For more information, see Pricing.	

Migration types

- Schema migration

  DTS supports schema migration for tables and indexes. DTS does not support schema migration for the following types of objects: views, synonyms, triggers, stored procedures, stored functions, packages, and user-defined data types. DTS has the following limits on schema migration for tables and indexes:

  - DTS does not support schema migration for nested tables. DTS converts clustered tables and index-organized tables into standard tables in the destination database.
  - DTS does not support schema migration for function-based indexes, domain indexes, bitmap indexes, or reverse indexes.

- Full data migration

  DTS migrates the historical data of specified objects from the self-managed Oracle database to the destination database in the RDS instance.

- Incremental data migration

DTS uses the round-robin algorithm to retrieve redo log files from the self-managed Oracle database. Then, DTS synchronizes incremental data from the self-managed Oracle database to the destination database in the RDS instance. Incremental data migration ensures service continuity when you migrate data from the self-managed Oracle database to the destination database in the RDS instance.

SQL operations that can be synchronized during incremental data migration

- INSERT, DELETE, and UPDATE
- CREATE TABLE

> **Note** DTS cannot synchronize the CREATE TABLE operations that are performed to create tables in which functions are nested.

- ALTER TABLE, ADD COLUMN, DROP COLUMN, RENAME COLUMN, and ADD INDEX
- DROP TABLE
- RENAME TABLE, TRUNCATE TABLE, and CREATE INDEX

Data type mappings

For more information, see Data type mappings between heterogeneous databases.

Before you begin

Log on to the self-managed Oracle database, create an account that you want to use to collect data, and grant permissions to the account.

> **Note** If you created an account that is granted the permissions listed in the following table, you can skip this step.

Database	Schema migration	Full data migration	Incremental data migration
Self-managed Oracle database	Permissions of the schema owner	Permissions of the schema owner	DBA
ApsaraDB RDS for MySQL instance	Write permissions on the destination database	Write permissions on the destination database	Write permissions on the destination database

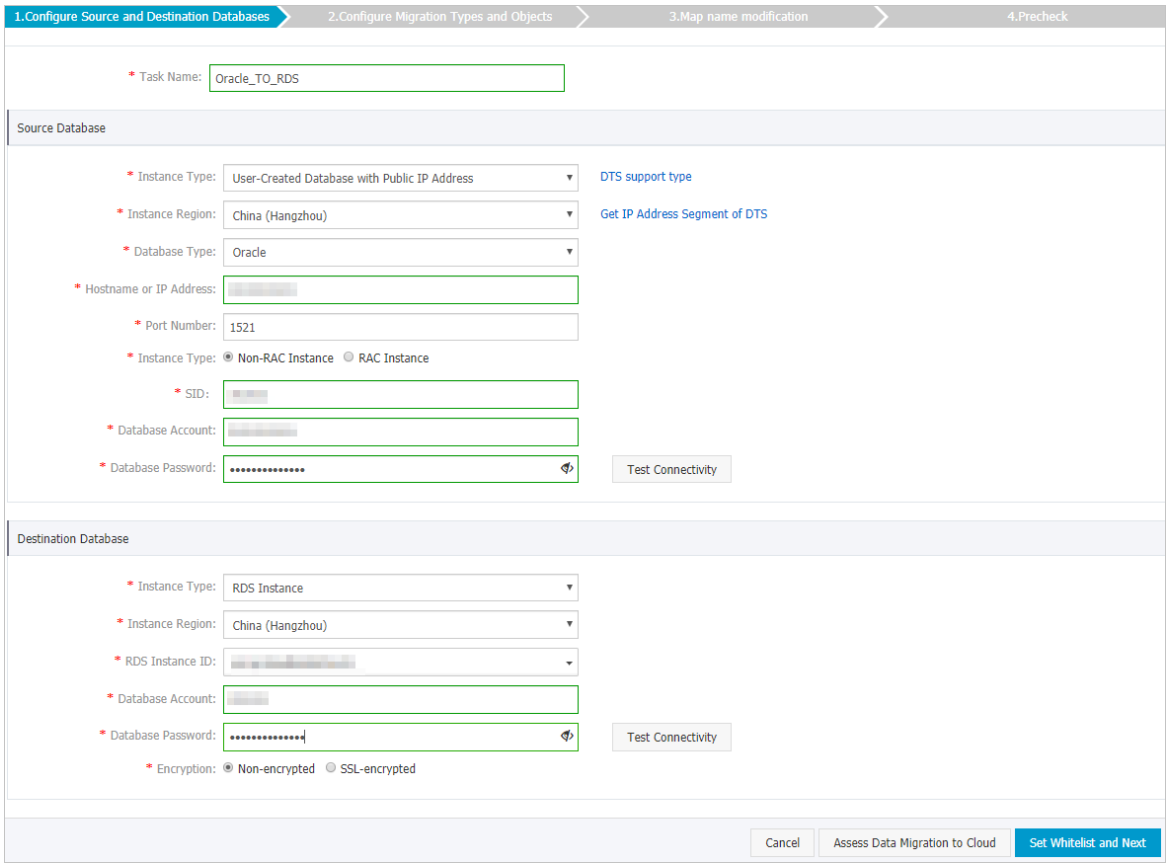
For more information about how to create an account and grant permissions to the account, see the following topics:

- Self-managed Oracle database: CREATE USER and GRANT.
- ApsaraDB RDS for MySQL instance: Create an account on an ApsaraDB RDS for MySQL instance and Modify the permissions of a standard account for an ApsaraDB RDS for MySQL instance.

Procedure

1. Log on to the DTS console.
2. In the left-side navigation pane, click **Data Migration**.

3. In the upper part of the **Migration Tasks** page, select the region where the RDS instance resides.
4. In the upper-right corner of the page, click **Create Migration Task**.
5. Configure the self-managed Oracle database and RDS instance.



Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify a name that can help you identify the task. You do not need to specify a unique task name.
	Instance Type	Select an instance type based on the deployment of the self-managed Oracle database. In this example, select **User-Created Database with Public IP Address**. **Note** If you select other instance types, you must set up the environment that is required for the self-managed Oracle database. For more information, see Preparation overview.

Section	Parameter	Description
Source Database	Instance Region	If you select **User-Created Database with Public IP Address** as the instance type, you do not need to configure the **Instance Region** parameter. **Note** If an IP address whitelist is configured for the self-managed Oracle database, you must add the CIDR blocks of DTS servers to the IP address whitelist of the database. You can click **Get IP Address Segment of DTS** next to **Instance Region** to obtain the CIDR blocks of DTS servers.
	Database Type	Select **Oracle**.
	Hostname or IP Address	Enter the endpoint that is used to connect to the self-managed Oracle database. In this example, enter the public IP address.
	Port Number	Enter the port number that is used to connect to the self-managed Oracle database. The port must be accessible over the Internet. The default port number is **1521**.
	Instance Type	◦ If you select **Non-RAC Instance**, you must configure the **SID** parameter. ◦ If you select **RAC or PDB Instance**, you must configure the **Service Name** parameter.
	Database Account	Enter the account that you created in the self-managed Oracle database. For more information about the permissions that are required for the account, see Before you begin.
	Database Password	Enter the password of the preceding account. **Note** After you specify the information about the self-managed Oracle database, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
Destination	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the RDS instance resides.
	RDS Instance ID	Select the ID of the RDS instance.
	Database Account	Enter the account that is used to connect to the RDS instance. For more information about the permissions that are required for the account, see Before you begin.

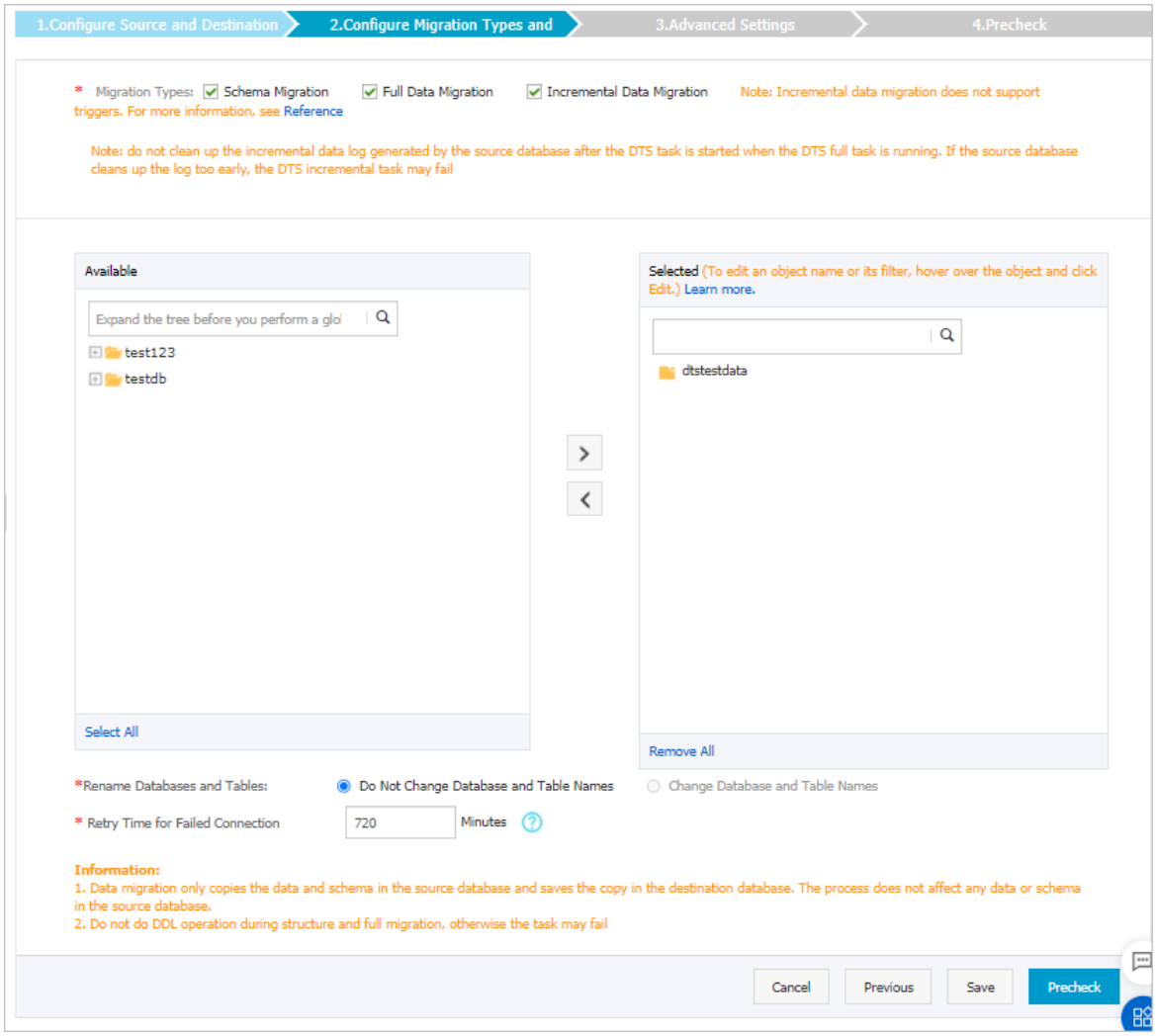
Database Section	Parameter	Description
	Database Password	Enter the password of the preceding account. **Note** After you specify the information about the RDS instance, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.

6. In the lower-right corner of the page, click **Set Whitelist and Next**.

Warning

- If the source or destination database instance is an Alibaba Cloud database instance, such as an ApsaraDB RDS for MySQL or ApsaraDB for MongoDB instance, or is a self-managed database hosted on Elastic Compute Service (ECS), DTS automatically adds the CIDR blocks of DTS servers to the whitelist of the database instance or ECS security group rules. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases. If the source or destination database is a self-managed database on data centers or is from other cloud service providers, you must manually add the CIDR blocks of DTS servers to allow DTS to access the database.
- If the CIDR blocks of DTS servers are automatically or manually added to the whitelist of the database instance or ECS security group rules, security risks may arise. Therefore, before you use DTS to migrate data, you must understand and acknowledge the potential risks and take preventive measures, including but not limited to the following measures: enhance the security of your account and password, limit the ports that are exposed, authenticate API calls, regularly check the whitelist or ECS security group rules and forbid unauthorized CIDR blocks, or connect the database to DTS by using Express Connect, VPN Gateway, or Smart Access Gateway.
- After the DTS task is completed or released, we recommend that you manually detect and remove the added CIDR blocks from the whitelist of the database instance or ECS security group rules.

7. Select the migration types and the objects that you want to migrate.



Setting	Description
Select the migration types	◦ If you want to perform only full data migration, select **Schema Migration** and **Full Data Migration**. ◦ If you want to ensure service continuity during data migration, select **Schema Migration**, **Full Data Migration**, and **Incremental Data Migration**. **Note** If you do not select **Incremental Data Migration**, make sure that no data is written to the self-managed Oracle database during full data migration. This ensures data consistency between the self-managed Oracle database and the RDS instance.

Setting	Description
Select the objects that you want to migrate	Select one or more objects in the **Available** section and click the icon to move the objects to the **Selected** section. **Note** • You can select columns, tables, or databases. • By default, the name of an object that is migrated to the RDS instance remains the same as that in the self-managed Oracle database. You can use the object name mapping feature to rename the objects that are migrated to the RDS instance. For more information, see Object name mapping.
Specify whether to rename objects	You can use the object name mapping feature to rename the objects that are migrated to the RDS instance. For more information, see Object name mapping.
Specify the retry time range for failed connections to the self-managed Oracle database or RDS instance	By default, if DTS fails to connect to the self-managed Oracle database or RDS instance, DTS retries within the following 12 hours. You can specify the retry time range based on your business requirements. If DTS is reconnected to the self-managed Oracle database and RDS instance within the specified time range, DTS resumes the data migration task. Otherwise, the data migration task fails. **Note** Within the time range in which DTS attempts to reconnect to the self-managed Oracle database and RDS instance, you are charged for the DTS instance. We recommend that you specify the retry time range based on your business requirements. You can also release the DTS instance at the earliest opportunity after the self-managed Oracle database and RDS instance are released.

8. Click **Precheck**.

   Note
   - A precheck is performed before the migration task starts. The migration task only starts after the precheck succeeds.
   - If the precheck fails, click the icon next to each failed check item to view the related details. Fix the issues as instructed and run the precheck again.

9. After the data migration task passes the precheck, click **Next**.
10. In the **Confirm Settings** dialog box, configure the **Channel Specification** parameter. Then, read and select **Data Transmission Service (Pay-as-you-go) Service Terms**.

11. Click **Buy and Start** to start the data migration task.
    - Full data migration

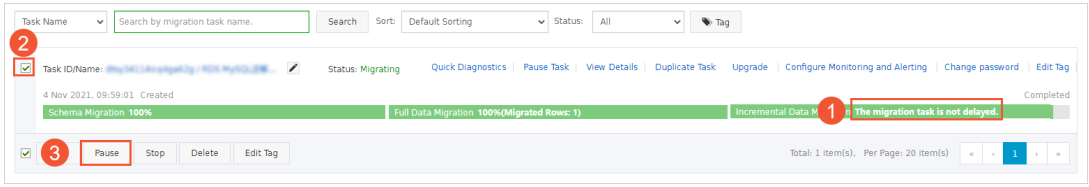
      Do not manually stop a full data migration task. If you manually stop a full data migration task, the data that is migrated to the RDS instance may be incomplete. You can wait until the full data migration task automatically stops.
    - Incremental data migration

      An incremental data migration task does not automatically stop. You must manually stop the task.

      > **Note** We recommend that you manually stop an incremental data migration task at an appropriate point in time. For example, you can stop the task during off-peak hours or before you switch your workloads over to the RDS instance.

      a. Wait until **Incremental Data Migration** and **The data migration task is not delayed** appear in the progress bar of the data migration task. Then, stop writing data to the self-managed Oracle database for a few minutes. The delay time of **incremental data migration** may be displayed in the progress bar.

      b. Wait until the status of **incremental data migration** changes to **The data migration task is not delayed** again. Then, manually stop the migration task.

12. Switch your workloads over to the RDS instance.

What to do next

The accounts that are used to migrate data are granted the read and write permissions. After the data migration is complete, you must delete the accounts of the self-managed Oracle database and the RDS instance to ensure database security.

Additional information

DTS supports reverse data transmission when you migrate data from a self-managed Oracle database to an ApsaraDB RDS for MySQL instance. You can use this feature to synchronize data changes from the ApsaraDB RDS for MySQL instance to the self-managed Oracle database. If you want to use the reverse data transmission feature, submit a ticket.

6.2.4. Migrate data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL instance

This topic describes how to migrate data from a self-managed MySQL database that is connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL database by using Data Transmission Service (DTS). DTS supports schema migration, full data migration, and incremental data migration. When you migrate data from a self-managed MySQL database, you can select all of the supported migration types to ensure service continuity.

Prerequisites

- The version of the self-managed MySQL database is 5.1, 5.5, 5.6, 5.7, or 8.0.
- The available storage space of the ApsaraDB RDS for MySQL instance is larger than the total size of the data in the self-managed MySQL database.
- The on-premises network to which the self-managed MySQL database belongs is connected to Alibaba Cloud over Express Connect, VPN Gateway, or Smart Access Gateway.

  > **Note** For more information, see Connect an on-premises database to DTS by using CEN.

Precautions

- DTS uses read and write resources of the source and destination databases during full data migration. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you migrate data, evaluate the impact of data migration on the performance of the source and destination databases. We recommend that you migrate data during off-peak hours. For example, you can migrate data when the CPU utilization of the source and destination databases is less than 30%.
- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- DTS uses the `ROUND(COLUMN,PRECISION)` function to retrieve values from columns of the FLOAT or DOUBLE data type. If you do not specify a precision, DTS sets the precision for the FLOAT data type to 38 digits and the precision for the DOUBLE data type to 308 digits. You must check whether the precision settings meet your business requirements.
- DTS automatically creates a destination database in the ApsaraDB RDS for MySQL instance. However, if the name of the source database is invalid, you must manually create a database in the ApsaraDB RDS for MySQL instance before you configure the data migration task.

  > **Note** For more information about the naming conventions of ApsaraDB RDS for MySQL databases and how to create a database, see Create a database on an ApsaraDB RDS for MySQL instance.

- If a data migration task fails, DTS automatically resumes the task. Before you switch your workloads to the destination instance, stop or release the data migration task. Otherwise, the data in the source instance will overwrite the data in the destination instance after the task is resumed.

Billing

Migration type	Task configuration fee	Internet traffic fee
Schema migration and full data migration	Free of charge.	Charged only when data is migrated from Alibaba Cloud over the Internet. For more information, see Pricing.
Incremental data migration	Charged. For more information, see Pricing.	

Migration types

- Schema migration

  DTS migrates the schemas of the required objects to the destination instance. DTS supports schema migration for the following types of objects: table, view, trigger, stored procedure, and function.

  > Note
  > - During schema migration, DTS changes the value of the SECURITY attribute from DEFINER to INVOKER for views, stored procedures, and functions.
  > - DTS does not migrate user information. To call a view, stored procedure, or function of the destination database, you must grant the read and write permissions to INVOKER.

- Full data migration

  DTS migrates historical data of the required objects from the self-managed MySQL database to the destination database in the ApsaraDB RDS for MySQL instance.

  > Note During full data migration, concurrent INSERT operations cause fragmentation in the tables of the destination database. After full data migration is completed, the tablespace of the destination database is larger than that of the source database.

- Incremental data migration

  After full data migration is completed, DTS retrieves binary log files from the self-managed MySQL database. Then, DTS synchronizes incremental data from the self-managed MySQL database to the destination ApsaraDB RDS for MySQL instance. Incremental data migration allows you to ensure service continuity when you migrate data from a self-managed MySQL database to Alibaba Cloud.

SQL operations that can be synchronized during incremental data migration

Operation type	SQL statement
DML	INSERT, UPDATE, DELETE, and REPLACE
DDL	• ALTER TABLE and ALTER VIEW • CREATE FUNCTION, CREATE INDEX, CREATE PROCEDURE, CREATE TABLE, and CREATE VIEW • DROP INDEX and DROP TABLE • RENAME TABLE • TRUNCATE TABLE

Permissions required for database accounts

Database	Schema migration	Full data migration	Incremental data migration
Self-managed MySQL database	The SELECT permission	The SELECT permission	The REPLICATION SLAVE, REPLICATION CLIENT, SHOW VIEW, and SELECT permissions
ApsaraDB RDS for MySQL instance	The read and write permissions	The read and write permissions	The read and write permissions

For more information about how to create and authorize a database account, see the following topics:

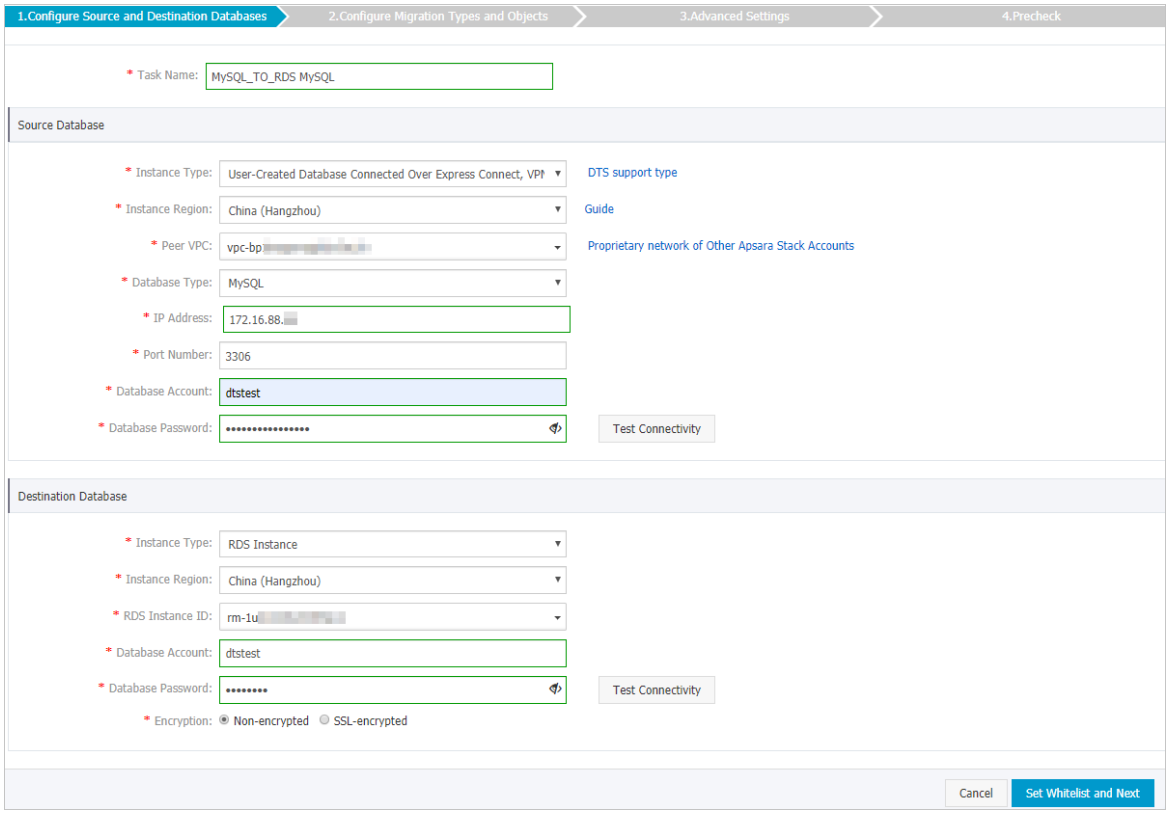
- Self-managed MySQL database: Create an account for a user-created MySQL database and configure binary logging
- ApsaraDB RDS for MySQL instance: Create an account on an ApsaraDB RDS for MySQL instance and Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance.

Before you begin

1. Create an account for a user-created MySQL database and configure binary logging.
2. Configure a route between DTS and Express Connect, VPN Gateway, or Smart Access Gateway.

Procedure

1. Log on to the DTS console.
2. In the left-side navigation pane, click **Data Migration**.
3. At the top of the **Migration Tasks** page, select the region where the destination cluster resides.
4. In the upper-right corner of the page, click **Create Migration Task**.
5. Configure the source and destination databases.



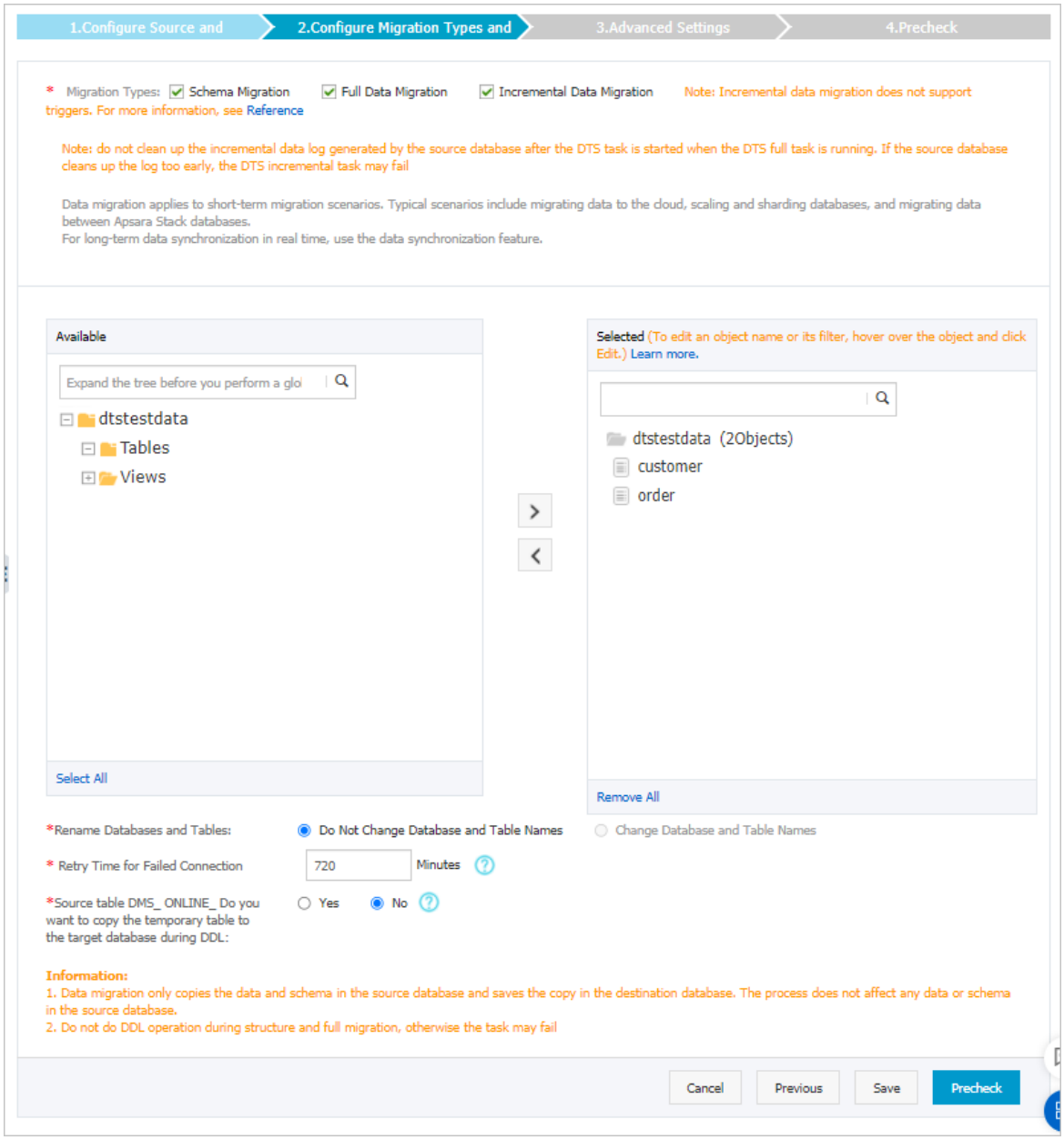
Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Database	Instance Type	Select **User-Created Database Connected over Express Connect, VPN Gateway, or Smart Access Gateway**.
	Instance Region	Select the region to which the virtual private cloud (VPC) that is connected to Express Connect, VPN Gateway, or Smart Access Gateway belongs.
	Peer VPC	Select the VPC that is connected to Express Connect, VPN Gateway, or Smart Access Gateway.
	Database Type	Select **MySQL**.
	IP Address	Enter the endpoint that is used to access the self-managed MySQL database.
	Port Number	Enter the service port number of the self-managed MySQL database. The default port number is **3306**.
	Database Account	Enter the account of the self-managed MySQL database. For more information about the permissions that are required for the account, see Permissions required for database accounts.

Section	Parameter	Description
	Database Password	Enter the password of the database account. **Note** After you specify the information about the self-managed Oracle database, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
Destination Database	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the destination RDS instance resides.
	RDS Instance ID	Select the ID of the destination RDS instance.
	Database Account	Enter the database account of the destination RDS instance. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the information about the RDS instance, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data migration task. For more information, see Configure SSL encryption on an ApsaraDB RDS for MySQL instance. **Note** The **Encryption** parameter is available only for regions in mainland China and the China (Hong Kong) region.

6. In the lower-right corner of the page, click **Set Whitelist and Next**.

> **Warning**
>
> - If the source or destination database instance is an Alibaba Cloud database instance, such as an ApsaraDB RDS for MySQL or ApsaraDB for MongoDB instance, or is a self-managed database hosted on Elastic Compute Service (ECS), DTS automatically adds the CIDR blocks of DTS servers to the whitelist of the database instance or ECS security group rules. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases. If the source or destination database is a self-managed database on data centers or is from other cloud service providers, you must manually add the CIDR blocks of DTS servers to allow DTS to access the database.
> - If the CIDR blocks of DTS servers are automatically or manually added to the whitelist of the database instance or ECS security group rules, security risks may arise. Therefore, before you use DTS to migrate data, you must understand and acknowledge the potential risks and take preventive measures, including but not limited to the following measures: enhance the security of your account and password, limit the ports that are exposed, authenticate API calls, regularly check the whitelist or ECS security group rules and forbid unauthorized CIDR blocks, or connect the database to DTS by using Express Connect, VPN Gateway, or Smart Access Gateway.
> - After the DTS task is completed or released, we recommend that you manually detect and remove the added CIDR blocks from the whitelist of the database instance or ECS security group rules.

7. Select the migration types and the objects to be migrated.



Setting	Description
Select the migration types	◦ To perform only full migration, select **Schema Migration** and **Full Data Migration**. ◦ To ensure service continuity during data migration, select **Schema Migration**, **Full Data Migration**, and **Incremental Data Migration**. **Notice** If **Incremental Data Migration** is not selected, we recommend that you do not write data to the source database during full data migration. This ensures data consistency between the source and destination databases.

Setting	Description
Select the objects to be migrated	Select one or more objects from the **Available** section and click the icon to move the objects to the **Selected** section. **Notice** ◦ You can select columns, tables, or databases as the objects to be migrated. ◦ By default, after an object is migrated to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are migrated to the destination database. For more information, see Object name mapping. ◦ If you use the object name mapping feature to rename an object, other objects that are dependent on the object may fail to be migrated.
Specify whether to rename objects	You can use the object name mapping feature to rename the objects that are migrated to the destination instance. For more information, see Object name mapping.
Specify the retry time for failed connections to the source or destination database	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data migration task. Otherwise, the data migration task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

Setting	Description
Specify whether to copy temporary tables to the destination database when DMS performs online DDL operations on the source table	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to migrate temporary tables generated by online DDL operations. ○ **Yes**: DTS migrates the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data migration task may be delayed. ○ **No**: DTS does not migrate the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is migrated. **Note** If you select No, the tables in the destination database may be locked.

8. In the lower-right corner of the page, click **Precheck**.

   Note
   - Before you can start the data migration task, a precheck is performed. You can start the data migration task only after the task passes the precheck.
   - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
     - You can troubleshoot the issues based on the causes and run a precheck again.
     - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

9. After the task passes the precheck, click **Next**.
10. In the **Confirm Settings** dialog box, specify the **Channel Specification** parameter and select **Data Transmission Service (Pay-As-You-Go) Service Terms**.
11. Click **Buy and Start** to start the data migration task.
    - Schema migration and full data migration

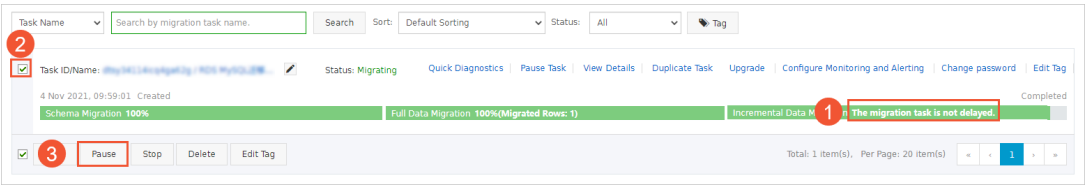
      We recommend that you do not manually stop the task during full data migration. Otherwise, the data migrated to the destination database will be incomplete. You can wait until the data migration task automatically stops.
    - Schema migration, full data migration, and incremental data migration

      The task does not automatically stop during incremental data migration. You must manually stop the task.

> **Notice** We recommend that you select an appropriate time to manually stop the data migration task. For example, you can stop the task during off-peak hours or before you switch your workloads to the destination cluster.

a. Wait until **Incremental Data Migration** and **The migration task is not delayed** appear in the progress bar of the migration task. Then, stop writing data to the source database for a few minutes. The delay time of **incremental data migration** may be displayed in the progress bar.

b. Wait until the status of **incremental data migration** changes to **The migration task is not delayed** again. Then, manually stop the migration task.



12. Switch your workloads to the ApsaraDB RDS for MySQL instance.

6.2.5. Migrate data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL instance across Alibaba Cloud accounts

This topic describes how to migrate data from a self-managed MySQL database that is connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL instance by using Data Transmission Service (DTS). In this scenario, the Express Connect circuit and the destination RDS instance are owned by different Alibaba Cloud accounts. DTS supports schema migration, full data migration, and incremental data migration. You can select all of the supported migration types to ensure service continuity.

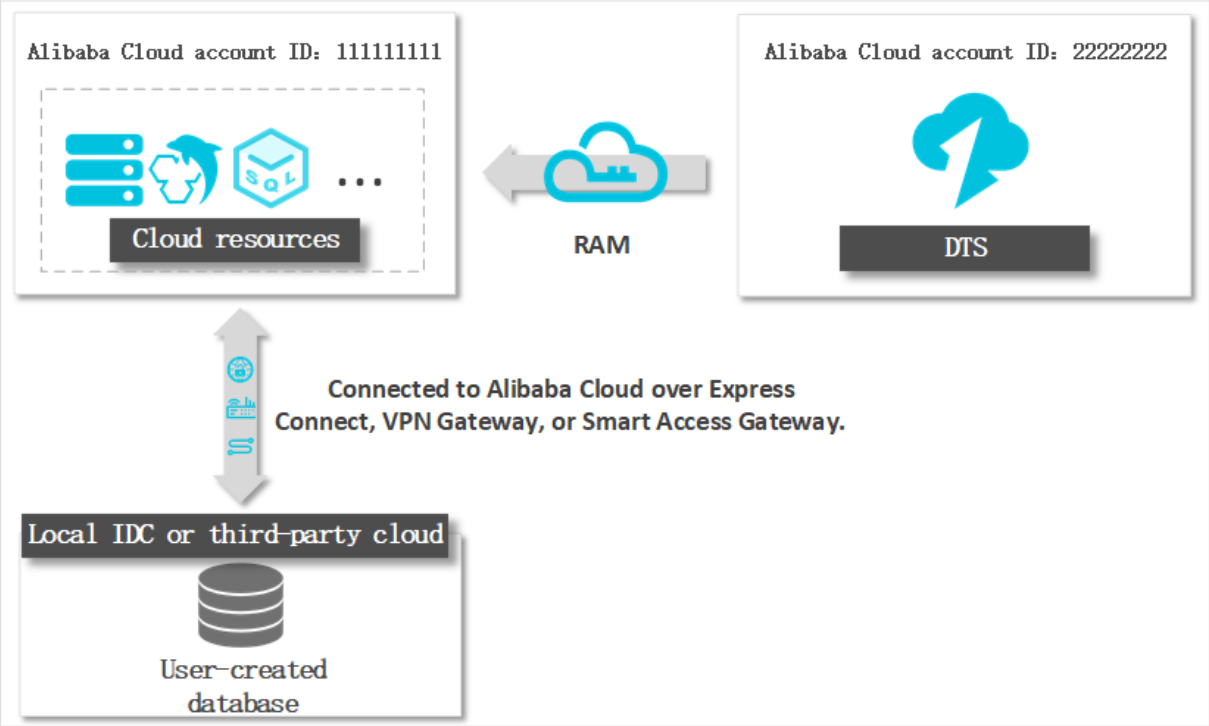
Prerequisites

- The version of the self-managed MySQL database is 5.1, 5.5, 5.6, 5.7, or 8.0.
- The available storage space of the ApsaraDB RDS for MySQL instance is larger than the total size of the data in the self-managed MySQL database.
- The on-premises network to which the self-managed MySQL database belongs is connected to Alibaba Cloud VPC over Express Connect, VPN Gateway, or Smart Access Gateway. The Express Connect circuit and the destination RDS instance are owned by different Alibaba Cloud accounts.

> **Note** For more information, see Connect an on-premises database to DTS by using CEN.

Context

The data center that hosts your database is connected to Alibaba Cloud VPC over Express Connect, VPN Gateway, or Smart Access Gateway. You need to migrate data from the on-premises database to an ApsaraDB RDS for MySQL instance across different Alibaba Cloud accounts. The following figure shows the architecture for this scenario.



Precautions

- DTS uses read and write resources of the source and destination databases during full data migration. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you migrate data, evaluate the impact of data migration on the performance of the source and destination databases. We recommend that you migrate data during off-peak hours. For example, you can migrate data when the CPU utilization of the source and destination databases is less than 30%.
- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- DTS uses the `ROUND(COLUMN,PRECISION)` function to retrieve values from columns of the FLOAT or DOUBLE data type. If you do not specify a precision, DTS sets the precision for the FLOAT data type to 38 digits and the precision for the DOUBLE data type to 308 digits. You must check whether the precision settings meet your business requirements.
- DTS automatically creates a destination database in the ApsaraDB RDS for MySQL instance. However, if the name of the source database is invalid, you must manually create a database in the ApsaraDB RDS for MySQL instance before you configure the data migration task.

> **Note** For more information about the naming conventions of ApsaraDB RDS for MySQL databases and how to create a database, see Create a database on an ApsaraDB RDS for MySQL instance.

- If a data migration task fails, DTS automatically resumes the task. Before you switch your workloads to the destination instance, stop or release the data migration task. Otherwise, the data in the source instance will overwrite the data in the destination instance after the task is resumed.

Billing

Migration type	Task configuration fee	Internet traffic fee
Schema migration and full data migration	Free of charge.	Charged only when data is migrated from Alibaba Cloud over the Internet. For more information, see Pricing.
Incremental data migration	Charged. For more information, see Pricing.	

Migration types

- Schema migration

  DTS migrates the schemas of the required objects to the destination instance. DTS supports schema migration for the following types of objects: table, view, trigger, stored procedure, and function.

  > **Note**
  > - During schema migration, DTS changes the value of the SECURITY attribute from DEFINER to INVOKER for views, stored procedures, and functions.
  > - DTS does not migrate user information. To call a view, stored procedure, or function of the destination database, you must grant the read and write permissions to INVOKER.

- Full data migration

  DTS migrates historical data of the required objects from the self-managed MySQL database to the destination database in the ApsaraDB RDS for MySQL instance.

  > **Note** During full data migration, concurrent INSERT operations cause fragmentation in the tables of the destination database. After full data migration is completed, the tablespace of the destination database is larger than that of the source database.

- Incremental data migration

  After full data migration is completed, DTS retrieves binary log files from the self-managed MySQL database. Then, DTS synchronizes incremental data from the self-managed MySQL database to the destination ApsaraDB RDS for MySQL instance. Incremental data migration allows you to ensure service continuity when you migrate data from a self-managed MySQL database to Alibaba Cloud.

SQL operations that can be synchronized during incremental data migration

Operation type	SQL statement
DML	INSERT, UPDATE, DELETE, and REPLACE

Operation type	SQL statement
DDL	• ALTER TABLE and ALTER VIEW • CREATE FUNCTION, CREATE INDEX, CREATE PROCEDURE, CREATE TABLE, and CREATE VIEW • DROP INDEX and DROP TABLE • RENAME TABLE • TRUNCATE TABLE

Permissions required for database accounts

Database	Schema migration	Full data migration	Incremental data migration
Self-managed MySQL database	The SELECT permission	The SELECT permission	The REPLICATION SLAVE, REPLICATION CLIENT, SHOW VIEW, and SELECT permissions
ApsaraDB RDS for MySQL instance	The read and write permissions	The read and write permissions	The read and write permissions

For information about how to create and authorize a database account, see the following topics:

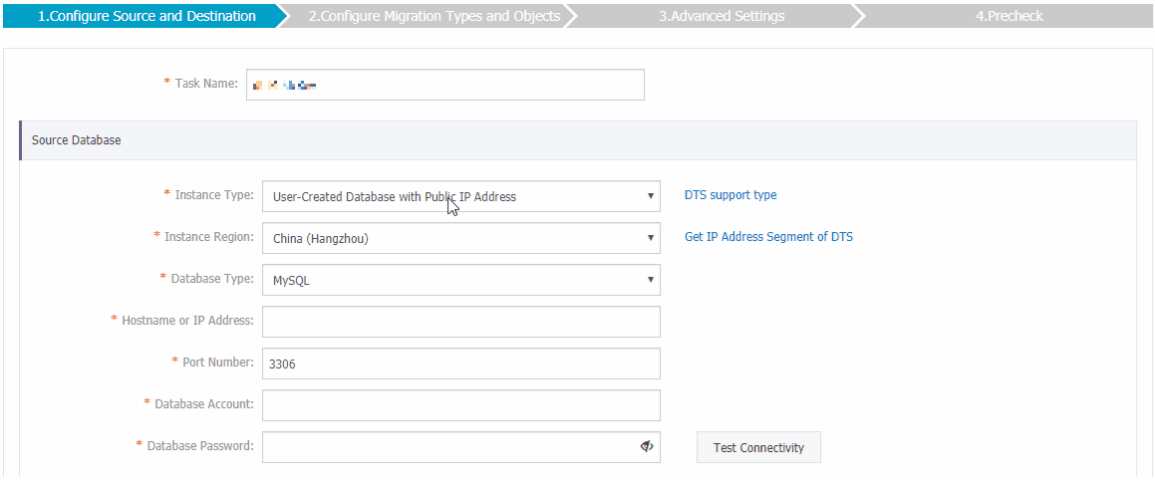
- Self-managed MySQL database: Create an account for a user-created MySQL database and configure binary logging
- ApsaraDB RDS for MySQL instance: Create an account on an ApsaraDB RDS for MySQL instance and Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance.

Before you begin

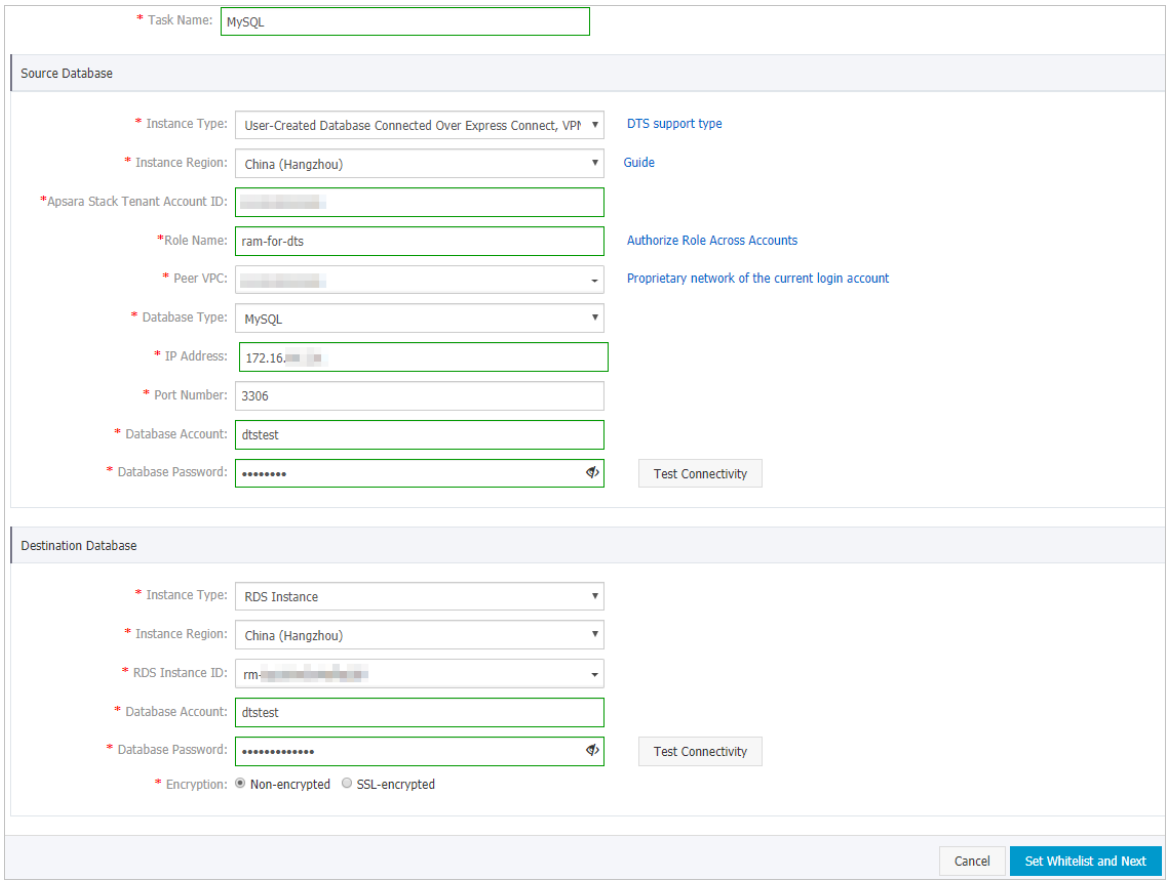
1. Create an account for a user-created MySQL database and configure binary logging.
2. Log on to the Alibaba Cloud Management Console by using the Alibaba Cloud account that owns the Express Connect circuit. Authorize DTS to access the network that is connected over Express Connect. For more information, see Configure a route between DTS and Express Connect, VPN Gateway, or Smart Access Gateway.
3. Create a RAM role and authorize the RAM role to access the resources of the Alibaba Cloud account. For more information, see Configure RAM authorization for data migration or synchronization from a self-managed database in a VPC across different Alibaba Cloud accounts.

Procedure

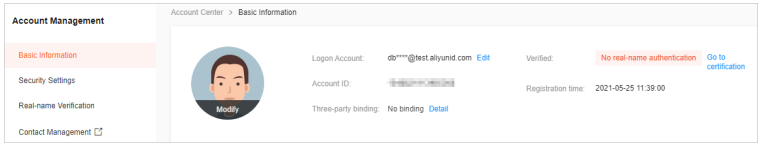
1. Use the Alibaba Cloud account that owns the destination RDS instance to log on to the DTS console.
2. In the left-side navigation pane, click **Data Migration**.
3. At the top of the **Migration Tasks** page, select the region where the destination cluster resides.
4. In the upper-right corner of the page, click **Create Migration Task**.
5. Select **User-Created Database Connected over Express Connect, VPN Gateway, or Smart Access Gateway** as the instance type. Then, click **VPC of Another Alibaba Cloud Account** next to the Peer VPC field.



6. Configure the source and destination databases.



Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
	Instance Type	Select **User-Created Database Connected over Express Connect, VPN Gateway, or Smart Access Gateway**.

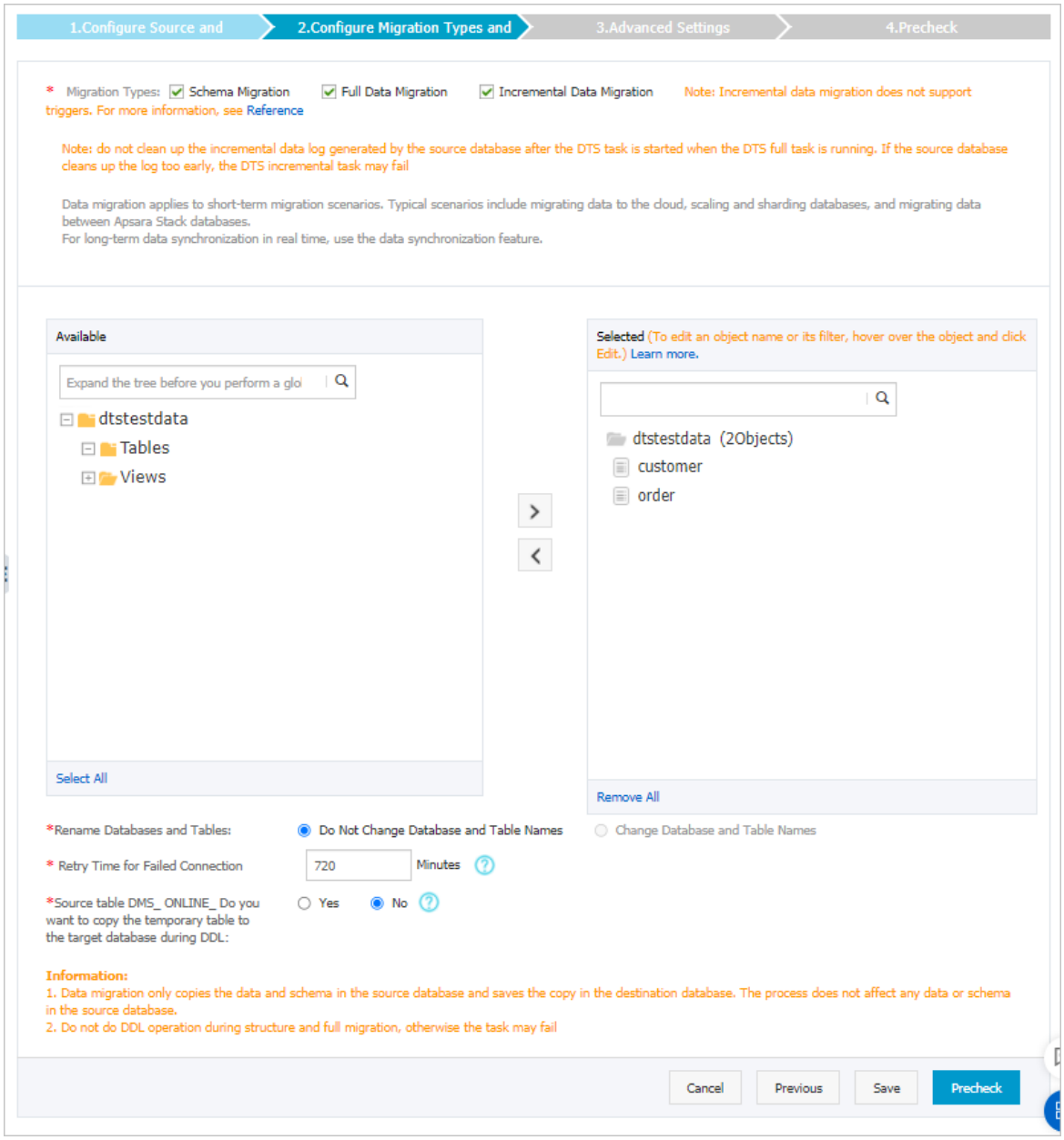
Section	Parameter	Description
Source Database	Instance Region	Select the region of the VPC that is connected to the self-managed MySQL database.
	Alibaba Cloud Account ID	Enter the ID of the Alibaba Cloud account that owns the Express Connect circuit. **Note** To obtain the ID of the Alibaba Cloud account that owns the Express Connect circuit, you must log on to the Account Management console by using this account. The account ID is displayed on the Security Settings page. 
	Role Name	Enter the name of the RAM role that you created earlier in Before you begin.
	Peer VPC	Select the ID of the VPC that is connected to the self-managed MySQL database.
	Database Type	Select **MySQL**.
	IP Address	Enter the endpoint that is used to access the self-managed MySQL database.
	Port Number	Enter the service port number of the self-managed MySQL database. The default port number is **3306**.
	Database Account	Enter the account of the self-managed MySQL database. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the information about the self-managed Oracle database, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the destination RDS instance resides.
	RDS Instance ID	Select the ID of the destination RDS instance.

Section	Parameter	Description
Destination Database	Database Account	Enter the database account of the destination RDS instance. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the information about the RDS instance, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data migration task. For more information, see Configure SSL encryption on an ApsaraDB RDS for MySQL instance. **Note** The **Encryption** parameter is available only for regions in mainland China and the China (Hong Kong) region.

7. In the lower-right corner of the page, click **Set Whitelist and Next**.

   Note DTS adds the CIDR blocks of DTS servers to the whitelist of the destination ApsaraDB RDS for MySQL instance. This ensures that DTS servers can connect to the destination RDS instance.

8. Select the migration types and the objects to be migrated.



Setting	Description
Select the migration types	◦ To perform only full migration, select **Schema Migration** and **Full Data Migration**. ◦ To ensure service continuity during data migration, select **Schema Migration**, **Full Data Migration**, and **Incremental Data Migration**. **Note** If **Incremental Data Migration** is not selected, we recommend that you do not write data to the source database during data migration. This ensures data consistency between the source and destination databases.

Setting	Description
Select the objects to be migrated	Select one or more objects from the **Available** section and click the icon to move the objects to the **Selected** section. **Note** ◦ You can select columns, tables, or databases as the objects to be migrated. ◦ By default, after an object is migrated to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are migrated to the destination database. For more information, see Object name mapping. ◦ If you use the object name mapping feature to rename an object, other objects that are dependent on the object may fail to be migrated.
Specify whether to rename objects	You can use the object name mapping feature to rename the objects that are migrated to the destination instance. For more information, see Object name mapping.
Specify the retry time for failed connections to the source or destination database	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data migration task. Otherwise, the data migration task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

Setting	Description
Specify whether to copy temporary tables to the destination database when DMS performs online DDL operations on the source table	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to migrate temporary tables generated by online DDL operations. ○ **Yes**: DTS migrates the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data migration task may be delayed. ○ **No**: DTS does not migrate the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is migrated. **Note** If you select No, the tables in the destination database may be locked.

9. In the lower-right corner of the page, click **Precheck**.

   Note
   - Before you can start the data migration task, a precheck is performed. You can start the data migration task only after the task passes the precheck.
   - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
     - You can troubleshoot the issues based on the causes and run a precheck again.
     - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

10. After the task passes the precheck, click **Next**.
11. In the **Confirm Settings** dialog box, specify the **Channel Specification** parameter and select **Data Transmission Service (Pay-As-You-Go) Service Terms**.
12. Click **Buy and Start** to start the data migration task.
    - Schema migration and full data migration

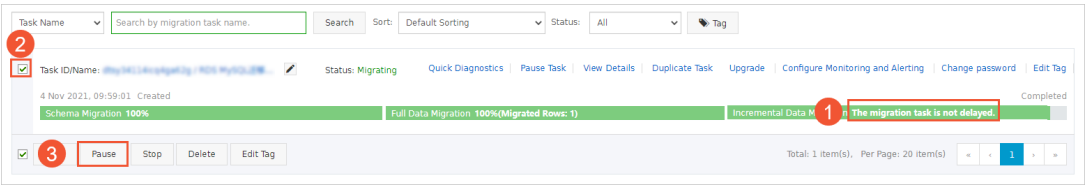
      We recommend that you do not manually stop the task during full data migration. Otherwise, the data migrated to the destination database will be incomplete. You can wait until the data migration task automatically stops.
    - Schema migration, full data migration, and incremental data migration

      The task does not automatically stop during incremental data migration. You must manually stop the task.

> **Notice** We recommend that you select an appropriate time to manually stop the data migration task. For example, you can stop the task during off-peak hours or before you switch your workloads to the destination cluster.

a. Wait until **Incremental Data Migration** and **The migration task is not delayed** appear in the progress bar of the migration task. Then, stop writing data to the source database for a few minutes. The delay time of **incremental data migration** may be displayed in the progress bar.

b. Wait until the status of **incremental data migration** changes to **The migration task is not delayed** again. Then, manually stop the migration task.



13. Switch your workloads to the destination ApsaraDB RDS for MySQL instance.

6.2.6. Migrate data from a self-managed Db2 database to an ApsaraDB RDS for MySQL instance

This topic describes how to migrate data from a self-managed Db2 database to an ApsaraDB RDS for MySQL instance by using Data Transmission Service (DTS). DTS supports schema migration, full data migration, and incremental data migration. When you migrate data from a self-managed Db2 database, you can select all of the supported migration types to ensure service continuity.

Prerequisites

- The version of the Db2 database is 9.7 to 11.5.

  > **Note** DTS supports data migration from a Db2 for i database of version 7.3 or 7.4 to an ApsaraDB RDS for MySQL instance. You can follow the procedure described in this topic to migrate data from a Db2 for i database to an ApsaraDB RDS for MySQL instance.

- The available storage space of the ApsaraDB RDS for MySQL instance is larger than the total size of the data in the Db2 database.

Precautions

- In this scenario, DTS cannot synchronize data definition language (DDL) operations.
- If the name of the source database is invalid, you must create a database in the ApsaraDB RDS for MySQL instance before you configure a data migration task.

  > **Note** For more information about how to create a database and the database naming conventions, see Create a database on an ApsaraDB RDS for MySQL instance.

- DTS uses read and write resources of the source and destination databases during full data migration. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become

unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you migrate data, evaluate the impact of data migration on the performance of the source and destination databases. We recommend that you migrate data during off-peak hours. For example, you can migrate data when the CPU utilization of the source and destination databases is less than 30%.

- If a data migration task fails, DTS automatically resumes the task. Before you switch your workloads to the destination instance, stop or release the data migration task. Otherwise, the data in the source database will overwrite the data in the destination instance after the task is resumed.
- DTS synchronizes incremental updates from a DB2 database to the destination database based on the Change Data Capture (CDC) replication technology of Db2. However, the CDC replication technology has its own limits. For more information, see General data restrictions for SQL Replication.

Billing

Migration type	Task configuration fee	Internet traffic fee
Schema migration and full data migration	Free of charge.	Charged only when data is migrated from Alibaba Cloud over the Internet. For more information, see Pricing.
Incremental data migration	Charged. For more information, see Pricing.	

Migration types

- Schema migration

  DTS migrates the schemas of the required objects to the destination instance. DTS supports schema migration for the following types of objects: table, index, and foreign key.

- Full data migration

  DTS migrates historical data of the required objects from the Db2 database to the destination database in the ApsaraDB RDS for MySQL instance.

- Incremental data migration

  After full data migration is complete, DTS synchronizes incremental data from the Db2 database to the destination database in the ApsaraDB RDS for MySQL instance. Incremental data migration allows you to ensure service continuity when you migrate data from a Db2 database.

Permissions required for database accounts

Database	Schema migration	Full data migration	Incremental data migration
Db2 database	The CONNECT and SELECT permissions	The CONNECT and SELECT permissions	The DBADM permission
ApsaraDB RDS for MySQL instance	The read and write permissions	The read and write permissions	The read and write permissions

For more information about how to create and authorize a database account, see the following topics:

- Db2 database: Creating group and user IDs for a Db2 database installation (Linux and UNIX) and Authorities overview
- ApsaraDB RDS for MySQL instance: Create an account on an ApsaraDB RDS for MySQL instance and Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance.

Data migration process

To prevent data migration failures caused by dependencies between objects, DTS migrates the schemas and data of the Db2 database in the following order:

1. Migrate the schemas and indexes.
2. Perform full data migration.
3. Migrate the schemas of foreign keys.
4. Perform incremental data migration.

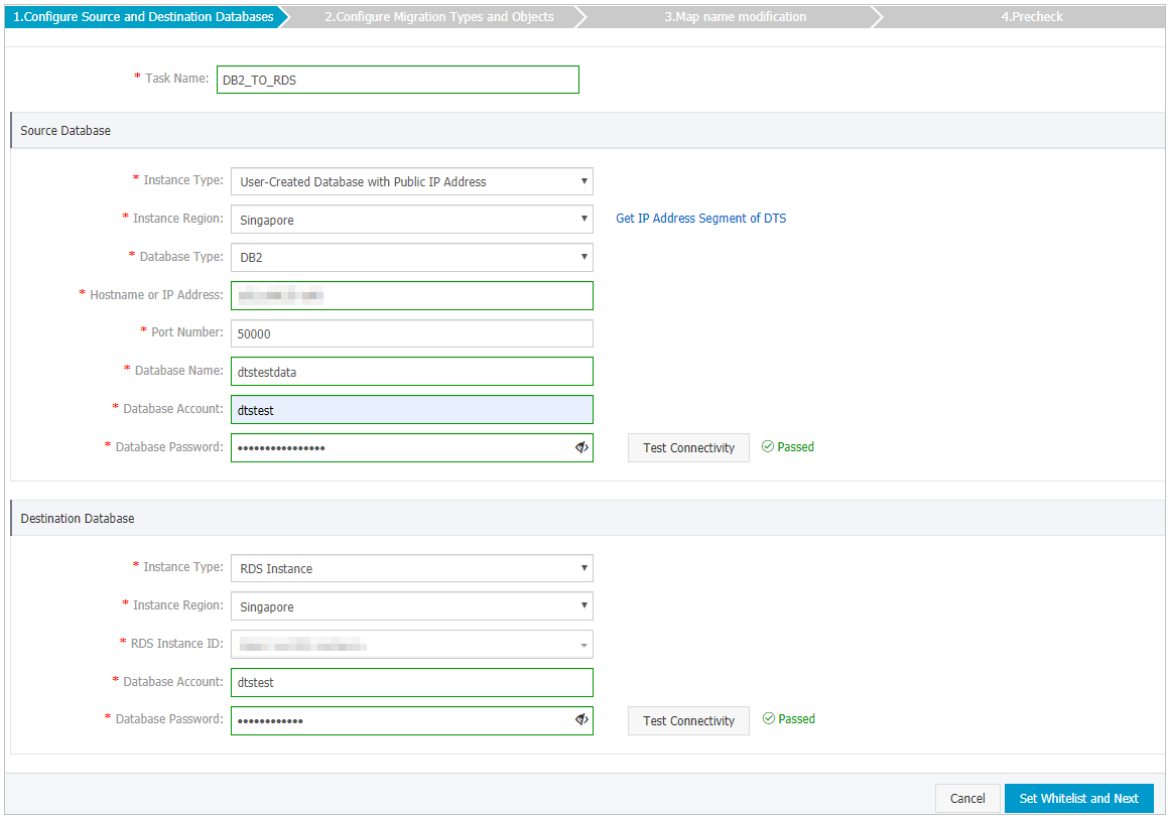
Before you begin

Before you configure an incremental data migration task, enable the archive log feature for the Db2 database. For more information, see Primary log archive method and Secondary log archive method.

> **Note** Skip this step if you perform only full data migration.

Procedure

1. Log on to the DTS console.
2. In the left-side navigation pane, click **Data Migration**.
3. In the upper part of the **Migration Tasks** page, select the region where the RDS instance resides.
4. In the upper-right corner of the page, click **Create Migration Task**.
5. Configure the **source and destination databases**.



Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
	Instance Type	Select an instance type based on the deployment of the source database. In this example, select **User-Created Database with Public IP Address**. **Note** If you select other instance types, you must deploy the network environment for the self-managed database. For more information, see Preparation overview.
	Instance Region	If the instance type is set to **User-Created Database with Public IP Address**, you do not need to specify the **instance region**. **Note** If a whitelist is configured for the Db2 database, you must add the CIDR blocks of DTS servers to the whitelist of the database. You can click **Get IP Address Segment of DTS** next to **Instance Region** to obtain the CIDR blocks of DTS servers.
	Database Type	Select **DB2**.
	Hostname or IP Address	Enter the endpoint that is used to connect to the Db2 database. In this example, enter the public IP address.

Section	Parameter	Description
Source Database	Port Number	Enter the service port number of the Db2 database. The default port number is **50000**. **Note** The service port of the Db2 database must be accessible over the Internet.
	Database Name	Enter the name of the Db2 database.
	Database Account	Enter the account of the Db2 database. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the Db2 database account. **Note** After you specify the source database parameters, click **Test Connectivity** next to **Database Password** to verify whether the specified parameters are valid. If the specified parameters are valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Modify the source database parameters based on the check results.
Destination Database	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the destination RDS instance resides.
	RDS Instance ID	Select the ID of the destination RDS instance.
	Database Account	Enter the database account of the destination RDS instance. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the destination database parameters, click **Test Connectivity** next to **Database Password** to verify whether the specified parameters are valid. If the specified parameters are valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Modify the destination database parameters based on the check results.

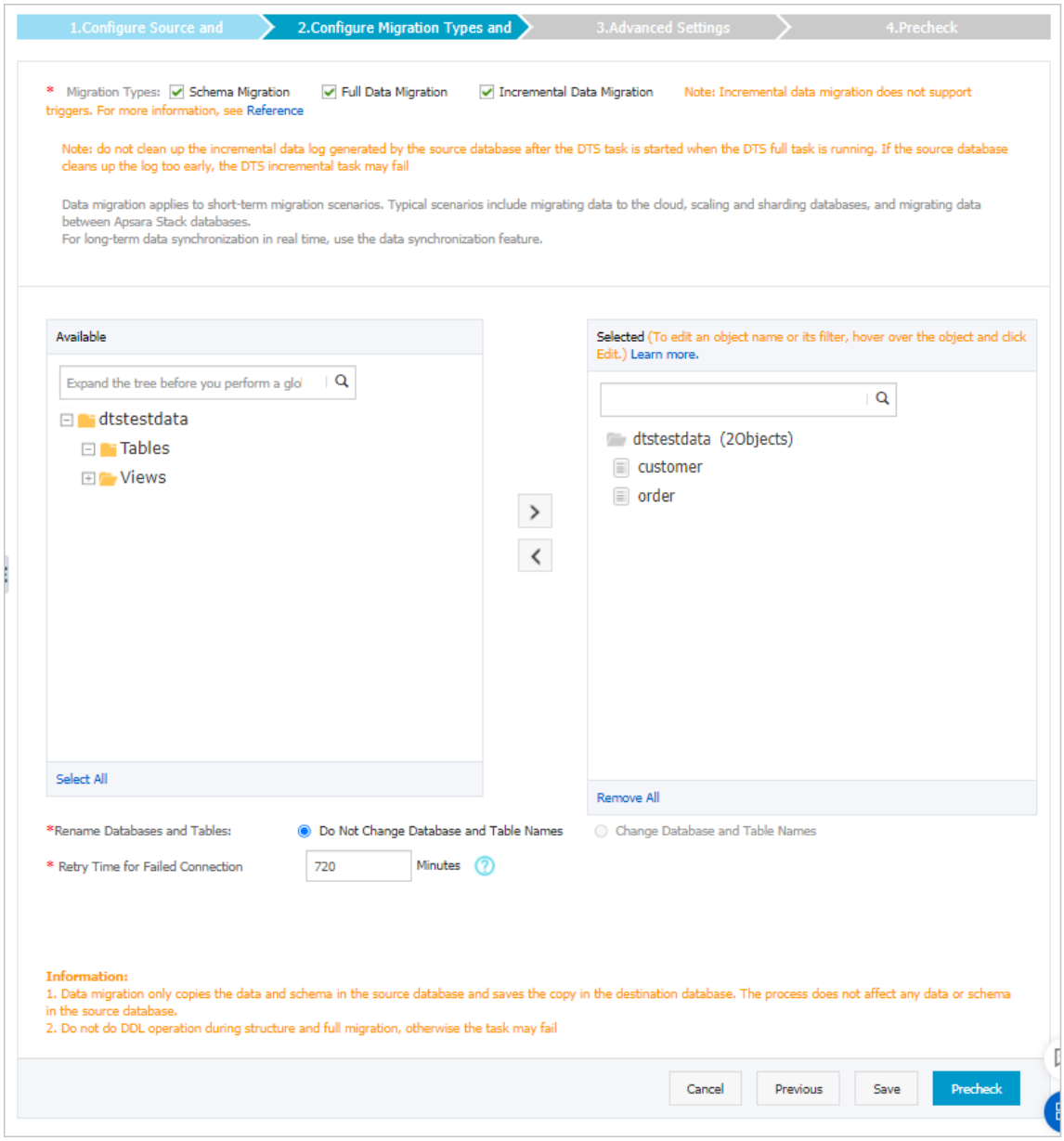
Section	Parameter	Description
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data migration task. For more information, see Configure SSL encryption on an ApsaraDB RDS for MySQL instance. **Note** The **Encryption** parameter is available only for regions in mainland China and the China (Hong Kong) region.

6. In the lower-right corner of the page, click **Set Whitelist and Next**.

Warning

- If the source or destination database instance is an Alibaba Cloud database instance, such as an ApsaraDB RDS for MySQL or ApsaraDB for MongoDB instance, or is a self-managed database hosted on Elastic Compute Service (ECS), DTS automatically adds the CIDR blocks of DTS servers to the whitelist of the database instance or ECS security group rules. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases. If the source or destination database is a self-managed database on data centers or is from other cloud service providers, you must manually add the CIDR blocks of DTS servers to allow DTS to access the database.
- If the CIDR blocks of DTS servers are automatically or manually added to the whitelist of the database instance or ECS security group rules, security risks may arise. Therefore, before you use DTS to migrate data, you must understand and acknowledge the potential risks and take preventive measures, including but not limited to the following measures: enhance the security of your account and password, limit the ports that are exposed, authenticate API calls, regularly check the whitelist or ECS security group rules and forbid unauthorized CIDR blocks, or connect the database to DTS by using Express Connect, VPN Gateway, or Smart Access Gateway.
- After the DTS task is completed or released, we recommend that you manually detect and remove the added CIDR blocks from the whitelist of the database instance or ECS security group rules.

7. Select the migration types and the objects to be migrated.



Setting	Description
Select the migration types	◦ To perform only full data migration, select **Schema Migration** and **Full Data Migration**. ◦ To ensure service continuity during data migration, select **Schema Migration**, **Full Data Migration**, and **Incremental Data Migration**. **Note** If **Incremental Data Migration** is not selected, we recommend that you do not write data to the Db2 database during data migration. This ensures data consistency between the source and destination databases.

Setting	Description
Select the objects to be migrated	Select one or more objects from the **Available** section and click the icon to move the objects to the **Selected** section. **Note** • You can select columns, tables, or databases as the objects to be migrated. • By default, after an object is migrated to the destination RDS instance, the name of the object remains the same as that in the Db2 database. You can use the object name mapping feature to change the names of the objects that are migrated to the destination RDS instance. For more information, see Object name mapping. • If you use the object name mapping feature on an object, other objects that are dependent on the object may fail to be migrated.
Specify whether to rename object names	You can use the object name mapping feature to change the names of the objects that are migrated to the destination instance. For more information, see Object name mapping.
Specify the retry time for failed connections to the source or destination database	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 12 hours. You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data migration task. Otherwise, the data migration task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

8. Click **Precheck**.

   Note
   - A precheck is performed before the migration task starts. The migration task only starts after the precheck succeeds.
   - If the precheck fails, click the icon next to each failed check item to view the related details. Fix the issues as instructed and run the precheck again.

9. After the data migration task passes the precheck, click **Next**.

10. In the **Confirm Settings** dialog box, configure the **Channel Specification** parameter. Then, read and select **Data Transmission Service (Pay-as-you-go) Service Terms**.
11. Click **Buy and Start** to start the data migration task.
    - Full data migration

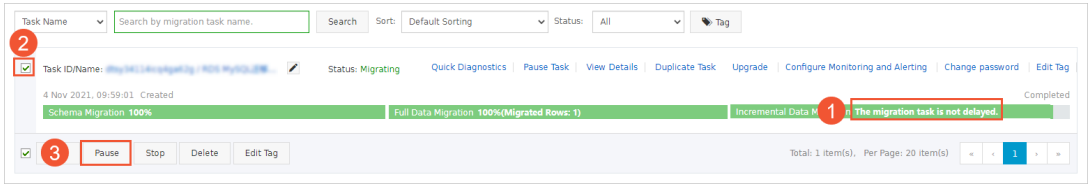
      Do not manually stop a full data migration task. If you manually stop a full data migration task, the data that is migrated to the RDS instance may be incomplete. You can wait until the full data migration task automatically stops.
    - Incremental data migration

      An incremental data migration task does not automatically stop. You must manually stop the task.

      > **Note** We recommend that you manually stop an incremental data migration task at an appropriate point in time. For example, you can stop the task during off-peak hours or before you switch your workloads over to the RDS instance.

      a. Wait until **Incremental Data Migration** and **The data migration task is not delayed** appear in the progress bar of the data migration task. Then, stop writing data to the self-managed Oracle database for a few minutes. The delay time of **incremental data migration** may be displayed in the progress bar.

      b. Wait until the status of **incremental data migration** changes to **The data migration task is not delayed** again. Then, manually stop the migration task.



12. Switch your workloads to the ApsaraDB RDS for MySQL instance.

6.2.7. Use mysqldump to migrate data from a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance

This topic describes how to migrate data from a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance by using the mysqldump plug-in. The mysqldump plug-in is easy to use but causes long downtime. The mysqldump plug-in is suitable for scenarios in which the data volume is small or long downtime does not have a negative impact on your business.

Prerequisites

IP address whitelists are configured, a public endpoint is obtained, and databases and accounts are created in the RDS for MySQL instance. For more information, see General workflow to use ApsaraDB RDS for MySQL.

Background information

ApsaraDB RDS for MySQL is fully compatible with open source MySQL. The process of migrating data from a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance is similar to the process of migrating data from one MySQL server to another MySQL server.

> **Note**
> - The mysqldump-based migration process is complex. We recommend that you use Data Transmission Service (DTS) to migrate data. For more information, see Overview of data migration methods.
> - For more information about the parameters of the mysqldump plug-in, see the official MySQL documentation.

Scenario

You want to migrate data from a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance.

Precautions

After the migration is complete, the names of all tables that are migrated from the self-managed MySQL instance are in lowercase on the ApsaraDB RDS for MySQL instance. You can use the following method to configure the names of tables on the ApsaraDB RDS for MySQL instance to be case-sensitive:

After the ApsaraDB RDS for MySQL instance is created, set the lower_case_table_names parameter to 0 for the instance in the ApsaraDB RDS console. For more information, see Modify the parameters of an ApsaraDB RDS for MySQL instance.

> **Note**
> - After you set the lower_case_table_names parameter to 0, do not change the value of this parameter to 1. If you change the value of this parameter to 1, the "`ERROR 1146 (42S02): Table doesn't exist`" error occurs. This error has a serious impact on your business.
> - If the ApsaraDB RDS for MySQL instance runs MySQL 8.0, you cannot reconfigure the lower_case_table_names parameter for the instance.

Procedure

1. Use the mysqldump plug-in to export the data, stored procedures, triggers, and functions of the self-managed MySQL instance.

   > **Note** When the export task is in progress, do not update the data. Wait until the export task is completed.

i. In the Linux command-line interface (CLI), run the following command to export the data as a file:

```
mysqldump -h 127.0.0.1 -u root -p --opt --default-character-set=utf8 --hex-blob <The name of the self-managed MySQL instance> --skip-triggers --skip-lock-tables > /tmp/<The name of the self-managed MySQL instance>.sql
```

Example:

```
mysqldump -h 127.0.0.1 -u root -p --opt --default-character-set=utf8 --hex-blob testdb --skip-triggers --skip-lock-tables > /tmp/testdb.sql
```

ii. In the Linux CLI, run the following command to export the stored procedures, triggers, and functions as a file:

```
mysqldump -h 127.0.0.1 -u root -p --opt --default-character-set=utf8 --hex-blob <The name of the self-managed MySQL instance> -R | sed -e 's/DEFINER[ ]*=[ ]*[^*]*\*/\*/' > /tmp/<The name of the self-managed MySQL instance>Trigger.sql
```

Example:

```
mysqldump -h 127.0.0.1 -u root -p --opt --default-character-set=utf8 --hex-blob testdb -R | sed -e 's/DEFINER[ ]*=[ ]*[^*]*\*/\*/' > /tmp/testdbTrigger.sql
```

> **Note** If the self-managed MySQL instance does not contain stored procedures, triggers, or functions, you can skip this step.

2. Upload the exported files to a specified path on an Elastic Compute Service (ECS) instance. In this example, the path is */tmp*.

> **Note** If the self-managed MySQL instance resides on an ECS instance, you can skip this step.

3. Run the following commands to import the exported files into the ApsaraDB RDS for MySQL instance:

```
mysql -h <The endpoint that is used to connect to the ApsaraDB RDS for MySQL instance>
-P <The port number that is used to connect to the ApsaraDB RDS for MySQL instance> -u
<The username of the account that is used to log on to the ApsaraDB RDS for MySQL insta
nce> -p <The name of the destination database on the ApsaraDB RDS for MySQL instance> <
/tmp/<The name of the source database on the self-managed MySQL instance>.sql
mysql -h <The endpoint that is used to connect to the ApsaraDB RDS for MySQL instance>
-P <The port number that is used to connect to the ApsaraDB RDS for MySQL instance> -u
<The username of the account that is used to log on to the ApsaraDB RDS for MySQL insta
nce> -p <The name of the destination database on the ApsaraDB RDS for MySQL instance> <
/tmp/<The name of the source database on the self-managed MySQL instance>Trigger.sql
```

> **Note**
>
> - The destination database on the ApsaraDB RDS for MySQL instance must be an existing database that you created. For more information about how to create a database, see Create a database on an ApsaraDB RDS for MySQL instance.
> - The account that is used to log on to the ApsaraDB RDS for MySQL instance must be a privileged account or a standard account that has the read and write permissions.

Examples:

```
mysql -h rm-bpxxxxx.mysql.rds.aliyuncs.com -P 3306 -u testuser -p testdb  < /tmp/testdb
.sql
mysql -h rm-bpxxxxx.mysql.rds.aliyuncs.com -P 3306 -u testuser -p testdb  < /tmp/testdb
Trigger.sql
```

4. After the import is complete, log on to the ApsaraDB RDS for MySQL instance and check whether the data is normal. For more information, see Use DMS to log on to an ApsaraDB RDS for MySQL instance.

6.3. Migrate data from a third-party cloud database to ApsaraDB for RDS

6.3.1. Migrate a MySQL database from Google Cloud to Alibaba Cloud

This topic describes how to migrate a MySQL database from Google Cloud to Alibaba Cloud and the corresponding precautions.

Prerequisites

- You have created an Alibaba Cloud RDS MySQL instance.
- You have created an account with read/write privileges.

Limits

- Structure migration does not support migration of events.
- For MySQL databases, DTS reads floating-point values (FLOAT and DOUBLE data types) with `round(c`

`olumn,precision)`. If the column definition does not specify the precision, the precision is 38 for FLOAT values and 308 for DOUBLE values.

- If the object name mapping function is used for an object, migration of objects relying on the object may fail.
- For incremental migration, you must enable binlog for the source MySQL instance.
- For incremental migration, binlog_format of the source database must be set to ROW.

> **Note** You can modify parameters of Google Cloud databases by choosing **Instance details > Configuration > Edit configuration > Add database flags**.

- For incremental migration, if the source database version is MySQL 5.6 or later, binlog_row_image must be set to FULL.
- For incremental migration, if the source instance has binlog file ID disorder caused by cross-host migration, the incremental migration may have data loss.

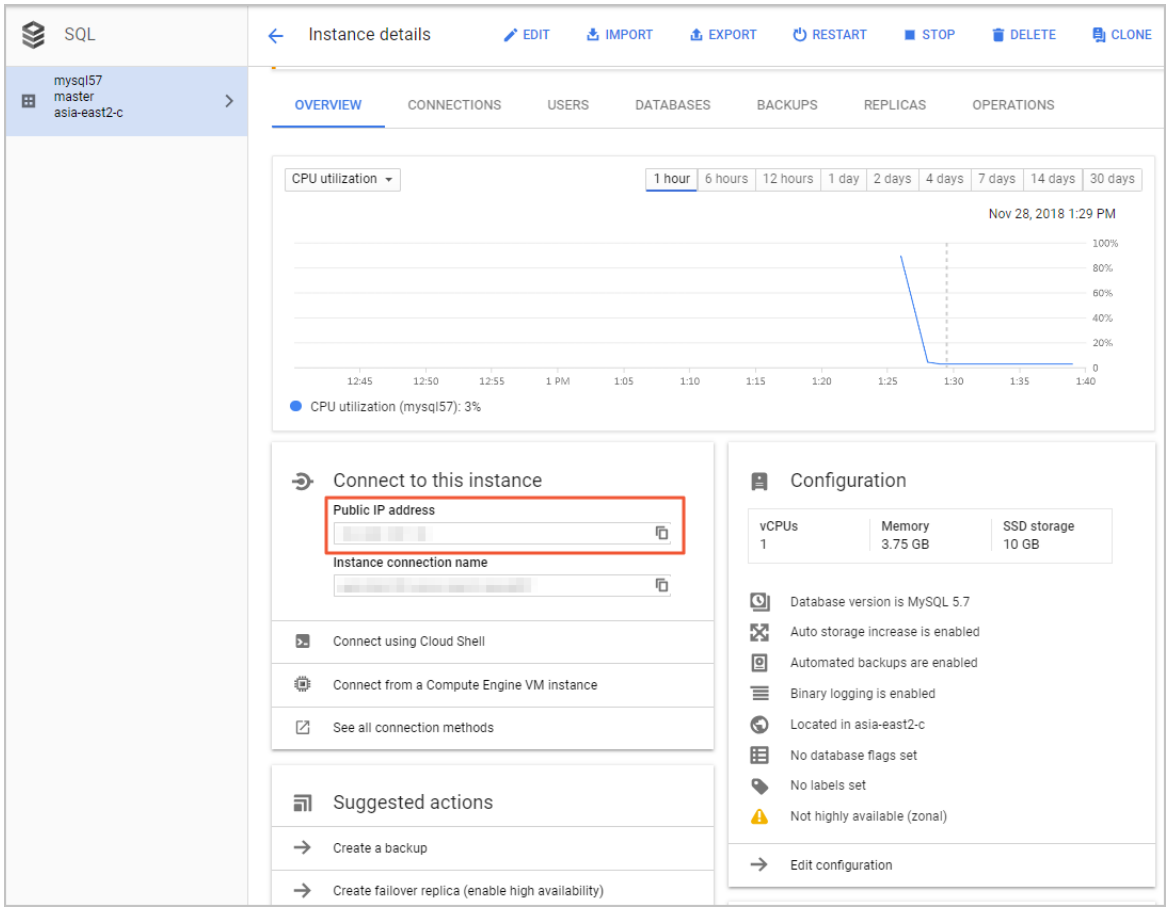
Precautions

DTS automatically attempts to recover abnormal tasks of the past seven days. This may cause the new data in the target instance to be overwritten by the source database data. Therefore, you must revoke the write permission of the DTS account that is used to access the target instance by running the `revoke` command.

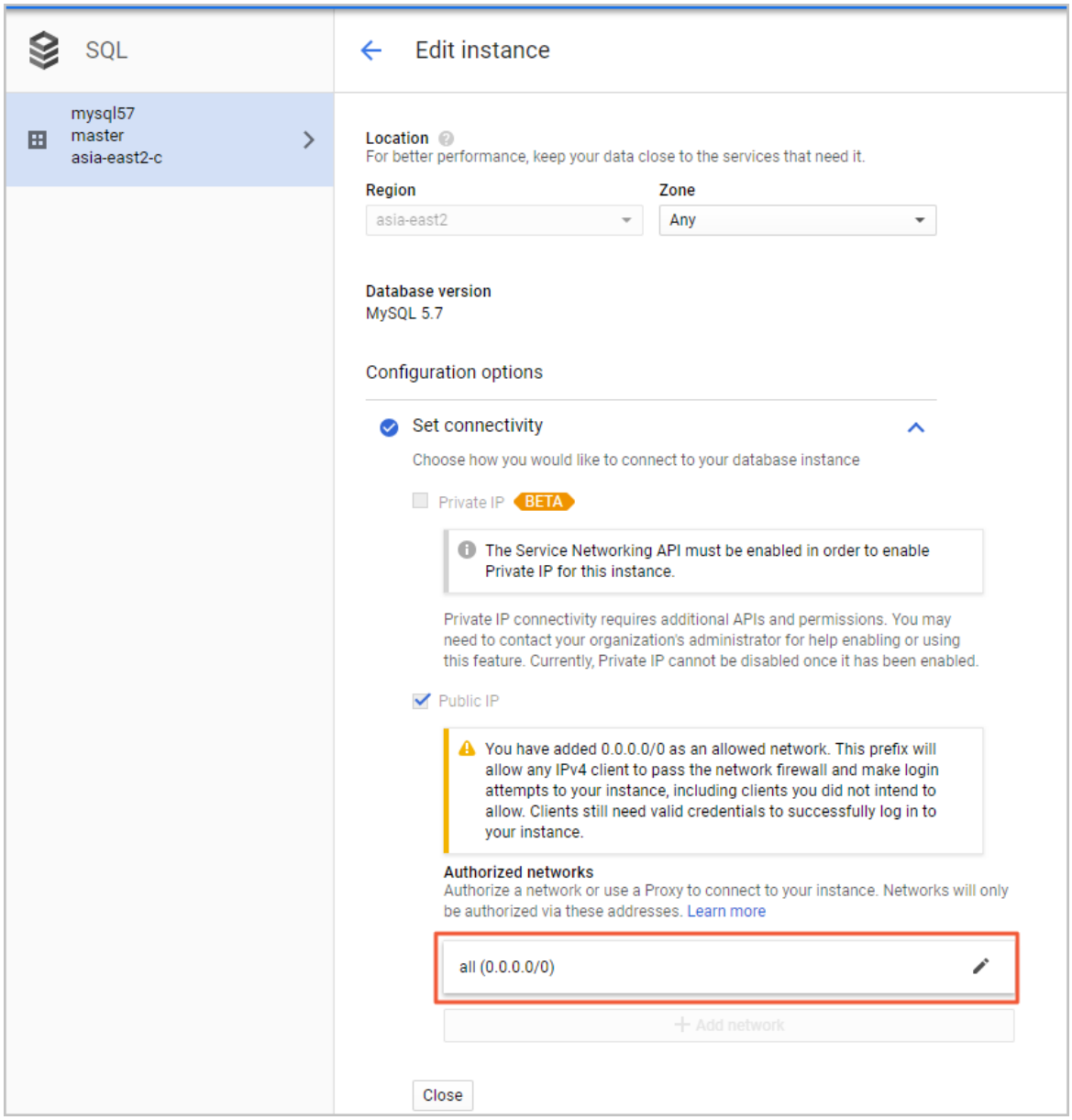
Procedure

1. Log on to your database instance on Google Cloud. On the **Instance details** page, view Public IP address.

   > **Note** If a public IP address is not enabled, perform related settings by going to **Configuration > Edit configuration > Set connectivity**.



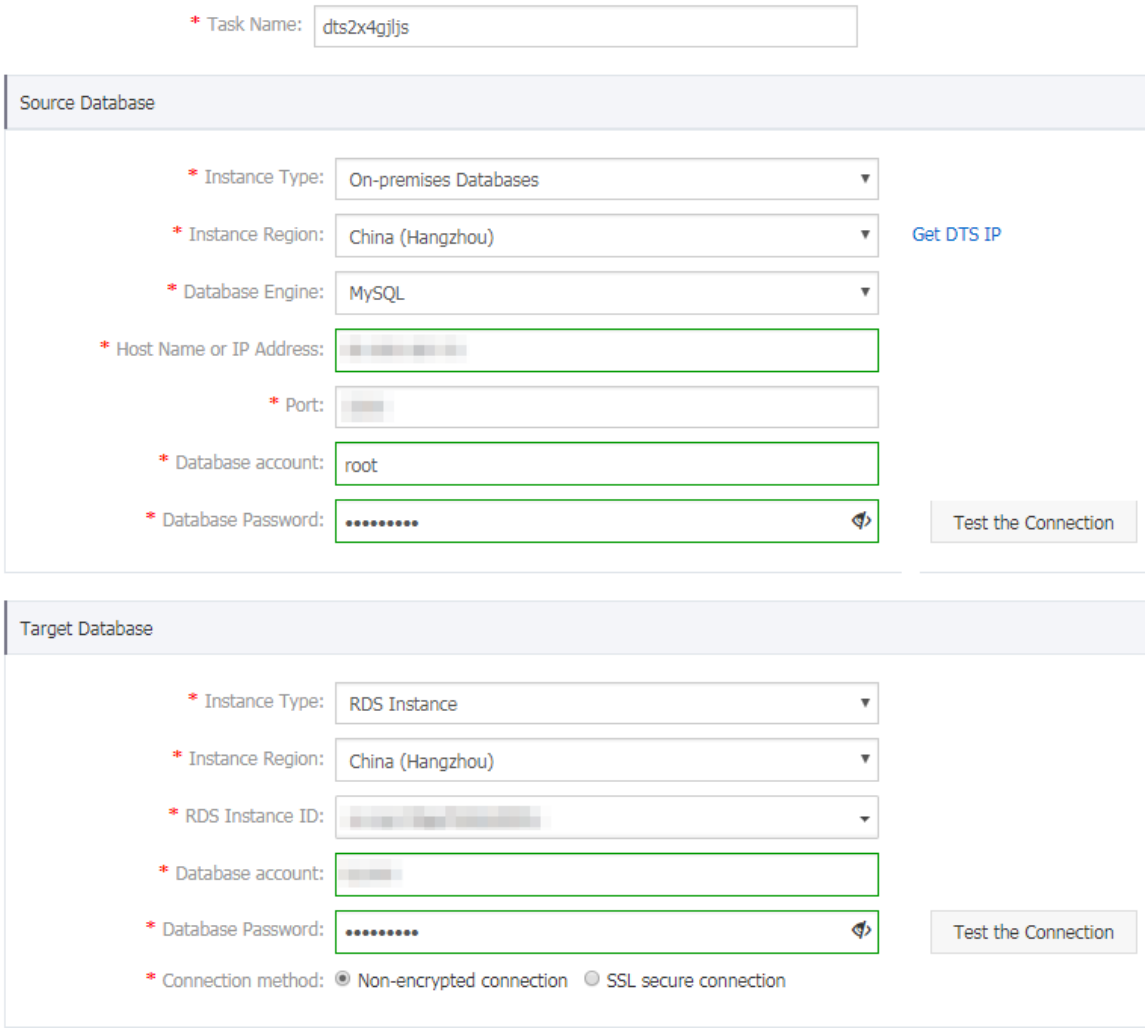
2. Choose **Configuration > Edit configuration > Set connectivity > Add network**, and then add the IP address of the region of the source database instance obtained from DTS.



3. Log on to the DTS console.
4. In the left-side navigation pane, click **Data Migration**. In the right pane, click **Create Migration Task** in the upper-right corner.
5. Enter information about the source and target databases. The following table describes the parameters.

Database type	Parameter	Description
	Instance Type	Type of the instance in the source database. Select *On-premises Databases*.

Database type	Parameter	Description
Source database (on Google Cloud)	Instance Region	If you have configured access control for your instance, you must allow the specified Internet IP segment of the region to access the instance before configuring a migration task. **Note** You can click **Get DTS IP** to view and copy the IP segment of the region.
	Database Engine	Source database type. Select *MySQL* .
	Host Name or IP Address	*Public IP address* of the database
	Port	Default port 3306
	Database account	Default superuser account *root*
	Database Password	Password of the root account
Target database (on Alibaba Cloud)	Instance Type	Type of the instance in the target database. Select *RDS Instance*.
	Instance Region	Region of the target instance
	RDS Instance ID	ID of the instance in the selected region. Select the ID of the target instance.
	Database account	An account with read and write permissions under the target instance
	Database Password	Account password
	Connection method	Select **Non-encrypted connection** or **SSL secure connection**. The latter greatly increases CPU consumption.

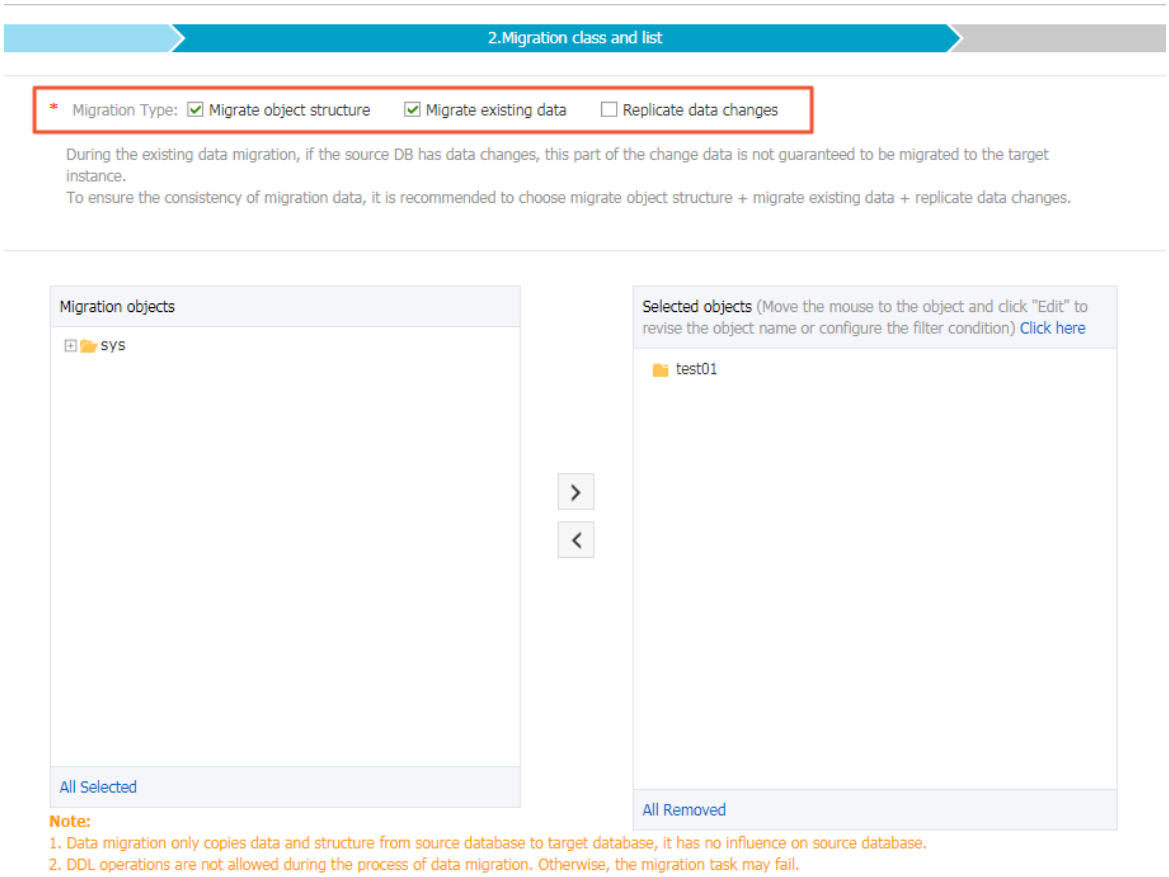


6. Click **Test the Connection** and confirm that the test results for both the source and target databases are *Test passed*.
7. Click **Authorize Whitelist and Enter into Next Step**.
8. Select the migration type. In the **Migration objects** area, select the target database and click

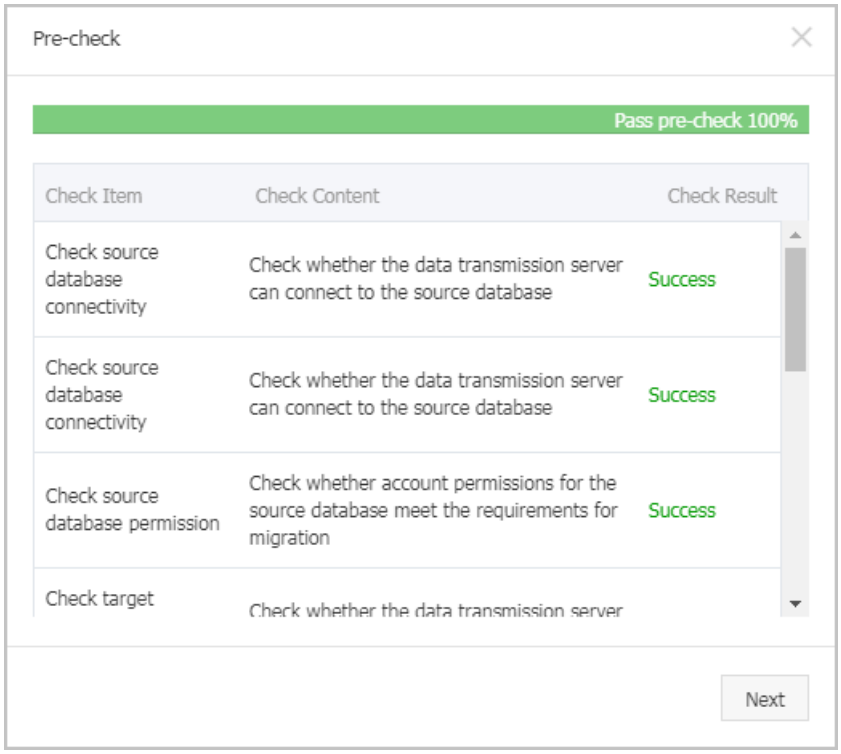
   to add the database to the **Selected objects** area.

> **Note** To maintain data consistency before and after migration, we recommend that you migrate the structure, full data, and incremental data.



9. Click **Pre-check** and wait until the pre-check ends.

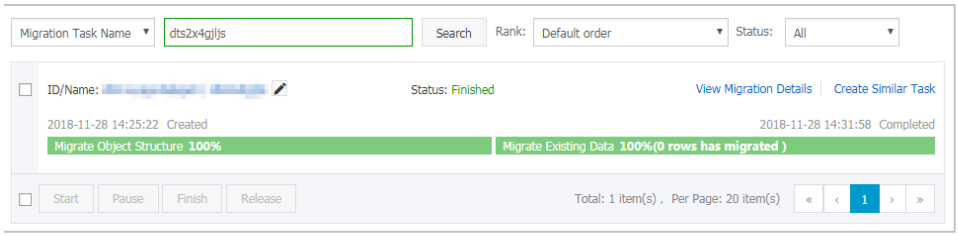
> **Note** If the check fails, you can rectify faults according to error items and restart the task.



10. Click **Next**. In the **Confirm Purchase Configuration** dialog box, read and select **Service Terms of Data Transmission (Pay-As-You-Go)**, then click **Buy and Start Now**.

> **Note** Currently, structure migration and full migration are free of charge, while incremental migration is charged by the hour according to link specifications.

11. Wait until the migration task is completed.



6.3.2. Migrate data from an Amazon RDS for MySQL instance to an ApsaraDB RDS for MySQL instance

This topic describes how to migrate data from an Amazon RDS for MySQL instance to an ApsaraDB RDS for MySQL instance by using Data Transmission Service (DTS). DTS supports schema migration, full data migration, and incremental data migration. When you configure a data migration task, you can select all of the supported migration types to ensure service continuity.

Prerequisites

- The **Public accessibility** option of the Amazon RDS for MySQL instance is set to **Yes**. The setting ensures that DTS can access the Amazon RDS for MySQL instance over the Internet.
- An ApsaraDB RDS for MySQL instance is created. For more information, see Create an ApsaraDB RDS for MySQL instance.
- The available storage space of the ApsaraDB RDS for MySQL instance is larger than the total size of the data in the Amazon RDS for MySQL instance.

Precautions

- DTS uses read and write resources of the source and destination databases during full data migration. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you migrate data, evaluate the impact of data migration on the performance of the source and destination databases. We recommend that you migrate data during off-peak hours. For example, you can migrate data when the CPU utilization of the source and destination databases is less than 30%.
- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- DTS uses the `ROUND(COLUMN,PRECISION)` function to retrieve values from columns of the FLOAT or DOUBLE data type. If you do not specify a precision, DTS sets the precision for the FLOAT data type to 38 digits and the precision for the DOUBLE data type to 308 digits. You must check whether the precision settings meet your business requirements.
- DTS automatically creates a destination database in the ApsaraDB RDS for MySQL instance. However, if the name of the source database is invalid, you must manually create a database in the ApsaraDB RDS for MySQL instance before you configure the data migration task.

  > **Note** For more information about the naming conventions of ApsaraDB RDS for MySQL databases and how to create a database, see Create a database on an ApsaraDB RDS for MySQL instance.

- If a data migration task fails, DTS automatically resumes the task. Before you switch your workloads to the destination instance, stop or release the data migration task. Otherwise, the data in the source instance will overwrite the data in the destination instance after the task is resumed.

Billing

Migration type	Task configuration fee	Internet traffic fee
Schema migration and full data migration	Free of charge.	Charged only when data is migrated from Alibaba Cloud over the Internet. For more information, see Pricing.
Incremental data migration	Charged. For more information, see Pricing.	

Migration types

- Schema migration

DTS migrates the schemas of the required objects to the destination instance. DTS supports schema migration for the following types of objects: table, view, trigger, stored procedure, and function. DTS does not support schema migration for events.

> Note
> - During schema migration, DTS changes the value of the SECURITY attribute from DEFINER to INVOKER for views, stored procedures, and functions.
> - DTS does not migrate user information. To call a view, stored procedure, or function of the destination database, you must grant the read and write permissions to INVOKER.

- Full data migration

  DTS migrates historical data of the required objects from the Amazon RDS for MySQL instance to the ApsaraDB RDS for MySQL instance.

  > Note
  > - During full data migration, concurrent INSERT operations cause fragmentation in the tables of the destination instance. After full data migration is complete, the tablespace of the destination instance is larger than that of the source instance.
  > - During schema migration and full data migration, do not perform data definition language (DDL) operations in the source database, for example, add a field. Otherwise, data migration may fail.

- Incremental data migration

  After full data migration is complete, DTS retrieves binary log files from the Amazon RDS for MySQL instance. Then, DTS synchronizes incremental data from the Amazon RDS for MySQL instance to the ApsaraDB RDS for MySQL instance. Incremental data migration allows you to ensure service continuity when you migrate data between MySQL databases.

Permissions required for database accounts

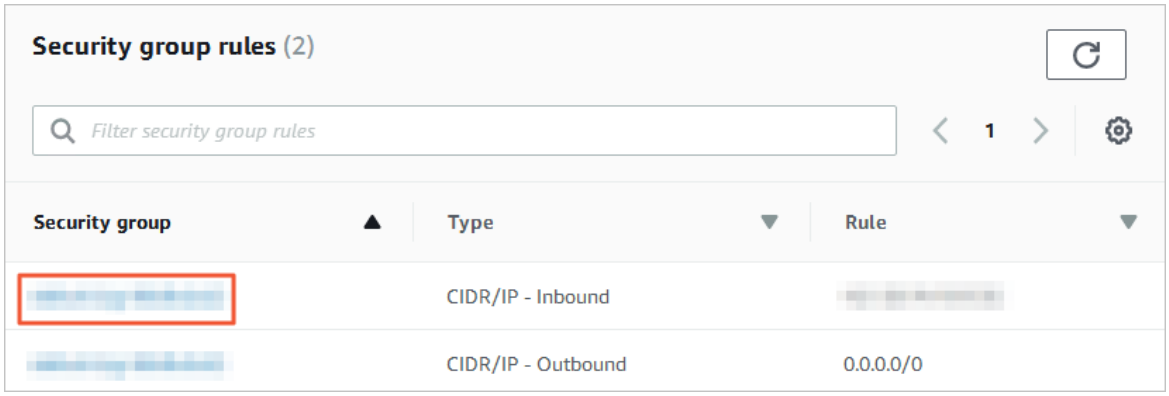
Database	Schema migration	Full data migration	Incremental data migration
Amazon RDS for MySQL	The SELECT permission	The SELECT permission	The REPLICATION CLIENT, REPLICATION SLAVE, SHOW VIEW, and SELECT permissions
ApsaraDB RDS for MySQL	The read and write permissions	The read and write permissions	The read and write permissions

For more information about how to create and authorize a database account, see the following topics:

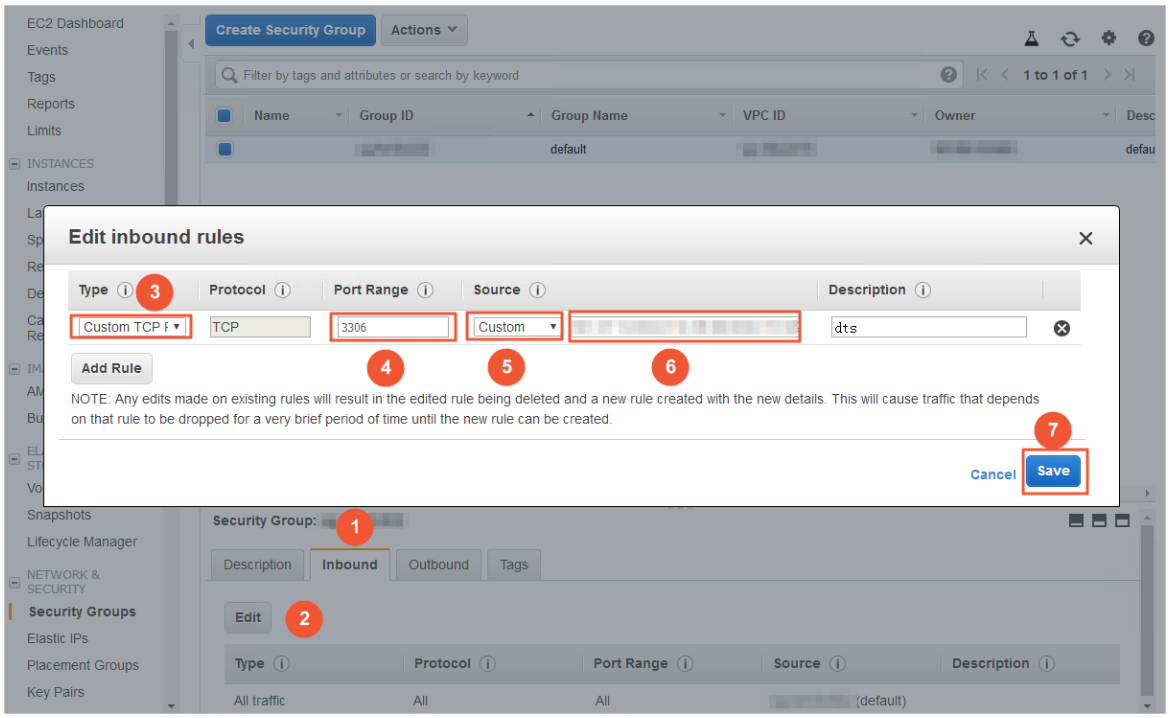
- Amazon RDS for MySQL instance: Create an account for a user-created MySQL database and configure binary logging
- ApsaraDB RDS for MySQL instance: Create an account on an ApsaraDB RDS for MySQL instance and Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance.

Before you begin

1. Log on to the Amazon RDS Management Console.
2. Go to the **Basic Information** page of the Amazon RDS for MySQL instance.
3. In the **Security group rules** section, click the name of the security group corresponding to the existing inbound rule.



4. On the **Security Groups** page, click the Inbound tab in the Security Group section. On the Inbound tab, click Edit to add the CIDR blocks of DTS servers in the corresponding region to the inbound rule. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases.



> **Note**
>
> - You need to add only the CIDR blocks of DTS servers that reside in the same region as the destination database. For example, the source database resides in the Singapore (Singapore) region and the destination database resides in the China (Hangzhou) region. You need to add only the CIDR blocks of DTS servers that reside in the China (Hangzhou) region.
> - You can add all of the required CIDR blocks to the inbound rule at a time.

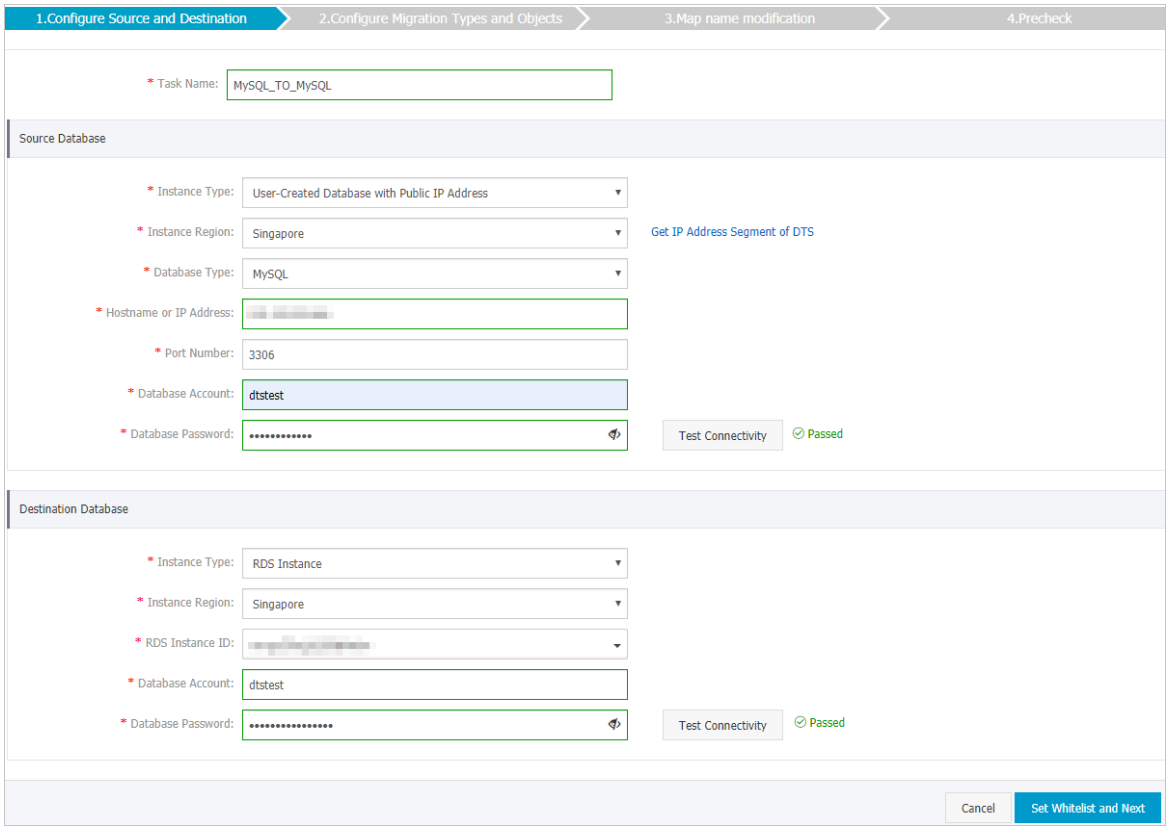
5. Log on to the Amazon RDS for MySQL database and specify the number of hours to retain binary log files. Skip this step if you do not need to perform incremental data migration.

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

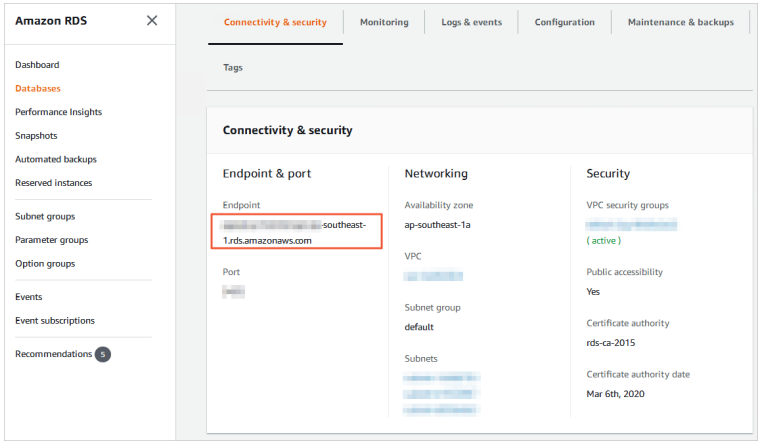
> **Note**
>
> - The preceding command sets the retention period of binary log files to 24 hours. The maximum value is 168 hours (7 days).
> - The binary logging feature of the Amazon RDS for MySQL instance must be enabled and the value of the binlog_format parameter must be set to row. If the MySQL version is 5.6 or later, the value of the binlog_row_image parameter must be set to full.

Procedure

1. Log on to the DTS console.
2. In the left-side navigation pane, click **Data Migration**.
3. At the top of the **Migration Tasks** page, select the region where the destination cluster resides.
4. In the upper-right corner of the page, click **Create Migration Task**.
5. Configure the source and destination databases.



Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
	Instance Type	Select **User-Created Database with Public IP Address**.
	Instance Region	If the instance type is set to **User-Created Database with Public IP Address**, you do not need to specify the **instance region**.
	Database Type	Select **MySQL**.

Section	Parameter	Description
Source Database	Hostname or IP Address	Enter the endpoint that is used to access the Amazon RDS for MySQL instance. **Note** You can obtain the endpoint on the **Basic Information** page of the Amazon RDS for MySQL instance. 
	Port Number	Enter the service port number of the Amazon RDS for MySQL instance. The default port number is **3306**.
	Database Account	Enter the database account of the Amazon RDS for MySQL instance. For information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the source database parameters, click **Test Connectivity** next to **Database Password** to verify whether the specified parameters are valid. If the specified parameters are valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Modify the source database parameters based on the check results.
	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the ApsaraDB RDS for MySQL instance resides.
	RDS Instance ID	Select the ID of the ApsaraDB RDS for MySQL instance.
	Database Account	Enter the database account of the ApsaraDB RDS for MySQL instance. For information about the permissions that are required for the account, see Permissions required for database accounts.

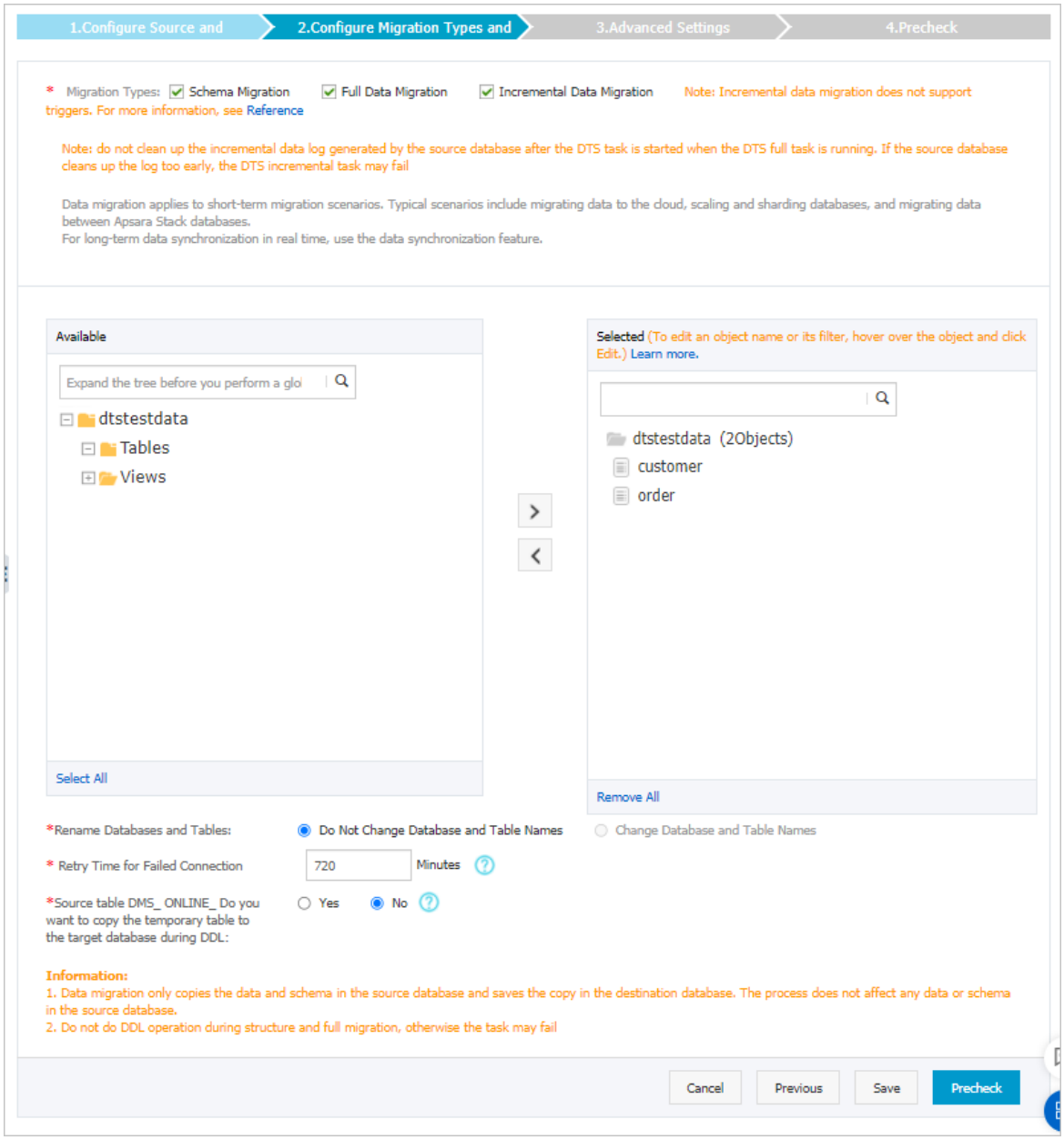
Section	Parameter	Description
Destination Database	Database Password	Enter the password of the database account. **Note** After you specify the destination database parameters, click **Test Connectivity** next to **Database Password** to verify whether the specified parameters are valid. If the specified parameters are valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Modify the destination database parameters based on the check results.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data migration task. For more information, see Configure SSL encryption on an ApsaraDB RDS for MySQL instance **Note** The **Encryption** parameter is available only for regions in mainland China and the China (Hong Kong) region.

6. In the lower-right corner of the page, click **Set Whitelist and Next**.

Warning

- If the source or destination database instance is an Alibaba Cloud database instance, such as an ApsaraDB RDS for MySQL or ApsaraDB for MongoDB instance, or is a self-managed database hosted on Elastic Compute Service (ECS), DTS automatically adds the CIDR blocks of DTS servers to the whitelist of the database instance or ECS security group rules. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases. If the source or destination database is a self-managed database on data centers or is from other cloud service providers, you must manually add the CIDR blocks of DTS servers to allow DTS to access the database.
- If the CIDR blocks of DTS servers are automatically or manually added to the whitelist of the database instance or ECS security group rules, security risks may arise. Therefore, before you use DTS to migrate data, you must understand and acknowledge the potential risks and take preventive measures, including but not limited to the following measures: enhance the security of your account and password, limit the ports that are exposed, authenticate API calls, regularly check the whitelist or ECS security group rules and forbid unauthorized CIDR blocks, or connect the database to DTS by using Express Connect, VPN Gateway, or Smart Access Gateway.
- After the DTS task is completed or released, we recommend that you manually detect and remove the added CIDR blocks from the whitelist of the database instance or ECS security group rules.

7. Select the migration types and the objects to be migrated.



Setting	Description

Setting	Description
Select the migration types	◦ To perform only full data migration, select **Schema Migration** and **Full Data Migration**. ◦ To ensure service continuity during data migration, select **Schema Migration**, **Full Data Migration**, and **Incremental Data Migration**. **Note** ◦ If **Incremental Data Migration** is not selected, we recommend that you do not write data to the source database during data migration. This ensures data consistency between the source and destination databases. ◦ During schema migration and full data migration, we recommend that you do not perform DDL operations on the required objects. Otherwise, the objects may fail to be migrated.
Specify whether to rename objects	You can use the object name mapping feature to rename the objects that are migrated to the destination instance. For more information, see Object name mapping.
Specify the retry time for failed connections to the source or destination database	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 12 hours. You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data migration task. Otherwise, the data migration task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

Setting	Description
Specify whether to copy temporary tables to the destination database when DMS performs online DDL operations on the source table	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to migrate temporary tables generated by online DDL operations. ○ **Yes**: DTS migrates the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data migration task may be delayed. ○ **No**: DTS does not migrate the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is migrated. **Note** If you select No, the tables in the destination database may be locked.

8. In the lower-right corner of the page, click **Precheck**.

   Note
   - Before you can start the data migration task, a precheck is performed. You can start the data migration task only after the task passes the precheck.
   - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
     - You can troubleshoot the issues based on the causes and run a precheck again.
     - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

9. After the task passes the precheck, click **Next**.
10. In the **Confirm Settings** dialog box, specify the **Channel Specification** parameter and select **Data Transmission Service (Pay-As-You-Go) Service Terms**.
11. Click **Buy and Start** to start the data migration task.
    - Schema migration and full data migration

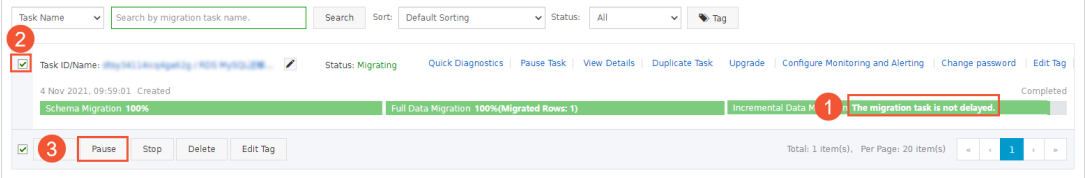
      We recommend that you do not manually stop the task during full data migration. Otherwise, the data migrated to the destination database will be incomplete. You can wait until the data migration task automatically stops.
    - Schema migration, full data migration, and incremental data migration

      The task does not automatically stop during incremental data migration. You must manually stop the task.

> **Notice** We recommend that you select an appropriate time to manually stop the data migration task. For example, you can stop the task during off-peak hours or before you switch your workloads to the destination cluster.

a. Wait until **Incremental Data Migration** and **The migration task is not delayed** appear in the progress bar of the migration task. Then, stop writing data to the source database for a few minutes. The delay time of **incremental data migration** may be displayed in the progress bar.

b. Wait until the status of **incremental data migration** changes to **The migration task is not delayed** again. Then, manually stop the migration task.



12. Switch your workloads to the ApsaraDB RDS for MySQL instance.

6.4. Migrate data between ApsaraDB RDS for MySQL instances

This topic describes how to migrate data between ApsaraDB RDS for MySQL instances by using Data Transmission Service (DTS). DTS supports schema migration, full data migration, and incremental data migration. When you configure a migration task, you can select all of these supported migration types. This allows you to migrate data without causing service interruptions to your application.

Prerequisites

The database types of the RDS instances meet the following requirements.

Source database	Destination database
ApsaraDB RDS for MySQL ApsaraDB RDS for MariaDB TX	ApsaraDB RDS for MySQL ApsaraDB RDS for MariaDB TX
ApsaraDB RDS for SQL Server	ApsaraDB RDS for SQL Server
ApsaraDB RDS for PostgreSQL	ApsaraDB RDS for PostgreSQL

Precautions

- Data migration does not affect the data of the source database. During data migration, DTS reads the data of the source database and copies the data to the destination database. DTS does not delete the data of the source database. For more information, see Design concept of data migration.
- DTS uses read and write resources of the source and destination databases during full data migration. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following

cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you migrate data, evaluate the impact of data migration on the performance of the source and destination databases. We recommend that you migrate data during off-peak hours. For example, you can migrate data when the CPU utilization of the source and destination databases is less than 30%.

- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- To ensure data consistency, we recommend that you do not write data to the source RDS instance during full data migration.
- If a data migration task fails, DTS automatically resumes the task. Before you switch your workloads to the destination instance, stop or release the data migration task. Otherwise, the data in the source instance will overwrite the data in the destination instance after the task is resumed.
- DTS automatically creates a database in the destination RDS instance. However, if the name of the source database is invalid, you must manually create a database in the destination RDS instance before you configure the data migration task.

  > **Note** For more information about the naming conventions of ApsaraDB RDS and how to create a database, see Create databases and accounts for an ApsaraDB RDS for MySQL instance.

- If you migrate data between ApsaraDB RDS for PostgreSQL instances, take note of the following limits: After your workloads are switched to the destination database, newly written sequences do not increment from the maximum value of the sequences in the source database. Therefore, you must query the maximum value of the sequences in the source database before you switch your workloads to the destination database. Then, you must specify the queried maximum value as the starting value of the sequences in the destination database. You can execute the following statements to query the maximum value of the sequences in the source database:

```
do language plpgsql $$
declare
  nsp name;
  rel name;
  val int8;
begin
  for nsp,rel in select nspname,relname from pg_class t2 , pg_namespace t3 where t2.relna
mespace=t3.oid and t2.relkind='S'
  loop
    execute format($_$select last_value from %I.%I$_$, nsp, rel) into val;
    raise notice '%',
    format($_$select setval('%I.%I'::regclass, %s);$_$, nsp, rel, val+1);
  end loop;
end;
$$;
```

Billing

Migration type	Task configuration fee	Internet traffic fee
Schema migration and full data migration	Free of charge.	Charged only when data is migrated from Alibaba Cloud over the Internet. For more

Migration type	Task configuration fee	Alibaba Cloud over the Internet. For more information, see Pricing.
Incremental data migration	Charged. For more information, see Pricing.	

Migration types

- Schema migration

  DTS migrates the schemas of the required objects from the source RDS instance to the destination RDS instance.

- Full data migration

  DTS migrates historical data of the required objects from the source RDS instance to the destination RDS instance.

- Incremental data migration

  After full data migration is completed, DTS synchronizes incremental data from the source RDS instance to the destination RDS instance. Incremental data migration allows you to ensure service continuity when you migrate data between RDS instances.

SQL operations that can be synchronized during incremental data migration

Scenario	Operation type	SQL statement
• Migrate data between ApsaraDB RDS for MySQL instances • Migrate data between ApsaraDB RDS for MariaDB TX instances • Migrate data from an ApsaraDB RDS for MariaDB TX instance to an ApsaraDB RDS for MySQL instance	DML	INSERT, UPDATE, DELETE, and REPLACE
	DDL	• ALTER TABLE and ALTER VIEW • CREATE FUNCTION, CREATE INDEX, CREATE PROCEDURE, CREATE TABLE, and CREATE VIEW • DROP INDEX and DROP TABLE • RENAME TABLE • TRUNCATE TABLE
	DML	INSERT, UPDATE, and DELETE **Note** If an UPDATE operation updates only the large fields, DTS does not synchronize the operation.

Scenario	Operation type	SQL statement
Migrate data between ApsaraDB RDS for SQL Server instances	DDL	• ALTER TABLE, including only ADD COLUMN, DROP COLUMN, and RENAME COLUMN • CREATE TABLE and CREATE INDEX Note If a CREATE TABLE operation creates a partitioned table or a table that contains functions, DTS does not synchronize the operation. • DROP TABLE • RENAME TABLE • TRUNCATE TABLE
Migrate data between ApsaraDB RDS for PostgreSQL instances Migrate data between ApsaraDB RDS for PPAS instances	DML	INSERT, UPDATE, and DELETE

Permissions required for database accounts

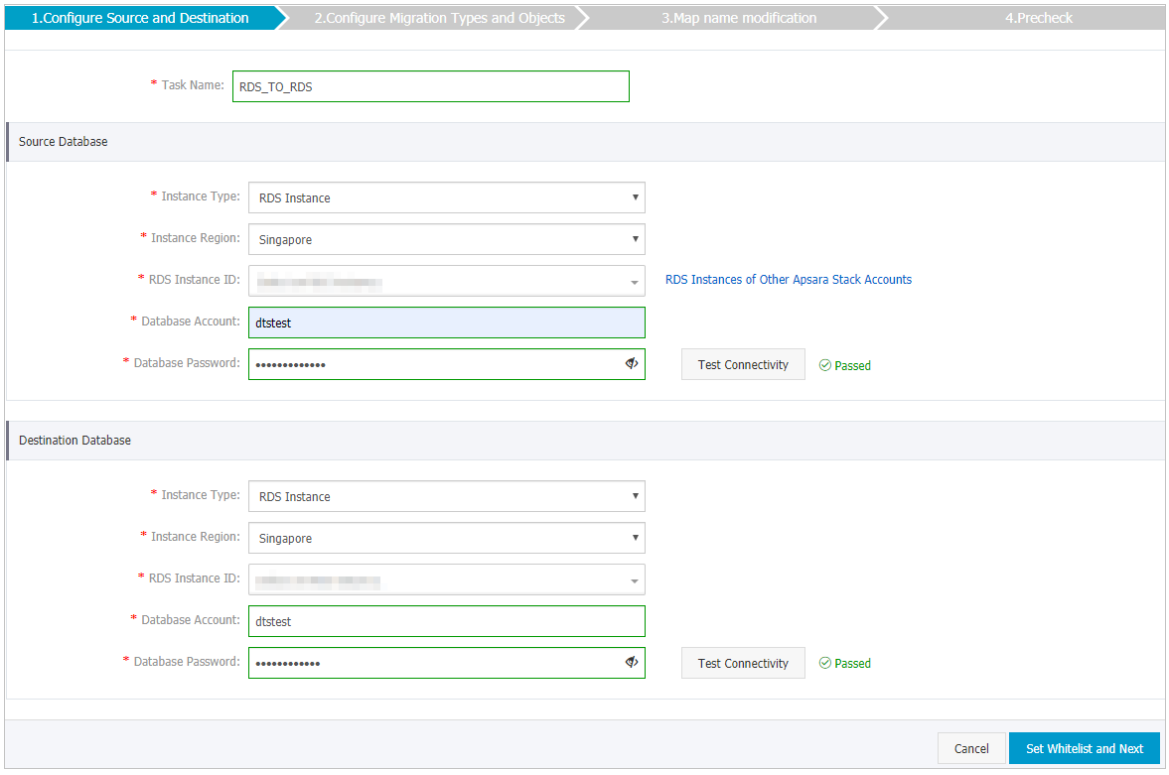
Scenario	Database	Schema migration	Full data migration	Incremental data migration
• Migrate data between ApsaraDB RDS for MySQL instances • Migrate data between ApsaraDB RDS for MariaDB TX instances • Migrate data from an ApsaraDB RDS for MariaDB TX instance to an ApsaraDB RDS for MySQL instance	Source instance	The SELECT permission	The SELECT permission	The REPLICATION SLAVE, REPLICATION CLIENT, SHOW VIEW, and SELECT permissions
	Destination instance	The read and write permissions	The read and write permissions	The read and write permissions

Scenario	Database	Schema migration	Full data migration	Incremental data migration
Migrate data between ApsaraDB RDS for SQL Server instances	Source instance	The SELECT permission	The SELECT permission	The owner permission on the objects to be migrated **Note** A privileged account has the required permissions.
	Destination instance	The read and write permissions	The read and write permissions	The read and write permissions
Migrate data between ApsaraDB RDS for PostgreSQL instances	Source instance	The USAGE permission on pg_catalog	The SELECT permission on the objects to be migrated	rds_superuser **Note** • A standard account of an ApsaraDB RDS for PostgreSQL instance has the required permissions. • If you receive a message indicating that the database account does not have the permissions of the superuser role, you must upgrade the kernel version of the RDS instance.

Scenario	Database	Schema migration	Full data migration	Incremental data migration
	Destination instance	The CREATE and USAGE permissions on the objects to be migrated	The permissions of the database owner, including the permissions to perform the INSERT, UPDATE, and DELETE operations **Note** A standard account of an ApsaraDB RDS for PostgreSQL instance has the required permissions.	The permissions of the database owner, including the permissions to perform the INSERT, UPDATE, and DELETE operations **Note** A standard account of an ApsaraDB RDS for PostgreSQL instance has the required permissions.

Procedure

1. Log on to the DTS console.
2. In the left-side navigation pane, click **Data Migration**.
3. In the upper part of the **Migration Tasks** page, select the region where the RDS instance resides.
4. In the upper-right corner of the page, click **Create Migration Task**.
5. Configure the source and destination databases.



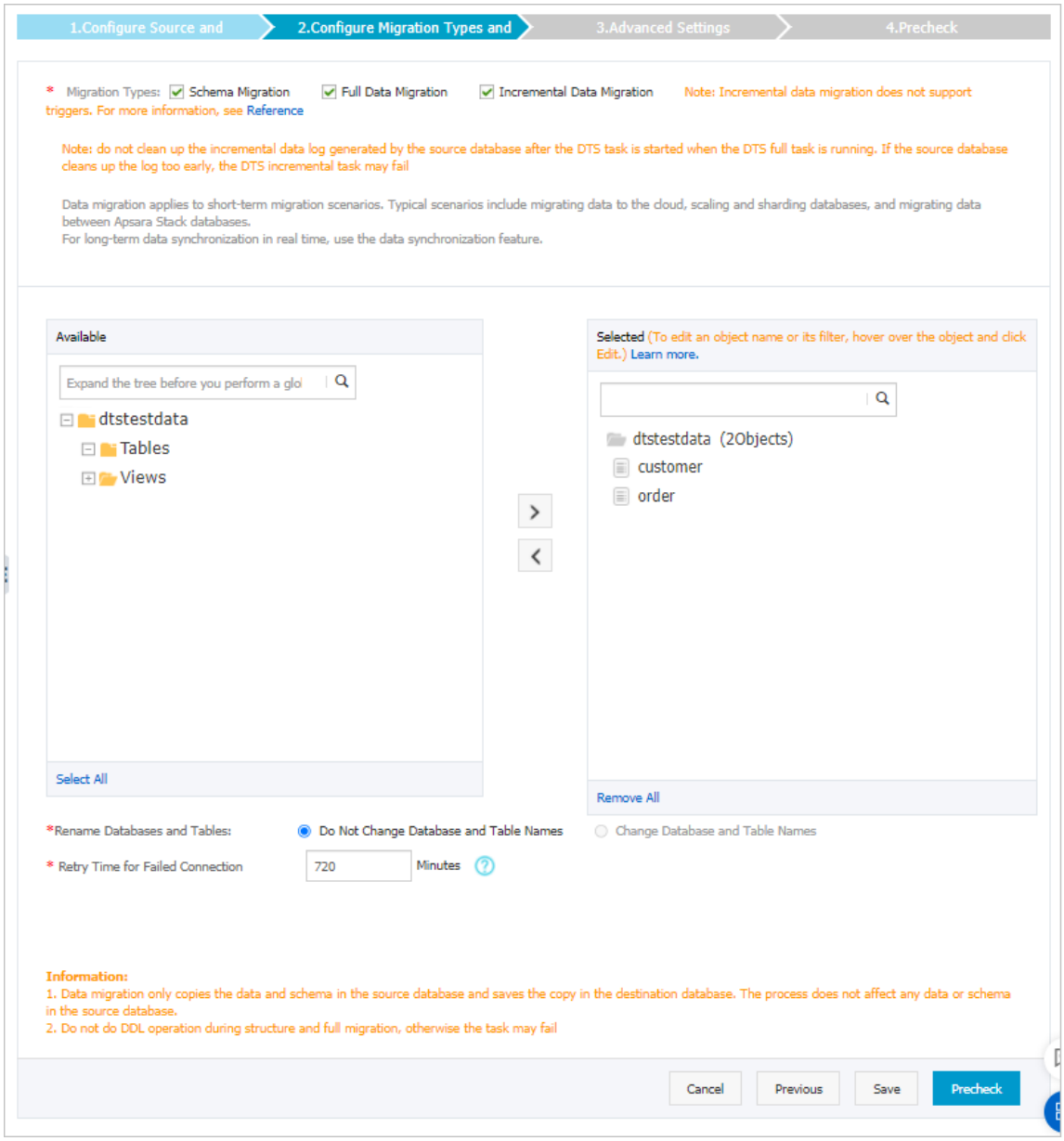
Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify an informative name to identify the task. You do not need to specify a unique task name.
Source Database	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the source RDS instance resides.
	RDS Instance ID	Select the ID of the source RDS instance. **Note** The source and destination RDS instances can be the same or different. You can use DTS to migrate data within an RDS instance or between two RDS instances.
	Database Name	Enter the name of the source database in the ApsaraDB RDS for PostgreSQL instance. **Note** This parameter is required only if the database engine of the RDS instance is **PostgreSQL**.
	Database Account	Enter the database account of the source RDS instance. For information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the information about the self-managed Oracle database, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data migration task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Note** This parameter is required only if the database engine of the RDS instance is **MySQL**. The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

Section	Parameter	Description
Destination Database	Instance Type	Select **RDS Instance**.
	Instance Region	Select the region where the destination RDS instance resides.
	RDS Instance ID	Select the ID of the destination RDS instance. **Note** The source and destination RDS instances can be the same or different. You can use DTS to migrate data within an RDS instance or between two RDS instances.
	The name of the database.	Enter the name of the destination database in the ApsaraDB RDS for PostgreSQL instance. The name of the destination database can be different from the name of the source database. **Note** This parameter is required only if the database engine of the RDS instance is **PostgreSQL**.
	Database Account	Enter the account that is used to connect to the RDS instance. For more information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account. **Note** After you specify the information about the RDS instance, you can click **Test Connectivity** next to **Database Password** to check whether the information is valid. If the information is valid, the **Passed** message appears. If the **Failed** message appears, click **Check** next to **Failed**. Then, modify the information based on the check results.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data migration task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Note** This parameter is required only if the database engine of the RDS instance is **MySQL**. The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

6. In the lower-right corner of the page, click **Set Whitelist and Next**.

> **Warning**
>
> - If the source or destination database instance is an Alibaba Cloud database instance, such as an ApsaraDB RDS for MySQL or ApsaraDB for MongoDB instance, or is a self-managed database hosted on Elastic Compute Service (ECS), DTS automatically adds the CIDR blocks of DTS servers to the whitelist of the database instance or ECS security group rules. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases. If the source or destination database is a self-managed database on data centers or is from other cloud service providers, you must manually add the CIDR blocks of DTS servers to allow DTS to access the database.
> - If the CIDR blocks of DTS servers are automatically or manually added to the whitelist of the database instance or ECS security group rules, security risks may arise. Therefore, before you use DTS to migrate data, you must understand and acknowledge the potential risks and take preventive measures, including but not limited to the following measures: enhance the security of your account and password, limit the ports that are exposed, authenticate API calls, regularly check the whitelist or ECS security group rules and forbid unauthorized CIDR blocks, or connect the database to DTS by using Express Connect, VPN Gateway, or Smart Access Gateway.
> - After the DTS task is completed or released, we recommend that you manually detect and remove the added CIDR blocks from the whitelist of the database instance or ECS security group rules.

7. Select the migration types and the objects to be migrated.



Setting	Description
Select the migration types	Select the migration types based on your business requirements. The migration types must be supported by the database engine. ○ To perform only full data migration, select **Schema Migration** and **Full Data Migration**. ○ To ensure service continuity during data migration, select **Schema Migration**, **Full Data Migration**, and **Incremental Data Migration**. **Note** If **Incremental Data Migration** is not selected, we recommend that you do not write data to the source RDS instance during data migration. This ensures data consistency between the source and destination instances.

Setting	Description
Select the objects to be migrated	Select one or more objects from the **Available** section and click the icon to move the objects to the **Selected** section. **Note** • You can select columns, tables, or databases as the objects to be migrated. If you select tables or columns as the objects to be migrated, DTS does not migrate other objects such as views, triggers, and stored procedures to the destination database. • By default, after an object is migrated to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are migrated to the destination database. For more information, see Object name mapping. • If you use the object name mapping feature to rename an object, other objects that are dependent on the object may fail to be migrated.
Specify whether to rename objects	You can use the object name mapping feature to rename the objects that are migrated to the destination instance. For more information, see Object name mapping.
Specify the retry time range for failed connections to the source or destination database	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time range based on your business requirements. If DTS reconnects to the source and destination databases within the specified time range, DTS resumes the data migration task. Otherwise, the data migration task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time range based on your business requirements. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

8. Click **Precheck**.

> **Note**
>
> - A precheck is performed before the migration task starts. The migration task only starts after the precheck succeeds.
> - If the precheck fails, click the
>
> icon next to each failed check item to view the related details. Fix the issues as instructed and run the precheck again.

9. After the data migration task passes the precheck, click **Next**.
10. In the **Confirm Settings** dialog box, configure the **Channel Specification** parameter. Then, read and select **Data Transmission Service (Pay-as-you-go) Service Terms**.
11. Click **Buy and Start** to start the data migration task.
    - Full data migration

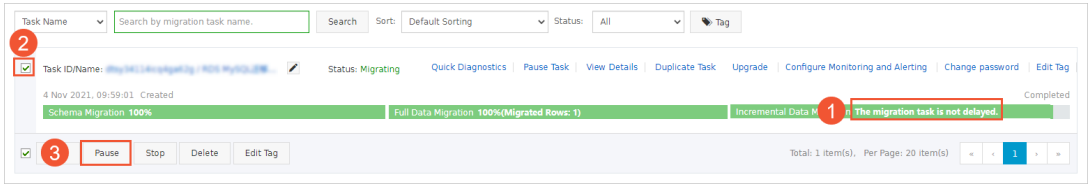
      Do not manually stop a full data migration task. If you manually stop a full data migration task, the data that is migrated to the RDS instance may be incomplete. You can wait until the full data migration task automatically stops.
    - Incremental data migration

      An incremental data migration task does not automatically stop. You must manually stop the task.

      > **Note** We recommend that you manually stop an incremental data migration task at an appropriate point in time. For example, you can stop the task during off-peak hours or before you switch your workloads over to the RDS instance.

      a. Wait until **Incremental Data Migration** and **The data migration task is not delayed** appear in the progress bar of the data migration task. Then, stop writing data to the self-managed Oracle database for a few minutes. The delay time of **incremental data migration** may be displayed in the progress bar.

      b. Wait until the status of **incremental data migration** changes to **The data migration task is not delayed** again. Then, manually stop the migration task.



7.Data synchronization

7.1. Overview of data synchronization

This topic provides an overview of the data synchronization solutions supported by ApsaraDB RDS for MySQL.

Scenario	Reference
Synchronize data between ApsaraDB RDS for MySQL instances	• Configure two-way data synchronization between MySQL instances • Configure one-way data synchronization between ApsaraDB RDS for MySQL instances • Synchronize data from a self-managed MySQL database hosted on ECS to an ApsaraDB RDS for MySQL instance • Synchronize data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL instance • Synchronize data between ApsaraDB RDS for MySQL instances that belong to different Alibaba Cloud accounts
Synchronize data from an ApsaraDB RDS for MySQL instance to other types of databases	Synchronize data from an ApsaraDB RDS for MySQL instance to a MaxCompute project

7.2. Synchronize data from MySQL to MySQL

7.2.1. Configure one-way data synchronization between ApsaraDB RDS for MySQL instances

Data Transmission Service (DTS) supports data synchronization between two MySQL databases. This topic describes how to configure one-way data synchronization between two ApsaraDB RDS for MySQL instances.

Prerequisites

- The source and destination RDS instances are created. For more information, see Create an ApsaraDB RDS for MySQL instance.
- The database type of the source and destination RDS instances is MySQL.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become

unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.

- We recommend that you do not use gh-ost or pt-online-schema-change to perform DDL operations on objects during data synchronization. Otherwise, data synchronization may fail.
- If you use only DTS to write data to the destination database, you can use Data Management (DMS) to perform online DDL operations during data synchronization. For more information, see Change schemas without locking tables.

  > **Warning** If you use tools other than DTS to write data to the destination database, we recommend that you do not use DMS to perform online DDL operations. Otherwise, data loss may occur in the destination database.

- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- During initial full data synchronization, concurrent INSERT operations cause fragmentation in the tables of the destination instance. After initial full data synchronization, the tablespace of the destination instance is larger than that of the source instance.

Supported synchronization topologies

- One-way one-to-one synchronization
- One-way one-to-many synchronization
- One-way cascade synchronization
- One-way many-to-one synchronization
- Two-way one-to-one synchronization

For more information, see Synchronization topologies.

SQL operations that can be synchronized

Limits

- Incompatibility with triggers

  If you select a database as the object and the database contains a trigger that updates a table, data inconsistency may occur. For more information about how to solve this issue, see Configure a data synchronization task for a source database that contains a trigger.

- Limits on RENAME TABLE operations

  RENAME TABLE operations may cause data inconsistency during synchronization. For example, if only Table A needs to be synchronized and it is renamed Table B, Table B cannot be synchronized to the destination database. To prevent this situation, you can select the entire database where Table A is located as the object when you configure the data synchronization task.

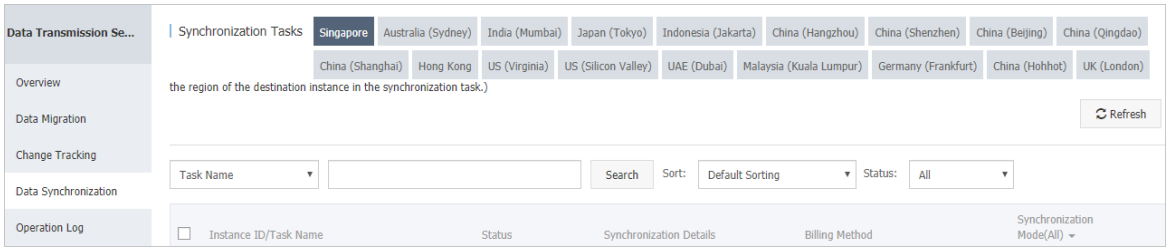
Procedure

1. Purchase a data synchronization instance. For more information, see Purchase a data synchronization

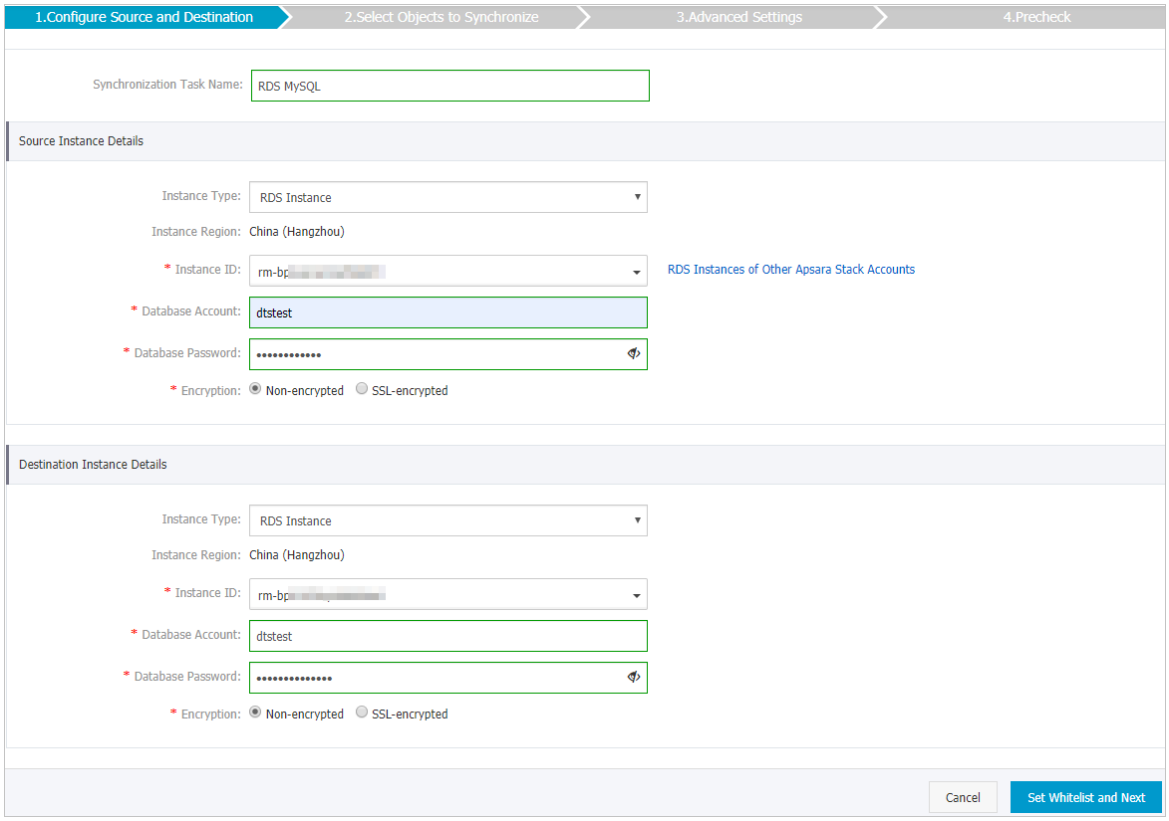
instance.

> **Note** Select **MySQL** for both the source instance and the destination instance and select **One-Way Synchronization** as the synchronization topology.

2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the data synchronization instance resides.



5. Find the data synchronization instance and click **Configure Synchronization Channel** in the Actions column.
6. Configure the source and destination instances.

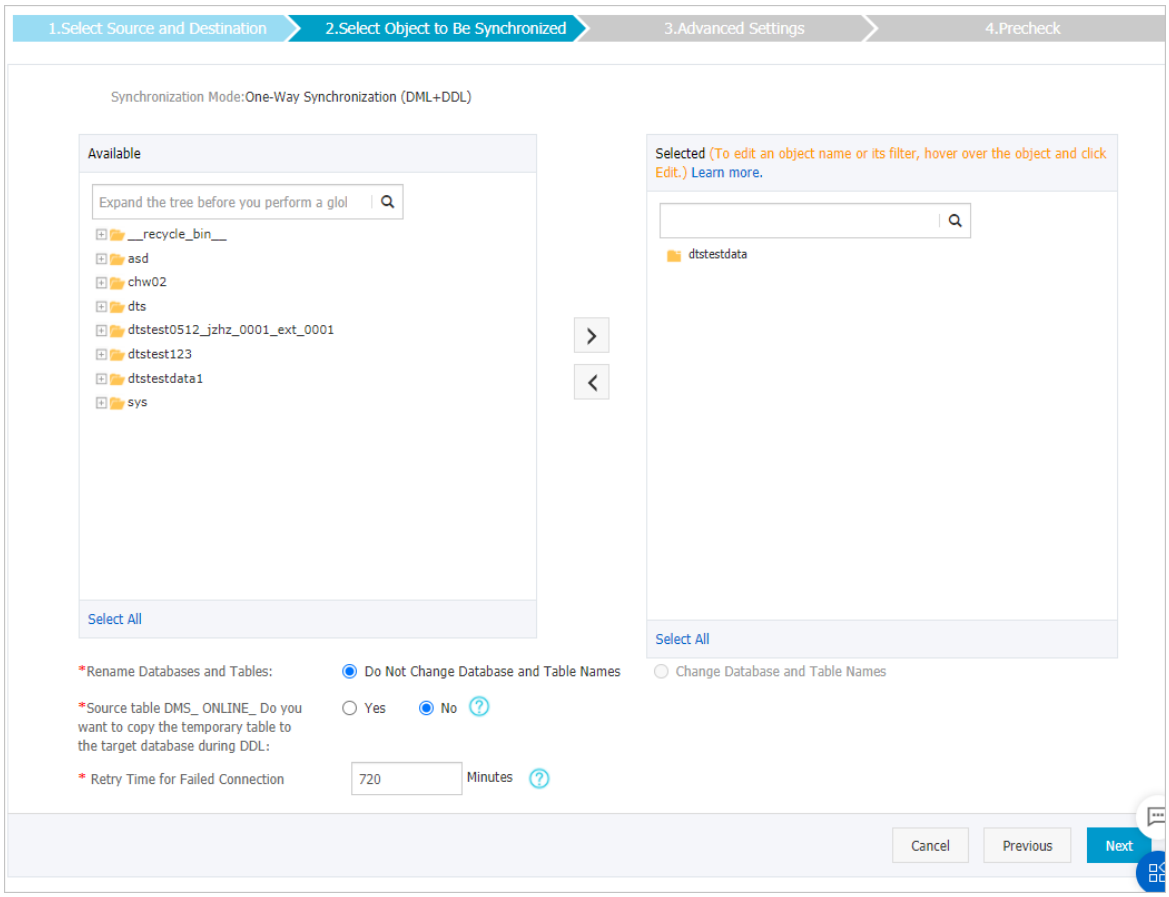


Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.

Section	Parameter	Description
Source Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the source RDS instance.
	Database Account	Enter the database account of the source RDS instance. **Note** If the database engine of the source RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the source database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance . **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.
Destination Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the destination RDS instance.
	Database Account	Enter the database account of the destination RDS instance. **Note** If the database engine of the destination RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the destination database account.

Section	Parameter	Description
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption on an ApsaraDB RDS for MySQL instance **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

7. In the lower-right corner of the page, click **Set Whitelist and Next**.
8. Select the synchronization policy and the objects to be synchronized.

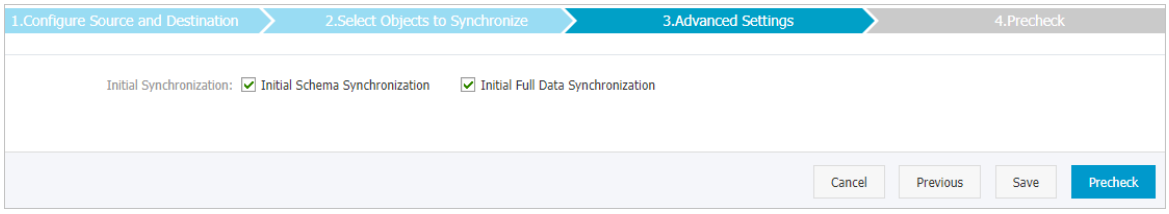


Setting	Description

Setting	Description
Select the processing mode of conflicting tables	○ **Pre-check and Intercept**: checks whether the destination database contains tables that have the same names as tables in the source database. If the source and destination databases do not contain identical table names, the precheck is passed. Otherwise, an error is returned during precheck and the data synchronization task cannot be started. **Note** You can use the object name mapping feature to change the names of the tables that are synchronized to the destination database. You can use this feature if the source and destination databases contain identical table names and the tables in the destination database cannot be deleted or renamed. For more information, see Rename an object to be synchronized. ○ **Ignore**: skips the precheck for identical table names in the source and destination databases. **Warning** If you select **Ignore**, data consistency is not guaranteed and your business may be exposed to potential risks. ■ DTS does not synchronize the data records that have the same primary keys as the data records in the destination database during initial full data synchronization. This occurs if the source and destination databases have the same schema. However, DTS synchronizes these data records during incremental data synchronization. ■ If the source and destination databases have different schemas, initial data synchronization may fail. In this case, only some columns are synchronized or the data synchronization task fails.
Select the objects to be synchronized	Select one or more objects from the **Available** section and click the › icon to move the objects to the **Selected** section. You can select tables or databases as the objects to be synchronized. **Note** ○ If you select a database as the object to be synchronized, all schema changes in the database are synchronized to the destination database. ○ By default, after an object is synchronized to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to change the names of the objects that are synchronized to the destination instance. For more information, see Rename an object to be synchronized.

Setting	Description
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Replicate Temporary Tables When DMS Performs DDL Operations	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ○ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ○ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. In the lower-right corner of the page, click **Next**.
10. Configure advanced settings.



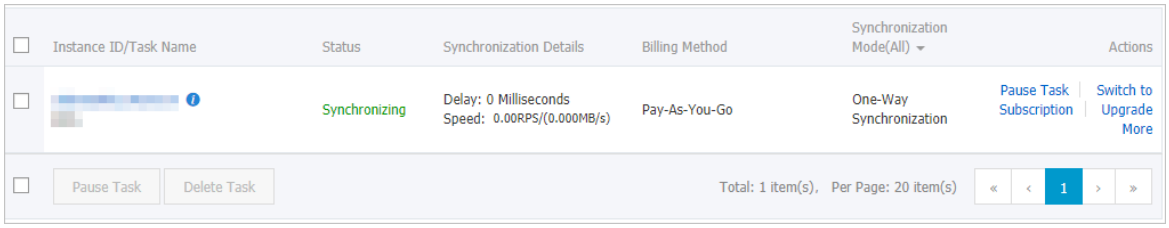
> **Note**
>
> - DTS performs initial synchronization when you start a synchronization channel. During initial synchronization, the schemas and data of the objects to be synchronized are replicated from the source instance to the destination instance. These schemas and data are then used as the baseline for subsequent incremental data synchronization.
> - Initial synchronization includes initial schema synchronization and initial full data synchronization. In most cases, you need to select both **Initial Schema Synchronization** and **Initial Full Data Synchronization**.

11. In the lower-right corner of the page, click **Precheck**.

> **Note**
>
> - Before you can start the data synchronization task, DTS performs a precheck. The data synchronization task can be started only after it passes the precheck.
> - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
>   - You can troubleshoot the issues based on the causes and run a precheck again.
>   - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

12. Close the **Precheck** dialog box after the following message is displayed: **The precheck is passed.** Then, the data synchronization task starts.
13. Wait until the initial synchronization is complete and the data synchronization task is in the **Synchronizing** state.

    You can view the status of the data synchronization task on the **Synchronization Tasks** page.



7.2.2. Configure two-way data synchronization between ApsaraDB RDS for MySQL instances

Data Transmission Service (DTS) supports real-time two-way data synchronization between two MySQL databases. This feature is applicable to scenarios such as active geo-redundancy (unit-based) and geo-disaster recovery. This topic describes how to configure two-way data synchronization between ApsaraDB RDS for MySQL instances. You can also follow the procedure to configure data synchronization tasks for self-managed MySQL databases.

Prerequisites

The source and destination ApsaraDB RDS for MySQL instances are created. For more information, see Create an ApsaraDB RDS for MySQL instance.

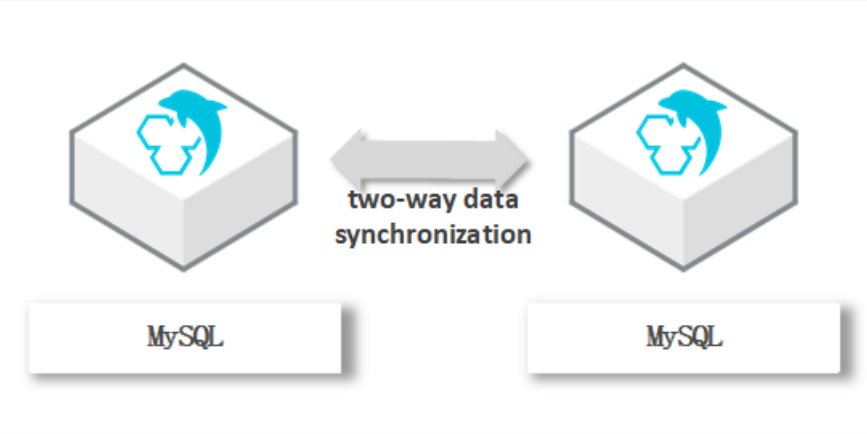
Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- We recommend that you do not use gh-ost or pt-online-schema-change to perform DDL operations on objects during data synchronization. Otherwise, data synchronization may fail.
- If you use only DTS to write data to the destination database, you can use Data Management (DMS) to perform online DDL operations during data synchronization. For more information, see DDL-based lockless change.

  > **Warning** If you use tools other than DTS to write data to the destination database, we recommend that you do not use DMS to perform online DDL operations. Otherwise, data loss may occur in the destination database.

Supported synchronization topologies

DTS supports two-way data synchronization only between two MySQL databases. DTS does not support two-way data synchronization between multiple MySQL databases.



Supported databases

The following table lists the types of MySQL databases that are supported by two-way data synchronization. This topic uses ApsaraDB RDS for MySQL instances as the data sources. You can also follow the procedure to configure two-way data synchronization for other types of MySQL databases.

Source database	Destination database
• ApsaraDB RDS for MySQL instance • Self-managed database that is hosted on Elastic Compute Service (ECS) • Self-managed database that is connected over Express Connect, VPN Gateway, or Smart Access Gateway • Self-managed database that is connected over Database Gateway • Self-managed database that is connected over Cloud Enterprise Network (CEN)	• ApsaraDB RDS for MySQL instance • Self-managed database that is hosted on Elastic Compute Service (ECS) • Self-managed database that is connected over Express Connect, VPN Gateway, or Smart Access Gateway • Self-managed database that is connected over Database Gateway • Self-managed database that is connected over Cloud Enterprise Network (CEN)

SQL operations that can be synchronized

Conflict detection

To ensure data consistency, make sure that data records with the same primary key, business primary key, or unique key are updated only on one of the synchronization nodes. If data records are updated on both nodes, DTS responds to conflicts based on the conflict resolution policy that you specify for the data synchronization task.

DTS checks and fixes conflicts to maximize the stability of two-way synchronization instances. DTS can detect the following types of conflicts:

- Uniqueness conflicts caused by INSERT operations

  INSERT operations that do not comply with the uniqueness constraint cannot be synchronized. For example, if a record with the same primary key value is inserted into the two synchronization nodes at almost the same time, one of the inserted records fails to be synchronized. The synchronization fails because a record with the same primary key value already exists in the other node.

- Inconsistent records caused by UPDATE operations
  - If the records to be updated do not exist in the destination instance, DTS converts the UPDATE operation into an INSERT operation. However, uniqueness conflicts may occur.
  - The primary keys or unique keys of the records to be inserted may conflict with those of existing records in the destination instance.

- Non-existent records to be deleted

  The records to be deleted do not exist in the destination instance. In this case, DTS ignores the DELETE operation regardless of the conflict resolution policy that you specify.

> **Notice**
>
> - During two-way synchronization, the system time of the source and destination instances may be different. Synchronization latency may occur. For these reasons, DTS cannot guarantee that the conflict detection mechanism can prevent all data conflicts. To perform two-way synchronization, make sure that records with the same primary key, business primary key, or unique key are updated only on one of the synchronization nodes.
> - DTS provides conflict resolution policies to prevent conflicts that may occur during data synchronization. You can select a conflict resolution policy when you configure two-way data synchronization.

Limits

- Incompatibility with triggers

  If you select a database as the object to be synchronized and the database contains a trigger that updates a table, data inconsistency may occur. For example, the source database contains Table A and Table B. If a data record is inserted into Table A, a trigger inserts a data record into Table B. In this case, after an INSERT operation is performed on Table A in the source instance, the data in Table B becomes inconsistent between the source and destination instances.

  To prevent this situation, before you synchronize data in Table B from the source instance, delete the trigger that is synchronized to the destination instance. For more information, see Configure a data synchronization task for a source database that contains a trigger.

- Limits on RENAME TABLE operations

  RENAME TABLE operations may cause data inconsistency between the source and destination databases. For example, if you select a table as the object and rename the table during data synchronization, the data of this table is not synchronized to the destination database. To prevent this situation, you can select the database to which this table belongs as the object when you configure the data synchronization task.

- Limits on DDL synchronization direction

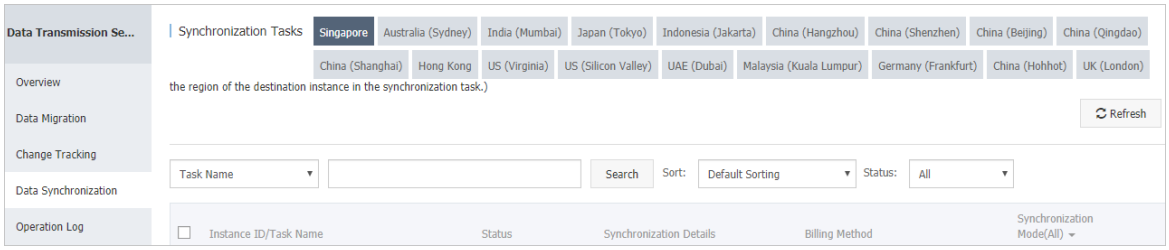
  To ensure the stability of two-way data synchronization, you can synchronize the DDL operations on a single table only in one direction. If DDL synchronization in a direction is configured, DDL synchronization in the opposite direction is not supported. Only data manipulation language (DML) operations can be synchronized in the opposite direction.

Procedure

1. Purchase an instance for two-way data synchronization. For more information, see Purchase a data synchronization instance.

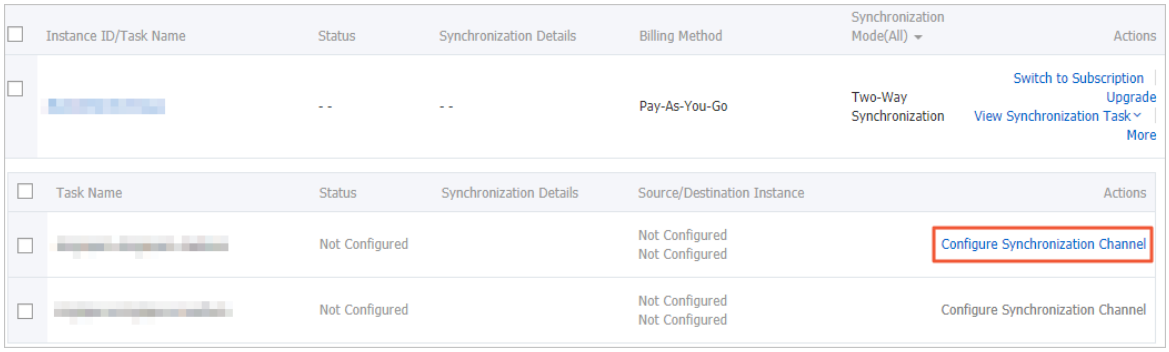
   > **Notice** On the buy page, set both Source Instance and Destination Instance to **MySQL** and set Synchronization Topology to **Two-way Synchronization**.

2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the destination instance resides.

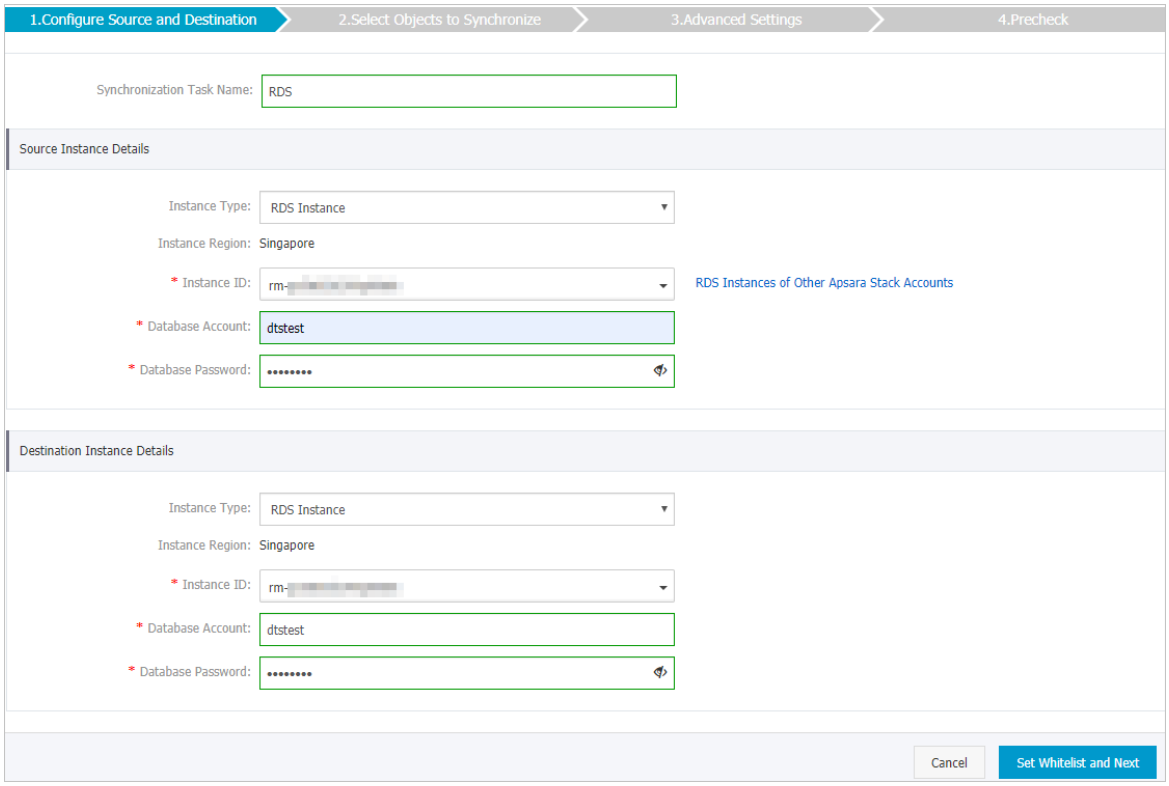


5. Find the data synchronization instance and click **Configure Synchronization Channel** in the Actions column of the first data synchronization task.

> **Notice** A two-way data synchronization instance contains two data synchronization tasks. You must set parameters for each task.



6. Configure source and destination instances.

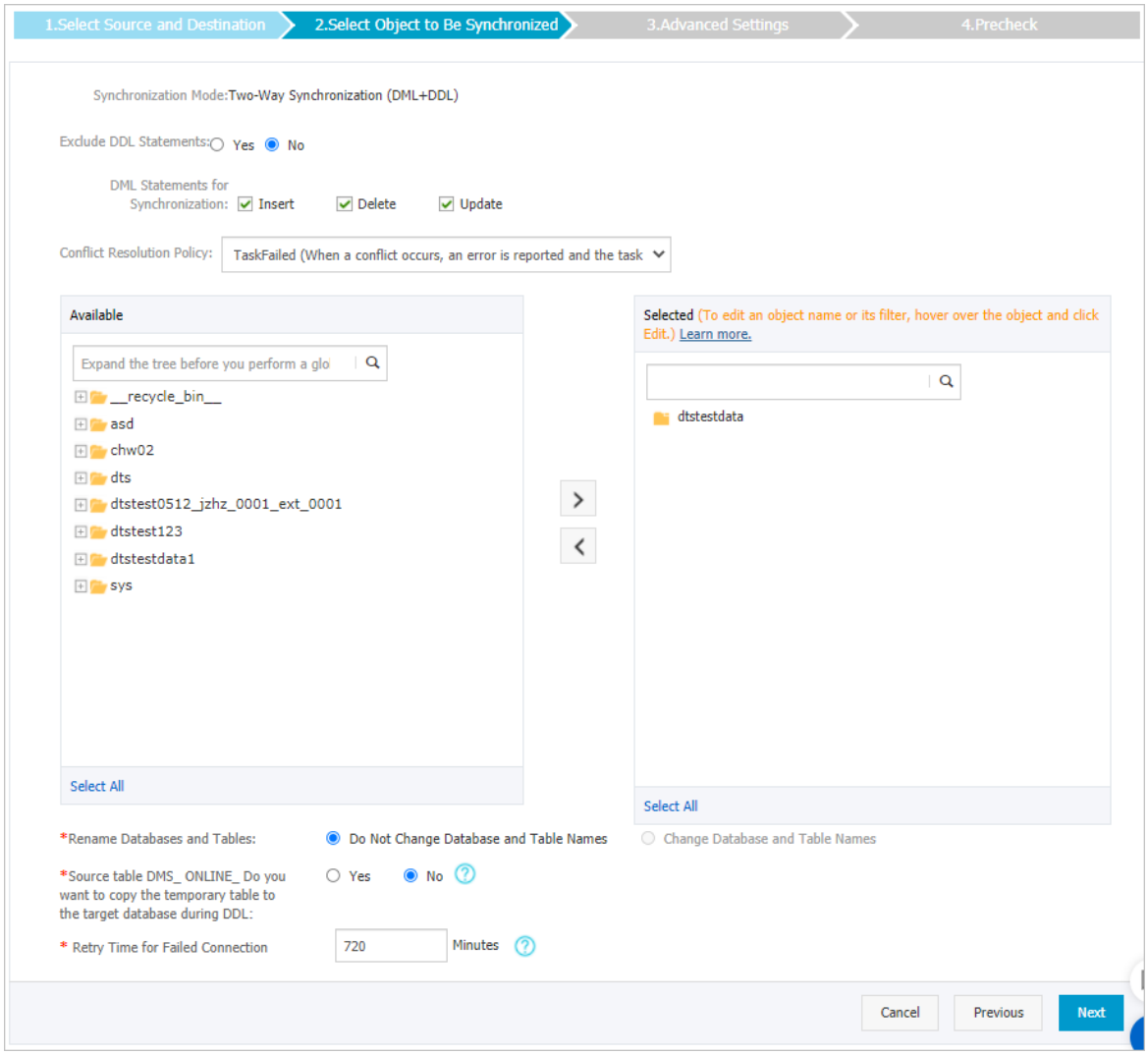


Section	Parameter	Description

Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the source RDS instance.
	Database Account	Enter the database account of the source RDS instance. **Notice** If the database engine of the source RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in mainland China and the China (Hong Kong) region.
	Instance Type	Select **RDS Instance**.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the destination RDS instance.

Section	Parameter	Description
Destination Instance Details	Database Account	Enter the database account of the destination RDS instance. **Notice** If the database engine of the destination RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption on an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

7. In the lower-right corner of the page, click **Set Whitelist and Next**.
8. Select the synchronization policy and the objects to be synchronized.



Setting	Parameter	Description
	Exclude DDL Statements	◦ To exclude DDL operations, select **Yes**. ◦ To include DDL operations, select **No**. **Notice** Limits on DDL synchronization direction: To ensure the stability of two-way data synchronization, you can synchronize DDL operations only in the forward direction.
	DML Statements for Synchronization	Select the types of DML operations that you want to synchronize. By default, the **INSERT**, **UPDATE**, and **DELETE** operations are selected. You can select the DML operation types based on your business requirements.

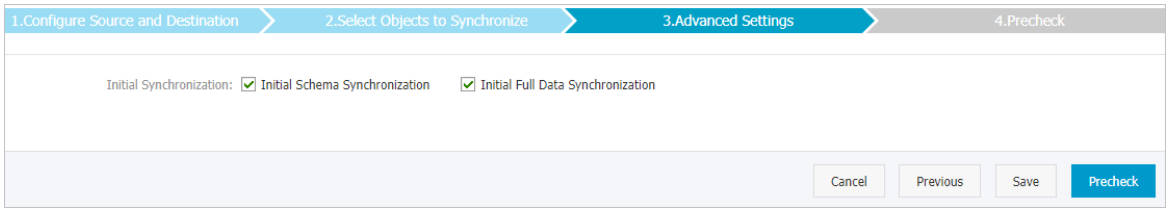
Setting	Parameter	Description
Select the synchronization policy	Conflict Resolution Policy	Select the resolution policy for synchronization conflicts. By default, **TaskFailed** is selected. You can select a conflict resolution policy based on your business requirements. ◦ **TaskFailed** The default conflict resolution policy. If a conflict occurs during data synchronization, the synchronization task reports an error and exits the process. The task enters a failed state and you must manually resolve the conflict. ◦ **Ignore** If a conflict occurs during data synchronization, the synchronization task ignores the current statement and continues the process. The conflicting records in the destination database are used. ◦ **Overwrite** If a conflict occurs during data synchronization, the conflicting records in the destination database are overwritten.

Setting	Parameter	Description
	Processing Mode of Conflicting Tables	◦ **Pre-check and Intercept**: checks whether the destination database contains tables that have the same names as tables in the source database. If the source and destination databases do not contain identical table names, the precheck is passed. Otherwise, an error is returned during precheck and the data synchronization task cannot be started. **Notice** You can use the object name mapping feature to rename the tables that are synchronized to the destination database. You can use this feature if the source and destination databases contain identical table names and the tables in the destination database cannot be deleted or renamed. For more information, see Rename an object to be synchronized. ◦ **Ignore**: skips the precheck for identical table names in the source and destination databases. **Warning** If you select **Ignore**, data consistency is not guaranteed and your business may be exposed to potential risks. ■ During initial data synchronization, DTS does not synchronize the data records that have the same primary keys as the data records in the destination database. This occurs if the source and destination databases have the same schema. However, DTS synchronizes these data records during incremental data synchronization. ■ If the source and destination databases have different schemas, initial data synchronization may fail. In this case, only some columns are synchronized or the data synchronization task fails.

Setting	Parameter	Description
Select the objects to be synchronized	N/A	Select one or more objects (tables or a database) from the **Available** section and click the › icon to move the objects to the **Selected** section. **Notice** ◦ If you select a database as the object to be synchronized, all schema changes in the database are synchronized to the destination database. ◦ By default, after an object is synchronized to the destination instance, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Rename an object to be synchronized.
Rename Databases and Tables	N/A	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Replicate Temporary Tables When DMS Performs DDL Operations	N/A	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ◦ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ◦ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.

Setting	Parameter	Description
Retry Time for Failed Connections	N/A	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. In the lower-right corner of the page, click **Next**.
10. Select the initial synchronization types.



During initial synchronization, DTS synchronizes the schemas and data of the required objects from the source instance to the destination instance. The schemas and data are the basis for subsequent incremental synchronization. Initial synchronization includes **initial schema synchronization** and **initial full data synchronization**. You must select both **Initial Schema Synchronization** and **Initial Full Data Synchronization** in most cases.

Notice If tables to be synchronized in one direction are also included in the objects to be synchronized in the opposite direction, DTS does not synchronize these tables during initial synchronization.

11. In the lower part of the page, click **Next: Precheck and Start Task**.

Notice

- Before you can start the data synchronization task, a precheck is performed. You can start the data synchronization task only after the task passes the precheck.
- If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
  - You can troubleshoot the issues based on the causes and run a precheck again.
  - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

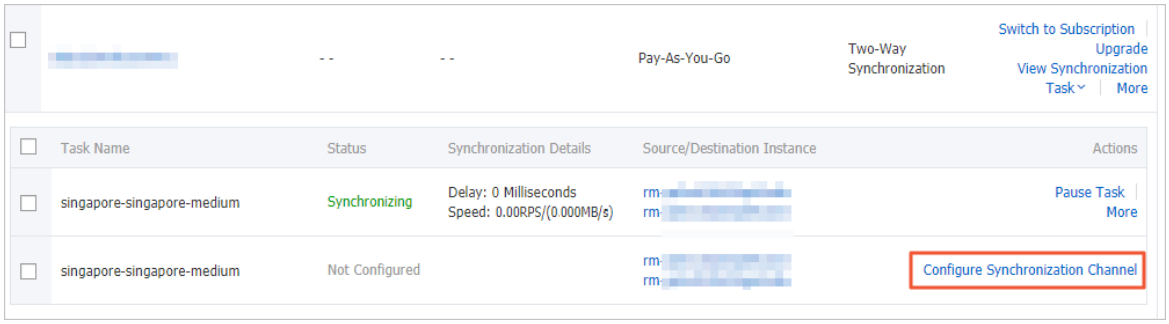
12. Close the **Precheck** dialog box after the following message is displayed: **The precheck is**

passed. Then, the data synchronization task starts.

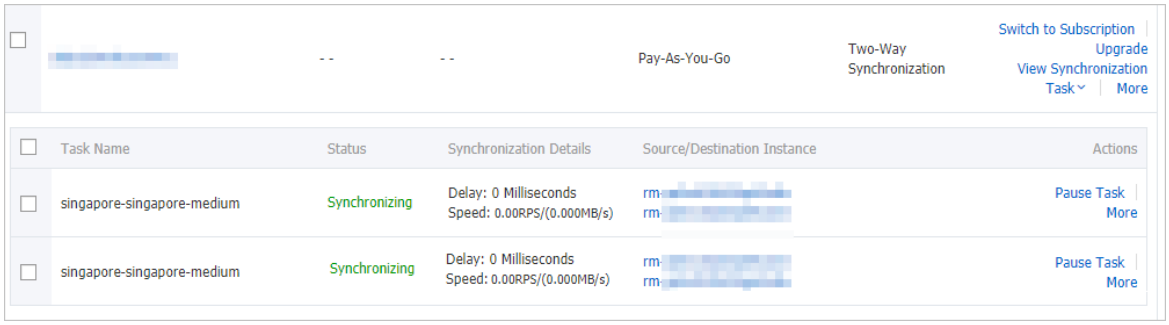
13. Wait until initial synchronization is completed and the data synchronization task is in the **Synchronizing** state.

    You can view the status of the data synchronization task on the **Synchronization Tasks** page.

14. Find the second data synchronization task and click **Configure Synchronization Channel** in the Actions column. Configure the task by following Steps 5 to 12.



15. After the second data synchronization task is configured, wait until both tasks are in the **Synchronizing** state. The two-way data synchronization tasks are configured.



7.2.3. Synchronize data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL instance

This topic describes how to synchronize data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to an ApsaraDB RDS for MySQL instance by using Data Transmission Service (DTS).

Prerequisites

- The destination RDS instance is created. For more information, see Create an ApsaraDB RDS for MySQL instance.
- The version of the self-managed MySQL database is 5.1, 5.5, 5.6, 5.7, or 8.0.
- The self-managed MySQL database is connected to Alibaba Cloud VPC over Express Connect, VPN Gateway, or Smart Access Gateway. For more information, see Connect an on-premises database to DTS by using CEN.

> **Note** DTS is allowed to access the VPC to which the self-managed MySQL database belongs. For more information, see Configure a route between DTS and Express Connect, VPN Gateway, or Smart Access Gateway.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- We recommend that you do not use gh-ost or pt-online-schema-change to perform DDL operations on objects during data synchronization. Otherwise, data synchronization may fail.
- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- During initial full data synchronization, concurrent INSERT operations cause fragmentation in the tables of the destination instance. After initial full data synchronization, the tablespace of the destination instance is larger than that of the source instance.

Supported synchronization topologies

- One-way one-to-one synchronization
- One-way one-to-many synchronization
- One-way cascade synchronization
- One-way many-to-one synchronization
- Two-way one-to-one synchronization

For more information about synchronization topologies, see Synchronization topologies.

SQL operations that can be synchronized

Limits

- Incompatibility with triggers

  If you select a database as the object to synchronize and the database contains a trigger that updates a table, data inconsistency may occur. For more information about how to solve this issue, see Configure a data synchronization task for a source database that contains a trigger.

- Limits on RENAME TABLE operations

  RENAME TABLE operations may cause data inconsistency between the source and destination databases. For example, if only Table A is selected as the object to synchronize and is renamed Table B, Table B cannot be synchronized to the destination database. To prevent this situation, you can select the entire database where Table A is located as the object to synchronize when you configure the data synchronization task.

Before you begin

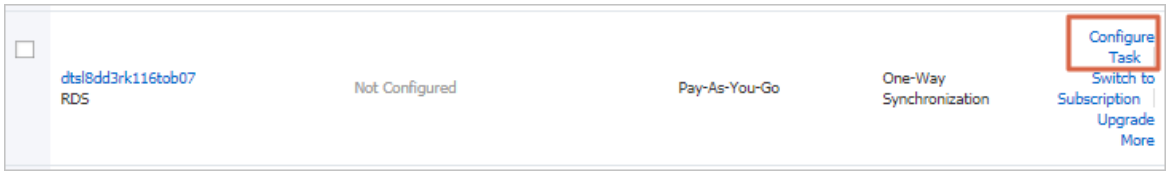
Before you configure the data synchronization task, you must create a database account and configure binary logging. For more information, see Create an account for a user-created MySQL database and configure binary logging.

Procedure

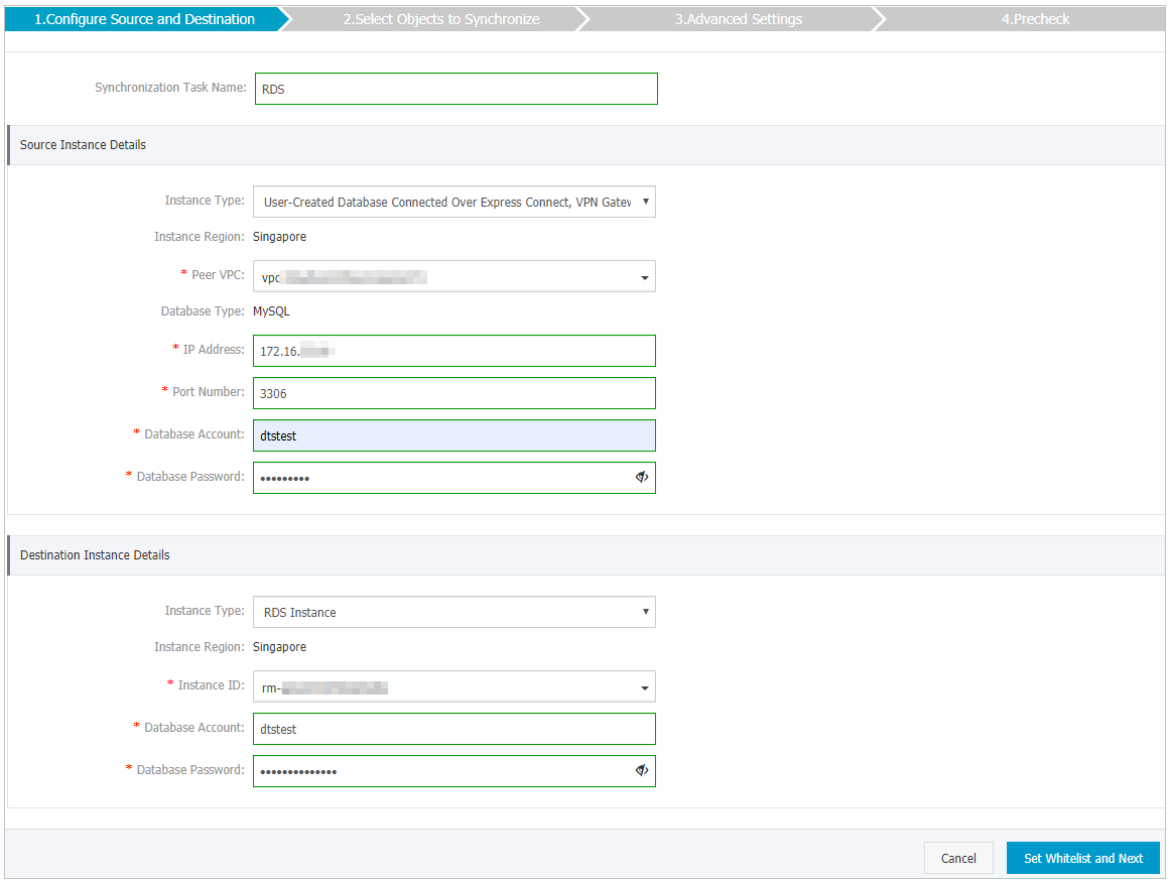
1. Purchase a data synchronization instance. For more information, see Purchase procedure.

   > **Note** Select **MySQL** for both the source instance and the destination instance. Select **One-Way Synchronization** as the synchronization topology.

2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the destination instance resides.
5. Find the data synchronization instance and click **Configure Synchronization Channel** in the Actions column.



6. Configure the source and destination instances.



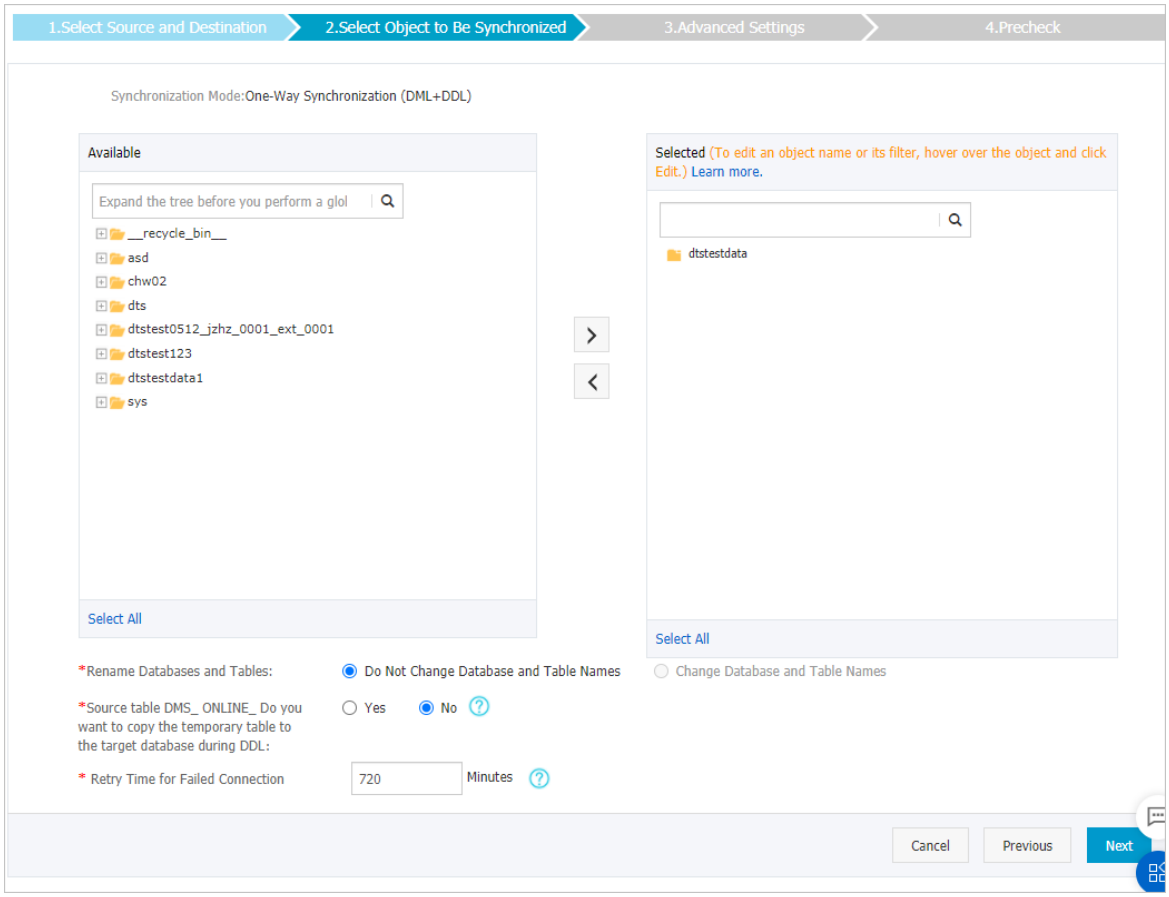
Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **User-Created Database Connected over Express Connect, VPN Gateway, or Smart Access Gateway**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	Peer VPC	Select the ID of the VPC that is connected to the self-managed MySQL database.
	Database Type	The value of this parameter is set to MySQL and cannot be changed.
	IP Address	Enter the server IP address of the self-managed MySQL database.
	Port Number	Enter the service port number of the self-managed MySQL database. The default port number is **3306**.
	Database Account	Enter the account of the self-managed MySQL database. The account must have the SELECT permission on the required objects, the REPLICATION CLIENT permission, the REPLICATION SLAVE permission, and the SHOW VIEW permission.
	Database Password	Enter the password of the source database account.
Destination Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Redis Instance ID	Select the ID of the destination RDS instance.
	Database Account	Enter the database account of the destination RDS instance. **Note** If the database engine of the destination RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the destination database account.

Section	Parameter	Description
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

7. In the lower-right corner of the page, click **Set Whitelist and Next**.

> **Note**

8. Select the synchronization policy and the objects to be synchronized.

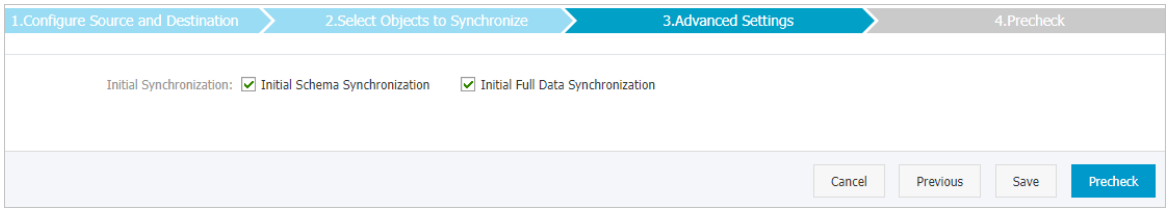


Setting	Description

Setting	Description
Select the objects to be synchronized	Select one or more objects from the **Available** section and click the icon to add the objects to the **Selected** section. You can select tables or databases as the objects to synchronize. **Note** ◦ If you select a database as the object to synchronize, all schema changes in the database are synchronized to the destination database. ◦ By default, after an object is synchronized to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Rename an object to be synchronized.
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Replicate Temporary Tables When DMS Performs DDL Operations	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ◦ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ◦ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.

Setting	Description
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

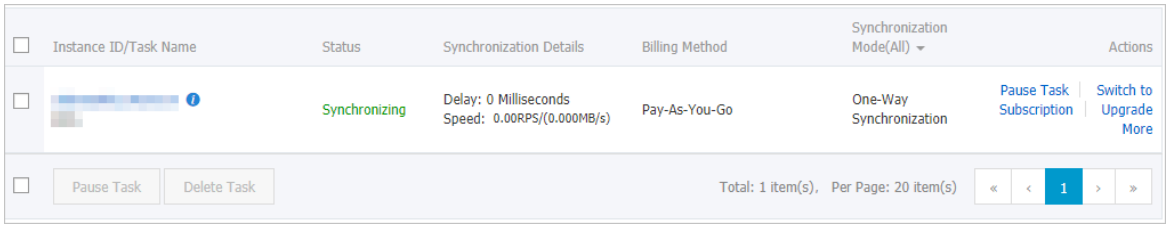
9. In the lower-right corner of the page, click **Next**.
10. Configure initial synchronization.



   - During an initial synchronization, DTS synchronizes the schemas and data of the required objects from the source instance to the destination instance. The schemas and data are the basis for subsequent incremental synchronization.
   - Initial synchronization includes initial schema synchronization and initial full data synchronization. In most cases, you need to select both **Initial Schema Synchronization** and **Initial Full Data Synchronization**.
11. In the lower part of the page, click **Next: Precheck and Start Task**.

    Notice
    - Before you can start the data synchronization task, a precheck is performed. You can start the data synchronization task only after the task passes the precheck.
    - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
      - You can troubleshoot the issues based on the causes and run a precheck again.
      - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.
12. Close the **Precheck** dialog box after the following message is displayed: **The precheck is passed.** Then, the data synchronization task starts.
13. Wait until the initial synchronization is complete and the data synchronization task is in the **Synchronizing** state.

You can view the status of the data synchronization task on the **Synchronization Tasks** page.



7.2.4. Synchronize data between ApsaraDB RDS for MySQL instances that belong to different Alibaba Cloud accounts

This topic describes how to synchronize data between ApsaraDB RDS for MySQL instances that belong to different Alibaba Cloud accounts by using Data Transmission Service (DTS).

Prerequisites

- The source and destination ApsaraDB RDS for MySQL instances are created. For more information, see Create an ApsaraDB RDS for MySQL instance.
- The database type of the source and destination RDS instances is MySQL.
- The source and destination ApsaraDB RDS for MySQL instances have internal endpoints.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- If you have selected one or more tables (not a database) for synchronization, do not use gh-ost or pt-online-schema-change to modify the tables during data synchronization. Otherwise, data synchronization may fail.

  Notice To avoid synchronization failure, you can use Data Management (DMS) to perform online DDL schema changes during data synchronization. For more information, see Change the table schema without locking.

- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- During initial full data synchronization, concurrent INSERT operations cause fragmentation in the tables of the destination instance. After initial full data synchronization, the tablespace of the destination instance is larger than that of the source instance.

Supported synchronization topologies

- One-way one-to-one synchronization
- One-way one-to-many synchronization
- One-way cascade synchronization
- One-way many-to-one synchronization

For more information, see Synchronization topologies.

SQL operations that can be synchronized

Limits

- Incompatibility with triggers

  If you select a database as the object to synchronize and the database contains a trigger that updates a table, data inconsistency may occur. For more information about how to solve this issue, see Configure a data synchronization task for a source database that contains a trigger.

- Limits on RENAME TABLE operations

  RENAME TABLE operations may cause data inconsistency between the source and destination databases. For example, if only Table A is selected as the object to synchronize and is renamed Table B, Table B cannot be synchronized to the destination database. To prevent this situation, you can select the entire database where Table A is located as the object to synchronize when you configure the data synchronization task.

Before you begin

Set the Alibaba Cloud account that owns the destination RDS instance as a trusted account. This allows DTS to access the cloud resources of the Alibaba Cloud account that owns the source RDS instance. For more information, see Configure RAM authorization for cross-account data migration and synchronization.

> **Note** To authorize the Alibaba Cloud account that owns the destination instance, you must log on to the Resource Access Management (RAM) console with the Alibaba Cloud account that owns the source instance. Then, you can create a data migration task or data synchronization task by using the Alibaba Cloud account that owns the destination instance.

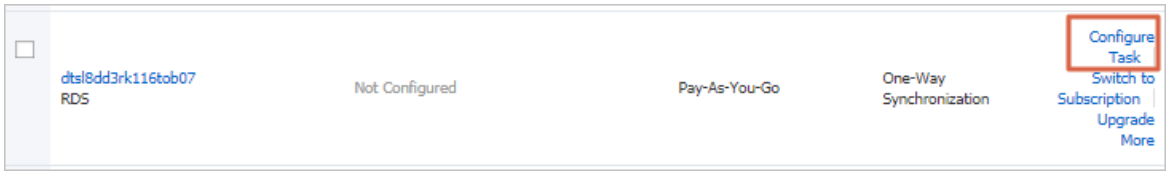
Procedure

1. Purchase a data synchronization instance by using the Alibaba Cloud account that owns the destination RDS instance. For more information, see Purchase a data synchronization instance.

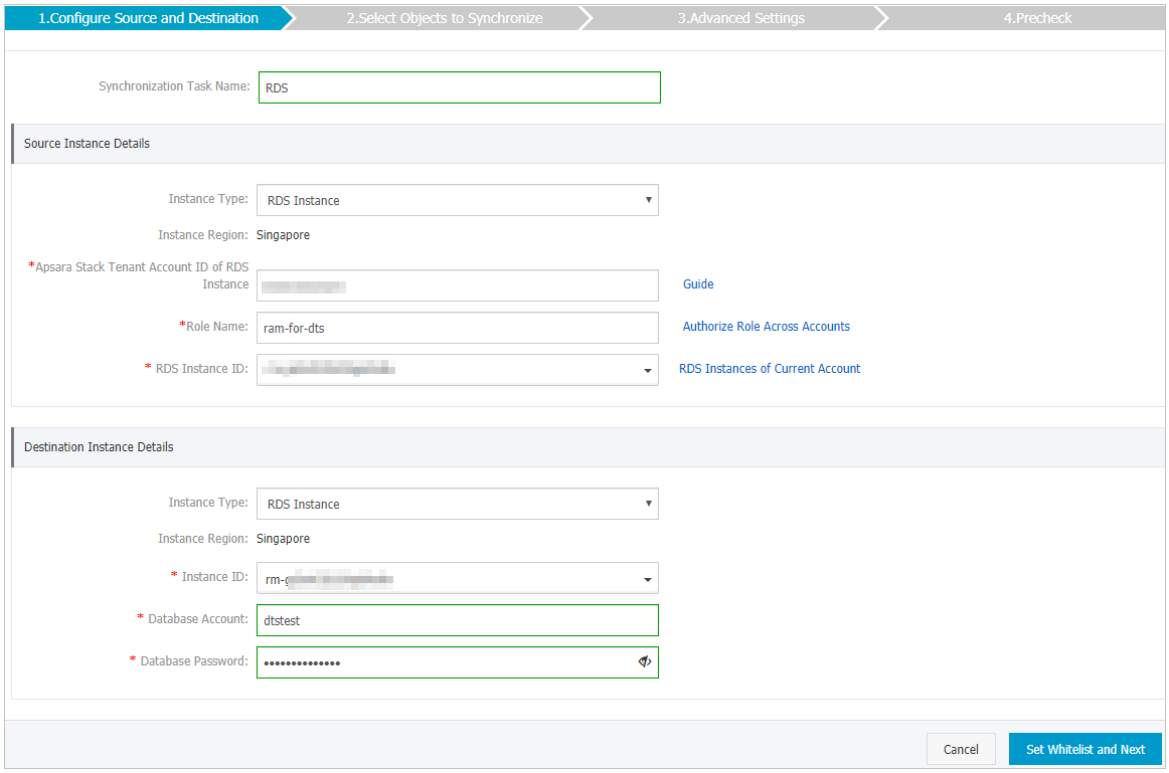
   > **Note** Select **MySQL** for both the source instance and the destination instance. Select **One-Way Synchronization** as the synchronization topology.

2. Use the Alibaba Cloud account that owns the destination RDS instance to log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the destination instance resides.
5. Find the data synchronization instance and click **Configure Synchronization Channel** in the

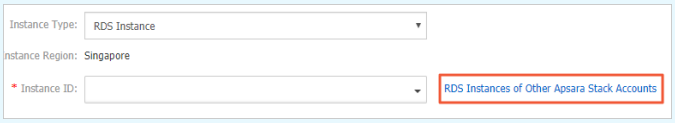
Actions column.



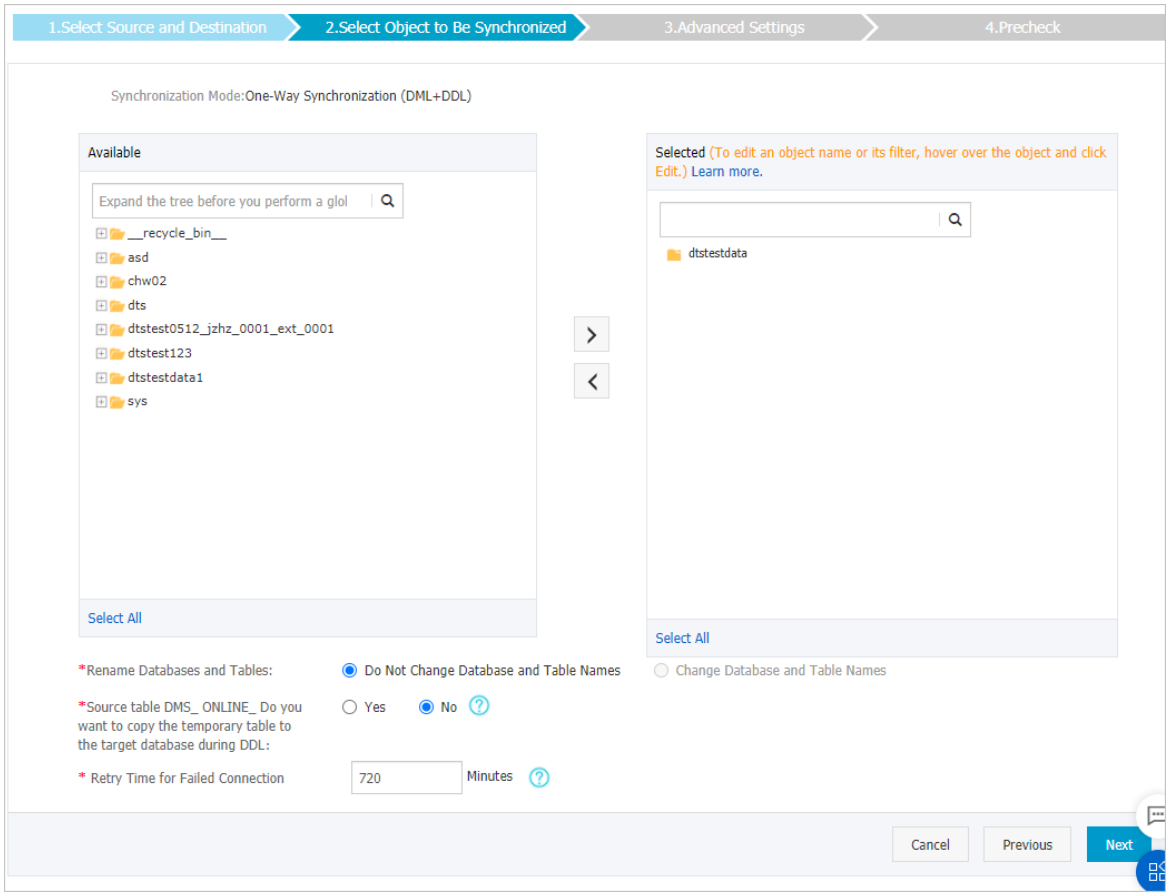
6. Configure the source and destination instances.



Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
	Instance Type	Select **RDS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.

Section	Parameter	Description
Source Instance Details	Alibaba Cloud Account ID of RDS Instance	Enter the ID of the Alibaba Cloud account that owns the source RDS instance. **Note** Before you configure this parameter, click **RDS Instances of Other Alibaba Cloud Accounts** in the **Source Instance Details** section. 
	Role Name	Enter the name of the RAM role that you configured earlier in Before you begin.
	RDS Instance ID	Select the ID of the source RDS instance.
Destination Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the destination RDS instance.
	Database Account	Enter the database account of the destination RDS instance. **Note** If the database engine of the destination RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the destination database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

7. In the lower-right corner of the page, click **Set Whitelist and Next**.
8. Select the synchronization policy and the objects to be synchronized.

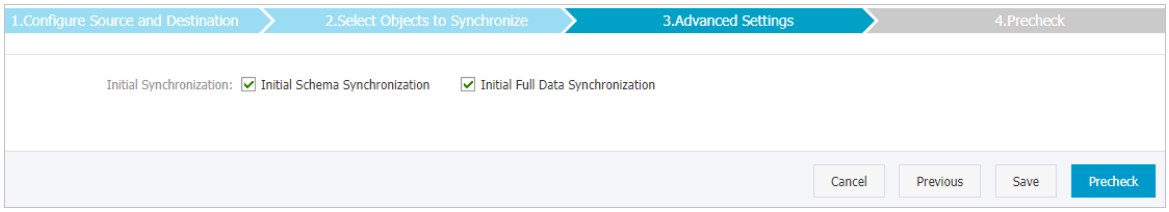


Setting	Description

Setting	Description
Select the objects to be synchronized	Select one or more objects from the **Available** section and click the icon to add the objects to the **Selected** section. You can select tables or databases as the objects to synchronize. **Note** ◦ If you select a database as the object to synchronize, all schema changes in the database are synchronized to the destination database. ◦ By default, after an object is synchronized to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Rename an object to be synchronized.
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Replicate Temporary Tables When DMS Performs DDL Operations	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ◦ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ◦ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.

Setting	Description
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. In the lower-right corner of the page, click **Next**.
10. Configure initial synchronization.

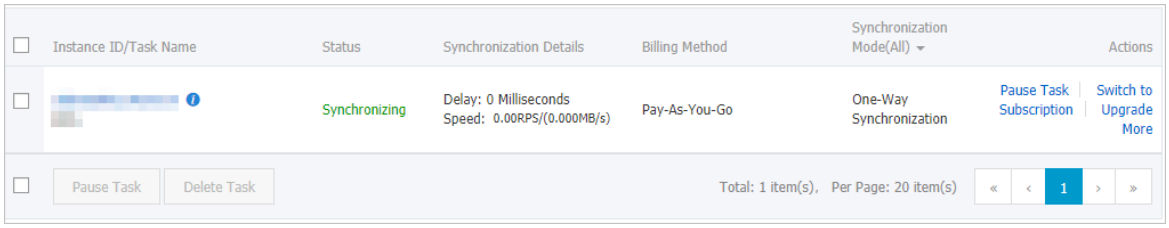


   - During an initial synchronization, DTS synchronizes the schemas and data of the required objects from the source instance to the destination instance. The schemas and data are the basis for subsequent incremental synchronization.
   - Initial synchronization includes initial schema synchronization and initial full data synchronization. In most cases, you need to select both **Initial Schema Synchronization** and **Initial Full Data Synchronization**.
11. In the lower part of the page, click **Next: Precheck and Start Task**.

> **Notice**
>
> - Before you can start the data synchronization task, a precheck is performed. You can start the data synchronization task only after the task passes the precheck.
> - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
>   - You can troubleshoot the issues based on the causes and run a precheck again.
>   - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

12. Close the **Precheck** dialog box after the following message is displayed: **The precheck is passed.** Then, the data synchronization task starts.
13. Wait until the initial synchronization is complete and the data synchronization task is in the **Synchronizing** state.

You can view the status of the data synchronization task on the **Synchronization Tasks** page.



7.2.5. Synchronize data from a self-managed MySQL database hosted on ECS to an ApsaraDB RDS for MySQL instance

This topic describes how to synchronize data from a self-managed MySQL database hosted on Elastic Compute Service (ECS) to an ApsaraDB RDS for MySQL instance by using Data Transmission Service (DTS).

Prerequisites

- The version of the self-managed MySQL database is 5.1, 5.5, 5.6, 5.7, or 8.0.
- The destination RDS instance is created. For more information, see Create an ApsaraDB RDS for MySQL instance.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- We recommend that you do not use gh-ost or pt-online-schema-change to perform DDL operations on objects during data synchronization. Otherwise, data synchronization may fail.
- If you use only DTS to write data to the destination database, you can use Data Management (DMS) to perform online DDL operations during data synchronization. For more information, see Change schemas without locking tables.

  > **Warning** If you use tools other than DTS to write data to the destination database, we recommend that you do not use DMS to perform online DDL operations. Otherwise, data loss may occur in the destination database.

- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.
- During initial full data synchronization, concurrent INSERT operations cause fragmentation in the tables of the destination cluster. After initial full data synchronization, the tablespace of the destination cluster is larger than that of the source database.

Supported synchronization topologies

- One-way one-to-one synchronization
- One-way one-to-many synchronization
- One-way many-to-one synchronization
- One-way cascade synchronization
- Two-way one-to-one synchronization

> **Note** For more information about two-way synchronization, see Configure two-way data synchronization between MySQL instances.

SQL operations that can be synchronized

Limits

- Incompatibility with triggers

  If you select a database as the object to synchronize and the database contains a trigger that updates a table, data inconsistency may occur. For more information about how to solve this issue, see Configure a data synchronization task for a source database that contains a trigger.

- Limits on RENAME TABLE operations

  RENAME TABLE operations may cause data inconsistency between the source and destination databases. For example, if only Table A is selected as the object to synchronize and is renamed Table B, Table B cannot be synchronized to the destination database. To prevent this situation, you can select the entire database where Table A is located as the object to synchronize when you configure the data synchronization task.

Before you begin

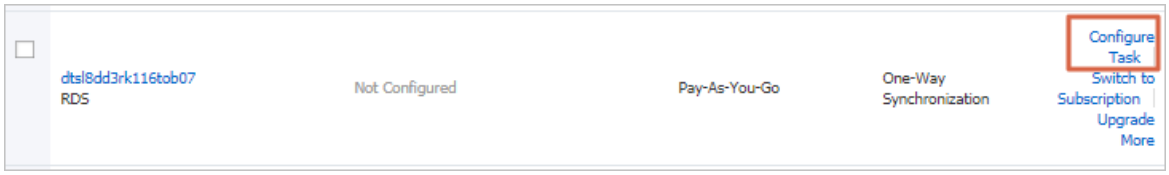
Before you configure the data synchronization task, you must create a database account and configure binary logging. For more information, see Create an account for a user-created MySQL database and configure binary logging.

Procedure

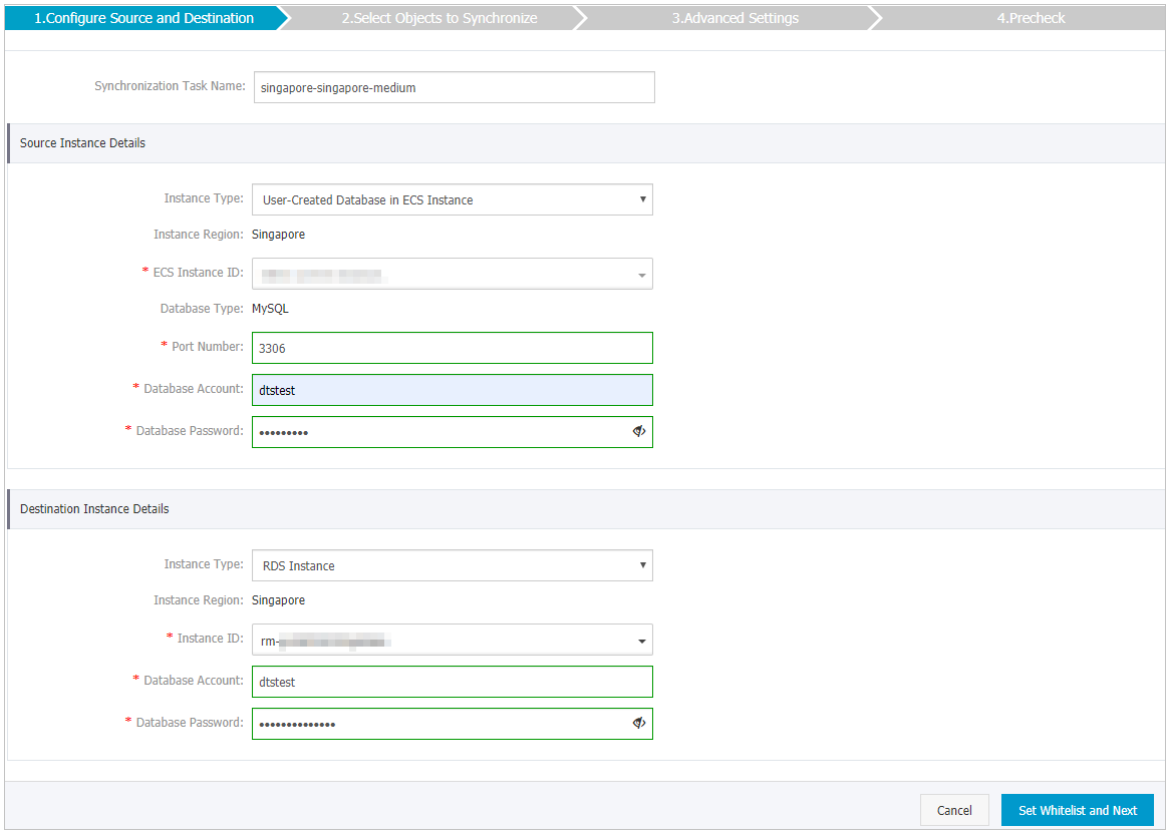
1. Purchase a data synchronization instance. For more information, see Purchase procedure.

   > **Note** Select **MySQL** for both the source instance and the destination instance. Select **One-Way Synchronization** as the synchronization topology.

2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the destination instance resides.
5. Find the data synchronization instance and click **Configure Synchronization Channel** in the Actions column.



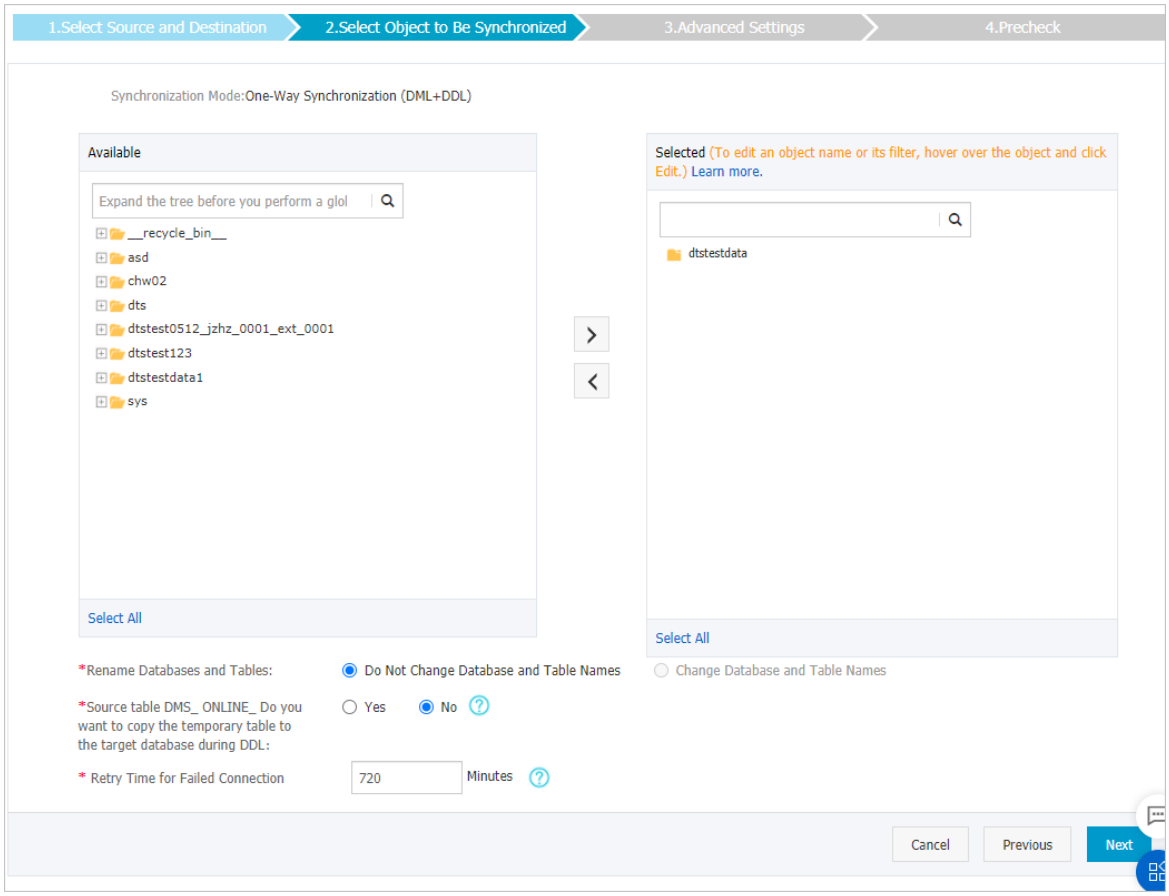
6. Configure the source and destination instances.



Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **User-Created Database in ECS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	ECS Instance ID	Select the ID of the ECS instance that hosts the self-managed MySQL database.
	Database Type	The value of this parameter is set to **MySQL** and cannot be changed.
	Port Number	Enter the service port number of the self-managed MySQL database. The default port number is **3306**.

Section	Parameter	Description
	Database Account	Enter the account of the self-managed MySQL database. The account must have the SELECT permission on the required objects, the REPLICATION CLIENT permission, the REPLICATION SLAVE permission, and the SHOW VIEW permission.
	Database Password	Enter the password for the account of the self-managed MySQL database.
Destination Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the destination RDS instance.
	Database Account	Enter the database account of the destination RDS instance. **Note** If the database engine of the destination RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the destination database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

7. In the lower-right corner of the page, click **Set Whitelist and Next**.
8. Select the synchronization policy and the objects to be synchronized.

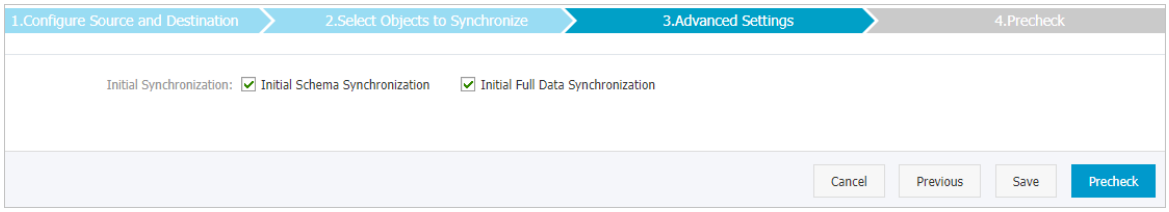


Setting	Description

Setting	Description
Select the objects to be synchronized	Select one or more objects from the **Available** section and click the icon to add the objects to the **Selected** section. You can select tables or databases as the objects to synchronize. **Note** - If you select a database as the object to synchronize, all schema changes in the database are synchronized to the destination database. - By default, after an object is synchronized to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Rename an object to be synchronized.
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Replicate Temporary Tables When DMS Performs DDL Operations	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. - **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. - **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.

Setting	Description
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. In the lower-right corner of the page, click **Next**.
10. Configure initial synchronization.

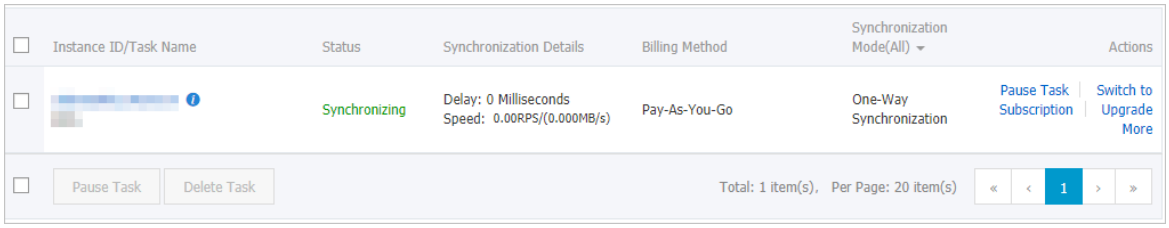


   - During initial synchronization, DTS synchronizes the schemas and data of the required objects from the source instance to the destination instance. The schemas and data are the basis for subsequent incremental synchronization.
   - Initial synchronization includes initial schema synchronization and initial full data synchronization. In most cases, you need to select both **Initial Schema Synchronization** and **Initial Full Data Synchronization**.
11. In the lower part of the page, click **Next: Precheck and Start Task**.

> **Notice**
>
> - Before you can start the data synchronization task, a precheck is performed. You can start the data synchronization task only after the task passes the precheck.
> - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
>   - You can troubleshoot the issues based on the causes and run a precheck again.
>   - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

12. Close the **Precheck** dialog box after the following message is displayed: **The precheck is passed.** Then, the data synchronization task starts.
13. Wait until the initial synchronization is complete and the data synchronization task is in the **Synchronizing** state.

You can view the status of the data synchronization task on the **Synchronization Tasks** page.



7.2.6. Synchronize data from an ApsaraDB RDS for MySQL instance to a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway

This topic describes how to synchronize data from an ApsaraDB RDS for MySQL instance to a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway by using Data Transmission Service (DTS).

Prerequisites

- The version of the self-managed MySQL database is 5.1, 5.5, 5.6, 5.7, or 8.0.

  > **Note** We recommend that you make sure the version of the source and destination MySQL databases is the same.

- The self-managed MySQL database is connected to Alibaba Cloud VPC over Express Connect, VPN Gateway, or Smart Access Gateway. For more information, see Connect an on-premises database to DTS by using CEN.

  > **Note** DTS is allowed to access the VPC to which the self-managed MySQL database belongs. For more information, see Configure a route between DTS and Express Connect, VPN Gateway, or Smart Access Gateway.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- During initial full data synchronization, concurrent INSERT operations cause fragmentation in the tables of the destination instance. After initial full data synchronization, the tablespace of the destination instance is larger than that of the source instance.

Limits

-
- The tables to be migrated in the source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.

Supported synchronization topologies

- One-way one-to-one synchronization
- One-way one-to-many synchronization
- One-way cascade synchronization
- One-way many-to-one synchronization
- Two-way one-to-one synchronization

For more information about synchronization topologies, see Synchronization topologies.

SQL operations that can be synchronized

Limits

- Incompatibility with triggers

  If you select a database as the object to synchronize and the database contains a trigger that updates a table, data inconsistency may occur. For more information about how to solve this issue, see Configure a data synchronization task for a source database that contains a trigger.

- Limits on RENAME TABLE operations

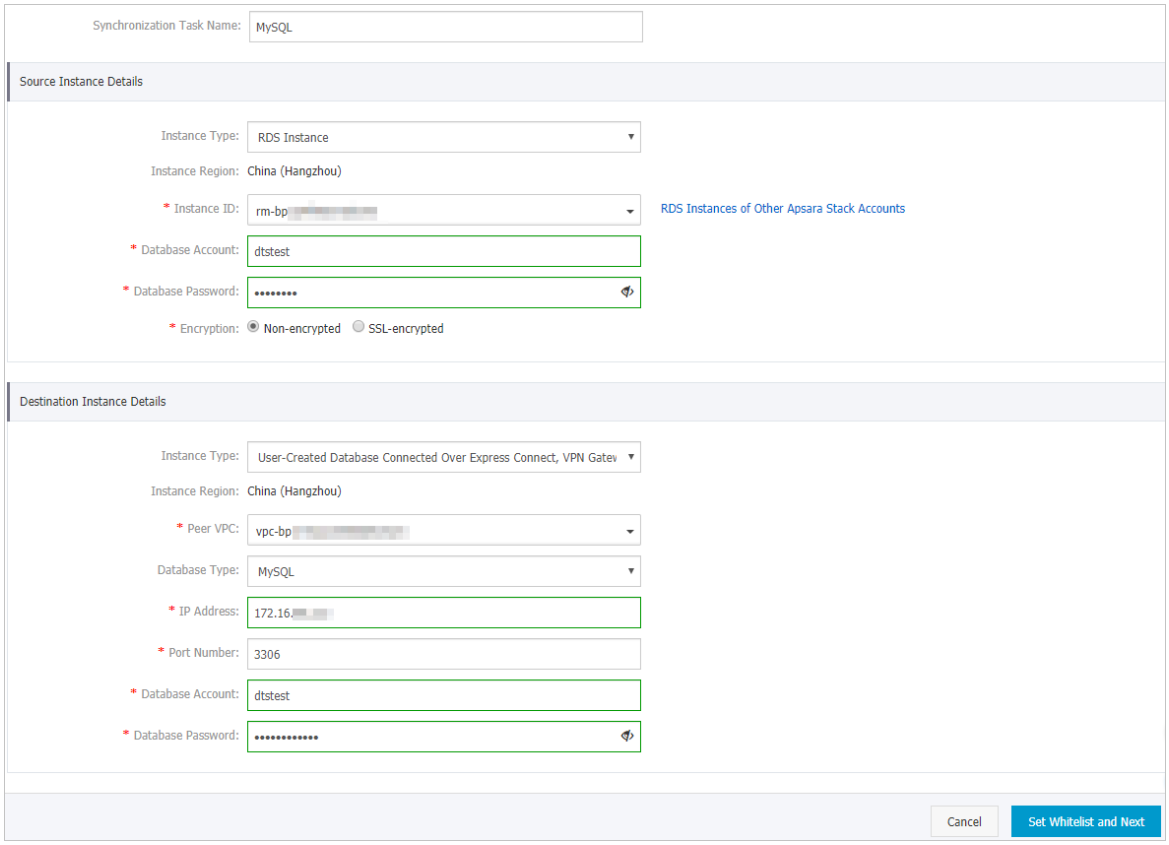
  RENAME TABLE operations may cause data inconsistency between the source and destination databases. For example, if only Table A is selected as the object to synchronize and is renamed Table B, Table B cannot be synchronized to the destination database. To prevent this situation, you can select the entire database where Table A is located as the object to synchronize when you configure the data synchronization task.

Procedure

1. Purchase a data synchronization instance. For more information, see Purchase procedure.

   > **Note** Select **MySQL** for both the source instance and the destination instance. Select **One-Way Synchronization** as the synchronization topology.

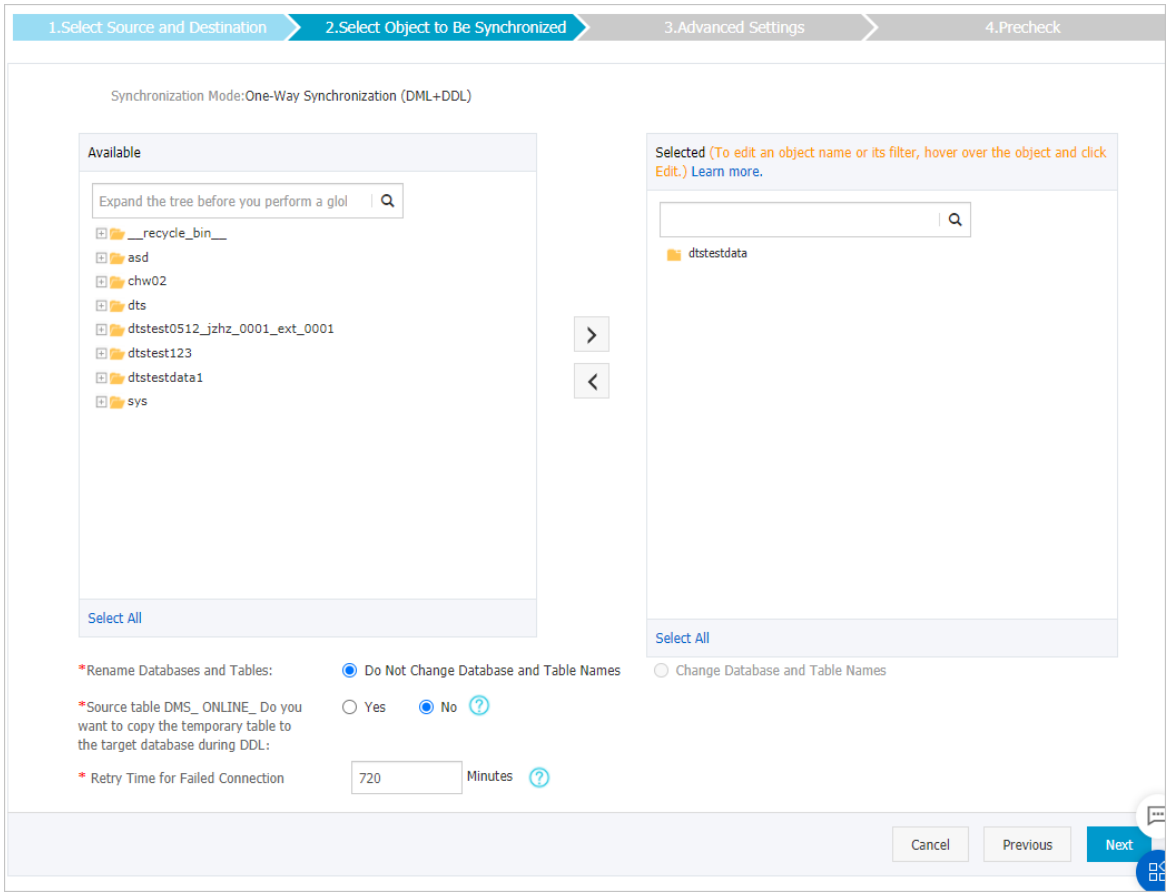
2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. In the upper part of the **Synchronization Tasks** page, select the region where the data synchronization instance resides.
5. Find the data synchronization instance and click **Configure Task** in the Actions column.
6. Configure the source and destination instances.



Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the source RDS instance.
	Database Account	Enter the database account of the source RDS instance. **Note** If the database type of the source RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the source database account.

Section	Parameter	Description
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.
Destination Instance Details	Instance Type	Select **User-Created Database Connected over Express Connect, VPN Gateway, or Smart Access Gateway**.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Peer VPC	Select the ID of the VPC that is connected to the self-managed MySQL database.
	Database Type	The value of this parameter is set to **MySQL** and cannot be changed.
	IP Address	Enter the server IP address of the self-managed MySQL database.
	Port Number	Enter the service port number of the self-managed MySQL database. The default port number is **3306**.
	Database Account	Enter the account of the self-managed MySQL database. **Note** The database account must have the SELECT permission on the objects to be synchronized, the REPLICATION CLIENT permission, the REPLICATION SLAVE permission, and the SHOW VIEW permission.
	Database Password	Enter the password of the destination database account.

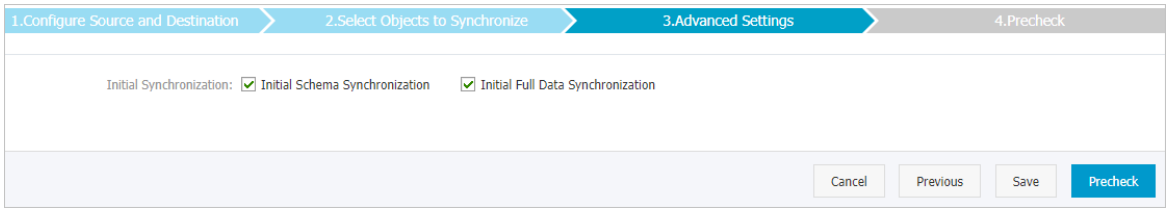
7. In the lower-right corner of the page, click **Set Whitelist and Next**.
8. Select the synchronization policy and the objects to synchronize.



Setting	Description
Select the objects to be synchronized	Select one or more objects from the **Available** section and click the icon to add the objects to the **Selected** section. You can select tables or databases as the objects to synchronize. **Note** ◦ If you select a database as the object to synchronize, all schema changes in the database are synchronized to the destination database. ◦ By default, after an object is synchronized to the destination database, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Rename an object to be synchronized.
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.

Setting	Description
Replicate Temporary Tables When DMS Performs DDL Operations	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ○ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ○ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. In the lower-right corner of the page, click **Next**.
10. Select the initial synchronization types.



Note

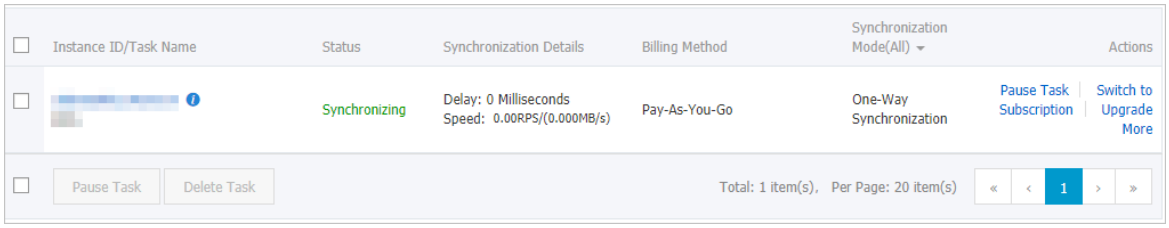
- During initial synchronization, DTS synchronizes the schemas and data of required objects from the source instance to the destination instance. The schemas and data are the basis for subsequent incremental synchronization.
- Initial synchronization includes initial schema synchronization and initial full data synchronization. In most cases, you need to select both **Initial Schema Synchronization** and **Initial Full Data Synchronization**.

11. In the lower-right corner of the page, click **Precheck**.

> **Note**
>
> - Before you can start the data synchronization task, DTS performs a precheck. You can start the data synchronization task only after the task passes the precheck.
> - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
>   - After you troubleshoot the issues based on the causes, you can run a precheck again.
>   - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

12. Close the **Precheck** dialog box after the following message is displayed: **Precheck Passed**. Then, the data synchronization task starts.
13. Wait until initial synchronization is complete and the data synchronization task enters the **Synchronizing** state.

    You can view the state of the data synchronization task on the **Synchronization Tasks** page.



7.3. Synchronize data from MySQL to other databases

7.3.1. Synchronize data from an ApsaraDB RDS for MySQL instance to a MaxCompute project

MaxCompute (formerly known as ODPS) is a fast and fully managed computing platform for large-scale data warehousing. MaxCompute can process exabytes of data. This topic describes how to synchronize data from an ApsaraDB RDS for MySQL instance to a MaxCompute project by using Data Transmission Service (DTS).

Prerequisites

The following operations are performed:

- Activate MaxCompute and DataWorks.
- Create a MaxCompute project.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- Only tables can be selected as the objects to be synchronized.
- We recommend that you do not use gh-ost or pt-online-schema-change to perform DDL operations on objects during data synchronization. Otherwise, data synchronization may fail.
- MaxCompute does not support the PRIMARY KEY constraint. If network errors occur, DTS may synchronize duplicate data records to MaxCompute.

Supported source database types

You can use DTS to synchronize data from the following types of MySQL databases:

- Self-managed database that is hosted on Elastic Compute Service (ECS)
- Self-managed database that is connected over Express Connect, VPN Gateway, or Smart Access Gateway
- Self-managed database that is connected over Database Gateway
- ApsaraDB RDS for MySQL instance that is owned by the same Alibaba Cloud account as the MaxCompute project or a different Alibaba Cloud account from the MaxCompute project

This topic uses an **ApsaraDB RDS for MySQL instance** as an example to describe how to configure a data synchronization task. You can also follow the procedure to configure data synchronization tasks for other types of MySQL databases.

> **Note** If your source database is a self-managed MySQL database, you must deploy the network environment for the source database. For more information, see Preparation overview.

SQL operations that can be synchronized

- DDL operation: ADD COLUMN

  > **Note** Only the following data types are supported: INTEGER, BIGINTEGER, IGINT, LONGSTRING, YEAR, TIME, DATA, TIMESTAMP, DATATIME, BYTE, BOOLEAN, DECIMAL, DOUBLE, and FLOAT.

- DML operations: INSERT, UPDATE, and DELETE

Synchronization process

1. Initial schema synchronization.

   DTS synchronizes the schemas of the required objects from the source database to MaxCompute. During initial schema synchronization, DTS adds the _base suffix to the end of the source table name. For example, if the name of the source table is customer, the name of the table in MaxCompute is customer_base.

2. Initial full data synchronization.

   DTS synchronizes the historical data of the table from the source database to the destination table in MaxCompute. For example, the customer table in the source database is synchronized to the customer_base table in MaxCompute. The data is the basis for subsequent incremental synchronization.

   > **Note** The destination table that is suffixed with _base is known as a full baseline table.

3. Incremental data synchronization.

   DTS creates an incremental data table in MaxCompute. The name of the incremental data table is suffixed with _log, for example, customer_log. Then, DTS synchronizes the incremental data that was generated in the source database to the incremental data table.

   > **Note** For more information, see Schema of an incremental data table.

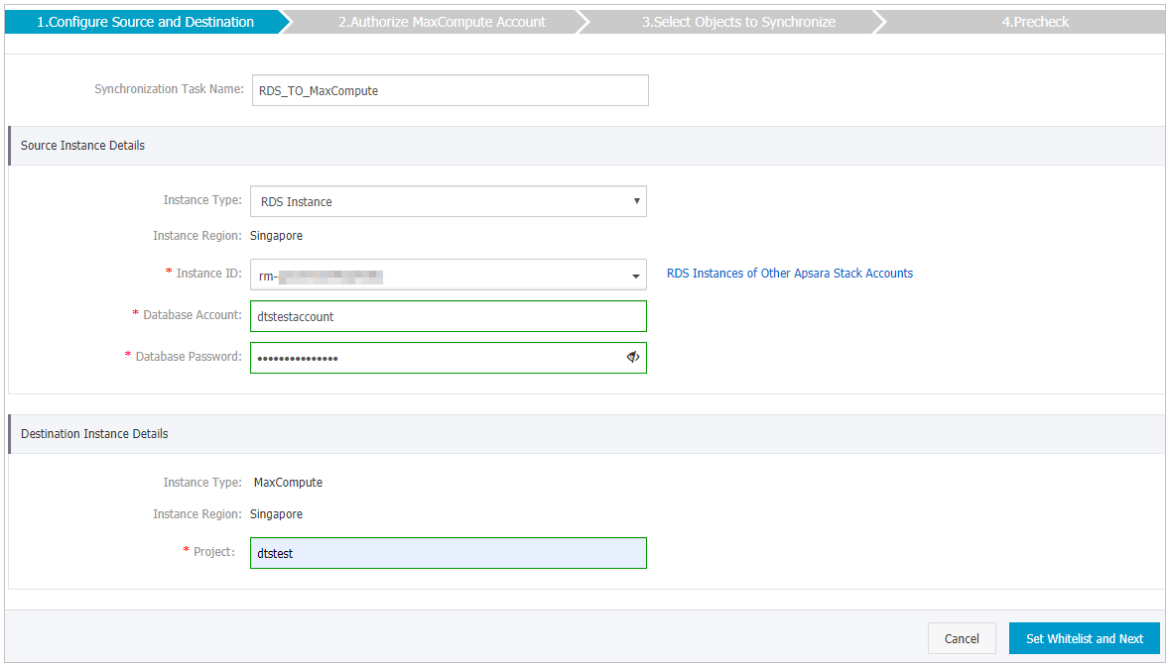
Procedure

> **Warning** To ensure that the synchronization account can be authorized, we recommend that you perform the following steps by using your Alibaba Cloud account.

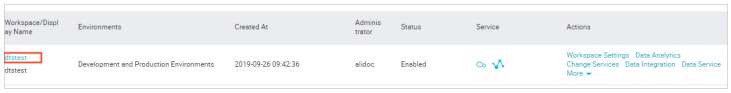
1. Purchase a data synchronization instance. For more information, see Purchase procedure.

   > **Note** On the buy page, set Source Instance to **MySQL**, set Destination Instance to **MaxCompute**, and set Synchronization Topology to **One-way Synchronization**.

2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the destination instance resides.
5. Find the data synchronization instance and click **Configure Synchronization Channel** in the Actions column.
6. Configure the source and destination instances.



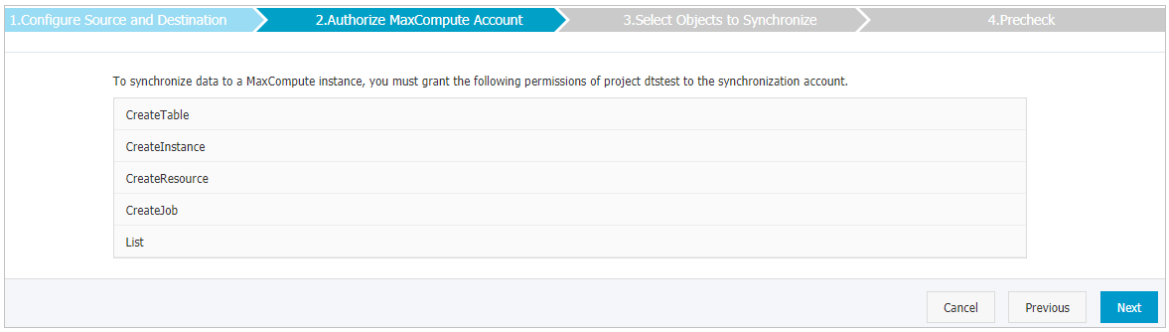
Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **RDS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the source RDS instance.
	Database Account	Enter the database account of the source RDS instance. **Note** If the database engine of the source RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.

Section	Parameter	Description
Destination Instance Details	Instance Type	This parameter is set to **MaxCompute** and cannot be changed.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Project	Enter the name of the MaxCompute **project**. You can search for a project on the Workspaces page in the DataWorks console. 

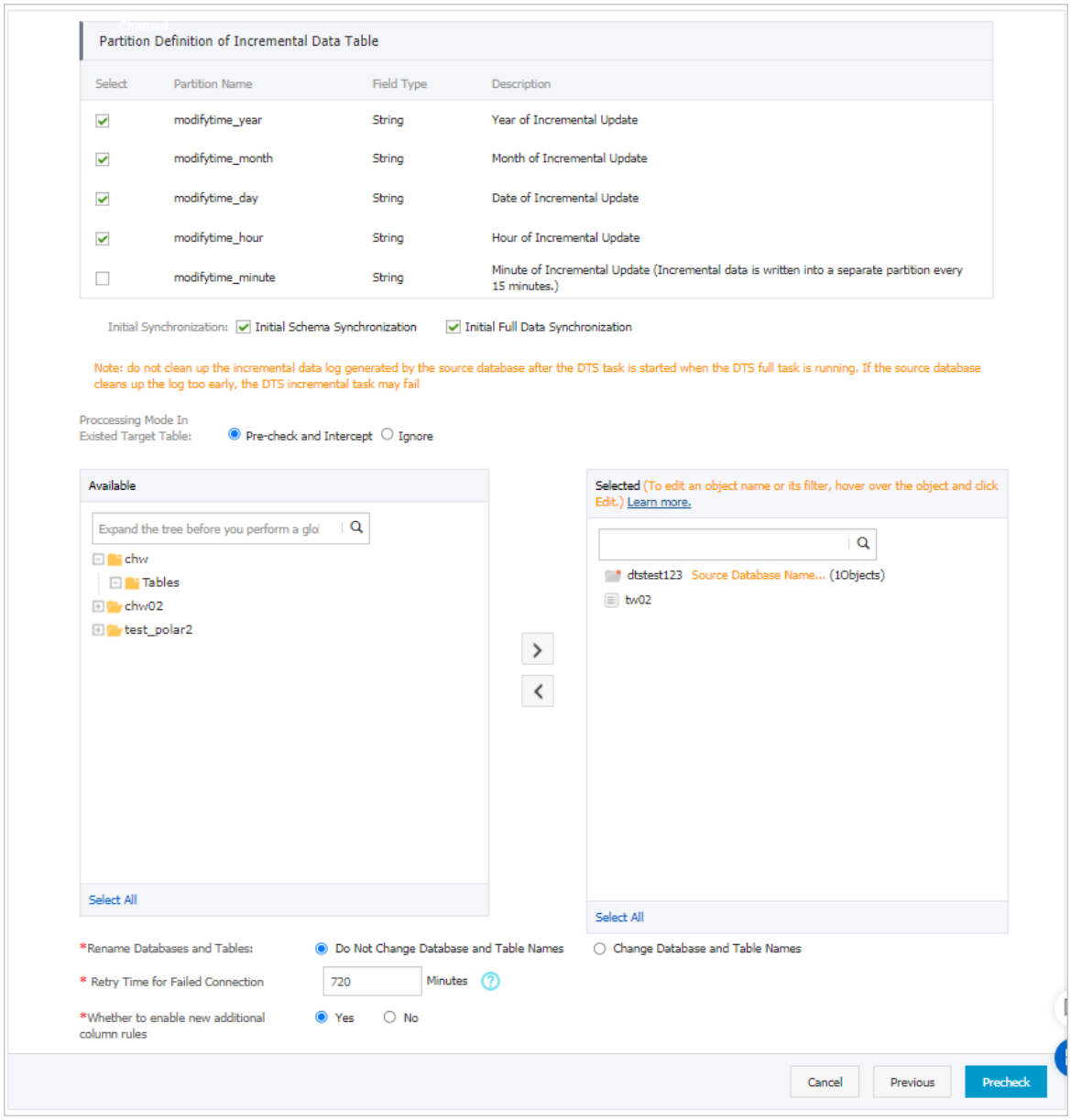
7. In the lower-right corner of the page, click **Set Whitelist and Next**.

> **Note** DTS adds the CIDR blocks of DTS servers to the whitelists of the RDS instance and the MaxCompute project. This ensures that DTS servers can connect to the source and destination instances.

8. In the lower-right corner of the page, click **Next**. In this step, the permissions on the MaxCompute project are granted to the synchronization account.



9. Select the synchronization policy and the objects to be synchronized.



Setting	Description
Partition Definition of Incremental Data Table	Select the partition names based on your business requirements. For more information, see Partition.
Initial Synchronization	Initial synchronization includes initial schema synchronization and initial full data synchronization. Select both **Initial Schema Synchronization** and **Initial Full Data Synchronization**. In this case, DTS synchronizes the schemas and historical data of the required objects and then synchronizes incremental data.

Setting	Description
Select the processing mode of conflicting tables	◦ **Pre-check and Intercept**: checks whether the destination database contains tables that have the same names as tables in the source database. If the source and destination databases do not contain identical table names, the precheck is passed. Otherwise, an error is returned during precheck and the data synchronization task cannot be started. **Note** You can use the object name mapping feature to rename the tables that are synchronized to the destination database. You can use this feature if the source and destination databases contain identical table names and the tables in the destination database cannot be deleted or renamed. For more information, see Rename an object to be synchronized. ◦ **Ignore**: skips the precheck for identical table names in the source and destination databases. **Warning** If you select **Ignore**, data consistency is not guaranteed and your business may be exposed to potential risks. ■ During initial data synchronization, DTS does not synchronize the data records that have the same primary keys as the data records in the destination database. This occurs if the source and destination databases have the same schema. However, DTS synchronizes these data records during incremental data synchronization. ■ If the source and destination databases have different schemas, initial data synchronization may fail. In this case, only some columns are synchronized or the data synchronization task fails.
Select the objects to be synchronized	Select one or more tables from the **Available** section and click the › icon to move the tables to the **Selected** section. **Note** ◦ You can select tables from multiple databases as the objects to be synchronized. ◦ By default, after an object is synchronized to the destination instance, the name of the object remains unchanged. You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Rename an object to be synchronized.

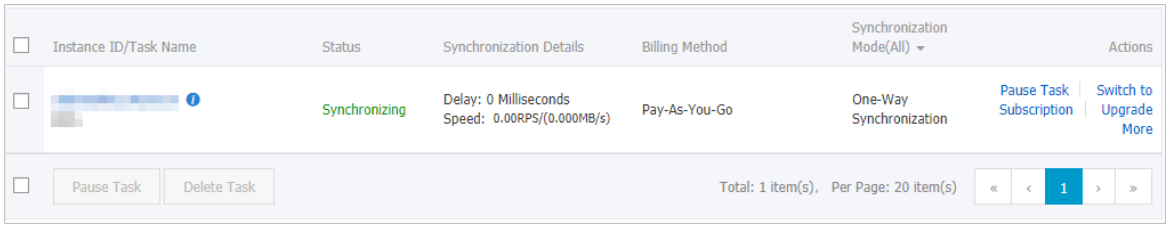
Setting	Description
Whether to enable the new naming rules for additional columns	After DTS synchronizes data to MaxCompute, DTS adds additional columns to the destination table. If the names of additional columns are the same as the names of existing columns in the destination table, data synchronization fails. Select **Yes** or **No** to specify **whether you want to enable the new naming rules for additional columns**. **Warning** Before you specify this parameter, check whether additional columns and existing columns in the destination table have name conflicts. For more information, see Naming rules for additional columns.
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Replicate Temporary Tables When DMS Performs DDL Operations	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ○ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ○ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

10. In the lower-right corner of the page, click **Precheck**.

> **Note**
>
> - Before you can start the data synchronization task, DTS performs a precheck. You can start the data synchronization task only after the task passes the precheck.
> - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
>   - After you troubleshoot the issues based on the causes, you can run a precheck again.
>   - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

11. Close the **Precheck** dialog box after the following message is displayed: **The precheck is passed.** Then, the data synchronization task starts.
12. Wait until initial synchronization is completed and the data synchronization task enters the **Synchronizing** state.

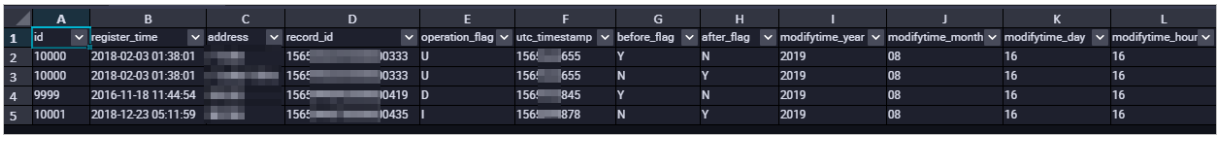
    You can view the state of the data synchronization task on the **Synchronization Tasks** page.



Schema of an incremental data table

> **Note** You must run the `set odps.sql.allow.fullscan=true;` command in MaxCompute to allow full table scan for the MaxCompute project.

DTS synchronizes incremental data that is generated in the source MySQL database to the incremental data table in MaxCompute. The incremental data table stores incremental data and specific metadata. The following figure shows the schema of an incremental data table.



	A	B	C	D	E	F	G	H	I	J	K	L
1	id	register_time	address	record_id	operation_flag	utc_timestamp	before_flag	after_flag	modifytime_year	modifytime_month	modifytime_day	modifytime_hour
2	10000	2018-02-03 01:38:01		1565 0333	U	1565 655	Y	N	2019	08	16	16
3	10000	2018-02-03 01:38:01		1565 0333	U	1565 655	N	Y	2019	08	16	16
4	9999	2016-11-18 11:44:54		1565 0419	D	1565 845	Y	N	2019	08	16	16
5	10001	2018-12-23 05:11:59		1565 0435	I	1565 878	N	Y	2019	08	16	16

> **Note** In the example, the `modifytime_year`, `modifytime_month`, `modifytime_day`, `modifytime_hour`, and `modifytime_minute` fields form the partition key. These fields are specified in the Select the synchronization policy and the objects to be synchronized step.

Schema of an incremental data table

Field	Description

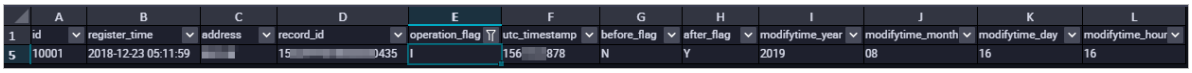
Field	Description
record_id	The ID of the incremental log entry. **Note** • The ID auto-increments for each new log entry. • If an UPDATE operation is performed, DTS generates two incremental log entries to record the pre-update and post-update values. The two incremental log entries have the same record ID.
operation_flag	The operation type. Valid values: • I: an INSERT operation • D: a DELETE operation • U: an UPDATE operation
utc_timestamp	The operation timestamp, in UTC. It is also the timestamp of the binary log file.
before_flag	Indicates whether the column values are pre-update values. Valid values: Y and N.
after_flag	Indicates whether the column values are post-update values. Valid values: Y and N.

Additional information about the before_flag and after_flag fields

The **before_flag** and **after_flag** fields of an incremental log entry are defined depending on the operation type.

- INSERT

  For an INSERT operation, the column values are the newly inserted record values (post-update values). The value of the before_flag field is N and the value of the after_flag field is Y.

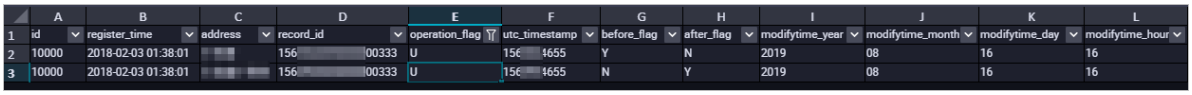


	A	B	C	D	E	F	G	H	I	J	K	L
1	id	register_time	address	record_id	operation_flag	utc_timestamp	before_flag	after_flag	modifytime_year	modifytime_month	modifytime_day	modifytime_hour
5	10001	2018-12-23 05:11:59	[redacted]	15[redacted]0435	I	156[redacted]878	N	Y	2019	08	16	16

- UPDATE

  DTS generates two incremental log entries for an UPDATE operation. The two incremental log entries have the same values for the record_id, operation_flag, and utc_timestamp fields.

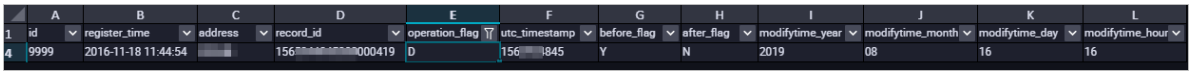
  The first log entry records the pre-update values, so the value of the before_flag field is Y and the value of the after_flag field is N. The second log entry records the post-update values, so the value of the before_flag field is N and the value of the after_flag field is Y.



	A	B	C	D	E	F	G	H	I	J	K	L
1	id	register_time	address	record_id	operation_flag	utc_timestamp	before_flag	after_flag	modifytime_year	modifytime_month	modifytime_day	modifytime_hour
2	10000	2018-02-03 01:38:01	[redacted]	156[redacted]00333	U	156[redacted]1655	Y	N	2019	08	16	16
3	10000	2018-02-03 01:38:01	[redacted]	156[redacted]00333	U	156[redacted]1655	N	Y	2019	08	16	16

- DELETE

  For a DELETE operation, the column values are the deleted record values (pre-update values). The value of the before_flag field is Y and the value of the after_flag field is N.



	A	B	C	D	E	F	G	H	I	J	K	L
1	id	register_time	address	record_id	operation_flag	utc_timestamp	before_flag	after_flag	modifytime_year	modifytime_month	modifytime_day	modifytime_hour
4	9999	2016-11-18 11:44:54	[redacted]	156[redacted]000419	D	156[redacted]1845	Y	N	2019	08	16	16

Merge a full baseline table and an incremental data table

After a data synchronization task is started, DTS creates a full baseline table and an incremental data table in MaxCompute. You can use SQL statements to merge the two tables. This allows you to obtain the full data at a specific time point.

This section describes how to merge data for a table named customer. The following figure shows the schema of the customer table.

	Field	Type	Null	Key	Default	Extra
1	id	int(11)	NO	PRI	null	
2	register_time	timestamp	YES		null	
3	address	varchar(32)	YES		null	

1. Create a table in MaxCompute based on the schema of the source table. The table is used to store the merged data.

   For example, you can obtain full data of the customer table at the `1565944878` time point. Run the following SQL statements to create the required table:

   ```
   CREATE TABLE `customer_1565944878` (
       `id` bigint NULL,
       `register_time` datetime NULL,
       `address` string);
   ```

   > **Note**
   > - You can use the ad-hoc query feature to run SQL statements. For more information, see (Optional) Use an ad-hoc query to run SQL statements.
   > - For more information about the data types that are supported by MaxCompute, see Data types.

2. Run the following SQL statements in MaxCompute to merge the full baseline table and the incremental data table and obtain full data at a specific time point:

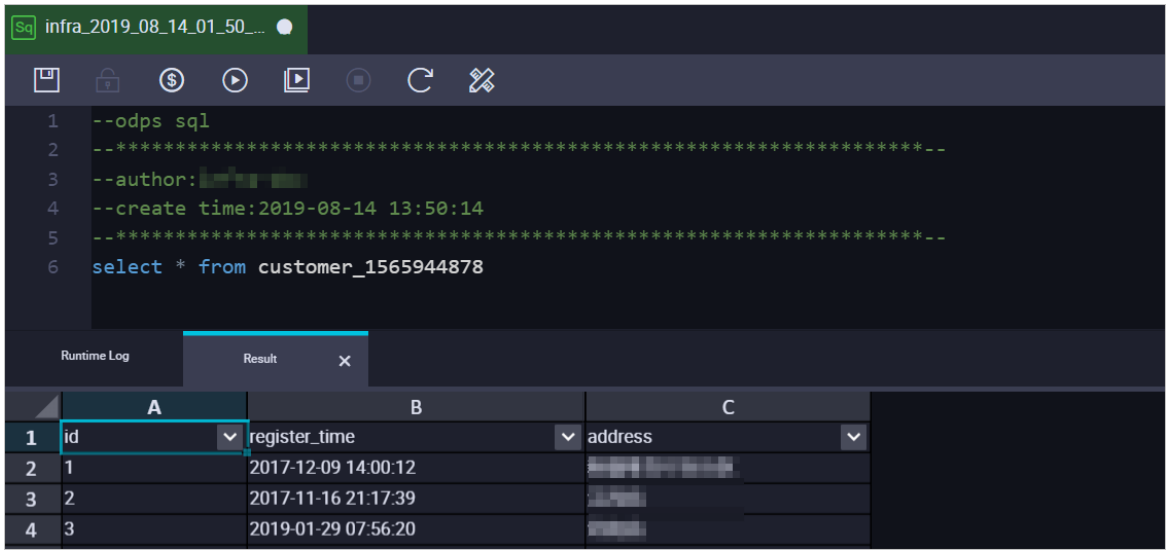
```
set odps.sql.allow.fullscan=true;
insert overwrite table <result_storage_table>
select <col1>,
       <col2>,
       <colN>
  from(
select row_number() over(partition by t.<primary_key_column>
 order by record_id desc, after_flag desc) as row_number, record_id, operation_flag, af
ter_flag, <col1>, <col2>, <colN>
  from(
select incr.record_id, incr.operation_flag, incr.after_flag, incr.<col1>, incr.<col2>,i
ncr.<colN>
  from <table_log> incr
 where utc_timestamp< <timestamp>
 union all
select 0 as record_id, 'I' as operation_flag, 'Y' as after_flag, base.<col1>, base.<col
2>,base.<colN>
  from <table_base> base) t) gt
where record_num=1
  and after_flag='Y'
```

> **Note**
>
> - <result_storage_table>: the name of the table that stores the merged data.
> - <col1>/<col2>/<colN>: the names of the columns in the table to be merged.
> - <primary_key_column>: the name of the primary key column in the table to be merged.
> - <table_log>: the name of the incremental data table.
> - <table_base>: the name of the full baseline table.
> - <timestamp>: the timestamp that is generated when full data is obtained.

Run the following SQL statements to obtain full data of the customer table at the `1565944878` time point:

```
set odps.sql.allow.fullscan=true;
insert overwrite table customer_1565944878
select id,
       register_time,
       address
  from(
select row_number() over(partition by t.id
 order by record_id desc, after_flag desc) as row_number, record_id, operation_flag, af
ter_flag, id, register_time, address
  from(
select incr.record_id, incr.operation_flag, incr.after_flag, incr.id, incr.register_tim
e, incr.address
  from customer_log incr
 where utc_timestamp< 1565944878
 union all
select 0 as record_id, 'I' as operation_flag, 'Y' as after_flag, base.id, base.register
_time, base.address
  from customer_base base) t) gt
 where gt.row_number= 1
   and gt.after_flag= 'Y';
```

3. Query the merged data from the customer_1565944878 table.



7.3.2. Synchronize data from an ApsaraDB RDS for MySQL instance to an AnalyticDB for MySQL cluster

is a real-time online analytical processing (RT-OLAP) service that is developed by Alibaba Cloud for online data analysis with high concurrency. AnalyticDB for MySQL can analyze petabytes of data from multiple dimensions at millisecond-level timing to provide data-driven insights into your business. This topic describes how to synchronize data from an ApsaraDB RDS for MySQL instance to an cluster by using Data Transmission Service (DTS). After you synchronize data, you can use AnalyticDB for MySQL to build internal business intelligence (BI) systems, interactive query systems, and real-time report systems.

Prerequisites

- The tables that you want to synchronize from the ApsaraDB RDS for MySQL instance contain primary keys.
- An cluster is created. For more information, see Create an cluster.
- The destination cluster has sufficient storage space.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- We recommend that you do not use gh-ost or pt-online-schema-change to perform data definition language (DDL) operations on the required objects during data synchronization. Otherwise, data may fail to be synchronized.
- Due to the limits of , if the disk space usage of the nodes in an cluster reaches 80%, the cluster is locked. We recommend that you estimate the required disk space based on the objects that you want to synchronize. You must ensure that the destination cluster has sufficient storage space.
- Prefix indexes cannot be synchronized. If the source database contains prefix indexes, data may fail to be synchronized.

SQL operations that can be synchronized

- DDL operations: CREATE TABLE, DROP TABLE, RENAME TABLE, TRUNCATE TABLE, ADD COLUMN, DROP COLUMN, and MODIFY COLUMN
- DML operations: INSERT, UPDATE, and DELETE

> **Note** If the data type of a field in the source table is changed during data synchronization, an error message is generated and the data synchronization task is stopped. You can submit a or troubleshoot the issue. For more information, see Troubleshoot the synchronization failure that occurs due to field type changes.

Permissions required for database accounts

Database	Required permissions
ApsaraDB RDS for MySQL	The SELECT permission on the objects to be synchronized, the REPLICATION CLIENT permission, the REPLICATION SLAVE permission, and the SHOW VIEW permission
	The read and write permissions on the objects to be synchronized

Data type mappings

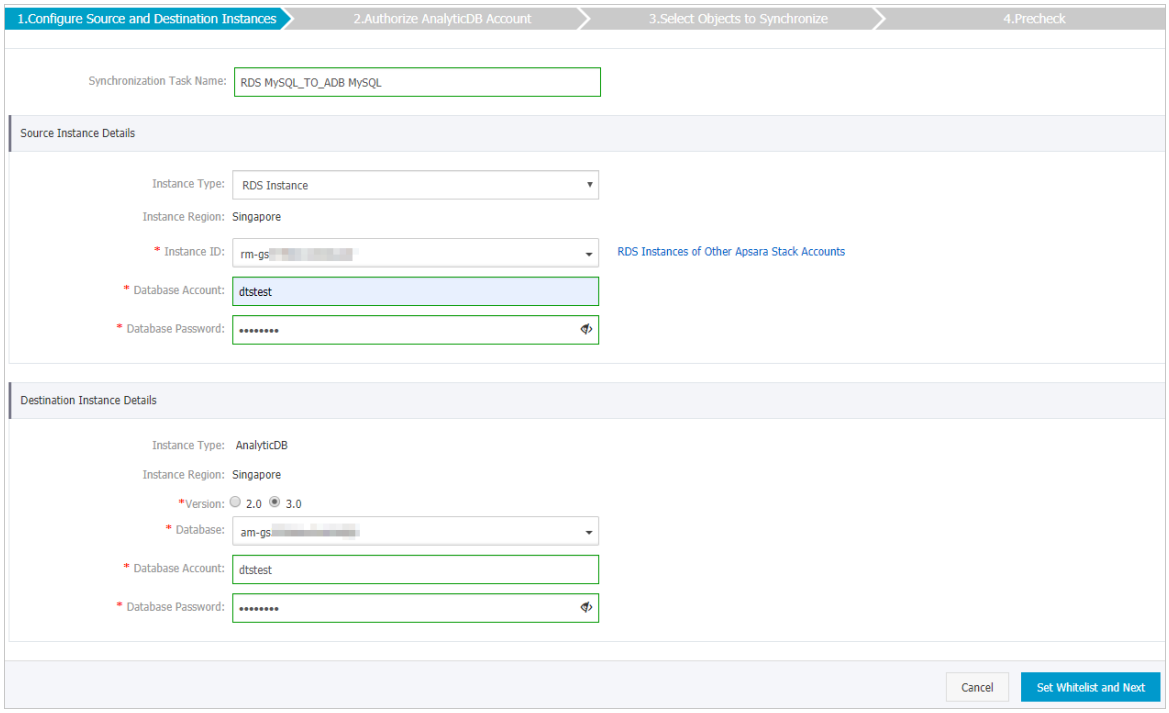
The data types of ApsaraDB RDS for MySQL and do not have one-to-one correspondence. During initial schema synchronization, DTS converts the data types of the source database into those of the destination database. For more information, see Data type mappings for schema synchronization.

Procedure

1. Purchase a data synchronization instance. For more information, see Purchase procedure.

   > **Note** On the buy page, set Source Instance to **MySQL**, set Target Instance to **AnalyticDB MySQL**, and set Synchronization Topology to **One-Way Synchronization**.

2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. In the upper part of the **Synchronization Tasks** page, select the region where the data synchronization instance resides.
5. Find the data synchronization instance and click **Configure Task** in the Actions column.
6. Configure the source and destination databases.

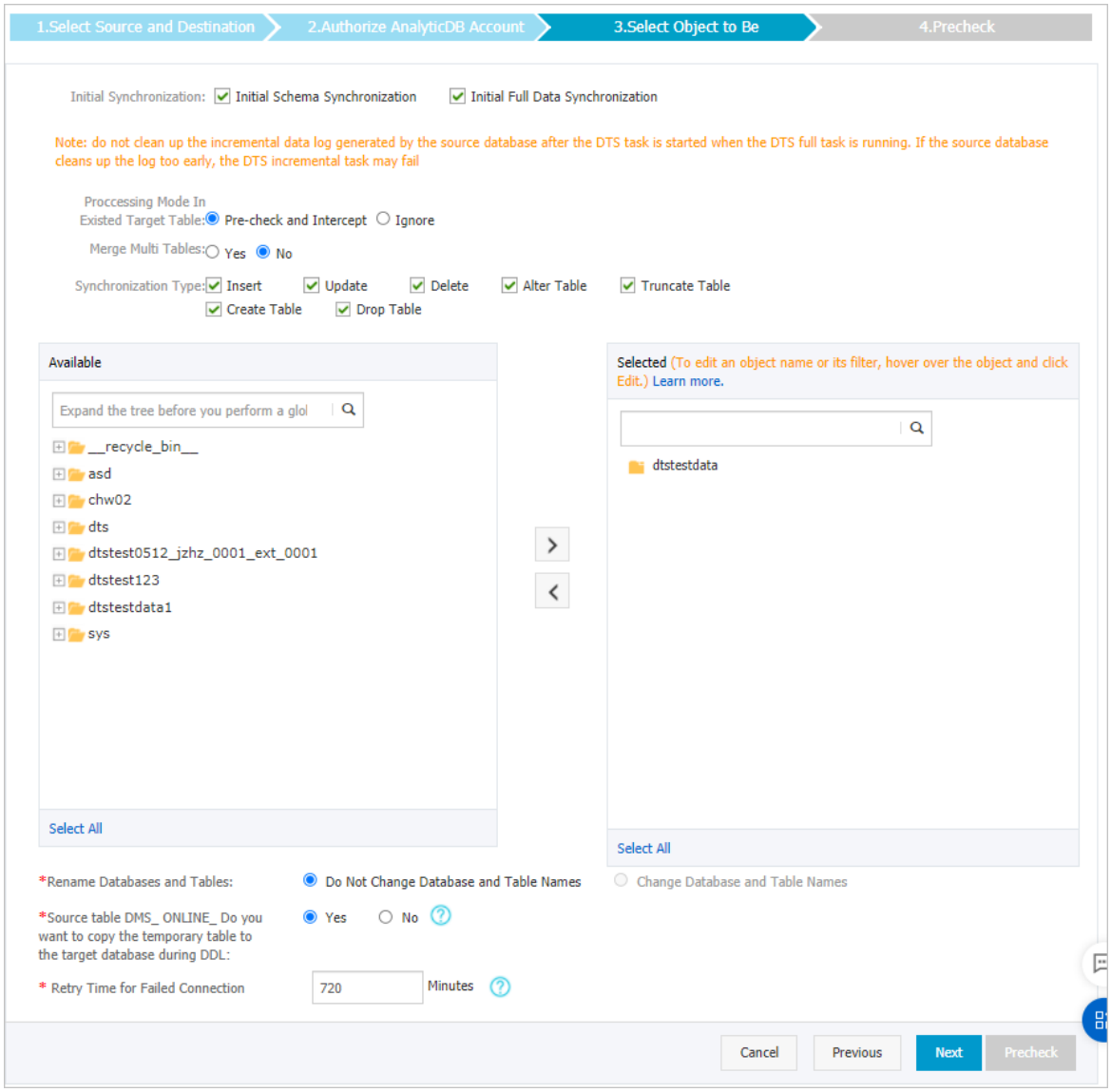


Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
	Instance Type	Select **RDS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.

Section	Parameter	Description
Source Instance Details	Instance ID	Select the ID of the source RDS instance.
	Database Account	Enter the database account of the source RDS instance. For information about the permissions that are required for the account, see Permissions required for database accounts. **Note** If the database engine of the source RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** or **database password**.
	Database Password	Enter the password of the database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Notice** The **Encryption** parameter is available only for regions in the Chinese mainland and the China (Hong Kong) region.
Destination Instance Details	Instance Type	The value of this parameter is set to **AnalyticDB** and cannot be changed.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Version	Select **3.0**.
	Database	Select the ID of the destination cluster.
	Database Account	Enter the database account of the cluster. For information about the permissions that are required for the account, see Permissions required for database accounts.
	Database Password	Enter the password of the database account.

7.

8. Select the synchronization policy and the objects to be synchronized.



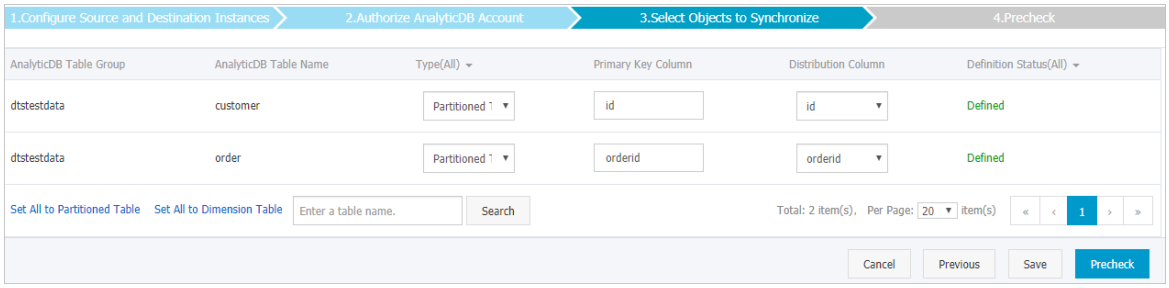
Parameter	Description
Initial Synchronization	You must select both **Initial Schema Synchronization** and **Initial Full Data Synchronization** in most cases. After the precheck, DTS synchronizes the schemas and data of the required objects from the source instance to the destination cluster. The schemas and data are the basis for subsequent incremental synchronization.

Parameter	Description
Processing Mode In Existed Target Table	◦ **Pre-check and Intercept**: checks whether the destination database contains tables that have the same names as tables in the source database. If the source and destination databases do not contain identical table names, the precheck is passed. Otherwise, an error is returned during precheck and the data synchronization task cannot be started. **Note** You can use the object name mapping feature to change the names of the tables that are synchronized to the destination database. You can use this feature if the source and destination databases contain identical table names and the tables in the destination database cannot be deleted or renamed. For more information, see Rename an object to be synchronized. ◦ **Ignore**: skips the precheck for identical table names in the source and destination databases. **Warning** If you select **Ignore**, data consistency is not guaranteed and your business may be exposed to potential risks. ■ If the source and destination databases have the same schema, DTS does not synchronize data records that have the same primary keys as data records in the destination database. ■ If the source and destination databases have different schemas, initial data synchronization may fail. In this case, only specific columns are synchronized or the data synchronization task fails.
Merge Multi Tables	◦ If you select **Yes**, DTS adds the `__dts_data_source` column to each table to record data sources. In this case, DDL operations cannot be synchronized. ◦ **No** is selected by default. In this case, DDL operations can be synchronized. **Note** You can merge the data source columns based on tasks rather than tables. To merge only the data source columns of specific tables, you can create two data synchronization tasks.
Synchronization Type	Select the types of operations that you want to synchronize based on your business requirements. All operation types are selected by default. For more information, see SQL operations that can be synchronized.

Parameter	Description
Select the objects to be synchronized	Select one or more objects from the **Available** section and click the icon to move the objects to the **Selected** section. You can select tables or databases as the objects to be synchronized. **Note** ◦ If you select a database as the object to be synchronized, all schema changes in the database are synchronized to the destination database. ◦ If you select a table as the object to be synchronized, only the ADD COLUMN operations that are performed on the table are synchronized to the destination database. ◦ By default, after an object is synchronized to the destination cluster, the name of the object remains unchanged. You can use the object name mapping feature to change the names of the objects that are synchronized to the destination cluster. For more information, see Rename an object to be synchronized.
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Replicate Temporary Tables When DMS Performs DDL Operations	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ◦ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ◦ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.

Parameter	Description
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. In the lower-right corner of the page, click **Next**.
10. Specify a type for the tables that you want to synchronize to the destination database.



Note After you select **Initial Schema Synchronization**, you must specify the **type**, **primary key column**, and **partition key column** for the tables that you want to synchronize to . For more information, see CREATE TABLE.

11. In the lower-right corner of the page, click **Precheck**.

Note

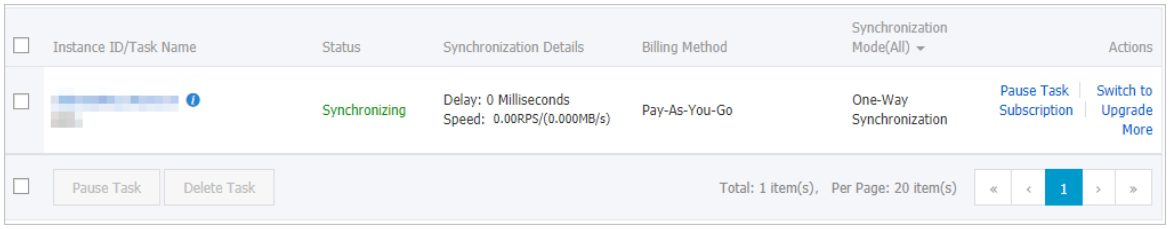
- Before you can start the data synchronization task, DTS performs a precheck. You can start the data synchronization task only after the task passes the precheck.
- If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
  - After you troubleshoot the issues based on the causes, you can run a precheck again.
  - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

12. Close the **Precheck** dialog box after the following message is displayed: **Precheck Passed**. Then,

the data synchronization task starts.

13. Wait until initial synchronization is complete and the data synchronization task enters the **Synchronizing** state.

    You can view the state of the data synchronization task on the **Synchronization Tasks** page.



Troubleshoot the synchronization failure that occurs due to field type changes

If the data type of a field in the source table is changed during data synchronization, an error message is generated and the data synchronization task is stopped. You can submit a or perform the following steps to troubleshoot the issue.

1. Create a table in the destination cluster based on the schema of source table that fails to be synchronized. For example, if a table named customer (Table A) fails to be synchronized, you can create a table named customer_new (Table B) in the destination cluster. Make sure that Table B has the same schema as Table A.
2. Run the INSERT INTO SELECT command to copy the data of Table A and insert the data into Table B. This ensures that the data of the two tables is consistent.
3. Rename or delete Table A. Then, change the name of Table B to customer.
4. Restart the data synchronization task in the DTS console.

7.3.3. Synchronize data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to a self-managed Kafka cluster

Kafka is a distributed message queue service that features high throughput and high scalability. Kafka is widely used for big data analytics such as log collection, data aggregation, streaming processing, and online and offline analysis. It is important for the big data ecosystem. This topic describes how to synchronize data from a self-managed MySQL database connected over Express Connect, VPN Gateway, or Smart Access Gateway to a self-managed Kafka cluster by using Data Transmission Service (DTS). The data synchronization feature allows you to extend message processing capabilities.

Prerequisites

- A Kafka cluster is created and the Kafka version is 0.10.1.0 to 2.7.0.
- The version of the self-managed MySQL database is 5.1, 5.5, 5.6, 5.7, or 8.0.
- The self-managed MySQL database is connected to Alibaba Cloud VPC over Express Connect, VPN Gateway, or Smart Access Gateway. For more information, see Connect an on-premises database to

DTS by using CEN.

Precautions

- DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.
- The source database must have PRIMARY KEY or UNIQUE constraints and all fields must be unique. Otherwise, the destination database may contain duplicate data records.

Limits

- Only tables can be selected as the objects to synchronize.
- DTS does not synchronize the data in a renamed table to the destination Kafka cluster. This applies if the new table name is not included in the objects to synchronize. If you want to synchronize the data in a renamed table to the destination Kafka cluster, you must **reselect the objects to be synchronized**. For more information, see Add an object to a data synchronization task.

Supported synchronization topologies

- One-way one-to-one synchronization
- One-way one-to-many synchronization
- One-way many-to-one synchronization
- One-way cascade synchronization

Before you begin

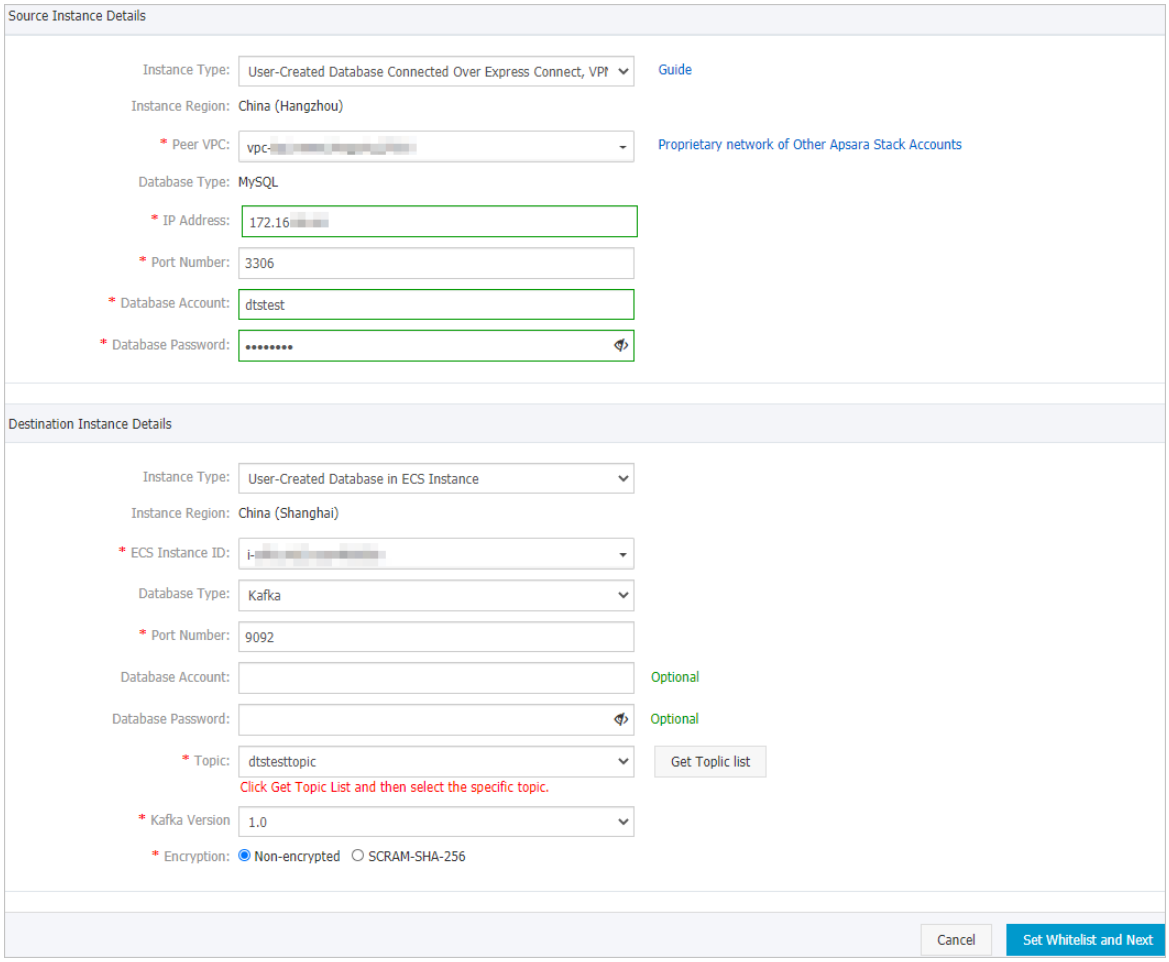
Before you configure the data synchronization task, you must create a database account and configure binary logging. For more information, see Create an account for a user-created MySQL database and configure binary logging.

Procedure

1. Purchase a data synchronization instance. For more information, see Purchase procedure.

   > **Note** On the buy page, set Source Instance to **MySQL**, Destination Instance to **Kafka**, and Synchronization Topology to **One-Way Synchronization**.

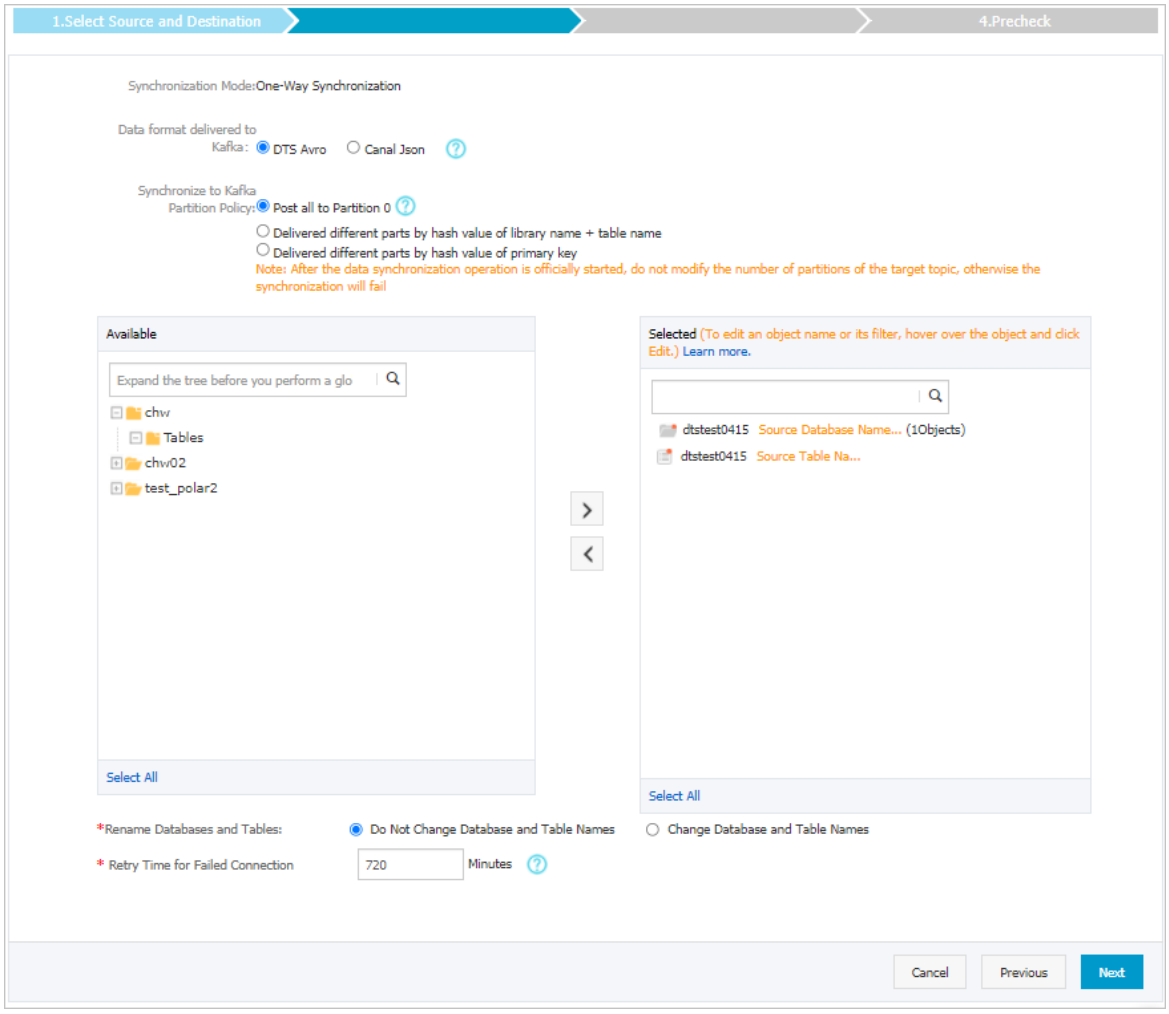
2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the destination instance resides.
5. Find the data synchronization instance and click **Configure Synchronization Channel** in the Actions column.
6. Configure the source and destination instances.



Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **User-Created Database Connected over Express Connect, VPN Gateway, or Smart Access Gateway**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	Peer VPC	Select the ID of the VPC that is connected to the self-managed MySQL database.
	Database Type	The value of this parameter is set to **MySQL** and cannot be changed.
	IP Address	Enter the server IP address of the self-managed MySQL database.
	Port Number	Enter the service port number of the self-managed MySQL database. The default port number is 3306.

Section	Parameter	Description
	Database Account	Enter the account of the self-managed MySQL database. The account must have the SELECT permission on the required objects, the REPLICATION CLIENT permission, the REPLICATION SLAVE permission, and the SHOW VIEW permission.
	Database Password	Enter the password of the source database account.
Destination Instance Details	Instance Type	Select an instance type based on the deployment of the Kafka cluster. In this example, select **User-Created Database in ECS Instance**. **Note** If you select other instance types, you must deploy the network environment for the Kafka cluster. For more information, see Preparation overview.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	ECS Instance ID	Select the ID of the Elastic Compute Service (ECS) instance that hosts the Kafka cluster.
	Database Type	Select **Kafka**.
	Port Number	Enter the service port number of the Kafka cluster. The default port number is 9092.
	Database Account	Enter the username that is used to log on to the Kafka cluster. If no authentication is enabled for the Kafka cluster, you do not need to enter the username.
	Database Password	Enter the password of the username. If no authentication is enabled for the Kafka cluster, you do not need to enter the password.
	Topic	Click **Get Topic List** and select a topic name from the drop-down list.
	Kafka Version	Select the version of the destination Kafka cluster.
	Encryption	Select **Non-encrypted** or **SCRAM-SHA-256** based on your business and security requirements.

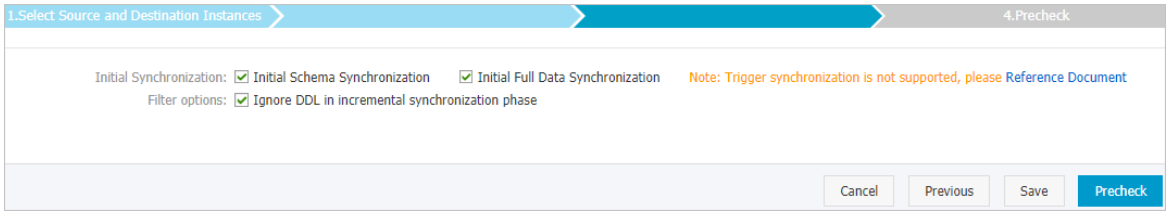
7. In the lower-right corner of the page, click **Set Whitelist and Next**.
8. Select the objects to synchronize.



Parameter	Description
Data Format in Kafka	The data that is synchronized to the Kafka cluster is stored in the Avro or Canal JSON format. For more information, see Data formats of a Kafka cluster.
Policy for Shipping Data to Kafka Partitions	The policy used to synchronize data to Kafka partitions. Select a policy based on your business requirements. For more information, see Specify the policy for synchronizing data to Kafka partitions.
Select the objects to synchronize	Select one or more tables from the **Available** section and click the icon to add the tables to the **Selected** section. **Note** DTS maps the table names to the topic name that you select in Step 6. If you want to rename the topic, you can use the object name mapping feature. For more information, see Rename an object to be synchronized.

Parameter	Description
Rename Databases and Tables	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.
Retry Time for Failed Connections	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. In the lower-right corner of the page, click **Next**.
10. Configure initial synchronization.



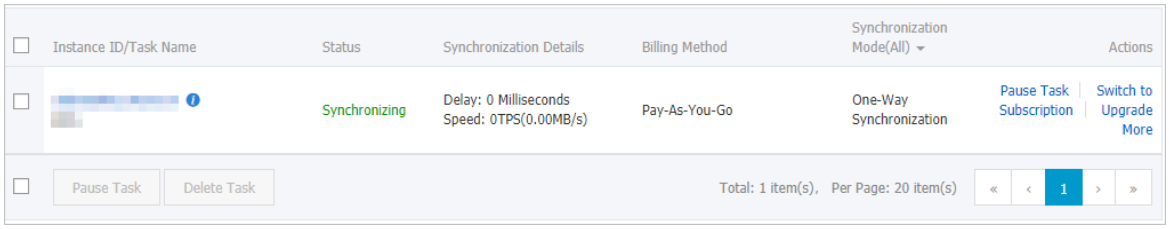
Setting	Description
Initial Synchronization	Select both **Initial Schema Synchronization** and **Initial Full Data Synchronization**. DTS synchronizes the schemas and historical data of the required objects and then synchronizes incremental data.
Filter options	**Ignore DDL in incremental synchronization phase** is selected by default. In this case, DTS does not synchronize DDL operations that are performed on the source database during incremental data synchronization.

11. In the lower-right corner of the page, click **Precheck**.

> **Note**
>
> - Before you can start the data synchronization task, DTS performs a precheck. You can start the data synchronization task only after the task passes the precheck.
> - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
>   - After you troubleshoot the issues based on the causes, you can run a precheck again.
>   - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

12. Close the **Precheck** dialog box after the following message is displayed: **Precheck Passed**. Then, the data synchronization task starts.

    You can view the state of the data synchronization task on the **Synchronization Tasks** page.



7.3.4. Synchronize data from an ApsaraDB RDS for MySQL instance to an AnalyticDB for PostgreSQL instance

This topic describes how to synchronize data from an ApsaraDB RDS for MySQL instance to an instance by using Data Transmission Service (DTS). The data synchronization feature provided by DTS allows you to transfer and analyze data with ease.

Prerequisites

- The tables that you want to synchronize from the ApsaraDB RDS for MySQL instance contain primary keys.
- The destination instance is created. For more information, see
- Create an AnalyticDB for PostgreSQL instance.

Precautions

DTS uses read and write resources of the source and destination databases during initial full data synchronization. This may increase the loads of the database servers. If the database performance is unfavorable, the specification is low, or the data volume is large, database services may become unavailable. For example, DTS occupies a large amount of read and write resources in the following cases: a large number of slow SQL queries are performed on the source database, the tables have no primary keys, or a deadlock occurs in the destination database. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination databases. We recommend that you synchronize data during off-peak hours. For example, you can synchronize data when the CPU utilization of the source and destination databases is less than 30%.

Limits

- You can select only tables as the objects to be synchronized.
- DTS does not synchronize the following types of data: BIT, VARBIT, GEOMETRY, ARRAY, UUID, TSQUERY, TSVECTOR, and TXID_SNAPSHOT.
- Prefix indexes cannot be synchronized. If the source database contains prefix indexes, data may fail to be synchronized.
- We recommend that you do not use gh-ost or pt-online-schema-change to perform DDL operations on objects during data synchronization. Otherwise, data synchronization may fail.

SQL operations that can be synchronized

- DML operations: INSERT, UPDATE, and DELETE
- DDL operation: ADD COLUMN

> **Note** The CREATE TABLE operation is not supported. To synchronize data from a new table, you must add the table to the selected objects. For more information, see Add an object to a data synchronization task.

Supported synchronization topologies

- One-way one-to-one synchronization
- One-way one-to-many synchronization
- One-way many-to-one synchronization

Term mappings

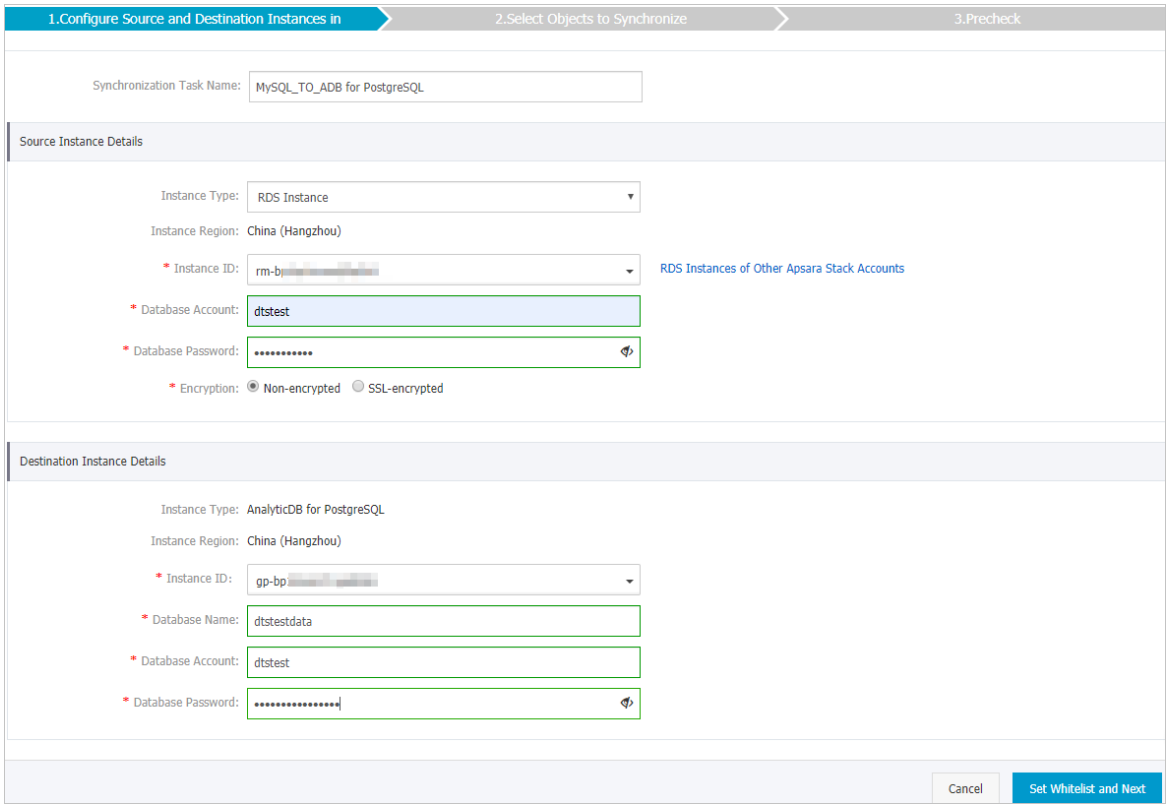
MySQL	
Database	Schema
Table	Table

Procedure

1. Purchase a data synchronization instance. For more information, see Purchase procedure.

> **Note** On the buy page, set Source Instance to **MySQL**, set Target Instance to **AnalyticDB for PostgreSQL**, and set Synchronization Topology to **One-Way Synchronization**.

2. Log on to the DTS console.
3. In the left-side navigation pane, click **Data Synchronization**.
4. At the top of the **Synchronization Tasks** page, select the region where the destination instance resides.
5. Find the data synchronization instance and click **Configure Synchronization Channel** in the Actions column.
6. Configure the source and destination instances.



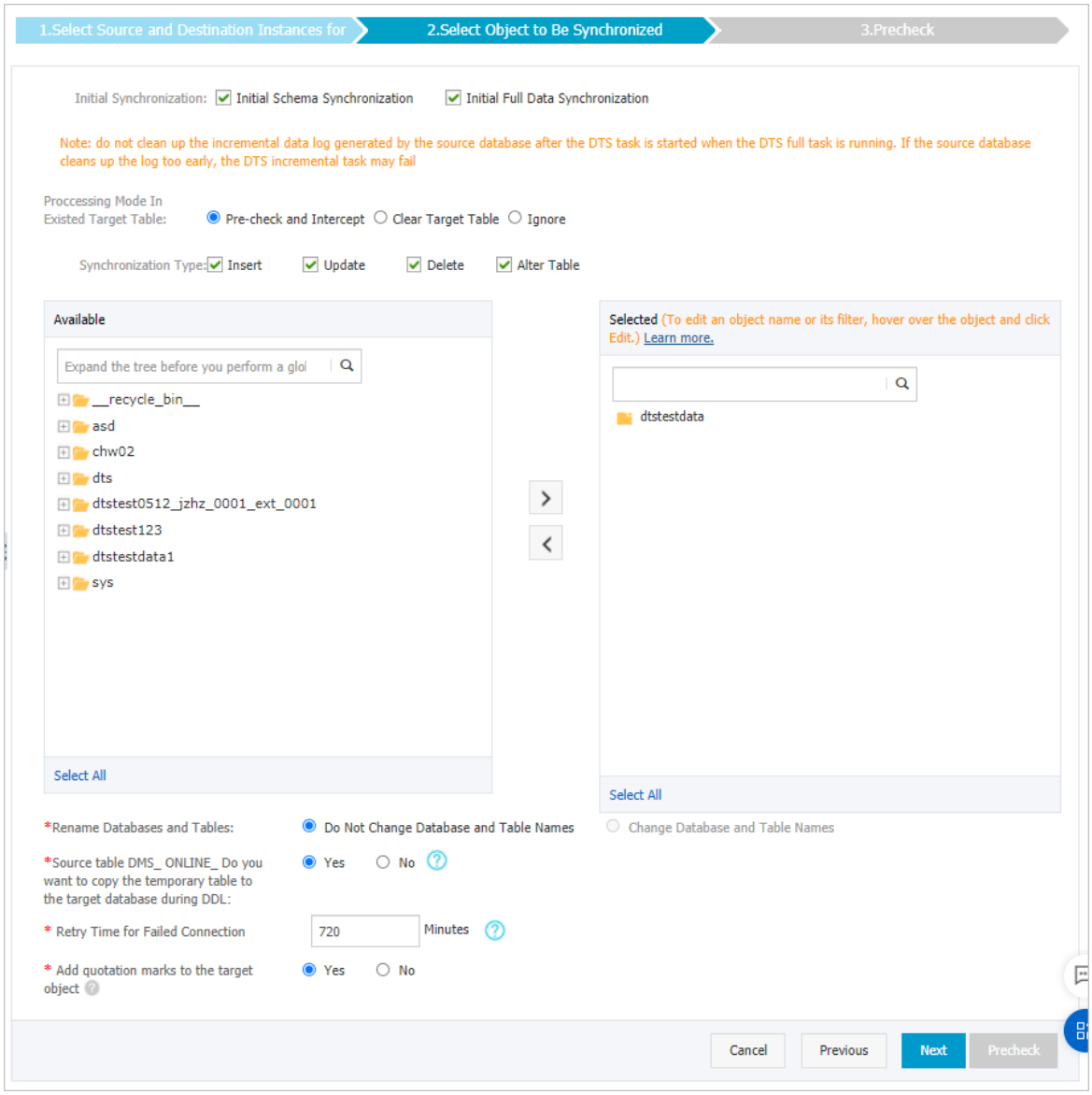
Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
	Instance Type	Select **RDS Instance**.
	Instance Region	The source region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the source RDS instance.

Section	Parameter	Description
Source Instance Details	Database Account	Enter the database account of the ApsaraDB RDS for MySQL instance. **Note** If the database engine of the source RDS instance is **MySQL 5.5** or **MySQL 5.6**, you do not need to configure the **database account** and **database password**.
	Database Password	Enter the password of the source database account.
	Encryption	Select **Non-encrypted** or **SSL-encrypted**. If you want to select **SSL-encrypted**, you must enable SSL encryption for the RDS instance before you configure the data synchronization task. For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance. **Note** The **Encryption** parameter is available only for regions in mainland China and the China (Hong Kong) region.
Destination Instance Details	Instance Type	The value of this parameter is set to **AnalyticDB for PostgreSQL** and cannot be changed.
	Instance Region	The destination region that you selected on the buy page. You cannot change the value of this parameter.
	Instance ID	Select the ID of the destination instance.
	Database Name	Enter the name of the destination database.
	Database Account	Enter the **initial account** of the instance. For more information, see Create a database account. **Note** You can also enter an account that has the RDS_SUPERUSER permission. For more information, see Manage users and permissions.
	Database Password	Enter the password of the destination database account.

7. In the lower-right corner of the page, click **Set Whitelist and Next**.

Note DTS adds the CIDR blocks of DTS servers to the whitelists of the ApsaraDB RDS for MySQL instance and the instance. This ensures that DTS servers can connect to the source RDS instance.

8. Select the synchronization policy and the objects to be synchronized.

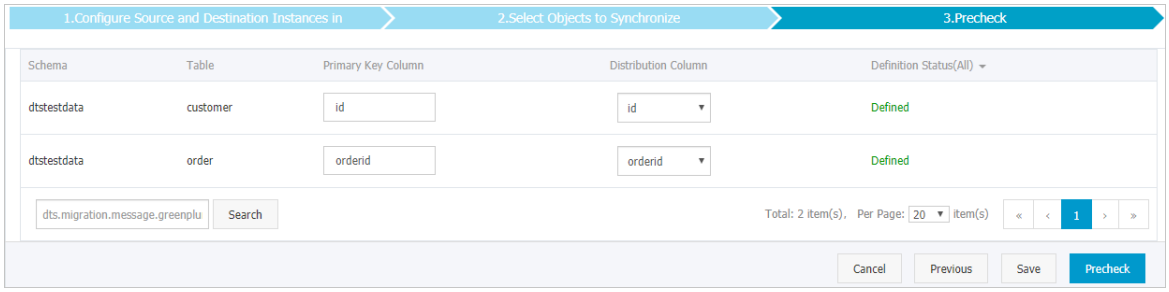


Setting	Parameter	Description
	Initial Synchronization	You must select both **Initial Schema Synchronization** and **Initial Full Data Synchronization** in most cases. After the precheck, DTS synchronizes the schemas and data of the required objects from the source instance to the destination instance. The schemas and data are the basis for subsequent incremental synchronization.

Setting	Parameter	Description
Select the synchronization policy	Processing Mode of Conflicting Tables	○ **Clear Target Table** Skips the **Schema Name Conflict** item during the precheck. Clears the data in the destination table before initial full data synchronization. If you want to synchronize your business data after testing the data synchronization task, you can select this mode. ○ **Ignore** Skips the **Schema Name Conflict** item during the precheck. Adds data to the existing data during initial full data synchronization. If you want to synchronize data from multiple tables to one table, you can select this mode.
	Synchronization Type	Select the types of operations that you want to synchronize based on your business requirements. ○ **Insert** ○ **Update** ○ **Delete** ○ **AlterTable**
Select the objects to be synchronized	N/A	Select one or more tables from the **Available** section and click the › icon to move the tables to the **Selected** section. **Note** ○ You can select only tables as the objects to be synchronized. ○ You can use the object name mapping feature to change the names of the columns that are synchronized to the destination database. For more information, see Rename an object to be synchronized.
Rename Databases and Tables	N/A	You can use the object name mapping feature to rename the objects that are synchronized to the destination instance. For more information, see Object name mapping.

Setting	Parameter	Description
Replicate Temporary Tables When DMS Performs DDL Operations	N/A	If you use Data Management (DMS) to perform online DDL operations on the source database, you can specify whether to synchronize temporary tables generated by online DDL operations. ○ **Yes**: DTS synchronizes the data of temporary tables generated by online DDL operations. **Note** If online DDL operations generate a large amount of data, the data synchronization task may be delayed. ○ **No**: DTS does not synchronize the data of temporary tables generated by online DDL operations. Only the original DDL data of the source database is synchronized. **Note** If you select No, the tables in the destination database may be locked.
Retry Time for Failed Connections	N/A	By default, if DTS fails to connect to the source or destination database, DTS retries within the next 720 minutes (12 hours). You can specify the retry time based on your needs. If DTS reconnects to the source and destination databases within the specified time, DTS resumes the data synchronization task. Otherwise, the data synchronization task fails. **Note** When DTS retries a connection, you are charged for the DTS instance. We recommend that you specify the retry time based on your business needs. You can also release the DTS instance at your earliest opportunity after the source and destination instances are released.

9. Specify the primary key column and distribution column of the table that you want to synchronize to the instance.



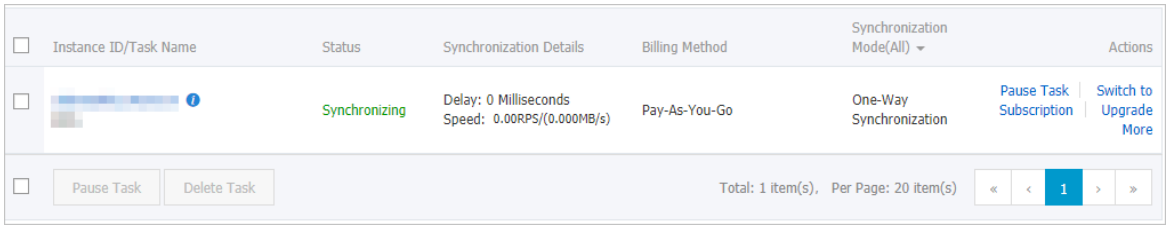
> **Note** The page in this step appears only if you select **Initial Schema Synchronization**. For more information about primary key columns and distribution columns, see Define constraints and Define table distribution.

10. In the lower-right corner of the page, click **Precheck**.

> **Note**
> - Before you can start the data synchronization task, DTS performs a precheck. You can start the data synchronization task only after the task passes the precheck.
> - If the task fails to pass the precheck, you can click the icon next to each failed item to view details.
>   - After you troubleshoot the issues based on the causes, you can run a precheck again.
>   - If you do not need to troubleshoot the issues, you can ignore failed items and run a precheck again.

11. Close the **Precheck** dialog box after the following message is displayed: **The precheck is passed.** Then, the data synchronization task starts.
12. Wait until the initial synchronization is complete and the data synchronization task is in the **Synchronizing** state.

    You can view the status of the data synchronization task on the **Synchronization Tasks** page.



8.Instance lifecycle

8.1. Create an ApsaraDB RDS for MySQL instance

This topic describes how to create an ApsaraDB RDS for MySQL instance.

> **Note** You are offered a reduced price on your first purchase of an RDS instance. For more information, visit the ApsaraDB RDS promotion page.

Prerequisites

The AliyunRDSFullAccess policy is attached to the RAM user that you used to create the RDS instance. For more information, see Use RAM for resource authorization.

Procedure

1. Go to the ApsaraDB RDS buy page.
2. Configure the **Billing Method** parameter.

Billing method	Description	Benefit
Subscription	A subscription instance is an instance for which you pay an upfront fee. If you want to use an instance for a long period of time, we recommend that you select the **Subscription** billing method. If you select the subscription billing method, configure the **Duration** parameter in the lower part of the page.	In most cases, the subscription billing method is more cost-effective than the pay-as-you-go billing method for long-term usage. Alibaba Cloud provides lower prices for longer subscription periods.
Pay-As-You-Go	You are charged on an hourly basis for a pay-as-you-go instance based on your actual resource usage. If you want to use an instance for a short period of time, we recommend that you select the **Pay-As-You-Go** billing method. You can create a **pay-as-you-go** RDS instance. After you confirm that the new RDS instance meets your business requirements, you can change the billing method of the RDS instance from pay-as-you-go to **subscription**.	You can release a pay-as-you-go RDS instance based on your business requirements. The billing cycle of a pay-as-you-go RDS instance immediately stops after you release the instance.

> **Note** You can view the price in the lower-right corner of the page. The price is displayed only after you configure all required parameters.

3. Configure the **Region** parameter.

   We recommend that you use an RDS instance that resides in the same region as on which your

application is deployed. If the RDS instance and the ECS instance reside in different regions, you cannot connect these instances over an internal network. In this case, these instances cannot deliver the optimal performance.

> **Note**
>
> - After an RDS instance is created, you cannot change the region of the RDS instance. If you want to connect an ECS instance and an RDS instance over an internal network, make sure that the RDS instance and the ECS instance reside in the same region.
> - For more information about how to view the region in which an ECS instance resides, see Get ready to use ApsaraDB RDS for MySQL.
> - If your application is deployed on an on-premises server or on-premises computer, we recommend that you select a region that is near your on-premises server or on-premises computer. This way, you can use the public endpoint of the RDS instance to connect to the RDS instance from your application.

the Elastic Compute Service (ECS) instance

4. Configure the **Database Engine** parameter.

   In this example, select **MySQL**.

   We recommend that you select MySQL 8.0 or MySQL 5.7 or select the database engine version that your self-managed MySQL instance runs. The default value of this parameter is 8.0.

5. Configure the **Edition** parameter. The default value of this parameter is **High-availability**.

Edition	Description	Benefit
Basic	In RDS Basic Edition, the database system consists of only a primary RDS instance.	RDS Basic Edition is cost-effective and is suitable for learning and testing scenarios. RDS instances that run RDS Basic Edition require a long period of time to restart or recover from faults.
High-availability	This is the recommended RDS edition. In RDS High-availability Edition, the database system consists of a primary RDS instance and a secondary RDS instance. You can create read-only RDS instances and attach the read-only RDS instances to the primary RDS instance.	RDS High-availability Edition is suitable for more than 80% of business scenarios that require production environments.
Enterprise	In RDS Enterprise Edition, the database system consists of a primary RDS instance, a secondary RDS instance, and a logger RDS instance. You can create read-only RDS instances and attach the read-only RDS instances to the primary RDS instance.	RDS Enterprise Edition is suitable for financial institutions that have high requirements for reliability.

> **Note**
>
> - The available RDS editions vary based on the region and database engine version that you select. If you select MySQL 5.6, Basic is not displayed.
> - For more information, see Overview of ApsaraDB RDS editions.

6. Configure the **Storage Type** parameter.

Comparison item	**ESSD** (recommended)	**Local SSD**
Scalability	★★★★★ ○ You can increase the storage capacity up to 32 TB. ○ No transient connections occur during storage expansion. ○ You can upgrade, downgrade, create, or release the RDS instance in minutes. ○ Automatic storage expansion is supported.	★★ ○ You can increase the storage capacity up to 6 TB. ○ Transient connections occur during storage expansion. ○ A few hours may be required to upgrade, downgrade, create, or release the RDS instance. ○ Automatic storage expansion is not supported.
Performance	★★★★★ ○ PL1<PL2<PL3 ○ An enhanced SSD (ESSD) of performance level 2 (PL2) provides twice the IOPS and throughput of an ESSD of performance level 1 (PL1). ○ An ESSD of PL3 provides 20 times the IOPS and 11 times the throughput of an ESSD of PL1.	★★★★★
Backup	★★★★★ ○ A backup can be completed in minutes or seconds. ○ The highest backup frequency is one backup every 15 minutes.	★★★ ○ A backup requires a long period of time to complete. ○ The highest backup frequency is one backup every day.

> **Note**
>
> - RDS instances that run MySQL 5.7 or MySQL 8.0 on RDS Basic Edition support only standard SSDs and ESSDs. RDS instances that run MySQL 8.0 or MySQL 5.7 on RDS Enterprise Edition support only local SSDs. RDS instances that run MySQL 5.6 or MySQL 5.5 on RDS High-availability Edition support only local SSDs. Serverless RDS instances support only standard SSDs and ESSDs.
> - The available features vary based on the storage type that you select. For more information, see MySQL 8.0.
> - For more information about different types of storage media, see Features.

7. Select the . zone
   - Select a zone.
     - No significant differences exist between the zones in the same region.
     - If the RDS instance resides in the same zone as the ECS instance on which your application is deployed, these instances can provide the optimal performance. If the RDS instance and the ECS instance reside in different zones in the same region, the performance of the RDS instance and the ECS instance is slightly lower than the performance of the RDS instance and the ECS instance that reside in the same zone.
   - Select a deployment method.
     - **Multi-zone Deployment**: The RDS instance and its secondary RDS instance reside in different zones to allow users to perform zone-disaster recovery. This is the recommended deployment method.
     - **Single-zone Deployment**: The RDS instance and its secondary RDS instance reside in the same zone.

     > **Note** If you select **Basic** for the Edition parameter, only the **Single-zone Deployment** method is supported.

8. Configure the **Instance Type** parameter.

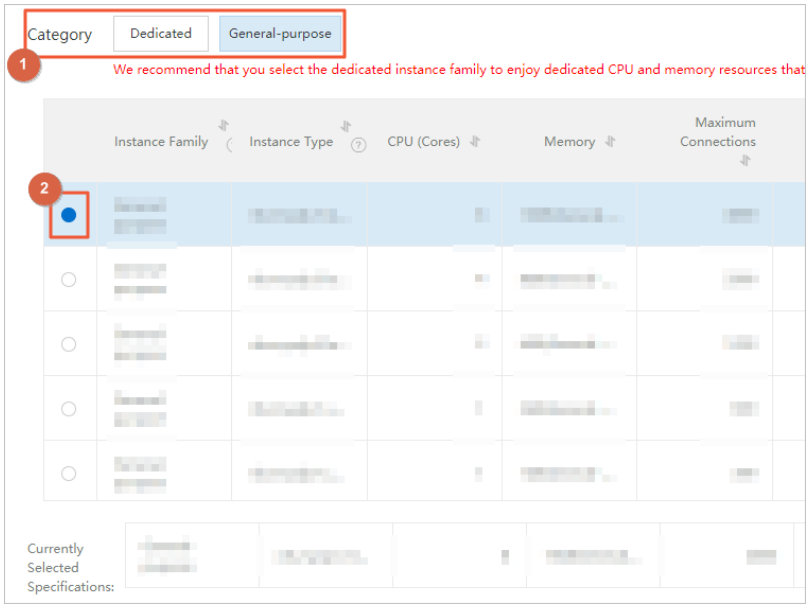
i. Select an **instance family**. You can select General-purpose or Dedicated.

Instance family	Description	Benefit
General-purpose	A general-purpose RDS instance occupies all the allocated memory and I/O resources. A general-purpose RDS instance shares CPU and storage resources with other general-purpose RDS instances that are deployed on the same host.	General-purpose RDS instances are cost-effective.
Dedicated	A dedicated RDS instance occupies all the allocated CPU, memory, storage, and I/O resources. **Note** The dedicated host instance family is the highest configuration of the dedicated instance family. A dedicated host RDS instance occupies all CPU, memory, storage, and I/O resources of the host on which the RDS instance is deployed.	A dedicated RDS instance provides higher performance and higher stability. **Note** An RDS instance that runs RDS Basic Edition does not support the dedicated instance family.

ii. Select an instance type.

- In a test environment, select an instance type that provides one or more CPU cores.
- In a production environment, select an instance type that provides four or more CPU cores.

Note For more information, see Primary ApsaraDB RDS for MySQL instance types.



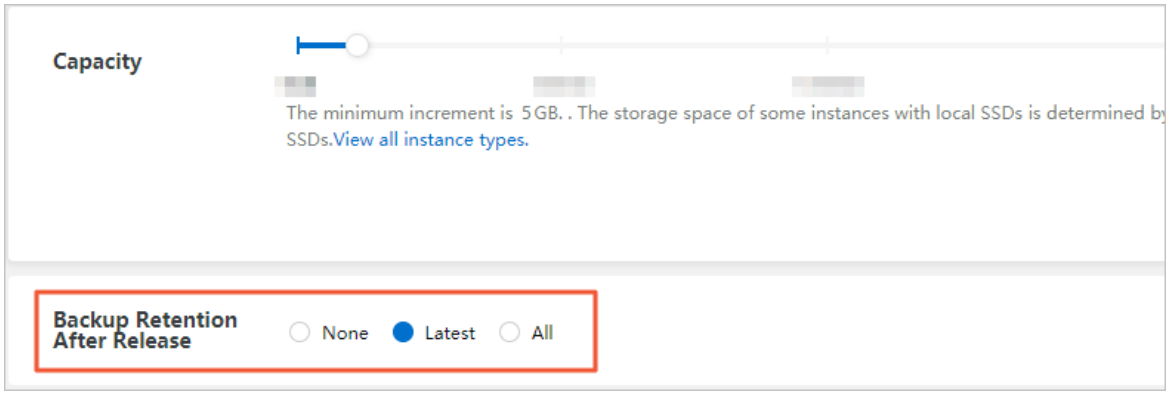
9. Configure the **Capacity** parameter.

The value range of the storage capacity varies based on the instance type and storage type that you select.

You can change the storage capacity at a step size of 5 GB.

10. Configure the following parameters. This step is required only if you select the **Subscription** billing method and the **Local SSD** storage type.

    We recommend that you set the **Backup Retention After Release** parameter to **Latest** or **All**. This way, you can retrieve the data of the RDS instance if the RDS instance is released due to overdue payments and data is lost.



11. In the lower-right corner of the page, click **Next: Instance Configuration**.

12. Configure the . Network Type

    i. **Network Type**: If your application is deployed on an ECS instance, the network type of the ECS instance and the RDS instance must be the same. Otherwise, you cannot connect the ECS instance and the RDS instance over an internal network.

> **Note**
>
> - For more information about how to view the network type of the ECS instance, see Get ready to use ApsaraDB RDS for MySQL.
> - If you do not want to connect the ECS instance and the RDS instance over an internal network, you can select the classic network type or the VPC network type.
> - Specific RDS instances do not support the classic network type.

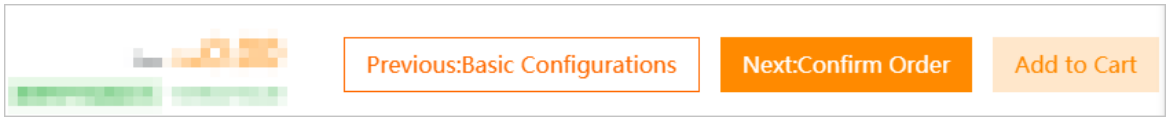
ii. **VPC** and **vSwitch**: If you select the VPC network type, you must also select a VPC and a vSwitch. We recommend that you select the VPC of the ECS instance on which your application is deployed. If the RDS instance and the ECS instance reside in different VPCs, you cannot connect these instances over an internal network.

> **Note**
> - For more information about how to view the VPC in which your ECS instance resides, see Get ready to use ApsaraDB RDS for MySQL.
> - You can connect the RDS instance and the ECS instance over an internal network even if the instances use different vSwitches in the same VPC.

13. Configure other custom parameters. **If you do not have special business requirements, you can use the default values of these parameters.**

Parameter	Description
Release Protection	Specifies whether to enable the release protection feature. The release protection feature is used to prevent a pay-as-you-go RDS instance from being released due to incorrect operations. For more information, see Enable or disable the release protection feature for an ApsaraDB RDS for MySQL instance.
Minor Version Upgrade Policy	The policy based on which the minor engine version of the RDS instance is updated. ○ **Automatic Upgrade**: ApsaraDB RDS automatically updates the minor engine version of the RDS instance to the most recent version during the scheduled maintenance window. For more information about how to change the maintenance window, see Set the maintenance window of an ApsaraDB RDS for MySQL instance. For more information about how to change the upgrade time, see Manage scheduled events. ○ **Manual Upgrade**: You must manually update the minor engine version of the RDS instance on the Basic Information page. If you do not want to use the latest minor engine version, select **Select Minor Version**. Then, you can select a minor engine version from the drop-down list that is displayed.
Resource Group	The resource group to which the RDS instance belongs. You can use the default resource group or select a custom resource group based on your business requirements.

14. In the lower-right corner of the page, click Next: Confirm Order.



15. Confirm the configuration of the RDS instance in the Parameters section, configure the **Purchase Plan** and **Duration** parameters, read and select **Terms of Service**, and then click **Pay Now**. You must configure the Duration parameter only if you select the subscription billing method for the RDS instance.

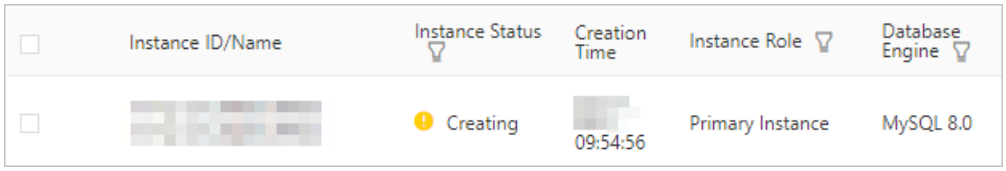
> **Note** If you select the subscription billing method for the RDS instance, we recommend that you select **Auto-Renew Enabled**. This way, you can prevent interruptions on your application even if you forget to renew the RDS instance.

The "**Congratulations.**" or "**The service is activated**" message is displayed in the ApsaraDB RDS console.

16. View the RDS instance.

    Go to the Instances page. In the top navigation bar, select the region where the RDS instance resides. Then, find the RDS instance based on the **Creation Time** parameter.

    ApsaraDB RDS requires 1 to 10 minutes to create an RDS instance. You can refresh the page to view the RDS instance that you created.



What to do next

Create databases and accounts for an ApsaraDB RDS for MySQL instance

FAQ

Why am I unable to find the RDS instance that I created?

Possible cause	Description	Suggestion
Incorrect region	The RDS instance does not reside in the region that you selected.	In the top navigation bar, select the region where the RDS instance resides. Then, you can find the RDS instance.
Insufficient resources	The zone that you selected cannot provide sufficient resources. If the RDS instance cannot be created, you can go to the Orders page in the Billing Management console to view the refunded fee.	We recommend that you select a different zone and try again.
RAM policies that do not allow users to create unencrypted RDS instances	RAM policies that do not allow users to create unencrypted RDS instances are attached to RAM users. • If you use the credentials of a RAM user to create an RDS instance that uses local SSDs, the RDS instance cannot be created. When you create an RDS instance that uses local SSDs, you cannot enable disk encryption. • If you use the credentials of a RAM user to create an RDS instance that uses standard SSDs or ESSDs and you do not enable disk encryption for the RDS instance, the RDS instance cannot be created. For more information, see Use RAM policies to manage the permissions of RAM users on ApsaraDB RDS instances.	When you create an RDS instance, select the standard SSD or ESSD storage type, select Disk Encryption, set the Key parameter, and then try again.

References

- For more information about how to create an RDS instance by calling an API operation, see Create an instance.
- For more information about how to create an RDS instance that runs a different database engine, see the following topics:
  - Create an ApsaraDB RDS for SQL Server instance
  - Create an ApsaraDB RDS for PostgreSQL instance
  - Create an ApsaraDB RDS for MariaDB TX instance

8.2. Restart an ApsaraDB RDS for MySQL instance

This topic describes how to manually restart an ApsaraDB RDS for MySQL instance. This applies if the number of connections exceeds the specified threshold or a performance issue occurs.

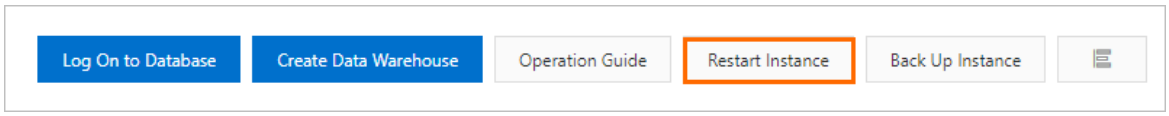
Impacts

A restart causes a network interruption that lasts about 30 seconds. Before you restart your RDS instance, we recommend that you make proper service arrangements. Proceed with caution.

> **Note** The Basic Edition does not provide a secondary RDS instance as a hot standby for the primary RDS instance. If the primary RDS instance unexpectedly exits, your database service may be unavailable for a long period of time. If you change the specifications or upgrade the database engine version of the primary RDS instance, your database service may also be unavailable for a long period of time. If you require high service availability, we recommend that you do not select the Basic Edition. For example, you can select the High-availability Edition. Some primary RDS instances support the upgrade from the Basic Edition to the High-availability Edition. For more information, see Upgrade an RDS instance to the High-availability Edition.

Procedure

1.
2. In the upper-right corner of the Basic Information page, click **Restart Instance**.



3. In the message that appears, click **OK**.

Related operations

Operation	Description
Restart an ApsaraDB for RDS instance	Restarts an ApsaraDB RDS instance.

8.3. Renew instance

8.3.1. Manually renew an ApsaraDB RDS for MySQL instance

This topic describes how to manually renew an ApsaraDB RDS for MySQL instance that uses the subscription billing method. We recommend that you manually renew your RDS instance before the expiration date. This allows you to prevent service interruptions and data losses.

For more information about the impacts that are caused by subscription expiration, see Unlock or rebuild an expired or overdue ApsaraDB RDS instance.

> **Note** RDS instances that use the pay-as-you-go billing method do not expire and therefore do not require renewal.

You can manually renew your RDS instance before your RDS instance expires. You can also manually renew your RDS instance within 15 days after it expires.

Method 1: Renew an RDS instance in the ApsaraDB RDS console

Renew a single RDS instance

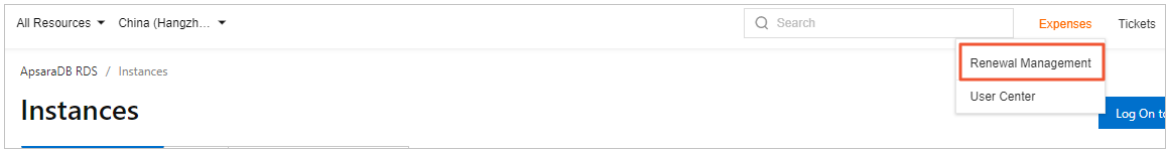
1.
2. In the **Status** section of the page that appears, click **Renew** on the right.
3. On the **Renew** page, configure the **Duration** parameter. You are offered lower prices for longer subscription periods.
4. Read and select Terms of Service, click **Pay Now**, and then complete the payment.

Renew multiple RDS instances at a time

1.
2. Select the RDS instances that you want to renew and click **Renew** below the instance list.
3. In the **Renew** dialog box, confirm the selected RDS instances and click **OK** to go to the **Renewal** page.
4. On the **Manual** tab, select the RDS instances and click **Batch Renew** in the lower part of the page.
5. Configure the **Duration** parameter of each RDS instance, click **Pay**, and then complete the payment.

Method 2: Renew the instance in the Billing Management console

1. Log on to the ApsaraDB RDS console.
2. In the top navigation bar, choose **Expenses > Renewal Management**.



3. On the **Manual** tab of the Renewal page, find the RDS instances that you want to renew. You can renew one or more RDS instances at a time.
   - **Renew a single RDS instance**

a. Find the RDS instance that you want to renew and click **Renew** in the Actions column.

> **Note** If the RDS instance is displayed on the **Auto** or **Nonrenewal** tab, you can click **Enable Manual Renewal** in the Actions column and then click **OK** in the message that appears to manually renew the RDS instance.

b. On the page that appears, configure the Duration parameter, click **Pay Now**, and then complete the payment.

- **Renew multiple RDS instances at a time**

a. Select the RDS instances that you want to renew and click **Batch Renew** in the lower part of the page.

b. Configure the **Duration** parameter of each RDS instance, click **Pay**, and then complete the payment.

Enable auto-renewal for an RDS instance

After automatic renewal is enabled for an RDS instance, you do not need to renew the RDS instance on a regular basis. This allows you to prevent service interruptions that are caused by subscription expiration. For information, see Enable auto-renewal for an ApsaraDB RDS for MySQL instance.

8.3.2. Enable auto-renewal for an ApsaraDB RDS for MySQL instance

This topic describes how to enable auto-renewal for an ApsaraDB RDS for MySQL instance that uses the subscription billing method. If you enable auto-renewal for your RDS instance, you do not need to manually renew your subscription or be concerned about service interruptions caused by subscription expiration.

If do not renew your RDS instance before the expiration date, your RDS instance expires. As a result, your workloads are interrupted and your data may be lost. For more information, see Unlock or rebuild an expired or overdue ApsaraDB for RDS instance.

> **Note** RDS instances that use the pay-as-you-go billing method do not expire and therefore do not require renewal.

Precautions

- If you enable auto-renewal, the first time when the system deducts the subscription fee from your Alibaba Cloud account comes at 08:00:00 three days before the expiration date. If the deduction fails, the system attempts to deduct the fee every day for the next two days.

  > **Note** Make sure that the balance of your Alibaba Cloud account is sufficient. Otherwise, the renewal fails. If all the three automatic fee deduction attempts fail, you must manually renew your RDS instance before the expiration date. This allows you to avoid service interruptions and data losses.

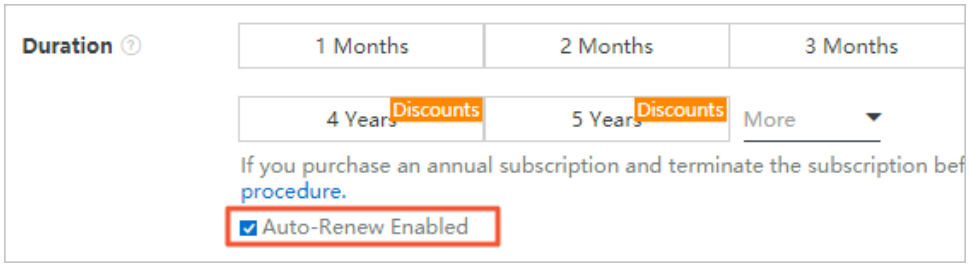
- If you manually renew your RDS instance before the system starts automatic fee deduction attempts, the system will automatically renew the instance next time before the expiration date.

- After you enable auto-renewal, it takes effect the next day. If your RDS instance is due to expire the next day, renew it manually to avoid service interruptions. For more information, see Manually renew an ApsaraDB RDS for MySQL instance.

Enable auto-renewal when you purchase an RDS instance

> **Note** If you select auto-renewal when you purchase an RDS instance, the system automatically renews the RDS instance based on the specified renewal cycle. The renewal cycle is one month or one year. For example, if you select auto-renewal when you purchase an RDS instance with a six-month subscription, the system automatically renews the RDS instance with a one-month subscription each time the instance is due to expire.

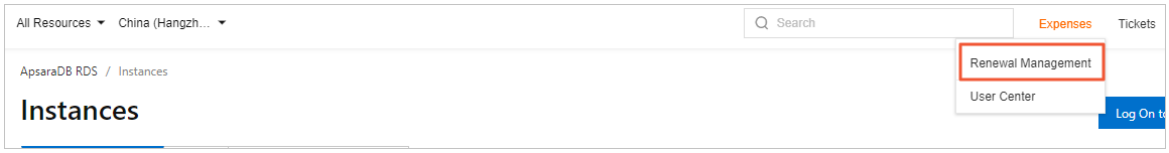
When you purchase a subscription RDS instance, select **Auto-Renew Enabled**.



Enable auto-renewal after you purchase an RDS instance

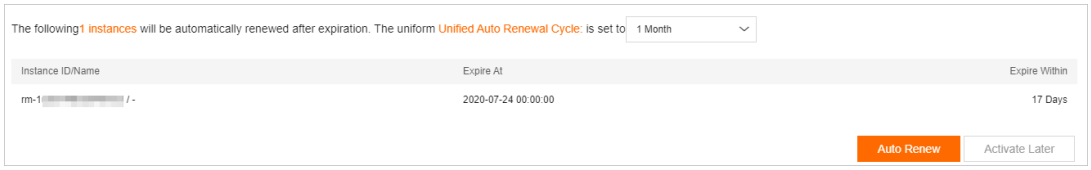
> **Note** After you enable auto-renewal for a created RDS instance, the system automatically renews the RDS instance based on the selected renewal cycle. For example, if you select a three-month renewal cycle, you are charged for a three-month subscription in each renewal cycle.

1. Log on to the ApsaraDB RDS console.
2. In the top navigation bar, choose **Expenses > Renewal Management**.



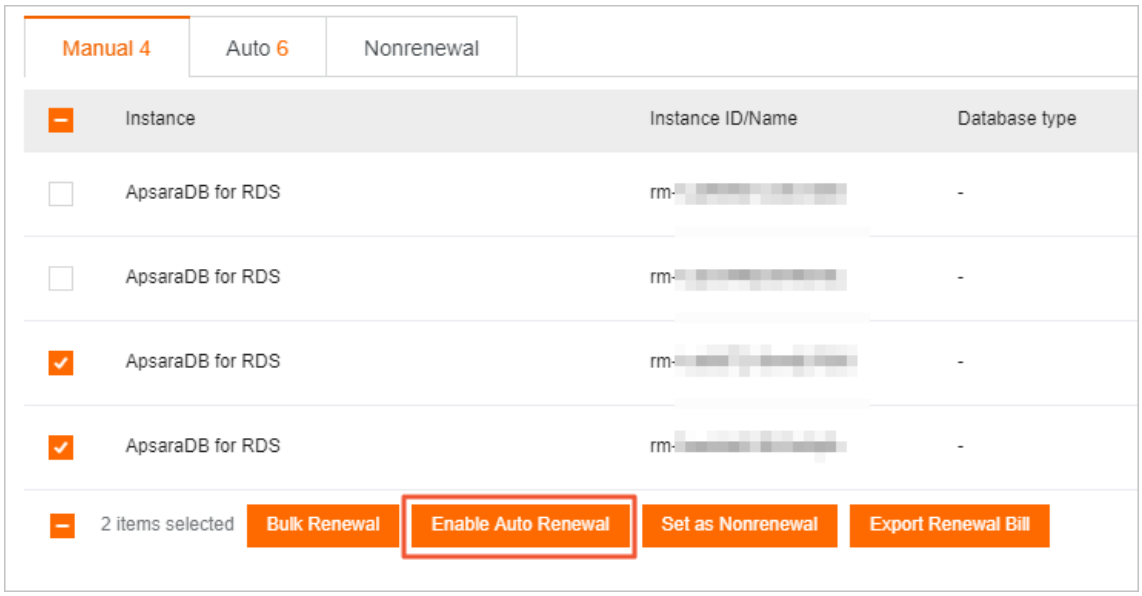
3. On the **Manual** or **Nonrenewal** tab, specify the filter conditions to find the RDS instance for which you want to enable auto-renewal. You can enable auto-renewal for one or more RDS instances at a time.
   - Enable auto-renewal for a single RDS instance.
     a. Find the RDS instance and in the Actions column click **Enable Auto Renewal**.

     b. In the dialog box that appears, specify the **Unified Auto Renewal Cycle** parameter and click **Auto Renew**.

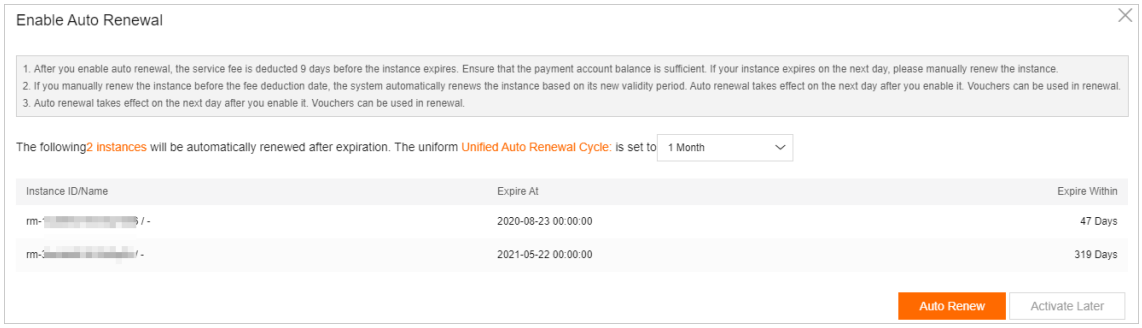


- Enable auto-renewal for multiple RDS instances.

  Select the RDS instances and click **Enable Auto Renewal** below the instance list.

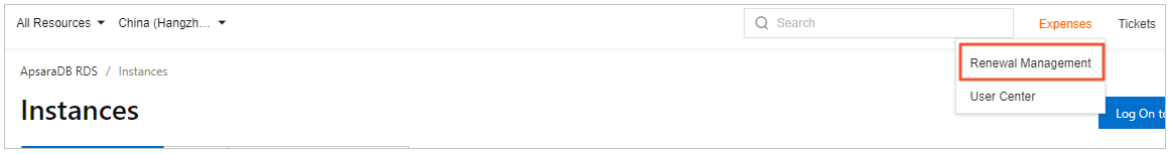


- In the dialog box that appears, specify the **Unified Auto Renewal Cycle** parameter and click **Auto Renew**.



Change the auto-renewal cycle

1. Log on to the ApsaraDB RDS console.
2. In the top navigation bar, choose **Expenses > Renewal Management**.

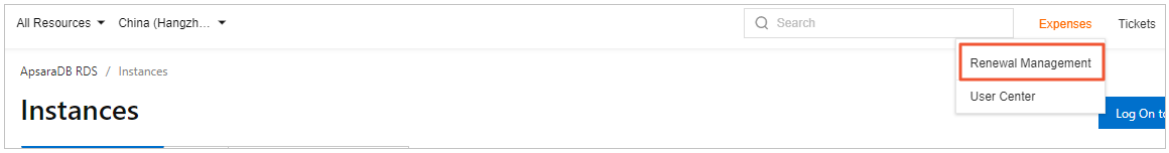


3. On the **Auto** tab, specify filter conditions to find the RDS instance for which you want to enable auto-renewal. Then, select the RDS instance and click **Edit Auto Renewal** in the Actions column.

4. In the dialog box that appears, change the auto-renewal cycle and click **OK**.

Disable auto-renewal

1. Log on to the ApsaraDB RDS console.
2. In the top navigation bar, choose **Expenses > Renewal Management**.



3. On the **Auto** tab, specify filter conditions to find the RDS instance for which you want to enable auto-renewal. Then, select the RDS instance and click **Enable Manual Renewal** in the Actions column.

4. In the message that appears, click **OK**.

Related operations

Operation	Description
Create an instance	Creates an ApsaraDB RDS instance. **Note** You can call this operation to enable auto-renewal for an RDS instance that you want to create.
Manually renew an ApsaraDB for RDS instance	Renews an ApsaraDB RDS instance. **Note** You can call this operation to enable auto-renewal for a created RDS instance.

8.4. Release or unsubscribe from an ApsaraDB RDS for MySQL instance

This topic describes how to release a pay-as-you-go ApsaraDB RDS for MySQL instance and how to unsubscribe from a subscription ApsaraDB RDS for MySQL instance.

Precautions

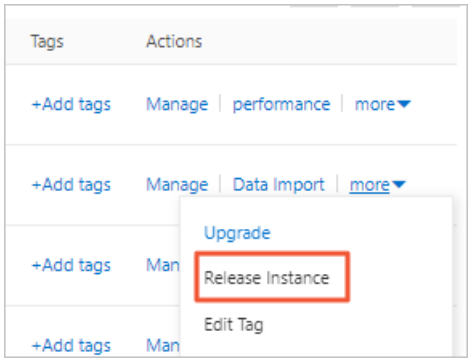
- After you release or unsubscribe from an RDS instance, the RDS instance and its data are immediately deleted. Before you release or unsubscribe from an RDS instance, we recommend that you back up the RDS instance and download the required backup files. For more information, see Enable automatic backups for an ApsaraDB RDS for MySQL instance and Download the backup files of an ApsaraDB RDS for MySQL instance.

> **Note**
>
> - If an RDS instance runs the RDS Basic Edition or High-availability Edition with standard SSDs or enhanced SSDs (ESSDs), you cannot download the backup files of the RDS instance.
> - If an RDS instance is equipped with local SSDs, you can use the **Backup Retention Policy After Release** parameter in the ApsaraDB RDS console to specify which backup files you want to retain after you release or unsubscribe from the RDS instance. This way, after you release or unsubscribe from the RDS instance, the data backup files of the RDS instance are not deleted. You can download the data backup files to your computer. For more information, see Retain data backup files after instance release.

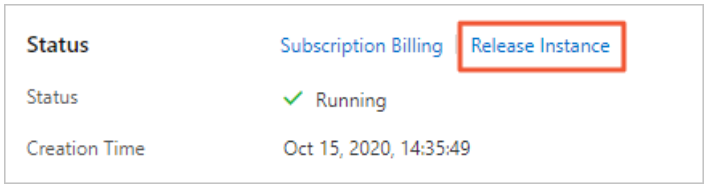
- If you release or unsubscribe from the last read-only RDS instance of a primary RDS instance, you must disable the read/write splitting feature for the primary RDS instance. For more information, see Disable read/write splitting for an RDS MySQL instance.
- After you release or unsubscribe from an RDS instance, ApsaraDB RDS moves the RDS instance to the recycle bin if the RDS instance meets specific requirements. For more information, see Manage ApsaraDB RDS for MySQL instances in the recycle bin
- After you release or unsubscribe from a primary RDS instance, the subscription read-only RDS instances and pay-as-you-go read-only RDS instances of the primary RDS instance are automatically deleted. In addition, the payments for the subscription read-only RDS instances of the primary RDS instance are refunded.

Release a pay-as-you-go RDS instance

1. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Instances**. In the top navigation bar, select the region where the RDS instance resides.
2. Use one of the following methods to open the **Release Instance** message:
   - Find the RDS instance. In the **Actions** column, choose **More > Release Instance**.



   - a. Find the RDS instance and click the instance ID.
     b. On the **Basic Information** page, click **Release Instance**.



3. In the message that appears, click **Confirm**.

Unsubscribe from a subscription RDS instance

If you want to unsubscribe from an RDS instance, submit a .

FAQ

- If I release or unsubscribe from a read-only RDS instance, are my workloads interrupted?

  Yes, if you release or unsubscribe from a read-only RDS instance, your workloads on the read-only RDS instance are interrupted. Before you release or unsubscribe from a read-only RDS instance, we recommend that you set the read weight of the read-only RDS instance to 0. For more information, see Modify the latency threshold and read weights of ApsaraDB RDS for MySQL instances.

  > **Note** The cached connections to the read-only RDS instance remain valid. If you want to route the read requests over the cached connections to the other read-only RDS instances, you must establish new connections.

- After I release or unsubscribe from my RDS instance, how do I retrieve the data of the RDS instance?

  If your RDS instance is configured to retain backup files after you release or unsubscribe from your RDS instance, you can go to the **Backup for Deleted Instances** tab of the Backups page in the ApsaraDB RDS console to restore the data of your RDS instance. For more information, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.

Related operations

Operation	Description
Release instance	Releases a pay-as-you-go ApsaraDB RDS instance. You cannot unsubscribe from a subscription ApsaraDB RDS instance by calling an API operation.

8.5. Manage ApsaraDB RDS for MySQL instances in the recycle bin

This topic describes how to manage the ApsaraDB RDS for MySQL instances that are moved to the recycle bin. You can unlock, rebuild, or destroy these instances in the recycle bin.

Functionality

All ApsaraDB RDS for MySQL instances that are manually released, automatically released due to expiration, or payment refunded are moved to the recycle bin. An RDS instance is not moved to the recycle bin in the following scenarios:

- The RDS instance is payment refunded or manually released within seven days after the instance is created.
- The RDS instance is a pay-as-you-go RDS instance and is automatically released due to overdue payments.
- The RDS instance runs the **RDS Cluster Edition** or is a **read-only RDS instance**.

Unlock an RDS instance whose payment is overdue

If a pay-as-you-go RDS instance is locked due to overdue payments, log on to the Billing Management console. Then, verify that a valid payment method is specified for your Alibaba Cloud account.

Unlock an RDS instance that has expired

If a subscription RDS instance is locked due to expiration, you can renew the instance in the recycle bin within the next 15 days. If you do not renew the instance within 15 days, the instance is released.

1.
2. In the left-side navigation pane, click **Locked Instances**. In the top navigation bar, select the region where the RDS instance resides.
3. Find the RDS instance and click **Unlock** to renew the RDS instance.

After the RDS instance is renewed, it is immediately restored to normal.

Rebuild an RDS instance

After a subscription RDS instance is automatically released due to expiration or a pay-as-you-go RDS instance that is created seven days ago or earlier is manually released, the backup files of the instance are retained for eight days. During the eight-day retention period, you can restore the data of the backup files to a new RDS instance by using the rebuild feature. After the eight-day retention period elapses, only the backup files that meet specified conditions are retained and all the other backup files are deleted.

> **Note**
>
> - The backup files that can be retained after the eight-day retention period elapses must meet the following conditions:
>   - The instance runs MySQL, PostgreSQL, or SQL Server, and the cross-region backup retention period that you specify has not elapsed. In this case, the cross-region backup files of the instance are retained within the specified cross-region backup retention period.
>   - The instance runs MySQL, and you have specified to retain the backup files of the instance even after the instance is released. In this case, the backup files of the instance are retained. For more information, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.
> - After an RDS instance is rebuilt, the new RDS instance does not inherit the ID and endpoint of the original RDS instance. However, you can still use the endpoint of the original RDS instance to make sure that your applications can connect with the new RDS instance. For more information about how to modify the endpoint, see Change the internal or public endpoint and port number of an RDS instance.

1. Log on to the ApsaraDB RDS console.
2. In the left-side navigation pane, click **Locked Instances**. In the top navigation bar, select the region where the RDS instance resides.
3. Find the RDS instance and click **Recreate Instance**.

By default, ApsaraDB RDS creates an RDS instance that has the same specifications in the same zone as the original RDS instance. You can also create an RDS instance that has different specifications in a different zone than the original RDS instance.

Destroy an RDS instance

If an RDS instance is locked due to overdue payments or expiration, you can destroy the RDS instance in the recycle bin.

> **Warning** After you destroy an RDS instance, only the cross-region backup files of the instance are retained. All regular data backup files, archived backup files, and log backup files of the instance are destroyed. Proceed with caution when you destroy an RDS instance. For more information about cross-region backup files, see Enable cross-region backups for an ApsaraDB RDS for MySQL instance.

1.
2. In the left-side navigation pane, click **Locked Instances**. In the top navigation bar, select the region where the RDS instance resides.
3. Find the instance and click **Destroy**.

References

- Unlock or rebuild an expired or overdue ApsaraDB RDS instance
- Release or unsubscribe from an ApsaraDB RDS for MySQL instance

9.Database connection

9.1. Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance

This topic describes how to configure IP address whitelists and use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.

Prerequisites

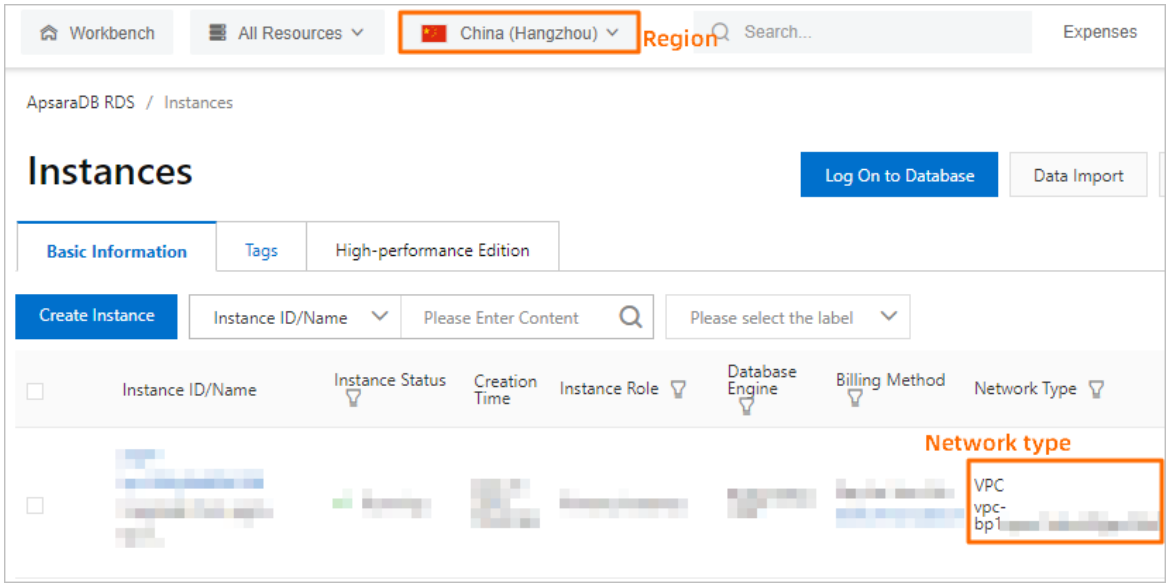
The operations that are described in the following topics are complete:

- Create an ApsaraDB RDS for MySQL instance
- Create databases and accounts for an ApsaraDB RDS for MySQL instance

Step 1: Check whether your application can connect to the RDS instance over an internal network

1. View the region of the instance on which your application is deployed. For more information, see Get ready to use ApsaraDB RDS for MySQL. Elastic Compute Service (ECS)network type
2. View the region and network type of the RDS instance.

   Log on to the ApsaraDB RDS console and go to the Instances page. In the top navigation bar, select the region where the RDS instance resides. Then, find the RDS instance and click the instance ID. On the page that appears, you can view the region, network type, and virtual private cloud (VPC) ID of the RDS instance.



3. Check whether the ECS instance and the RDS instance meet the following conditions for communication over an internal network:

   i. The ECS instance and the RDS instance reside in the same region.

ii. The ECS instance and the RDS instance reside in the same type of network. If the ECS instance and the RDS instance both reside in VPCs, these instances must reside in the same VPC.

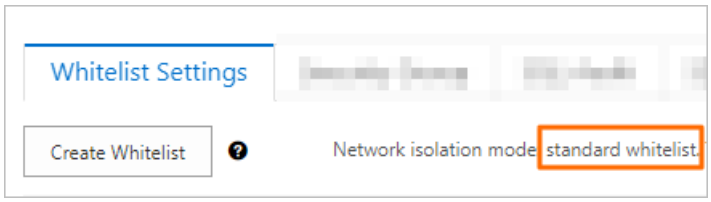
> **Note** If one of the preceding conditions is not met, the ECS instance cannot communicate with the RDS instance over an internal network.

Step 2: Configure IP address whitelists for the RDS instance

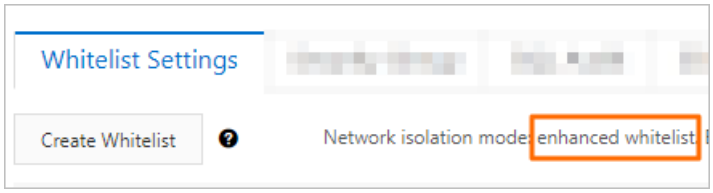
1.
2. In the left-side navigation pane, click **Data Security**.
3. View the network isolation mode of the RDS instance.

> **Note** Existing RDS instances may run in enhanced whitelist mode. New RDS instances run in standard whitelist mode.

Standard whitelist mode

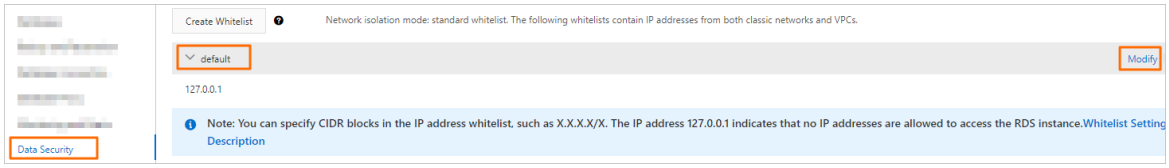


Enhanced whitelist mode



4. Click **Modify** to the right of the IP address whitelist named **default**.

> **Note** You can also click **Create Whitelist** to create an IP address whitelist.

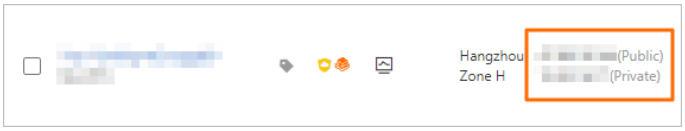


5. Add the IP address of the server on which your application is deployed to the default IP address whitelist.

   The server can communicate with the RDS instance only after you add the IP address of the server to the default IP address whitelist.

The following table describes various connection scenarios. You can obtain the required IP address

based on your connection scenario and add the IP address to an IP address whitelist of the RDS instance. Obtain IP addresses

Connection scenario	IP address to be obtained	How to obtain the IP address
the conditions for communication over an internal network	The private IP address of the ECS instance	i. Log on to the ECS console and go to the Instances page. ii. In the top navigation bar, select the region where the ECS instance resides. iii. View the public IP address and private IP address of the ECS instance. 
You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance do not meet the conditions for communication over an internal network.	The public IP address of the ECS instance	
You want to connect to the RDS instance from an on-premises device.	The public IP address of the on-premises device	On the on-premises device, use a search engine such as Google to search for **IP**. **Note** The IP address that you obtain by using this method may be inaccurate. For more information about how to obtain the accurate IP address of an on-premises device, see Why am I unable to connect to my ApsaraDB RDS for MySQL or ApsaraDB RDS for MariaDB instance from a local server over the Internet?

> **Note**
>
> - If you add multiple IP addresses and CIDR blocks to an IP address whitelist, you must separate the IP addresses and CIDR blocks with commas (,) and leave no spaces before and after each comma.
> - You can add a maximum of 1,000 IP addresses and CIDR blocks in total for each RDS instance. If you want to add a large number of IP addresses, we recommend that you merge the IP addresses into CIDR blocks, such as 10.10.10.0/24.
> - If an RDS instance runs in standard whitelist mode, you do not need to take note of special considerations when you configure IP address whitelists for the RDS instance. **If an RDS instance runs in enhanced whitelist mode, you must take note of the following considerations when you configure IP address whitelists for the RDS instance:**
>   - Add **public IP addresses** or the private IP addresses of -hosted ECS instances to IP address whitelists of the **classic network type**. classic network
>   - Add the private IP addresses of VPC-hosted ECS instances to IP address whitelists of the **VPC network type**.

6. Click **OK**.

Step 3: Connect to the RDS instance

To connect to the RDS instance by using the CLI, perform the following steps:

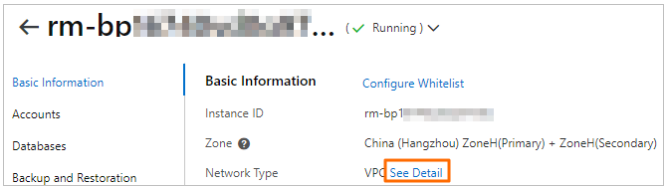
1. Log on to the server from which you want to connect to the RDS instance. For example, the server can be an ECS instance or an on-premises device.

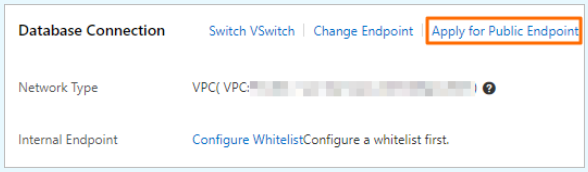
   > **Note** For more information about how to log on to an ECS instance, see the "**Connect to an instance**" section in Create and manage an ECS instance by using the ECS console (express version).

2. Run the following command:

   ```
   mysql -hEndpoint -PPort number -uUsername -p      //Take note that the uppercase letter P precedes the lowercase letter p.
   ```

   - Endpoint and port number: Enter the endpoint and port number that are used to connect to the RDS instance.

<table>
<tr><th>Connection scenario</th><th>Endpoint to be obtained</th><th>How to obtain the endpoint</th></tr>
<tr><td>You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance meet the conditions for communication over an internal network. For more information, see the "Step 1: Check whether your application can connect to the RDS instance over an internal network" section of this topic.</td><td>The internal endpoint of the RDS instance</td><td rowspan="2">a.
b. In the Basic Information section of the page that appears, click See Details to the right of the Network Type parameter to view the endpoint and port number that are used to connect to the RDS instance.
 

Note
■ Before you can view the endpoint and port number that are used to connect to the RDS instance, you must configure IP address whitelists for the RDS instance.
■ A public endpoint is displayed only after you click Apply for Public Endpoint to apply for a public endpoint for the RDS instance.
 
</td></tr>
<tr><td>You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance do not meet the conditions for communication over an internal network.</td><td>The public endpoint of the RDS instance</td></tr>
</table>

Connection scenario	Endpoint to be obtained	How to obtain the endpoint
Connect to the RDS instance from an on-premises device.		

- Username and password: Obtain the username and password of the account that is used to connect to the RDS instance from the page. Accounts

  Example

  ```
  root@ -> mysql -h .mysql.rds.aliyuncs.com -P3306 -u -p
  Enter password:
  ```

  Successful connection

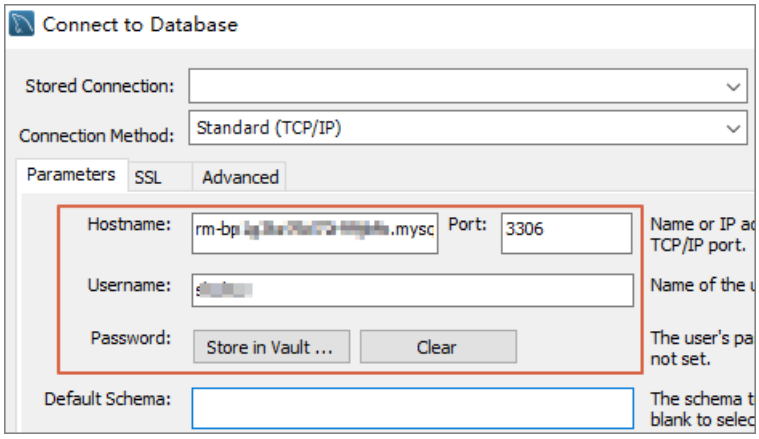
  ```
  Welcome to the MySQL monitor.  Commands end with ; or \g.
  Your MySQL connection id is 51325
  Server version: 8.0.18 Source distribution
  ```

  > **Note** If connection errors occur, you can troubleshoot the errors by following the instructions provided in Common connection errors.

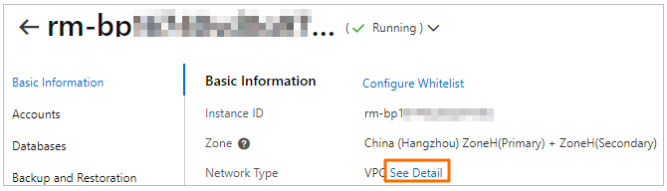
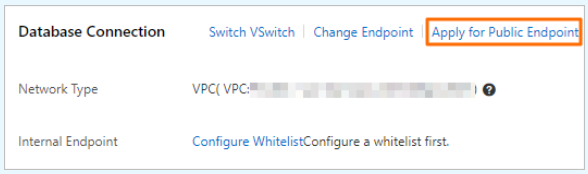
To connect to the RDS instance by using a database client, perform the following steps:

You can use a general-purpose MySQL client to connect to the RDS instance. In this example, MySQL Workbench is used. The methods of using other database clients to connect to the RDS instance are similar.

1. Go to the MySQL Community Downloads page, select the MySQL Workbench software package that can be used with your operating system, and then click **Download**.
2. Install MySQL Workbench.
3. Start MySQL Workbench and choose **Database > Connect to Database**.
4. Enter the information that is used to connect to the RDS instance.



- **Hostname** and **Port**: Enter the endpoint and port number that are used to connect to the RDS instance.

Connection scenario	Endpoint to be obtained	How to obtain the endpoint
You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance meet the conditions for communication over an internal network. For more information, see the "Step 1: Check whether your application can connect to the RDS instance over an internal network" section of this topic.	The internal endpoint of the RDS instance	a. b. In the Basic Information section of the page that appears, click **See Details** to the right of the Network Type parameter to view the endpoint and port number that are used to connect to the RDS instance.  **Note** ■ Before you can view the endpoint and port number that are used to connect to the RDS instance, you must configure IP address whitelists for the RDS instance. ■ A public endpoint is displayed only after you click **Apply for Public Endpoint** to apply for a public endpoint for the RDS instance. 
You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance do not meet the conditions for communication over an internal network.	The public endpoint of the RDS instance	

Connection scenario	Endpoint to be obtained	How to obtain the endpoint
Connect to the RDS instance from an on-premises device.		

- **Username** and **Password**: Obtain the username and password of the account that is used to connect to the RDS instance from the page. Accounts

Common connection errors

Error message	Cause and solution
mysql command not found	MySQL is not installed. Run the following commands to install MySQL: • If you use a CentOS operating system, run the `yum install mysql` command. • If you use an Ubuntu operating system, run the `apt-get update` command and then the `apt install mysql-server` command.
SSL connection error: SSL is required but the server doesn't support it	You are using the latest version of MySQL Workbench. In this version, standard TCP/IP connections require SSL encryption. However, the connected server does not support SSL encryption. In this case, you can download an earlier version of MySQL Workbench to establish regular connections.
Can't connect to MySQL server on 'rm-bp1xxxxxxxxxxxxx.mysql.rds.aliyuncs.com'(10060) Cannot Connect to Database Server Your connection attempt failed for user 'xx" to the MySQL server	• In most cases, this error occurs because the IP address whitelists that you configured are inappropriate. For more information, see the "**Step 2: Configure IP address whitelists for the RDS instance** section of this topic. • In a few cases, this error occurs because the RDS instance and the ECS instance do not meet the but you attempt to connect to the internal endpoint of the RDS instance. conditions for communication over an internal network
Access denied for user 'xxxxx'@'xxxxx'(using password:YES)	This error occurs because the username and password that you entered are incorrect. You can obtain the correct username and password from the page. Accounts
Unknown MySQL server host 'xxxxxxxxx'(11001)	This error occurs because the endpoint that you entered is invalid. Valid endpoints are in the rm-xxxxxx.mysql.rds.aliyuncs.com format.

References

- For more information about how to troubleshoot connection errors, see What do I do if I cannot connect an ECS instance to an ApsaraDB for RDS instance?
- For more information about how to connect to an RDS instance in a more convenient and efficient manner, see Use DMS to log on to an ApsaraDB RDS for MySQL instance.
- For more information about how to connect to an RDS instance that runs a different database engine, see the following topics:
  - Connect to an ApsaraDB RDS for SQL Server instance
  - Connect to an ApsaraDB RDS for PostgreSQL instance
  - Connect to an ApsaraDB RDS for MariaDB TX instance

9.2. Apply for or release a public endpoint for an ApsaraDB RDS for MySQL instance

ApsaraDB RDS supports two types of endpoints: internal endpoints and public endpoints. By default, you are provided with an internal endpoint that is used to connect to your ApsaraDB RDS for MySQL instance. If you want to connect to your RDS instance over the Internet, you must apply for a public endpoint.

For more information about how to apply for or release public endpoints for RDS instances that run other database engines, see the following topics:

- Apply for or release a public endpoint for an ApsaraDB RDS for SQL Server instance
- Apply for or release a public endpoint for an ApsaraDB RDS for PostgreSQL instance
- Apply for or release a public endpoint for an ApsaraDB RDS for MariaDB TX instance

Internal and public endpoints

Endpoint type	Description
Internal endpoint	• An internal endpoint is provided by default. You do not need to apply for this endpoint. In addition, you cannot release this endpoint. However, you can change the network type of your RDS instance. • If an Elastic Compute Service (ECS) instance resides in the same region and has the same network type as your RDS instance, these instances can communicate over an internal network. If your application is deployed on such an ECS instance, you do not need to apply for a public endpoint. For more information, see Change the network type of an ApsaraDB RDS for MySQL instance. • For security and performance purposes, we recommend that you connect to your RDS instance by using the internal endpoint.

Endpoint type	Description

Public endpoint	• You must manually apply for a public endpoint. You can release this endpoint if it is no longer required.
	• If you cannot connect to your RDS instance by using the internal endpoint, you must apply for a public endpoint. This includes the following scenarios:
	  ◦ Connect to your RDS instance from an ECS instance that resides in a different region or has a different network type from your RDS instance. For more information, see Change the network type of an ApsaraDB RDS for MySQL instance.
	  ◦ Connect to your RDS instance from a device that resides outside Alibaba Cloud.
	Note
	• You are not charged for the public endpoint or the traffic that is consumed.
	• If you connect to your RDS instance by using the public endpoint, security is compromised. Proceed with caution.
	• We recommend that you migrate your application to an ECS instance that resides in the same region and has the same network type as your RDS instance. This allows you to connect to your RDS instance by using the internal endpoint. The connection expedites transmission and improves security.

Procedure

1.
2. In the left-side navigation pane, click **Database Connection**.
3. Apply for or release a public endpoint for your RDS instance:
   - If you have not applied for a public endpoint, you can click **Apply for Public Endpoint**.
   - If you have applied for a public endpoint, you can click **Release Public Endpoint**.
4. In the message that appears, click **OK**.

FAQ

- Can I change the endpoints and ports of my RDS instance?

  No, you cannot change the endpoints of your RDS instance. You can change the prefixes of the endpoints. In addition, you can change the ports of your RDS instance. For more information, see View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance.

- Can I configure the endpoints of my RDS instances to static IP addresses?

  No, you cannot configure the endpoints of your RDS instance to static IP addresses. Both primary/secondary switchovers and specification changes may cause changes to the IP addresses. Therefore, we recommend that you connect to your RDS instance by using an endpoint. This allows you to minimize the impact on your workloads and relieves the need to modify the configuration data on your application.

- How do I connect to my RDS instance by using the public endpoint?

  For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.

Related operations

Operation	Description
Apply for a public endpoint	Applies for a public endpoint for an ApsaraDB RDS instance.
Release a public endpoint	Releases the public endpoint of an ApsaraDB RDS instance.

9.3. Use DMS to log on to an ApsaraDB RDS for MySQL instance

This topic describes how to log on to an ApsaraDB RDS for MySQL instance by using Data Management (DMS).

Prerequisites

An Alibaba Cloud account or a RAM user that has relevant permissions on your RDS instance is prepared. For more information about how to apply for permissions, see Permission management.

Context

DMS offers an integrated solution that supports data management, schema management, server management, user authorization, security audit, trend analysis, data tracking, business intelligence (BI) charting, and performance analysis and optimization.

Procedure

1.
2. In the upper-right corner of the page that appears, click **Log On to Database** to go to the RDS Database Logon page of the DMS console.
3. In the **Log on to Database Instance** dialog box, enter the username and password of the account that is used for the logon and click **Login**.

   > **Note**
   > - The account used for the logon must have permissions on the required database. Otherwise, the required database is not displayed in the left-side navigation pane. For more information about how to modify the permissions of an account, see Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance.
   > - For more information about how to create an account, see Create databases and accounts for an ApsaraDB RDS for MySQL instance.

4. In the left-side navigation pane, click **Instances Connected**, click the ID of your RDS instance, and then double-click the name of the specified database to switch to that database.

> **Note** You can also log on to the DMS console and add your RDS instance to DMS. Then, you can switch to the specified database of your RDS instance in the DMS console. For more information, see Register an ApsaraDB instance.

9.4. Use an application to connect to an ApsaraDB RDS for MySQL instance

This topic describes how to connect to an ApsaraDB RDS for MySQL instance by using a Java, Python, or C application.

Parameter description

The following table describes the parameters in the sample code.

Parameter	Description
Host	The internal or public endpoint of the RDS instance. • If the application runs on an Elastic Compute Service (ECS) instance that resides in the same region and has the same network type as the RDS instance, use the internal endpoint. For example, if the ECS and RDS instances both reside in virtual private clouds (VPCs) of the China (Hangzhou) region, you can use the internal endpoint to establish a secure and efficient connection. • In the other scenarios, use the public endpoint. For more information about how to view the internal and public endpoints and port numbers of an RDS instance, see View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance.
Port	The port number of the RDS instance. If you want to connect to the RDS instance over an internal network, enter the internal port number of the RDS instance. If you want to connect to the RDS instance over the Internet, enter the public port number of the RDS instance.
myDatabase	The name of the destination database on the RDS instance.
myUsername	The username of the account that is used to connect to the RDS instance.
myPassword	The password of the preceding account.

Sample code

- Java sample code

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
public class DatabaseConnection
{
    public static void main(String args[]) {
        String connectionUrl= "jdbc:mysql://<Host>:<Port>/<myDatabase>";
        ResultSet resultSet;
        try (Connection connection=DriverManager.getConnection(connectionUrl,"<myUsername
>","<myPassword>");
             Statement statement = connection.createStatement()) {
            String selectSql = "SELECT * FROM `courses`";            //Enter the SQL stat
ement that you want to execute.
            resultSet = statement.executeQuery(selectSql);
            while (resultSet.next()) {
                System.out.println(resultSet.getString("name"));
            }
        }
        catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

- Python sample code

```
import pymysql
connection = pymysql.connect(host='<Host>',
                      port=<Port>,
                      user='<myUsername>',
                      passwd='<myPassword>',
                      db='<myDatabase>')
try:
    with connection.cursor() as cursor:
        sql = "SELECT * FROM `courses`"        //Enter the SQL statement that you want to
execute.
        cursor.execute(sql)
        for result in cursor:
             print(result)
finally:
    connection.close()
```

- C sample code:

```
#include <stdio.h>
#include <mysql.h>
#include <string.h>
void main(void)
{
    MYSQL *t_mysql;
    MYSQL_RES       *res = NULL;
    MYSQL_ROW       row;
    char            *query_str = NULL;
    int             rc, i, fields;
    int             rows;
    char select[] = "select * from courses";    //Enter the SQL statement that you want t
o execute.
    t_mysql = mysql_init(NULL);
    if(NULL == t_mysql){
        printf("init failed\n");
    }
    if(NULL == mysql_real_connect(t_mysql, <Host>, <myUsername>, <myPassword>, <myDatabas
e>,
            <Port>, NULL, 0)){
        printf("connect failed\n");
    }
    if(mysql_real_query(t_mysql, select, strlen(select)) != 0){
        printf("select failed\n");
    }
    res = mysql_store_result(t_mysql);
    if (NULL == res) {
         printf("mysql_restore_result(): %s\n", mysql_error(t_mysql));
         return -1;
    }
    fields = mysql_num_fields(res);
    while ((row = mysql_fetch_row(res))) {
        for (i = 0; i < fields; i++) {
            printf("%s\t", row[i]);
        }
        printf("\n");
    }
    mysql_close(t_mysql);
}
```

Troubleshooting

If the connection fails, we recommend that you troubleshoot the failure based on the returned error information. For more information, see Resolve the issue that you cannot connect to an RDS instance.

9.5. View and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance

When you connect to an ApsaraDB RDS for MySQL instance, you must enter the internal or public endpoint and port number of the instance. This topic describes how to view and change the internal and public endpoints and port numbers of an ApsaraDB RDS for MySQL instance in the ApsaraDB RDS console.

View the internal and public endpoints and port numbers of an RDS instance in the new ApsaraDB RDS console

1.
2. In the **Basic Information** section of the Basic Information page, click **See Detail** next to Network Type. In the pane that appears, view the internal and public endpoints and port numbers of the RDS instance.

> **Note**
> - The internal and public endpoints of an RDS instance are displayed only after you configure IP address whitelists for the instance. For more information, see Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance.
> - The public endpoint of an RDS instance is displayed only after you apply for a public endpoint for the instance. For more information, see Apply for or release a public endpoint for an ApsaraDB RDS for MySQL instance.

Change the internal or public endpoint and port number of an RDS instance

1.
2. In the left-side navigation pane, click **Database Connection**.
3. Click **Change Endpoint**.
4. In the dialog box that appears, select a connection type, enter the prefix of the new endpoint, specify the port number, and then click **OK**.

> **Note**
> - The prefix can contain lowercase letters, digits, and hyphens (-). The prefix must start with a lowercase letter and end with a lowercase letter or a digit.
> - The prefix must contain at least 8 characters, and the total length of the endpoint cannot exceed 63 characters. The total length includes the prefix and suffix of the endpoint.
> - The port number must be within the range of 1000 to 65534.

FAQ

- After I change an endpoint or a port number of my RDS instance, do I need to update the endpoint or port number information in my application?

  Yes, after you change an endpoint or a port number of your RDS instance, you must update the endpoint or port number information on your application. If you do not update the information, your application cannot connect to your RDS instance.

- After I change an endpoint or a port number of my RDS instance, does the change immediately take

effect? Do I need to restart my RDS instance?

After you change an endpoint or a port number of your RDS instance, the change immediately takes effect. You do not need to restart your RDS instance.

- After I change or release an endpoint of my RDS instance, can I use the endpoint for another RDS instance?

  Yes, after you change or release an endpoint of your RDS instance, you can use the endpoint of your RDS instance for another RDS instance.

- Does a primary/secondary switchover trigger changes to the endpoints of my RDS instance?

  No, a primary/secondary switchover does not trigger changes to the endpoints of your RDS instance. However, the IP addresses that are associated with the endpoints change. Your application can still connect to your RDS instance by using the endpoints.

References

For more information about the endpoints that are used to connect to the dedicated proxy of an RDS instance, see Manage the dedicated proxy endpoints of an ApsaraDB RDS for MySQL instance.

9.6. Change the network type of an ApsaraDB RDS for MySQL instance

This topic describes how to change the network type of an ApsaraDB RDS for MySQL instance from classic network to virtual private cloud (VPC) based on your business requirements.

Prerequisites

Your RDS instance resides in the classic network.

For more information about how to change the network type of an RDS instance that runs a different database engine, see the following topics:

- Change the network type of an ApsaraDB RDS for SQL Server instance
- Change the network type of an ApsaraDB RDS for PostgreSQL instance

Impacts

The following table describes the impacts that a change in the network type may bring if the database proxy feature is enabled for your RDS instance. For more information, see Introduction to database proxies.

Proxy type	Impact

Proxy type	Impact
Shared proxy	After you change the network type of your RDS instance, the network type of the read/write splitting endpoint changes. For more information, see Read/write splitting. **Note** Since April 1, 2021, Alibaba Cloud has stopped the updates and maintenance for the shared proxy feature of ApsaraDB RDS for MySQL. For more information, see [Notice] End of updates and maintenance for the shared proxy service. We recommend that you upgrade the database proxy of your RDS instance from shared proxy to dedicated proxy at the earliest opportunity to use more features of higher quality. For more information, see Upgrade the database proxy of an ApsaraDB RDS for MySQL instance from a shared proxy to a dedicated proxy.
Dedicated proxy	After you change the network type of your RDS instance, the network type of the read/write splitting endpoint remains unchanged. For more information, see Read/write splitting. You can create proxy endpoints of different network types. For example, you can create a proxy endpoint of the classic network type and a proxy endpoint of the VPC network type on the same RDS instance. For more information, see Manage the dedicated proxy endpoints of an ApsaraDB RDS for MySQL instance.

Note You can view the type of proxy that is enabled for your RDS instance on the **Database Proxy** page in the ApsaraDB RDS console. For more information, see Upgrade the database proxy of an ApsaraDB RDS for MySQL instance from a shared proxy to a dedicated proxy.

Network types

- Classic network: RDS instances in the classic network are not isolated. To block unauthorized access to these instances, you must configure IP address whitelists or security groups.
- VPC: Each VPC is an isolated virtual network. VPCs provide higher security than the classic network. We recommend that you select the VPC network type.

  You can configure route tables, CIDR blocks, and gateways in a VPC. In addition, you can connect your data center to a VPC by using Express Connect circuits or VPNs. The data center and the VPC comprise a virtual data center. You can use the virtual data center to migrate your workloads to the cloud with no downtime.

Note

- You can select the classic or VPC network type and switch your RDS instance between these network types free of charge.
- After you change the network type of your RDS instance, you must add IP addresses to the IP address whitelists of the required network types. This applies if your RDS instance runs in enhanced whitelist mode. For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.

View the network type

1.
2. In the left-side navigation pane, click **Database Connection**. On the page that appears, view the network type of the RDS instance.

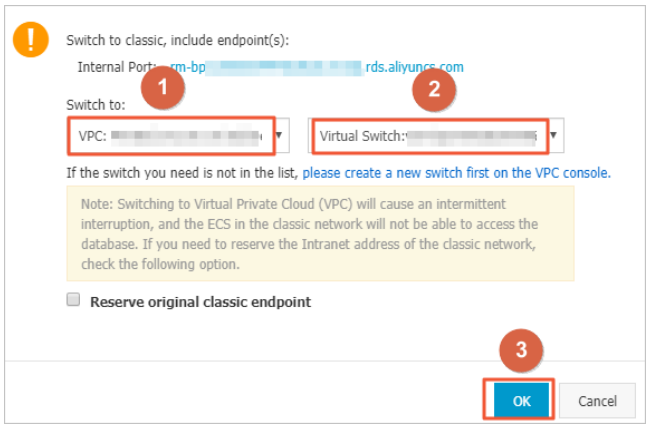
Change the network type from classic network to VPC

Procedure

1.
2. In the left-side navigation pane, click **Database Connection**.
3. Click **Switch to VPC**.

   > **Note** If the preceding button cannot be found, you must check whether the RDS instance meets all prerequisites that are specified in this topic.

4. In the dialog box that appears, select a VPC and a vSwitch and specify whether to retain the classic network endpoint.
   - Select a VPC. We recommend that you select the VPC where the Elastic Compute Service (ECS) instance that you want to connect resides. If the ECS instance and the RDS instance reside in different VPCs, these instances cannot communicate over an internal network unless you use Cloud Enterprise Network (CEN) or VPN Gateway to enable network communication between the VPCs of these instances. For more information, see Overview of CEN or Establish IPsec-VPN connections between two VPCs.
   - Select a vSwitch. If no vSwitches are available in the selected VPC, create a vSwitch in the zone where the RDS instance resides. For more information, see Create a vSwitch.



   - Clear or select the **Reserve original classic endpoint** check box. For more information, see the following table.

Operation	Description
Clear the Reserve original classic endpoint check box	The classic network endpoint is not retained and changes to a VPC endpoint. When you change the network type from classic network to VPC, a transient connection that lasts approximately 30 seconds occurs and ECS instances that reside in the classic network are immediately disconnected from the RDS instance.

Operation	Description
Select the Reserve original classic endpoint check box	The classic network endpoint is retained, and a new VPC endpoint is generated. In this case, the RDS instance runs in hybrid access mode. Classic network-housed ECS instances and VPC-housed ECS instances can connect to the RDS instance over an internal network. For more information, see Configure the hybrid access solution for an ApsaraDB RDS for MySQL instance. When you change the network type from classic network to VPC, no transient connection occurs. The connection between each classic network-housed ECS instance and the RDS instance remains available until the classic network endpoint expires. Before the classic network endpoint expires, add the VPC endpoint to your application that runs on a VPC-hosted ECS instance. This allows ApsaraDB RDS to migrate your workloads to the selected VPC with no downtime. For more information, see Configure the hybrid access solution for an ApsaraDB RDS for MySQL instance.

5. Add the private IP address of the required VPC-hosted ECS instance to an IP address whitelist of the VPC network type on the RDS instance. This way, the ECS instance can access the RDS instance over an internal network. If no IP address whitelists of the VPC network type are available, create one.

   > **Note** You can go to the **Instance Details** tab of an ECS instance in the ECS console to view the private IP address of the ECS instance.

6. Add the VPC endpoint of the RDS instance to the required VPC-hosted ECS instance.
   - If you selected the Reserve original classic endpoint check box, you must add the VPC endpoint to your application that runs on the required VPC-hosted ECS instance before the classic network endpoint expires.
   - If you cleared the Reserve original classic endpoint check box, the connection between each classic network-hosted ECS instance and the RDS instance over an internal network is immediately closed after the network type is changed. You must add the VPC endpoint of the RDS instance to your application that runs on the required VPC-hosted ECS instance.

   > **Note** If the RDS instance resides in a VPC and you want to connect a classic network-hosted ECS instance to the RDS instance over an internal network, you can use ClassicLink to establish a connection. Alternatively, you can migrate the ECS instance to the same VPC as the RDS instance. For more information, see Overview of ClassicLink.

FAQ

- How do I change the VPC of my RDS instance?
  - If your RDS instance supports changes to VPCs and vSwitches, you can directly change the VPC of the RDS instance. For more information, see 切换专有网络VPC和虚拟交换机.
  - If your RDS instance does not support direct changes to VPCs, perform the following steps:

    Purchase a new RDS instance that resides in the required VPC. Then, migrate the data of the original RDS instance to the new RDS instance. For more information, see Migrate data between ApsaraDB RDS for MySQL instances.

- Can I connect to my RDS instance from an ECS instance over the Internet?

  Yes, you can connect to your RDS instance from an ECS instance over the Internet if the IP address of the ECS instance is added to an IP address whitelist of the RDS instance, regardless of whether your application resides in a VPC or the classic network. For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.

- Can I change the network type of a read-only RDS instance and retain the classic network endpoint?

  Yes, you can change the network type of a read-only RDS instance and retain the classic network endpoint.

Related operations

Operation	Description
Change the network type of an ApsaraDB RDS instance	Changes the network type of an ApsaraDB RDS instance.

9.7. Configure the hybrid access solution for an ApsaraDB RDS for MySQL instance

This topic describes how to configure the hybrid access solution for an ApsaraDB RDS for MySQL instance. This solution allows you to retain both the classic network endpoint and virtual private cloud (VPC) endpoint of your RDS instance. This way, you can migrate your RDS instance from the classic network to a VPC without network interruptions.

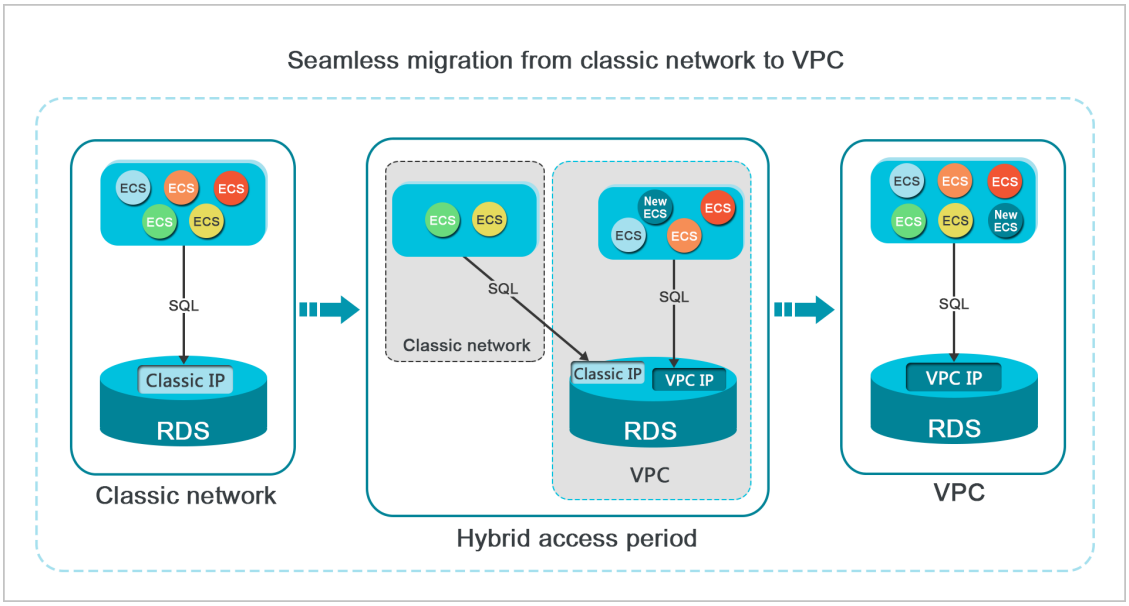
Background information

When you migrate your RDS instance from the classic network to a VPC, the internal classic network endpoint of the instance changes to the internal VPC endpoint. In this case, the endpoint itself remains unchanged, but the IP address that is bound to the endpoint changes. This change causes a transient connection error of up to 30 seconds, and all classic network-housed Elastic Compute Service (ECS) instances can no longer connect to your RDS instance over an internal network. To allow you to migrate your RDS instance from the classic network to a VPC without network interruptions, ApsaraDB RDS provides the hybrid access solution.

Hybrid access refers to the ability of your RDS instance to be connected by both classic network-housed ECS instances and VPC-housed ECS instances. During the hybrid access period, ApsaraDB RDS retains the internal classic network endpoint and generates an internal VPC endpoint. This prevents transient connection errors when you migrate your RDS instance from the classic network to a VPC.

For security and performance purposes, we recommend that you use only the internal VPC endpoint. Therefore, ApsaraDB RDS allows the configured hybrid access solution to remain valid only for a specified period of time. When the hybrid access period elapses, ApsaraDB RDS releases the internal classic network endpoint. In this case, your applications cannot connect to your RDS instance by using the internal classic network endpoint. You must add the internal VPC endpoint to all your applications during the hybrid access period. This ensures a smooth network migration and avoids interruptions to your workloads.

For example, a company uses the hybrid access solution to migrate their RDS instance from the classic network to a VPC. During the hybrid access period, some applications connect to the RDS instance by using the internal VPC endpoint, whereas the others connect to the RDS instance by using the internal classic network endpoint. When all applications of the company can connect to the RDS instance by using the internal VPC endpoint, the internal classic network endpoint can be released.



Limits

During the hybrid access period, your RDS instance does not support the following operations:

- Change to the classic network type
- Migration to another zone
- Change between the High-availability Edition and the Enterprise Edition

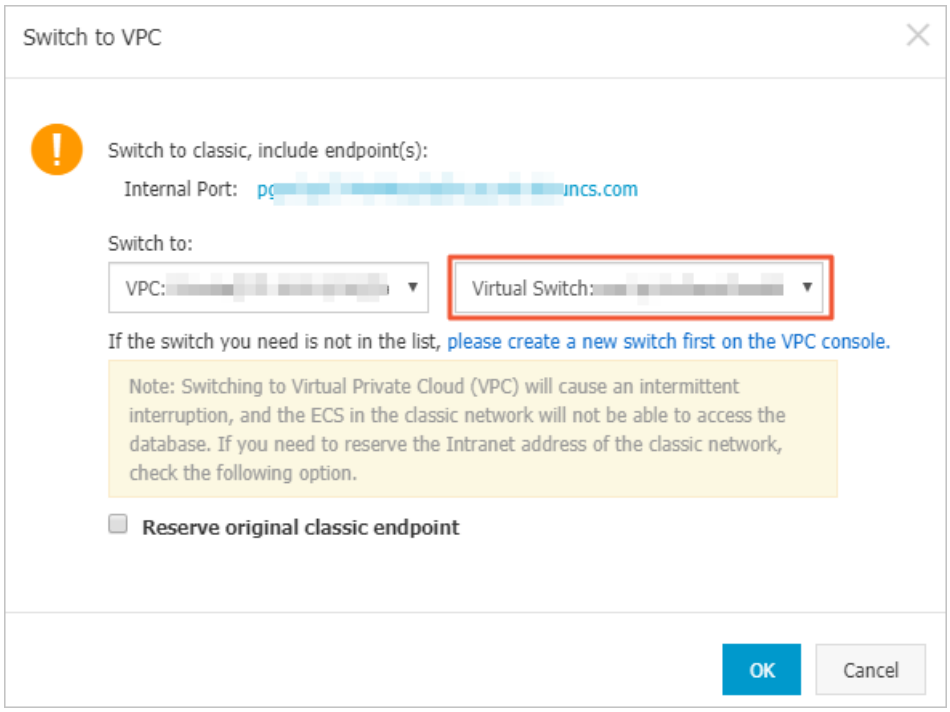
Prerequisites

- Your RDS instance has the classic network type.
- The zone where your RDS instance resides provides available VPCs and vSwitches. For more information about how to create VPCs and vSwitches, see Manage a VPC.

Change the network type from classic network to VPC

1.
2. In the left-side navigation pane, click **Database Connection**.
3. Click **Switch to other VPC**.
4. In the dialog box that appears, select a VPC and a vSwitch and specify whether to retain the classic network endpoint.
   - Select a VPC. We recommend that you select the VPC where the required ECS instance resides. If the ECS and RDS instances reside in different VPCs, these instances cannot communicate over an internal network unless you create a Cloud Enterprise Network (CEN) instance or an IPsec-VPN connection between the VPCs of these instances. For more information, see Use CEN to enable intra-region network communication and Establish IPsec-VPN connections between two VPCs.
   - Select a vSwitch. If no vSwitches are available in the selected VPC, create a vSwitch in the same

zone as your RDS instance. For more information, see Create a vSwitch.



- Clear or select the **Reserve original classic endpoint** option. For more information, see the following table.

Action	Description
Clear the Reserve original classic endpoint option	The classic network endpoint is not retained and changes to a VPC endpoint. When you change the network type from classic network to VPC, a transient connection error of 30 seconds occurs. In this case, the connection between each classic network-housed ECS instance and your RDS instance is closed.
Select the Reserve original classic endpoint option	The classic network endpoint is retained, and a new VPC endpoint is generated. In this case, your RDS instance runs in hybrid access mode. Both classic network-housed ECS instances and VPC-housed ECS instances can connect to your RDS instance over an internal network. When you change the network type from classic network to VPC, no transient connection errors occur. The connection between each classic network-housed ECS instance and your RDS instance remains available until the classic network endpoint expires. Before the classic network endpoint expires, you must add the VPC endpoint to the required VPC-housed ECS instance. This allows ApsaraDB RDS to migrate your workloads to the selected VPC without interruptions.

5. Add the private IP address of the required VPC-housed ECS instance to an IP address whitelist of the VPC network type. This allows the ECS instance to connect to your RDS instance over an internal network. If no IP address whitelists of the VPC network type are available, create such an IP address whitelist.
6. - If you have selected the Reserve original classic endpoint option, you must add the generated

VPC endpoint to each VPC-housed ECS instance before the classic network endpoint expires.

- If you have cleared the Reserve original classic endpoint option, the connection between each classic network-housed ECS instance and your RDS instance over an internal network is immediately closed after the network type is changed to VPC. You must add the generated VPC endpoint to each VPC-housed ECS instance.

> **Note** If you want to connect a classic network-housed ECS instance to your VPC-housed RDS instance over an internal network, you can use ClassicLink to establish a connection. Alternatively, you can migrate the ECS instance to the same VPC as your RDS instance. For more information, see Overview.

Change the expiration date of the internal classic network endpoint

During the hybrid access period, you can change the expiration date of the classic network endpoint at any time based on your business requirements. The expiration date is immediately recalculated starting from the day when you make the change. Assume that the classic network endpoint is configured to expire on August 18, 2017, and you extend the validity period of the classic network endpoint by 14 days on August 15, 2017. In this case, ApsaraDB RDS releases the classic network endpoint on August 29, 2017.

Perform the following steps:

1.
2. In the left-side navigation pane, click **Database Connection**.
3. On the **Database Connection** tab, click **Change Expiration Time**.

4. On the **Change Expiration Time** page, select an expiration date and click **OK**.

9.8. Troubleshoot failures in connecting to an ApsaraDB RDS for MySQL instance

This topic describes how to troubleshoot failures in connecting to an ApsaraDB RDS for MySQL instance from an Elastic Compute Service (ECS) instance.

When you set up a test environment to debug your business, you may fail to connect to your RDS instance from your ECS instance. The connection failures may occur due to various reasons. For example, the network type of your RDS instance is different from the network type of your ECS instance, or the IP address of your ECS instance is not added to an IP address whitelist of your RDS instance. This topic describes the most common causes of connection failures and the methods that you can use to troubleshoot the connection failures.

Different network types

- The ECS instance resides in a virtual private cloud (VPC), and the RDS instance resides in the classic network.

- Method 1: This is the recommended method. Migrate the RDS instance to the VPC to which the ECS instance belongs. For more information, see Switch the network type.

  > **Note** If the ECS instance and the RDS instance both reside in VPCs, they must reside in the same VPC to communicate with each other over an internal network.

- Method 2: Purchase an ECS instance that resides in the classic network, and connect to the RDS instance from the ECS instance that you purchase. ECS instances cannot be migrated from VPCs to the classic network. Take note that a VPC provides higher security than the classic network. We recommend that you use VPCs.
- Method 3: Connect to the RDS instance from the ECS instance by using the public endpoint of the RDS instance. This method cannot ensure optimal performance, security, or stability.

- The ECS instance resides in the classic network, and the RDS instance resides in a VPC.
  - Method 1: This is the recommended method. Migrate the ECS instance to the VPC to which the RDS instance belongs.

    > **Note** If the ECS instance and the RDS instance both reside in VPCs, they must reside in the same VPC to communicate with each other over an internal network.

  - Method 2: Migrate the RDS instance to the classic network. Take note that a VPC provides higher security than the classic network. We recommend that you use VPCs.
  - Method 3: Use the ClassicLink feature to establish an internal network connection between the ECS instance and the RDS instance.
  - Method 4: Connect to the RDS instance from the ECS instance by using the public endpoint of the RDS instance. This method cannot ensure optimal performance, security, or stability.

Different VPCs

A VPC is an isolated network environment that is built on Alibaba Cloud. VPCs are logically isolated from each other. Therefore, when the ECS instance and the RDS instance both reside VPCs, they must reside in the same VPC to communicate with each other over an internal network.

- Method 1: This is the recommended method. Migrate the RDS instance to the VPC to which the ECS instance belongs.

  You must change the network type of the RDS instance from VPC to classic network and then change the network type of the RDS instance from classic network back to VPC. When you change the network type of the RDS instance from classic network back to VPC, you must select the VPC to which the ECS instance belongs. For more information, see 切换专有网络VPC和虚拟交换机 or Switch the network type.

- Method 2: Use Cloud Enterprise Network (CEN) to establish a connection between the VPC of the ECS instance and the VPC of the RDS instance.
- Method 3: Connect to the RDS instance from the ECS instance over the Internet. This method cannot ensure optimal performance, security, or stability.

Different regions

If the ECS instance and the RDS instance reside in different regions, these instances cannot communicate with each other over an internal network.

- Method 1: Apply for a refund for the original RDS or ECS instance. Then, purchase a new RDS or ECS

instance based on your business requirements.

- Method 2: Change the network types of the ECS instance and the RDS instance to VPC. Then, use CEN to establish a connection between the VPCs of the ECS instance and the VPC of the RDS instance.
- Method 3: Connect to the RDS instance from the ECS instance over the Internet. This method cannot ensure optimal performance, security, or stability.

Incorrect IP address whitelist settings

- On the **Whitelist Settings** tab of the **Data Security** page, the IP address whitelist labeled default contains only the IP address 127.0.0.1. The IP address 127.0.0.1 indicates that no devices are allowed to access the RDS instance. You must obtain the IP address of the ECS instance and add the IP address to an IP address whitelist of the RDS instance. For more information, see Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance.
- The 0.0.0.0 entry is added to an IP address whitelist of the RDS instance.

  > **Note** If you want to allow all devices to access the RDS instance, you must add the 0.0.0.0/0 entry to an IP address whitelist of the RDS instance. Proceed with caution when you add this entry.

- The enhanced whitelist mode is enabled for the RDS instance. In this case, take note of the following limits:
  - If the RDS instance resides in a VPC and is connected by using its internal endpoint, the private IP address of the ECS instance must be added to the IP address whitelist labeled default VPC.
  - If the RDS instance resides in the classic network and is connected by using its internal endpoint, the private IP address of the ECS instance must be added to the IP address whitelist labeled default Classic Network.
  - If the RDS instance resides in the classic network and is connected over the Internet, the public IP address of the ECS instance must be added to the IP address whitelist labeled default Classic Network.
- The public IP address that you add to an IP address whitelist is invalid due to the following reasons:
  - The public IP address dynamically changes.
  - The tool or website that is used to query public IP addresses returns inaccurate results.

  For more information, see the following topics:

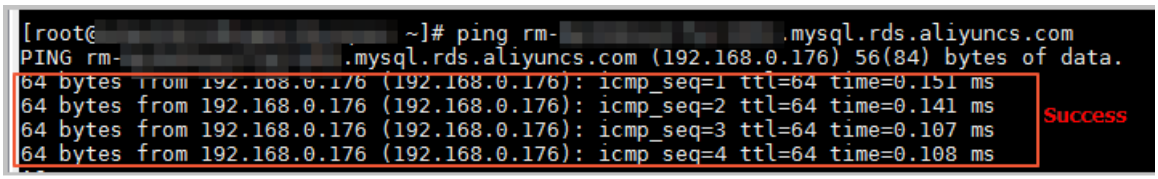
  - Why am I unable to connect to my ApsaraDB RDS for MySQL or ApsaraDB RDS for MariaDB instance from a local server over the Internet?
  - How SQL Server determines the public IP address of an external Server or client
  - How do I locate the IP address connected to an RDS for PostgreSQL instance?

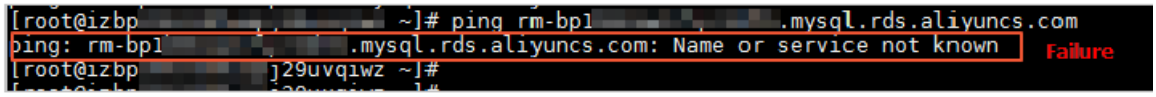
Domain name resolution failures or errors

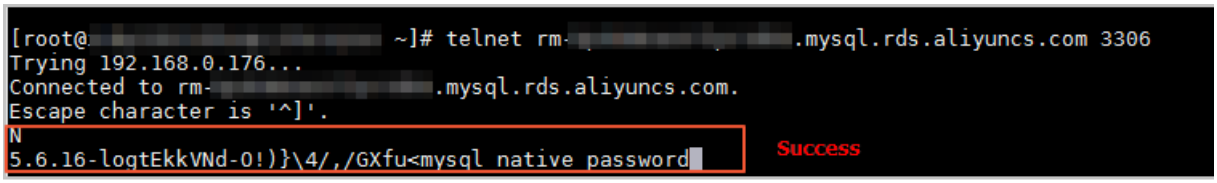
If the DNS servers are faulty or the configurations of the network interface controller are modified, domain names may fail to be resolved or may be resolved into incorrect IP addresses. In this case, you can run the `ping` command or the `telnet` command to check the connectivity to the RDS instance.

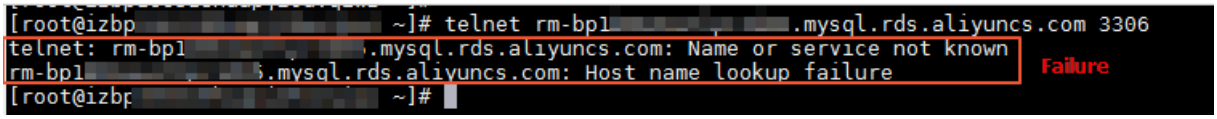
```
ping <Domain name>
telnet <Domain name> <Port number>
```

Examples:









If the RDS instance fails the connectivity test, perform the following operations to modify the configuration file of the network interface controller:

1. Open the configuration file of the network interface controller in edit mode.

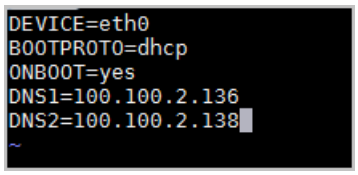
   ```
   vi /etc/sysconfig/network-scripts/<The name of the configuration file of the network interface controller>
   ```

   > **Note** The network interface controller in the preceding command refers to the network interface controller of the ECS instance. You can run the `ifconfig` command to check the extension in the name of the configuration file of the network interface controller. The default extension is ifcfg-eth0.

2. Add the following configurations to the end of the configuration file.

   ```
   DNS1=100.100.2.136
   DNS2=100.100.2.138
   ```

   > **Note** If the DNS1 and DNS2 configuration items exist in the configuration file, you must change the values of these configuration items to the values that are shown in the preceding configurations.

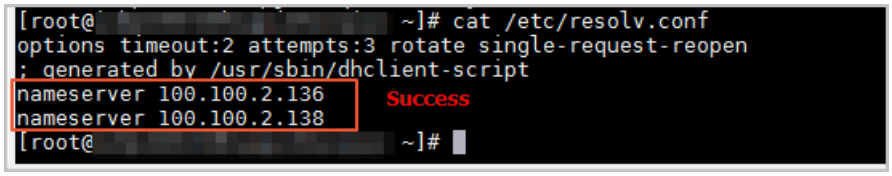
   

3. Run the following command to restart the network service:

```
systemctl restart network
```

4. Run the following command to check whether the modification is successful:

```
cat /etc/resolv.conf
```



10.Database proxy (read/write splitting)

10.1. Release notes of dedicated proxy versions

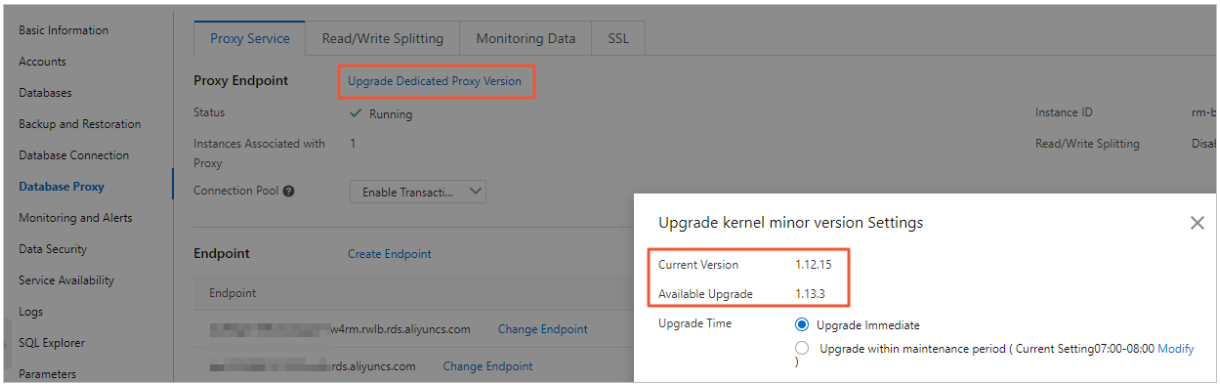
This topic describes the release notes for dedicated proxy versions.

> **Note**
>
> - For more information about how to upgrade the dedicated proxy version of an ApsaraDB RDS for MySQL instance, see Upgrade the dedicated proxy version of an ApsaraDB RDS for MySQL instance.
> - For more information about the minor engine versions of ApsaraDB RDS for MySQL, see AliSQL小版本Release Notes.

View the dedicated proxy version of an RDS instance

If the RDS instance is not using the latest dedicated proxy version, click **Upgrade Dedicated Proxy Version** on the **Database Proxy** page of the RDS instance in the ApsaraDB RDS console. In the dialog box that appears, check the in-use dedicated proxy version (**Current Version**) and the available dedicated proxy version (**Available Upgrade**).

> **Note** If the RDS instance is using the latest dedicated proxy version, the Upgrade Dedicated Proxy Version button is not displayed. Users can also view the dedicated proxy version of the RDS instance by calling an API operation. For more information, see Query database proxy details.



Updates to dedicated proxy versions

The following table contains only the mainstream dedicated proxy versions rather than all dedicated proxy versions.

Dedicated proxy version	Description
1.13.25	Bug fixes: • The bug that causes memory leaks in the dedicated proxy of an RDS instance due to authentication failures is fixed. • The bug that causes the dedicated proxy of an RDS instance to unexpectedly exit is fixed. This bug is triggered if an application connects to the dedicated proxy by using multiple endpoints.
1.13.22	• New features: ◦ The `SELECT LAST_INSERT_ID()` statement can be executed after the INSERT statement to obtain the unique ID of a sharded table. ◦ The FOUND_ROWS function is supported. ◦ The requests for running the COM_STATISTICS command can be routed to read-only RDS instances. ◦ The optimized transaction connection pool feature is supported. ◦ The requests for invoking the GEO function can be routed to read-only RDS instances. ◦ A few internal metrics are added to monitor the performance of RDS instances. • Bug fixes: ◦ The bug that causes requests to be routed to locked RDS instances is fixed. ◦ The bug that causes routing errors due to the incorrect parsing of some statements is fixed. ◦ The bug that causes failures in running the stmt_exec command is fixed. ◦ The bug that causes failures in executing the `LOAD DATA INFILE` statement is fixed.
1.13.17	Bug fixes: A few internal bugs are fixed.

Dedicated proxy version	Description
1.13.5	• New features: ◦ The Force Node Connection feature is provided to route all requests to a specified node. For example, the `/*force_proxy_internal*/set force_node = 'pi-123';` setting specifies that all requests over a specifiied connection are routed to the pi-123 node. **Note** If the specified node is faulty, the "`set force node 'pi-123' is not found, please check.`" error is returned. ◦ The Force Node Query feature is provided to route a specified request to a specified node. For example, the `/*force_node='pi-123'*/ show processlist;` setting specifies that a specified request is routed to the pi-123 node. **Note** If the specified node is faulty, the "`'force hint server node is not found, please check'.`" error is returned. ◦ A few internal metrics are added to monitor the performance of RDS instances. • Bug fixes: ◦ The bug that causes requests to be routed to read-only RDS instances is fixed. This bug is triggered if the statements in the requests contain the MODE keyword. ◦ The bug that causes unbalanced loads is fixed.
1.12.10	• New features: SSL encryption can be enabled for dedicated proxy endpoints. • Bug fixes: ◦ The bug that causes exceptions in establishing SSL connections to an RDS instance is fixed. This bug is triggered if the RDS instance runs MySQL 8.0. ◦ The bug that prevents the enabled dedicated proxies from properly routing new requests to an RDS instance is fixed. This bug is triggered if the RDS instance is restored from an abnormal state to a normal state.
1.12.7	• New features: ◦ The `SHOW FULL PROCESSLIST` statement is supported. ◦ The syntax for XA transactions is supported. • Bug fixes: ◦ The bug that causes errors in executing the `SHOW PROCESSLIST` statement on an RDS instance is fixed. This bug is triggered if the RDS instance runs MySQL 8.0. ◦ A few bugs that affect the transaction connection pool feature are fixed. ◦ A few bugs that cause connection failures are fixed.

Dedicated proxy version	Description
1.11.12	• New features: The transaction connection pool feature is supported. For more information, see Set the connection pool type of an ApsaraDB RDS for MySQL instance. • Bug fixes: ○ The bug that prevents new requests over the previous persistent connections to an RDS instance from being routed to the RDS instance is fixed. This bug is triggered if the RDS instance is restored from an abnormal state to a normal state. This bug is fixed by optimizing the mechanism that is used to balance loads among persistent connections. ○ The bug that prevents the PREPARE statement from being sent in unicast mode is fixed by optimizing the syntax of the PREPARE statement. ○ The bug that causes failures in connecting MySQL 5.7 databases to MySQL 5.6 databases is fixed. This bug is triggered if the Deprecate EOF feature is enabled. ○ The bug that causes disconnections to an RDS instance is fixed. This bug is triggered if stored procedures are invoked to modify the RDS instance. ○ The bug that causes a client to report the `Packets out of order` error is fixed. This bug is triggered if the size per line of large packets in a result set exceeds 16 MB. ○ The bug that prevents the timely termination of transactions on a read-only RDS instance is fixed. This bug is triggered if the transactions are started by the `SET autocommit=0` statement. ○ The bug that causes a statement to be routed to a read-only RDS instance is fixed. This bug is triggered if `LOCK IN SHARE MODE` is specified in the statement. ○ The bug that causes the `SELECT handler FROM abc` statement to be routed to a read-only RDS instance is fixed. This bug is triggered if FOR UPDATE is specified in the statement. ○ The bug that causes failures in authenticating a user from more than one host is fixed.
1.10.7	Bug fixes: A few bugs that affect the session connection pool feature are fixed.

Dedicated proxy version	Description

1.9.23	• New features:
	  ◦ The root account is granted the permissions to establish connections.
	  ◦ SSL connections are supported.
	• Bug fixes:
	  ◦ The bug that causes failures in running the `change user` command is fixed.
	  ◦ The bug that causes failures in running the `load file` command is fixed.
	  ◦ The bug that causes a client to report the "`Exception: Packets out of order`" error is fixed. This bug is triggered if the client receives packets that are in an unexpected sequence.
	  ◦ The bug that causes a read-only RDS instance to disconnect when its primary RDS instance becomes abnormal is fixed.

1.9.14	• New features:
	The `/*FORCE_SLAVE*/` and /*FORCE_MASTE*/ hints are supported.
	• Bug fixes:
	  ◦ The bug that causes the system to return garbled characters is fixed. This bug is triggered if the value of the charset parameter is invalid.
	  ◦ The bug that causes the system to return an invalid string for the MySQL version is fixed.

10.2. What are database proxies?

This topic introduces the database proxies of ApsaraDB RDS for MySQL.

ApsaraDB RDS for MySQL provides two types of database proxies: dedicated proxies and shared proxies. A database proxy resides between a database system and an application. The database proxy receives requests from the application and routes the requests to the primary RDS instance and read-only RDS instances in the database system. The database proxy is easy to use and maintain and provides high availability and high performance. The dedicated proxy also provides advanced features, such as automatic read/write splitting, transaction splitting, and connection pooling.

> **Note** Only dedicated proxies are supported. For more information, see the "Appendix: Introduction to shared proxies" section of this topic.

Scenarios

- The primary RDS instance is heavily loaded due to a large number of requests that are encapsulated in transactions.
- The primary RDS instance is heavily loaded due to an excessively large number of connections.
- Most of your workloads require short-lived connections.
- **Read-only workloads** and workloads that need to be **isolated**.

> **Note** For example, your database system consists of one primary RDS instance and four read-only RDS instances, and you have two applications, Application A and Application B. Application A initiates only read requests, and Application B initiates both read and write requests. In this case, you can use two read-only instances to create Proxy Terminal A with the **Read-only** attribute. Then, you can use the other two read-only instances to create Proxy Terminal B with the **Read/Write** attribute. This way, Application A and Application B are isolated from each other in your database system.

Introduction to proxy terminals

Proxy terminals are developed by Alibaba Cloud. You can use proxy terminals to customize the endpoints that are used to connect to an RDS instance. Each RDS instance supports up to seven proxy terminals. You can modify the read and write attributes of each proxy terminal based on your various business requirements.

Read and write attributes

You can set the read and write attributes of each proxy terminal.

- **Read/Write**: This attribute is used to support the read/write splitting feature. The read/write splitting feature allows you to linearly scale the volume of workloads that can be processed by your database system. For more information, see What is read/write splitting?

  If you select this attribute for a proxy terminal, you must make sure that the proxy terminal is associated with at least one primary RDS instance and one read-only RDS instance. All write requests are routed to the primary RDS instance. In this case, the proxy terminal supports features such as transaction splitting and connection pool. For more information, see Use the transaction splitting feature on an ApsaraDB RDS for MySQL instance and Set the connection pool type of an ApsaraDB RDS for MySQL instance.

- **Read-only**: This attribute is used to process only read requests. For example, if your application provides only the report service, you can select this attribute.

  If you select this attribute for a proxy terminal, you must make sure that the proxy terminal is associated with at least one read-only RDS instance. The proxy terminal does not route requests to the primary RDS instance. In addition, the proxy terminal does not support features such as transaction splitting and connection pool. For more information, see Use the transaction splitting feature on an ApsaraDB RDS for MySQL instance and Set the connection pool type of an ApsaraDB RDS for MySQL instance.

  If you select the **Read-only** attribute for a proxy terminal, the proxy terminal assigns connections to the associated read-only RDS instances based on a round-robin algorithm. Each database client is assigned only one connection to one read-only RDS instance. The connection to the primary RDS instance is not assigned by the proxy terminal. The total number of available connections is the sum of connections that are established to all the read-only RDS instances.

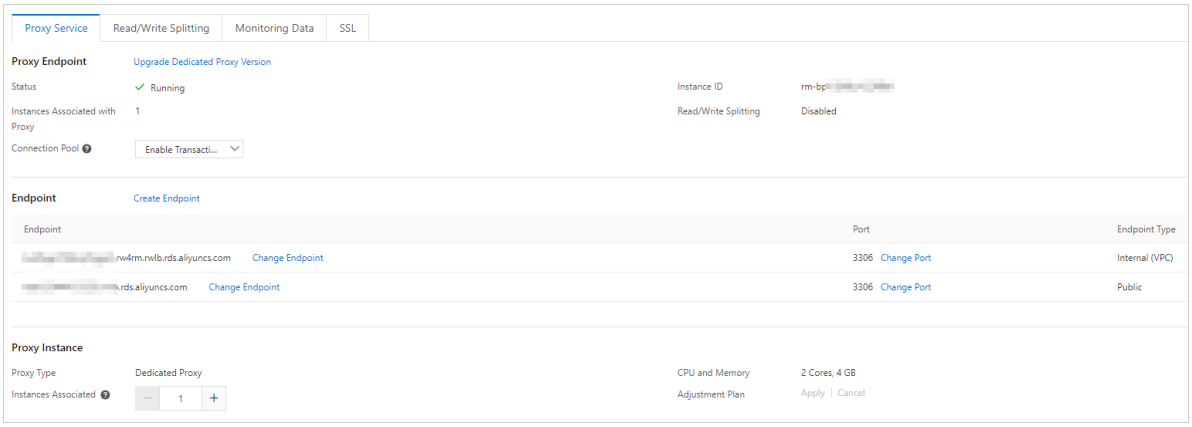
> **Note** For more information about how to modify the read and write attributes of a proxy terminal, see Enable the dedicated proxy feature for an ApsaraDB RDS for MySQL instance.

Overview

Database proxies provide the following features:

- You can perform various operations on the **Proxy** tab. For example, you can upgrade the dedicated

proxy version and create a proxy endpoint on this tab.



Type	Parameter	Description
Proxy Endpoint	**Status**	The status of the dedicated proxy.
	Instance ID	The ID of the RDS instance.
	Associated Proxy Instances	The number of proxy instances that are associated with the dedicated proxy. You can increase the processing capability of the dedicated proxy by enabling more proxy instances.
Endpoint	**Proxy Terminal**	The name of a proxy terminal. You can create multiple proxy endpoints for each proxy terminal. For more information, see What is read/write splitting?
	Endpoint	The endpoint that is used to connect to the dedicated proxy. The dedicated proxy provides a default proxy endpoint to which the proxy terminal feature is bound. You can create, modify, or delete a proxy endpoint. For more information, see Manage the dedicated proxy endpoints of an ApsaraDB RDS for MySQL instance.
	Port	The port number that is bound to a proxy endpoint. **Note** To change a port number, you must find the proxy endpoint to which the port number is bound and click **Change** on the right. A valid port number ranges from 1000 to 5999.
	Network Type	The network type of a proxy endpoint. You cannot change the network type of a proxy endpoint.
	Proxy Type	Only **Dedicated Proxy** is supported.
	CPU and Memory	The specifications that are provided per proxy instance. Each proxy instance can only have 2 cores and 4 GB of memory.

Type	Parameter	Description
Proxy Instance	Enabled Proxy Instance	The number of proxy instances that are enabled for your RDS instance. ApsaraDB RDS can balance the loads among the proxy instances that are enabled. You can enable up to 60 proxy instances for your RDS instance. **Note** We recommend that you set the number of proxy instances to `one-eighth of the total number of cores that are configured for your RDS instance and read-only RDS instances`. If the result is not an integer, you must round the result up to the nearest integer. Up to 60 proxy instances are supported. For example, if your RDS instance has 8 cores and read-only RDS instances have 4 cores, the recommended number of proxy instances is 2 based on the following calculation: `(8 + 4)/8 = 1.5`. The result 1.5 is rounded up to 2.

- Proxy terminal (original read/write splitting)

  ApsaraDB RDS automatically routes read and write requests to the proxy endpoints that you configure. Then, these endpoints route the read and write requests to the primary RDS instance and read-only RDS instances based on the read weights of these instances. For more information, see What is read/write splitting?
- Connection pool: This feature is used to mitigate the heavy loads on your database system. Heavy load issues are caused by excessive connections or frequent short-lived connections such as PHP-based connections. For more information, see Set the connection pool type of an ApsaraDB RDS for MySQL instance.
- Transaction splitting: This feature allows ApsaraDB RDS to route the read requests prior to write operations within a transaction to the read-only RDS instances. This reduces the loads on your primary RDS instance. For more information, see Use the transaction splitting feature on an ApsaraDB RDS for MySQL instance.
- You can view the CPU utilization for the proxy instances that are enabled. The CPU utilization information helps you obtain the loads on the proxy instances and adjust the number of proxy instances based on the monitoring data. For more information, see View the proxy monitoring data of an ApsaraDB RDS for MySQL instance.
- SSL encryption: This feature is used to encrypt the data that is destined for the protected proxy endpoint. This ensures the security of data in transit. For more information, see Configure SSL encryption for a proxy endpoint on an ApsaraDB RDS for MySQL instance.

Usage notes

For more information, see Usage notes for database proxies

How to enable dedicated proxies

For more information, see Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance

Appendix: Introduction to shared proxies

ApsaraDB RDS for MySQL provides two types of proxies: dedicated proxies and shared proxies.

A shared proxy is also called a multi-tenant proxy. A shared proxy allows the database systems of different users to share proxy computing resources. In this case, these database systems may compete for proxy computing resources. This decreases the stability of the shared proxy. In addition, a shared proxy does not support the advanced features that are provided by the dedicated proxy. These advanced features include connection pool, SSL encryption, and transaction splitting. However, a shared proxy supports read/write splitting. The network type of a read/write splitting endpoint changes with the network type of the primary RDS instance. For more information, see Change the network type of an ApsaraDB RDS for MySQL instance. Shared proxies are no longer available. For more information, see [Notice] End of updates and maintenance for the shared proxy feature.

A dedicated proxy is also called a single-tenant proxy. A dedicated proxy allows your database system to use dedicated proxy computing resources. Therefore, a dedicated proxy has the following benefits over a shared proxy:

- The dedicated proxy provides higher stability.
- The dedicated proxy provides higher isolation.
- The dedicated proxy provides better performance. A maximum of 60 proxy instances can be created for each RDS instance. A SysBench test shows that each proxy instance supports 20,000 to 50,000 queries per second (QPS) in online transaction processing (OLTP) scenarios. We recommend that you estimate the maximum QPS that is supported by your RDS instance based on your actual stress test results.

  > **Note** The maximum number of connections to the dedicated proxy is not limited. This number varies based on the specifications of the primary and read-only RDS instances in your database system.

- The dedicated proxy supports scaling. You can increase the number of proxy instances based on your business requirements. For more information, see Adjust the number of dedicated proxies on an ApsaraDB RDS for MySQL instance.
- The dedicated proxy supports performance monitoring. You can adjust the number of proxy instances based on the monitoring data and your business planning. For more information, see View the monitoring data of dedicated proxies on an ApsaraDB RDS for MySQL instance.
- If you change the network type of a primary RDS instance, the network type of the read/write splitting endpoint remains unchanged. For more information, see Change the network type of an ApsaraDB RDS for MySQL instance and What is read/write splitting?.
- A unified proxy endpoint is provided. This eliminates the need to modify the endpoint information on your application and reduces maintenance costs. The proxy endpoint remains valid until you release the proxy instances. For more information, see Manage the dedicated proxy endpoints of an ApsaraDB RDS for MySQL instance. The proxy endpoint remains connected unless you release the enabled proxy instances. For example, you may enable the read/write splitting feature during peak hours, and then release read-only RDS instances and disable the read/write splitting feature during off-peak hours. In these cases, the proxy endpoint remains connected, and you do not need to update the endpoint information on your application.
- The dedicated proxy supports automatic read/write splitting. This reduces maintenance costs. For more information, see Enable the read/write splitting feature for an ApsaraDB RDS for MySQL

instance. In normal cases, to implement read/write splitting, you must manually add the endpoints of the primary and read-only RDS instances to your application. This applies if the read-only RDS instances are available. If you enable the dedicated proxy feature, you need to add only a unified proxy endpoint to your application. This way, all the requests from your application are sent to the proxy endpoint. Then, the proxy endpoint routes read requests to the read-only RDS instances and write requests to the primary RDS instance based on the read weights of these instances. You do not need to update the configuration data on your application even if you create more read-only RDS instances or if you delete existing read-only RDS instances.

- The dedicated proxy provides more advanced features, such as connection pool and transaction splitting. For more information, see Set the connection pool type of an ApsaraDB RDS for MySQL instance and Enable the transaction splitting feature for an ApsaraDB RDS for MySQL instance

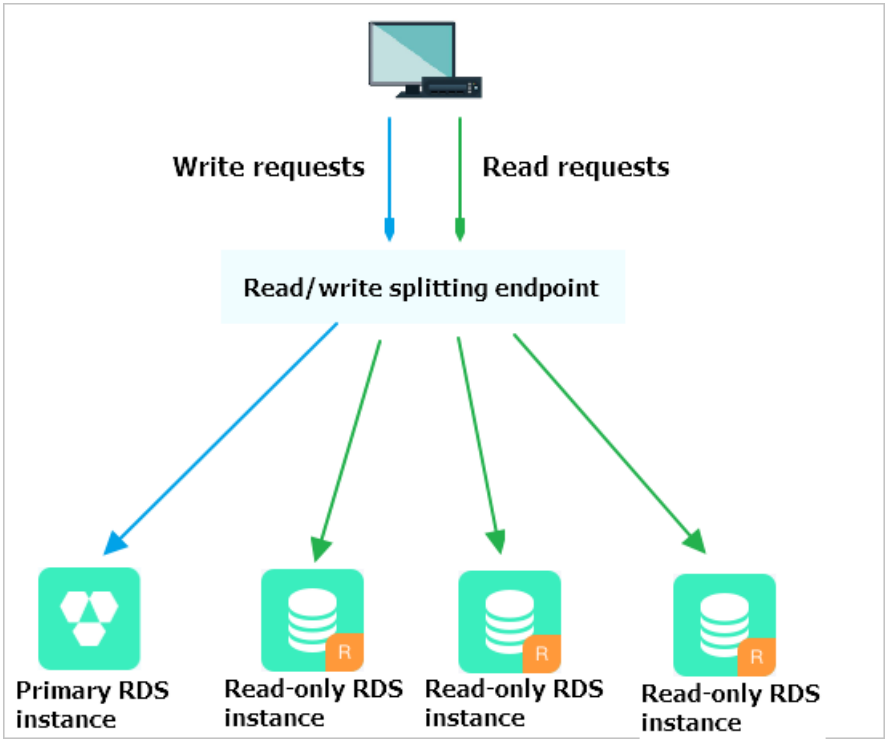
> **Note** Shared proxies are not supported for new RDS instances due to the known limits. Alibaba Cloud continues to provide support for the shared proxies of existing RDS instances. If the shared proxy of your RDS instance is enabled, we recommend that you upgrade the instance from shared proxy to dedicated proxy. This allows you to use more advanced features. For more information, see Upgrade the database proxy of an ApsaraDB RDS for MySQL instance from a shared proxy to a dedicated proxy.

10.3. What is read/write splitting?

This topic introduces the read/write splitting feature of ApsaraDB RDS for MySQL. This feature allows ApsaraDB RDS to route read and write requests to a dedicated proxy endpoint. Then, the dedicated proxy endpoint routes the read and write requests to the primary RDS instance and read-only RDS instances of your database system.

Background information

If your database system receives a large number of read requests and a small number of write requests, the primary RDS instance may fail to process read requests and your workloads may be interrupted. After read-only RDS instances are created, you can enable the read/write splitting feature. Then, you can use a dedicated proxy endpoint to perform read/write splitting. You need only to add the dedicated proxy endpoint to your application. After your application is connected to the dedicated proxy endpoint, ApsaraDB RDS routes write requests to the primary RDS instance and read requests to the read-only RDS instances based on the read weights of these instances. For more information, see Introduction to database proxies.



Benefits

- Unified endpoint to facilitate maintenance

  If you do not enable the read/write splitting feature, you can perform read/write splitting only after you add the endpoints of the primary RDS instance and read-only RDS instances to your application.

  If you enable the read/write splitting feature, you can use a dedicated proxy endpoint to perform read/write splitting. You need only to add the dedicated proxy endpoint to your application. After your application is connected to the dedicated proxy endpoint, your database system routes read and write requests to the primary RDS instance and read-only RDS instances based on the read weights of these instances. This reduces maintenance costs.

  You can also create read-only RDS instances to increase the read capability of your database system without the need to modify the configuration data on your application.

- Native link to increase performance and reduce maintenance costs

  If you build your own proxy layer on the cloud to perform read/write splitting, data must be parsed and forwarded by multiple components before the data reaches your database system. As a result, response latencies increase. The read/write splitting feature is embedded in the ApsaraDB RDS ecosystem to reduce response latencies, increase processing speeds, and reduce maintenance costs.

- Configurable read weights and thresholds to ensure suitability in various scenarios

  You can specify the read weights of the primary RDS instance and read-only RDS instances. You can also specify the latency threshold for data replication to the read-only RDS instances.

- Instance-level health checks to ensure high availability

The read/write splitting feature enables ApsaraDB RDS to actively check the health statuses of the primary RDS instance and read-only RDS instances. If a read-only RDS instance unexpectedly exits or its data replication latency exceeds the specified threshold, ApsaraDB RDS stops routing read requests to the instance. ApsaraDB RDS redirects the read requests that are destined for the faulty read-only RDS instance to other healthy RDS instances in your database system. This ensures service availability even if an individual read-only RDS instance fails. After the faulty read-only RDS instance is recovered, ApsaraDB RDS resumes routing read requests to the instance.

> **Note** We recommend that you create at least two read-only RDS instances to mitigate the impacts of single points of failure (SPOFs).

Logic used to route requests

- The following requests are routed only to the primary RDS instance:
  - Requests that are used to execute INSERT, UPDATE, DELETE, and SELECT FOR UPDATE statements.
  - All requests that are used to perform DDL operations, such as the DDL operations that are performed to create databases or tables, delete databases or tables, and change schemas or permissions.
  - All requests that are encapsulated in transactions.
  - Requests that are used to invoke user-defined functions.
  - Requests that are used to run stored procedures.
  - Requests that are used to execute EXECUTE statements.
  - Requests that are used to run multi-statement queries. For more information, see Multi-statement.
  - Requests that involve temporary tables.
  - Requests that are used to execute SELECT last_insert_id() statements.
  - All requests that are used to query or reconfigure user variables.
  - Requests that are used to execute KILL statements in SQL. These statements are different from the KILL commands in Linux.
- The following requests are routed to the primary RDS instance or read-only RDS instances:
  - Requests that are used to execute SELECT statements that are not encapsulated in transactions.
  - Requests that are used to execute COM_STMT_EXECUTE statements.
- The following requests are routed to the primary RDS instance and read-only RDS instances:
  - All requests that are used to reconfigure system variables.
  - Requests that are used to execute USE statements.
  - Requests that are used to execute SHOW PROCESSLIST statements.

    > **Note** After a SHOW PROCESSLIST statement is executed, the dedicated proxy returns all processes that run on the primary RDS instance and read-only RDS instances in your database system.

  - Requests that are used to execute COM_STMT_PREPARE statements.
  - Requests that are used to execute COM_CHANGE_USER, COM_QUIT, and COM_SET_OPTION statements.

Perform read/write splitting

For more information, see Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance.

Usage notes

For more information, see Usage notes for database proxies.

FAQ

For more information, see FAQ about dedicated proxies.

10.4. Billing rules for dedicated proxy instances that are enabled on an ApsaraDB RDS for MySQL instance

This topic describes the billing rules for dedicated proxy instances that are enabled on an ApsaraDB RDS for MySQL instance.

Background information

- Proxy instances, read-only RDS instances, and primary RDS instances are separately billed.
- If the primary RDS instance is released, the enabled proxy instances are automatically released. You are no longer charged for the dedicated proxy feature.

Billing

A maximum of 60 proxy instances can be created for each RDS instance. These proxy instances provide higher processing capabilities. A SysBench test shows that each proxy instance supports 20,000 to 50,000 queries per second (QPS) in online transaction processing (OLTP) scenarios. We recommend that you estimate the maximum QPS that is supported by your RDS instance based on your actual stress test results.

> **Note** If you upgrade the database proxy of your primary RDS instance from a shared proxy to a dedicated proxy, you can use the dedicated proxy feature free of charge for at least one year. For more information, see Upgrade the database proxy of an ApsaraDB RDS for MySQL instance from a shared proxy to a dedicated proxy.

Proxy instances support only the pay-as-you-go billing method. The following table describes the prices of proxy instances in different Alibaba Cloud regions.

Region		Price
Name	Code	USD/Hour/Proxy
China (Hangzhou)	cn-hangzhou	0.173
China (Shanghai)	cn-shanghai	0.173
China (Qingdao)	cn-qingdao	0.173

Region		Price
Name	Code	USD/Hour/Proxy
China (Beijing)	cn-beijing	0.173
China (Zhangjiakou)	cn-zhangjiakou	0.120
China (Hohhot)	cn-huhehaote	0.173
China (Shenzhen)	cn-shenzhen	0.173
China (Heyuan)	cn-heyuan	0.173
China (Chengdu)	cn-chengdu	0.173
China (Hong Kong)	cn-hongkong	0.297
Japan (Tokyo)	ap-northeast-1	0.288
India (Mumbai)	ap-south-1	0.231
Singapore (Singapore)	ap-southeast-1	0.271
Australia (Sydney)	ap-southeast-2	0.273
Malaysia (Kuala Lumpur)	ap-southeast-3	0.253
Indonesia (Jakarta)	ap-southeast-5	0.271
Germany (Frankfurt)	eu-central-1	0.243
UK (London)	eu-west-1	0.280
UAE (Dubai)	me-east-1	0.377
US (Virginia)	us-east-1	0.237
US (Silicon Valley)	us-west-1	0.284

10.5. Usage notes for database proxies

This topic describes the notes that you must understand before you use the dedicated proxy feature of ApsaraDB RDS for MySQL.

- Proxy instances, read-only RDS instances, and primary RDS instances are separately billed.
- When you change the specifications of the primary RDS instance or a read-only instance, a transient connection may occur.
- If your application connects to your database system by using a proxy endpoint and the transaction splitting feature is not enabled, all requests that are encapsulated in transactions are routed to the

primary RDS instance.

- If you create or restart a read-only instance after you enable the dedicated proxy feature, only the requests sent over new connections are routed to the new or restarted read-only instance.
- Dedicated proxy endpoints do not support compression.
- If a proxy endpoint is used to implement read/write splitting, the read consistency of the requests that are not encapsulated in transactions cannot be ensured. If you want to ensure the read consistency of these requests, you must encapsulate these requests in transactions or add hints. For more information, see Execute hints on an ApsaraDB RDS for MySQL instance.
- If your application connects to your database system by using a proxy endpoint, the `SHOW PROCESSLIST` statement returns a result set for each query. The result set consists of the query results from the primary RDS instance and read-only RDS instances.
- The connection pool feature is enabled by default. Therefore, the `SHOW PROCESSLIST` statement may return idle connections. For more information, see Set the connection pool type of an ApsaraDB RDS for MySQL instance.
- If you execute multi-statements or call stored procedures, all subsequent requests over the current connection are routed to the primary RDS instance. To use the read/write splitting feature again, you must close the current connection and establish a new connection.
- The dedicated proxy uses the 1:N connection model. After your application initiates a connection request, the dedicated proxy replicates the established connection to the primary RDS instance and all the read-only RDS instances. The maximum number of connections that are allowed to the dedicated proxy is not limited. The maximum number of connections varies based on the specifications of the primary RDS instance and read-only RDS instances. If you do not enable the transaction connection pool feature, the dedicated proxy establishes a separate connection from each client to the primary RDS instance and each of the read-only RDS instances. After you enable the dedicated proxy feature, we recommend that you specify the same maximum number of connections for the primary RDS instance and read-only RDS instances. If the maximum number is different for the primary RDS instances and read-only RDS instances, the maximum number of connections that are allowed to the dedicated proxy is subject to the smallest number of connections among these instances.
- The **max_prepared_stmt_count** parameter must be set to the same value for the primary RDS instance and read-only RDS instances.
- If you use the MySQL CLI to establish a connection for which hints are added, you must add the `-c` parameter to the hints. If you do not add this parameter to a hint, the MySQL CLI filters the hint out. For more information about the hint syntax, see Execute hints on an ApsaraDB RDS for MySQL instance.
- If the primary RDS instance is locked, the enabled proxy instances are not released but can process only read requests.
- If the primary RDS instance is released, the enabled proxy instances are automatically released. You are no longer charged for the dedicated proxy feature.

10.6. FAQ about dedicated proxies

This topic provides answers to some commonly asked questions about dedicated proxies.

After I update the minor engine version of my RDS instance that runs MySQL 5.7, why is the dedicated proxy service still unavailable?

After you update the minor engine version of your RDS instance that runs MySQL 5.7, you must disable the original read/write splitting feature before you can use the dedicated proxy service.

After I enable the dedicated proxy service, do I need to use a dedicated proxy endpoint or the read/write splitting endpoint to implement read/write splitting?

After you enable the dedicated proxy service, the read/write splitting endpoint is the same as the dedicated proxy endpoint that you are using. The backend uses a dedicated proxy endpoint to implement read/write splitting.

I use a dedicated proxy endpoint to connect my application to my database system. What do I do if a connection timeout error is reported?

We recommend that you increase the value of the wait_timeout parameter and then try again.

After I enable the dedicated proxy service, does ApsaraDB RDS reclaim the original endpoints of the primary and read-only RDS instances in my database system?

No, after you enable the dedicated proxy service, ApsaraDB RDS does not reclaim the original endpoints of the primary and read-only RDS instances in your database system.

Does the number of dedicated proxy endpoints vary based on the number of dedicated proxies?

No, the number of dedicated proxy endpoints does not vary based on the number of dedicated proxies. You can apply for more than one dedicated proxy endpoint even if you have enabled only one dedicated proxy. For more information, see Create a proxy endpoint for a dedicated proxy.

Which architecture do dedicated proxies work in? And does this architecture provide a failover mechanism?

Dedicated proxies work in the high availability architecture. In this architecture, each dedicated proxy has a secondary dedicated proxy as a standby. If a dedicated proxy becomes faulty, ApsaraDB RDS fails over your workloads to the secondary dedicated proxy of the faulty dedicated proxy.

10.7. Proxy Terminal

10.7.1. Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance

This topic describes how to enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance. The dedicated proxy feature provides advanced features such as read/write splitting, connection pooling, transaction splitting, and SSL encryption.

Prerequisites

- Your RDS instance is a primary instance that runs one of the following MySQL versions and RDS

editions:

- MySQL 8.0 with a minor engine version of 20191204 or later on RDS Enterprise Edition
- MySQL 8.0 with a minor engine version of 20190915 or later on RDS High-availability Edition
- MySQL 5.7 with a minor engine version of 20191128 or later on RDS Enterprise Edition
- MySQL 5.7 with a minor engine version of 20190925 or later on RDS High-availability Edition
- MySQL 5.6 with a minor engine version of 20200229 or later on RDS High-availability Edition

> **Note** To view the minor engine version of your RDS instance, you must log on to the ApsaraDB RDS console and go to the **Basic Information** page. In the **Configuration Information** section of the page, you can check whether the **Upgrade Kernel Version** button is displayed. If the button is displayed, you can click the button to view and update the minor engine version of your RDS instance. If the button is not displayed, your RDS instance runs the latest minor engine version. For more information, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.

- A read-only RDS instance is created for your RDS instance. For more information, see Create a read-only ApsaraDB RDS for MySQL instance.
- Your RDS instance does not reside in Zone C or Zone D of the China (Hangzhou) region.

> **Note** If your RDS instance resides in Zone C or Zone D of the China (Hangzhou) region, you must migrate your RDS instance to other zones before you enable the dedicated proxy feature for your RDS instance. For more information, see Migrate an ApsaraDB RDS for MySQL instance across zones in the same region.

Billing rules

For more information, see Billing rules for dedicated proxy instances that are enabled on an ApsaraDB RDS for MySQL instance.

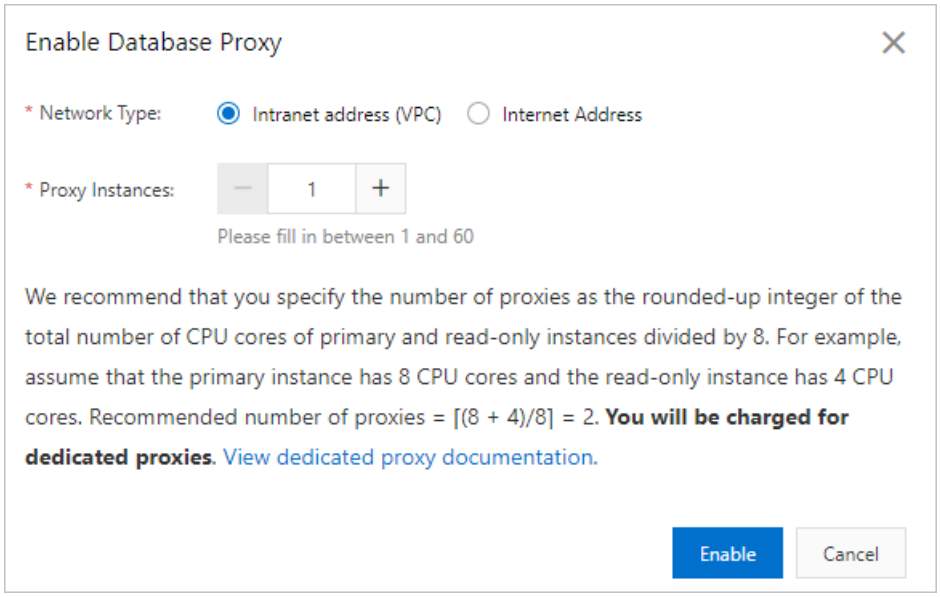
Limits

- If you enable the dedicated proxy feature for your RDS instance, your RDS instance does not support compression protocols.
- If you enable the dedicated proxy feature for your RDS instance, your RDS instance does not support vSwitch changes.

Step 1: Enable the dedicated proxy feature

This section describes how to enable the dedicated proxy feature for your RDS instance in the ApsaraDB RDS console. You can also enable the dedicated proxy feature when you create a read-only RDS instance for your RDS instance. For more information, see Create a read-only ApsaraDB RDS for MySQL instance.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. On the page that appears, click **Enable Proxy**. In the dialog box that appears, configure the **Network Type** parameter and the **Proxies** parameter and click **Enable**.



Note

- **Before you can use the advanced features, such as read/write splitting, that are provided by the dedicated proxy feature, you must configure a proxy terminal for your RDS instance.** For more information, see Step 2: Configure a proxy terminal.
- The default network type of a proxy endpoint varies based on the configuration of your RDS instance. For more information, see Manage the dedicated proxy endpoints of an ApsaraDB RDS for MySQL instance.
- We recommend that you set the number of proxy instances to `one-eighth of the total number of cores that are configured for your RDS instance and its read-only RDS instances`. If the result is not an integer, you must round up the result to the nearest integer. You can specify up to 60 proxy instances.

  For example, if your RDS instance has 8 cores and its read-only RDS instance has 4 cores, we recommend that you specify two proxy instances based on the following calculation: `(8 + 4)/8 = 1.5`. The result 1.5 is rounded up to 2.

Step 2: Configure a proxy terminal

Before you can use the advanced features that are provided by the dedicated proxy feature, you must configure a proxy terminal for your RDS instance.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. On the **Proxy Terminal (Original Read/Write Splitting)** tab, click **Configure Proxy Terminal**.
4. In the dialog box that appears, configure the following parameters and click **OK**.

Parameter	Description

Parameter	Description
Custom Proxy Terminal	The name of the proxy terminal. The name can be up to 30 characters in length.
Read/Write Attribute	The read and write attribute of the proxy terminal. Valid values: ○ Read/Write (Primary Instance Connected to Receive Write Requests): The proxy terminal connects to the primary RDS instance and the read-only RDS instances, and can receive write requests. This is the default attribute. ○ Read-only (Primary Instance Not Connected to Receive Write Requests): The proxy terminal connects only to the read-only RDS instances and cannot receive write requests. For more information, see What is read/write splitting?
Connection Pool	Specifies whether to enable the connection pool feature and specifies the type of connection pool that you want to enable. Valid values: ○ Transaction Connection Pool: If tens of thousands of or more connections need to be established, select this value. This is the default value. ○ Session Connection Pool: If only short-lived connections over PHP need to be established, set this parameter to this value. ○ Disable Connection Pool: If you want to disable the connection pool feature, set this parameter to this value. For more information, see Set the connection pool type of an ApsaraDB RDS for MySQL instance. **Note** This parameter is displayed only if you set the **Read/Write Attribute** parameter to **Read/Write (Primary Instance Connected to Receive Write Requests)**.
Latency Threshold	The maximum latency that is allowed for data replication from the primary RDS instance to the read-only RDS instances. If the latency of data replication to a read-only RDS instance exceeds the value of this parameter, ApsaraDB RDS no longer routes read requests to the read-only RDS instance regardless of the read weight of the read-only RDS instance. Valid values: 0 to 3600. Unit: seconds. The read-only RDS instances may replicate data from the primary RDS instance at a specific latency. The latency varies based on the statuses of the SQL statements that are executed. We recommend that you set this parameter to a value that is greater than or equal to 30. **Note** This parameter is displayed only if you set the **Read/Write Attribute** parameter to **Read/Write (Primary Instance Connected to Receive Write Requests)**.

Parameter	Description
Transaction Splitting	Specifies whether to enable the transaction splitting feature. After you enable the transaction splitting feature, ApsaraDB RDS can route the read requests prior to write operations in transactions to the read-only RDS instances. This way, the loads on the primary RDS instance are reduced. This feature is enabled by default. **Note** This parameter is displayed only if you set the **Read/Write Attribute** parameter to **Read/Write (Primary Instance Connected to Receive Write Requests)**.
Read Weight Distribution	The method that is used to assign read weights. A higher read weight indicates more read requests that need to be processed. For example, three read-only RDS instances are attached to the primary RDS instance, the read weight of the primary RDS instance is 0, and the read weights of the three read-only RDS instances are 100, 200, and 200. In this case, the primary RDS instance processes only write requests, and the three read-only RDS instances process all read requests based on the 1:2:2 ratio. ◦ **Automatic**: ApsaraDB RDS assigns a read weight to each RDS instance in your database system based on the specifications of the RDS instance. After you create a read-only RDS instance, ApsaraDB RDS automatically assigns a read weight to the read-only RDS instance and adds the read-only RDS instance to the read/write splitting link. For more information, see Rules of weight allocation by the system. ◦ **Custom**: You must manually specify a read weight for each RDS instance in your database system. Valid values: 0 to 10000. The read weight of a read-only RDS instance defaults to 0. After you create a read-only RDS instance, you must manually specify a read weight for the read-only RDS instance based on your business requirements. **Note** ◦ If a data replication latency is specified for a read-only RDS instance, you cannot specify a read weight for the read-only RDS instance. For more information, see Set the data replication latency of a read-only ApsaraDB RDS for MySQL instance. ◦ After you reconfigure this parameter, the new read weights immediately take effect and no transient connections occur. In addition, the existing connections remain open. Only the requests that are sent over new connections are routed based on the new weights.

After you configure a proxy terminal, you must add the specified endpoint of the proxy terminal to your application. This endpoint is also known as a proxy endpoint. Then, ApsaraDB RDS can route write requests to the primary RDS instance and read requests to the read-only RDS instances based on the read weights of these instances.

Step 3: Optional. Create a proxy terminal

Each RDS instance supports up to seven proxy terminals. You can create multiple proxy terminals, which help you apply different read and write policies to different clients.

Prerequisites

Multiple proxy instances are enabled, and the number of proxy instances that you enabled is greater than the number of proxy terminals that you created. For more information, see Adjust the number of dedicated proxies on an ApsaraDB RDS for MySQL instance.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. In the upper-right corner of the page, click **Create Proxy Terminal**.
4.

After you create a proxy terminal, you must add the specified endpoint of the proxy terminal to your application. This endpoint is also known as a proxy endpoint. Then, ApsaraDB RDS can route write requests to the primary RDS instance and read requests to the read-only RDS instances based on the read weights of these instances.

Related operations

Operation	Description
ModifyDBProxy	Enables or disables the dedicated proxy feature for an ApsaraDB RDS instance.
DescribeDBProxy	Queries the details about the dedicated proxy of an ApsaraDB RDS instance.

10.7.2. Set the connection pool type of an ApsaraDB RDS for MySQL instance

This topic describes how to set the connection pool type of an ApsaraDB RDS for MySQL instance. Connection pools are provided in the dedicated proxy to reduce the heavy loads that are caused by excessive connections or frequent short-lived connections such as PHP-based connections.

Prerequisites

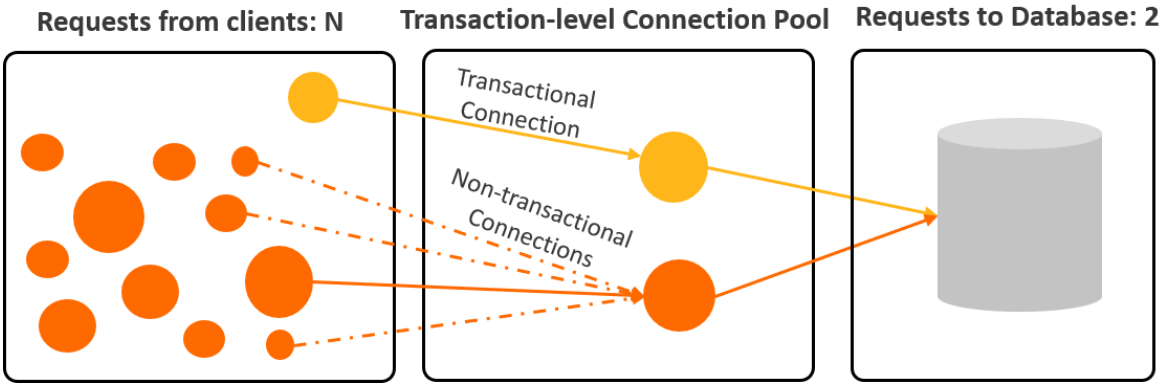
Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance

Context

The dedicated proxy supports the following two types of connection pools:

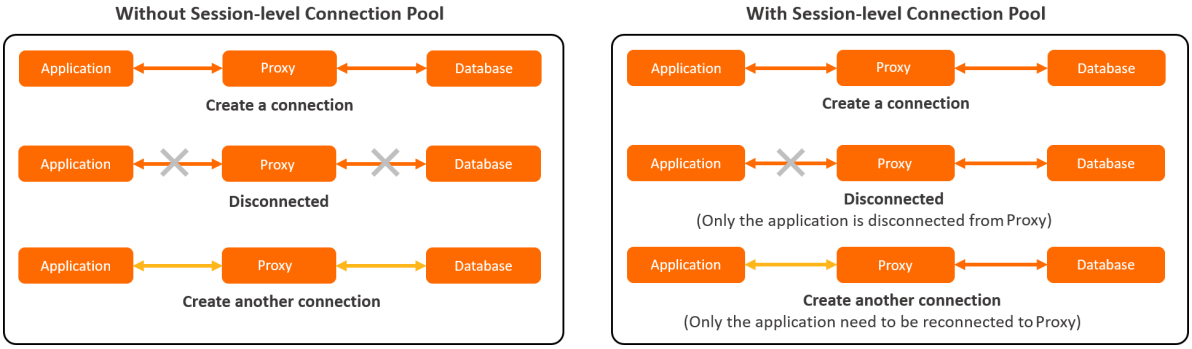
- **Transaction connection pool**

  This is the default connection pool type. A transaction connection pool is used to reduce the number of direct connections to your database system and reduce the heavy loads that are caused by frequent short-lived connections. If your application establishes tens of thousands of connections to your database system, we recommend that you select this connection pool type.



> **Note**
>
> - After you enable a transaction connection pool, your application can establish thousands of connections to the dedicated proxy. However, the dedicated proxy may establish only a few dozen or a few hundred connections to your database system.
> - **The maximum number of connections to the dedicated proxy is not limited. This maximum number varies only based on the specifications of the primary and read-only RDS instances in your database system.** If you do not enable a transaction connection pool, the dedicated proxy establishes a connection to each of the primary and read-only RDS instances after the dedicated proxy receives a request from your application.
>
>   If you enable a transaction connection pool, your application connects to the dedicated proxy when it initiates a request. This way, the dedicated proxy does not immediately establish a connection to each of the primary and read-only RDS instances. Instead, the dedicated proxy searches the transaction connection pool for an available connection that matches the request. A connection matches the request if the values of the `user` parameter, `dbname` parameter, and system variable are the same in the connection and the request. If the dedicated proxy cannot find an available connection, the dedicated proxy establishes a new connection. If the dedicated proxy can find an available connection, the dedicated proxy reuses the available connection. After the transaction that is specified in the request is complete, the dedicated proxy releases the connection to the transaction connection pool.
> - For more information about the limits of the **transaction connection pool** feature, see Limits of transaction connection pools.

- **Session connection pool**



If your application establishes short-lived connections to your database system, we recommend that you select this connection pool type.

A session connection pool is used to reduce the heavy loads that are caused by frequent short-lived connections. When your application becomes disconnected, ApsaraDB RDS checks whether the closed connection is idle. If the connection is idle, ApsaraDB RDS retains the connection in the session connection pool for a short period of time. When your application reinitiates a request, the dedicated proxy searches the session connection pool for an available connection that matches the request. A connection matches the request if the values of the user, clientip, and dbname parameters are the same in the connection and the request. If the dedicated proxy can find an available connection, the dedicated proxy reuses the available connection. This way, the overhead that is caused by frequent connections is reduced. If the dedicated proxy cannot find an available connection, the dedicated proxy establishes a new connection.

> Note
>
> - A session connection pool cannot reduce concurrent connections to your database system. However, a session connection pool can decrease the frequency at which your application establishes connections to your database system. This way, the overhead from the main MySQL thread is reduced and your database system can process requests more efficiently. However, the idle connections in the session connection pool temporarily consume the connection quota.
> - A session connection pool cannot reduce piled-up connections that are caused by a large number of slow SQL statements. To reduce piled-up connections, you must fix the issues that cause slow SQL statements.

Precautions

- The connection pool feature does not allow you to configure an account to have different permissions on different IP addresses. If you configure an account to have different permissions on different IP addresses, permission errors may occur when the existing connections are reused. For example, an account has permissions on database_a when it logs on from the 192.168.1.1 IP address, but the account does not have permissions on database_a when it logs on from the 192.168.1.2 IP address. In this case, permission errors may occur if you enable the connection pool feature.
- The connection pool feature that is provided in the dedicated proxy of your database system does not affect the connection pool feature that is provided in your application. If your application provides a connection pool, you do not need to enable the connection pool feature for your database system.

Limits of transaction connection pools

- When one of the following operations is performed over a connection, the dedicated proxy locks the connection. The dedicated proxy does not release the connection to the connection pool until the operation is complete.
  - Execute the PREPARE statement.
  - Create a temporary table.
  - Reconfigure a user variable.
  - Process large packets, such as the packets whose sizes exceed 16 MB.
  - Execute the LOCK TABLE statement.
  - Run a multi-statement query.

  - Call a stored procedure.

- The FOUND_ROWS, ROW_COUNT, and LAST_INSERT_ID functions are not supported. You can call these functions, but the results that are returned by these functions may be inaccurate.
  - If the dedicated proxy version that you use is V1.13.11 or later, you can execute the `SELECT FOUND_ROWS()` statement after the `SELECT SQL_CALC_FOUND_ROWS * FROM t1 LIMIT *` statement. However, we recommend that you do not perform the preceding operation. We recommend that you replace the `SELECT FOUND_ROWS()` statement with the `SELECT COUNT(*) FROM tb1` statement. For more information, see FOUND_ROWS().
  - If the dedicated proxy version that you use is V1.13.11 or later, you can execute the `SELECT LAST_INSERT_ID()` statement after the `INSERT` statement. This way, you can ensure the accuracy of query results.
- If you configure the `wait_timeout` parameter, the value of the `wait_timeout` parameter may not take effect on your application. This is because ApsaraDB RDS selects a connection from the connection pool whenever your application initiates a request. When the time that is specified by the `wait_timeout` parameter elapses, only the connections to the primary and read-only RDS instances are closed and the connections to your application remain open.
- The transaction connection pool matches requests with connections based on the following four variables: `sql_mode`, `character_set_server`, `collation_server`, and `time_zone`. If a request includes other session-level system variables, you must explicitly execute the SET statement on your application to configure these variables after the requested connection is established. Otherwise, a connection whose system variables are reconfigured may be selected from the transaction connection pool and reused.
- You can execute the `SELECT CONNECTION_ID()` statement to query the thread ID of a connection. This way, you can check whether the connection is reused.
- If the existing connections are reused, the IP address and port number that are returned by the `SHOW PROCESSLIST` statement or the SQL Explorer and Audit feature may differ from the actual IP address and port number of the database client on which your application runs. For more information, see Use the SQL Explorer and Audit feature on an ApsaraDB RDS for MySQL instance.
- The dedicated proxy merges the results that are obtained by the `SHOW PROCESSLIST` statement from all the primary and read-only RDS instances. Then, the dedicated proxy returns a result set to your application. If you enable a transaction connection pool, the thread ID of the connection between your application and the dedicated proxy differs from the thread ID of the connection between the dedicated proxy and your database system. As a result, the kill command may report an error even if the command is successfully run. In this case, you can execute the `SHOW PROCESSLIST` statement again to check whether the specified process is terminated.

Select a connection pool type

You can determine whether to enable the connection pool feature and specify the type of connection pool that you want to enable based on your business requirements:

- If your application establishes tens of thousands of connections to your database system or uses serverless computing to support a linear increase in the number of connections along with scaling and your application is not subject to the preceding limits of transaction connection pools, we recommend that you enable a transaction connection pool for your database system.
- If your application establishes only short-lived connections to your database system and is subject to the preceding limits of transaction connection pools, we recommend that you enable a session connection pool for your database system.

- If your application establishes a small number of connections to your database system and most of the connections are long-lived connections or if your application provides a connection pool, you do not need to enable the connection pool feature for your database system.

Change the connection pool type

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. Click the **Proxy Terminal (Original Read/Write Splitting)** tab. Then, select the type of connection pool that you want to enable from the **Connection Pool** drop-down list.

> **Note** The new type of connection pool is applied only to new connections.

Related operations

Operation	Description
DescribeDBProxy	Queries details about the dedicated proxy instances that are enabled on an ApsaraDB RDS instance.
DescribeDBProxyEndpoint	Queries the proxy endpoints that are used to connect to the dedicated proxy of an ApsaraDB RDS instance.
ModifyDBProxyEndpoint	Modifies a proxy endpoint that is used to connect to the dedicated proxy of an ApsaraDB RDS instance.

10.7.3. Use the transaction splitting feature on an ApsaraDB RDS for MySQL instance

This topic describes how to enable and disable the transaction splitting feature that is provided in the database proxy of a primary ApsaraDB RDS for MySQL instance. This feature allows ApsaraDB RDS to route the read requests prior to write operations in a transaction to the read-only RDS instances of your database system. This reduces the loads on the primary RDS instance.

Prerequisites

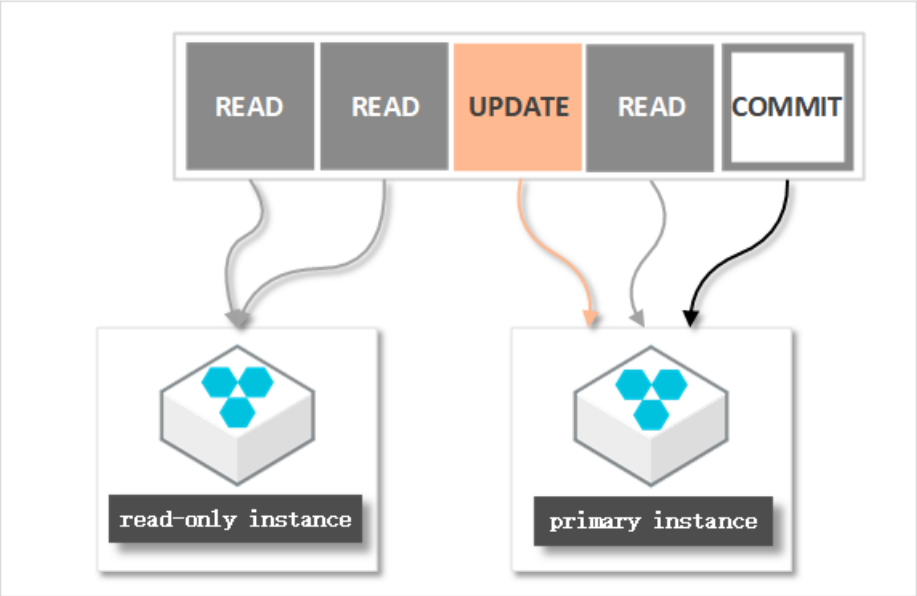
Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance

Context

By default, the dedicated proxy sends all requests that are encapsulated in transactions to the primary RDS instance. This ensures the correctness of the transactions. However, in some frameworks, the autocommit mode is disabled by using the `set autocommit=0;` command. As a result, all requests are encapsulated in the transactions that are not automatically committed. This causes heavy loads on the primary RDS instance. In this case, you can enable the transaction splitting feature.

By default, the transaction splitting feature is enabled with the default isolation level READ COMMITTED. If the autocommit mode is disabled, ApsaraDB RDS starts a transaction only for write operations. In addition, before the transaction starts, ApsaraDB RDS routes all read requests to the read-only RDS instances by using a load balancer.

> **Note**
>
> - Explicit transactions cannot be split. These explicit transactions include the transactions that are started by using BEGIN or START statements.
> - After you enable the transaction splitting feature, global consistency cannot be ensured. Before you enable this feature, we recommend that you evaluate whether this feature is suitable for your workloads.



Procedure

You can enable or disable the transaction splitting feature based on your business requirements.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. On the **Proxy Terminal (Original Read/Write Splitting)** tab, click **Enable** or **Disable** to the right of **Transaction Splitting**.

> **Note** After you enable or disable the transaction splitting feature, the new setting is applied only to new connections.

Related operations

Operation	Description
DescribeDBProxy	Queries details about the dedicated proxy instances that are enabled on an ApsaraDB RDS instance.

Operation	Description
DescribeDBProxyEndpoint	Queries the endpoints that are used to connect to the dedicated proxy of an ApsaraDB RDS instance.
ModifyDBProxyEndpoint	Modifies a proxy endpoint that is used to connect to the dedicated proxy of an ApsaraDB RDS instance.

10.7.4. Manage the dedicated proxy endpoints of an ApsaraDB RDS for MySQL instance

This topic describes how to manage the dedicated proxy endpoints of an ApsaraDB RDS for MySQL instance. After the dedicated proxy feature is enabled, a default dedicated proxy endpoint is generated. The proxy terminal feature is bound to this endpoint. You can create, modify, or delete a dedicated proxy endpoint.

Prerequisites

Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance

Network types

Dedicated proxy endpoints come in three network types: **Internal (VPC)**, **Internal (Classic Network)**, and **Public**.

After the dedicated proxy feature is enabled, a default dedicated proxy endpoint is generated. You can retain the default network type of the default dedicated proxy endpoint. You can also create more dedicated proxy endpoints. The available network types vary based on the configuration of your RDS instance.

Instance configuration	Network type of the default dedicated proxy endpoint	Network type of a new dedicated proxy endpoint
Standard SSDs (VPC)	**Internal (VPC)**	**Public**
ESSDs (VPC)		
Local SSDs (VPC)	**Internal (VPC)** **Public**	**Internal (VPC)** **Internal (Classic Network)** **Public**
Local SSDs (classic network)	**Internal (Classic Network)** **Public**	**Internal (Classic Network)** **Public**

> **Note** Each RDS instance can have only one dedicated proxy endpoint of each network type. The network type of a dedicated proxy endpoint is specified by the **Endpoint Type** parameter. For example, only one dedicated proxy endpoint of the **Internal (VPC)** network type is allowed.

Create a dedicated proxy endpoint

After the dedicated proxy feature is enabled, a default dedicated proxy endpoint is generated. For more information, see Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance. You can create more dedicated proxy endpoints.

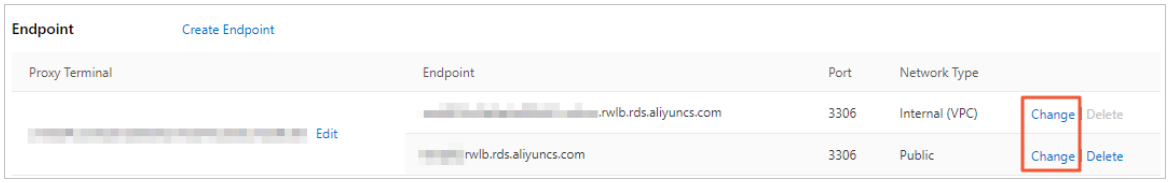
1. Go to the **Database Proxy** page.
   i.
   ii. Find your RDS instance and click the ID of the instance. In the left-side navigation pane, click **Database Proxy**.
2. In the **Endpoint** section of the Proxy tab, click **Create Endpoint**.
3. Configure the Proxy Terminal, Endpoint, Port, and Network Type parameters. Then, click **OK**. For more information, see Introduction to proxy terminals.

> **Note**
> - Each RDS instance can have only one dedicated proxy endpoint of each network type. The network type of a dedicated proxy endpoint is specified by the **Endpoint Type** parameter.
> - The prefix of a dedicated proxy endpoint must be 1 to 40 characters in length and can contain letters, digits, and hyphens (-). The prefix must start with a lowercase letter.
> - The port number that is bound to a dedicated proxy endpoint must be within the range of 1000 to 5999.

Change a dedicated proxy endpoint or the port number that is bound to the endpoint

1. Go to the **Database Proxy** page.
   i.
   ii. Find your RDS instance and click the ID of the instance. In the left-side navigation pane, click **Database Proxy**.
2. Find the dedicated proxy endpoint and click **Change** on the right. In the dialog box that appears, change the dedicated proxy endpoint or the associated port number and click **OK**.

> **Note**
> - The prefix of a dedicated proxy endpoint must be 1 to 40 characters in length and can contain letters, digits, and hyphens (-). The prefix must start with a lowercase letter.
> - The port number that is bound to a dedicated proxy endpoint must be within the range of 1000 to 5999.



Delete a dedicated proxy endpoint

> **Note**
> - If your RDS instance uses standard SSDs or enhanced SSDs (ESSDs), you cannot delete a dedicated proxy endpoint whose network type is **Internal (VPC)**.
> - If your RDS instance uses local SSDs, you must retain at least one dedicated proxy endpoint.

1. Go to the **Database Proxy** page.
   i.
   ii. Find your RDS instance and click the ID of the instance. In the left-side navigation pane, click **Database Proxy**.
2. Find the dedicated proxy endpoint and click **Delete** on the right. In the dialog box that appears, click **OK**.

> **Note** You can delete only the dedicated proxy endpoints that you create.

References

- Introduction to proxy terminals
- What is read/write splitting?

Related operations

Operation	Description
DescribeDBProxy	Queries details about the dedicated proxy of an ApsaraDB RDS instance.
CreateDBProxyEndpointAddress	Creates a dedicated proxy endpoint for an ApsaraDB RDS instance.
ModifyDBProxyEndpointAddress	Modifies a dedicated proxy endpoint of an ApsaraDB RDS instance.
DeleteDBProxyEndpointAddress	Deletes a dedicated proxy endpoint of an ApsaraDB RDS instance.

10.7.5. Configure SSL encryption for a proxy endpoint on an ApsaraDB RDS for MySQL instance

This topic describes how to configure Secure Sockets Layer (SSL) encryption for a proxy endpoint on an ApsaraDB RDS for MySQL instance. The dedicated proxy of your RDS instance provides advanced features, such as proxy terminal, connection pool, and transaction splitting. You can use SSL encryption to protect the data that is destined for a proxy endpoint.

Prerequisites

- Your RDS instance runs one of the following MySQL versions and RDS editions:
  - MySQL 8.0 on RDS High-availability Edition with local SSDs (The minor engine version is 20200831 or later.)
  - MySQL 5.7 on RDS High-availability Edition with local SSDs (The minor engine version is 20200831 or later.)
  - MySQL 5.6 on RDS High-availability Edition with local SSDs (The minor engine version is 20200831 or later.)

  > **Note** If your RDS instance is attached with read-only RDS instances, the read-only RDS instances must meet the requirements that are described in Update the minor engine version of an ApsaraDB RDS for MySQL instance.

- The dedicated proxy is enabled. For more information, see Enable the dedicated proxy service for an ApsaraDB RDS for MySQL instance.
- The dedicated proxy version of your RDS instance is V1.12.8 or later. For more information, see Upgrade the dedicated proxy version of an ApsaraDB RDS for MySQL instance.
- The total length of the proxy endpoint that you want to protect does not exceed 64 characters.

Precautions

- SSL encryption can be configured for only one proxy endpoint per proxy terminal.
- If you enable or disable SSL encryption, change the protected proxy endpoint, or update the validity period of the SSL certificate, your RDS instance restarts. Proceed with caution.

Enable SSL encryption

> **Notice** This operation triggers a restart of your RDS instance. Proceed with caution.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. Click the **Proxy Terminal (Original Read/Write Splitting)** tab.
4. Find the proxy terminal to which the proxy endpoint that you want to protect belongs. Turn on the switch next to **SSL Certificate Information**. In the dialog box that appears, select the proxy endpoint that you want to protect, and click **OK**.

Change the protected proxy endpoint

> **Notice** This operation triggers an update to the validity period of the SSL certificate. This operation also triggers a restart of your RDS instance. Proceed with caution.

1.

2. In the left-side navigation pane, click **Database Proxy**.
3. Click the **Proxy Terminal (Original Read/Write Splitting)** tab.
4. Find the proxy terminal to which the protected proxy endpoint belongs. Click Change Protected Endpoint to the right of **Protected Endpoint**. In the dialog box that appears, select a new proxy endpoint and click **OK**.

Update the validity period of the SSL certificate

> **Notice** This operation triggers a restart of your RDS instance. Proceed with caution.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. Click the **Proxy Terminal (Original Read/Write Splitting)** tab.
4. Find the proxy terminal to which the protected proxy endpoint belongs. Click **Update Expiration Time** to the right of SSL Certificate Information. In the message that appears, click **OK**.

Disable SSL encryption

> **Notice** This operation triggers a restart of your RDS instance. Proceed with caution.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. Click the **Proxy Terminal (Original Read/Write Splitting)** tab.
4. Find the proxy terminal to which the protected proxy endpoint belongs. Turn off the switch next to **SSL Certificate Information**. In the message that appears, click **OK**.

Related operations

Operation	Description
ModifyDbProxyInstanceSsl	Configures SSL encryption for a proxy endpoint of an RDS instance.
GetDbProxyInstanceSsl	Queries the SSL encryption settings for a proxy endpoint of an RDS instance.

10.7.6. View the proxy monitoring data of an ApsaraDB RDS for MySQL instance

This topic describes how to view the proxy monitoring data of an ApsaraDB RDS for MySQL instance. The monitoring data provides the CPU utilization for the proxy instances that are enabled. You can obtain the loads on the enabled proxy instances and adjust the number of proxy instances based on the monitoring data.

Prerequisites

Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance

Procedure

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. Click the **Monitoring Data** tab.
4. Select a time range. Then, you can view the **CPU Utilization** metric over the selected time range.

> **Note** The CPU Utilization (%) metric indicates the CPU utilization for the proxy instances that are enabled.

Related operations

Operation	Description
DescribeDBProxyPerformance	Queries the performance data of the dedicated proxy of an ApsaraDB RDS instance.

10.7.7. Adjust the number of dedicated proxies on an ApsaraDB RDS for MySQL instance

This topic describes how to adjust the number of dedicated proxies on an ApsaraDB RDS for MySQL instance based on monitoring data and business planning.

Prerequisites

The dedicated proxy service is enabled for your RDS instance. For more information, see Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance.

Precautions

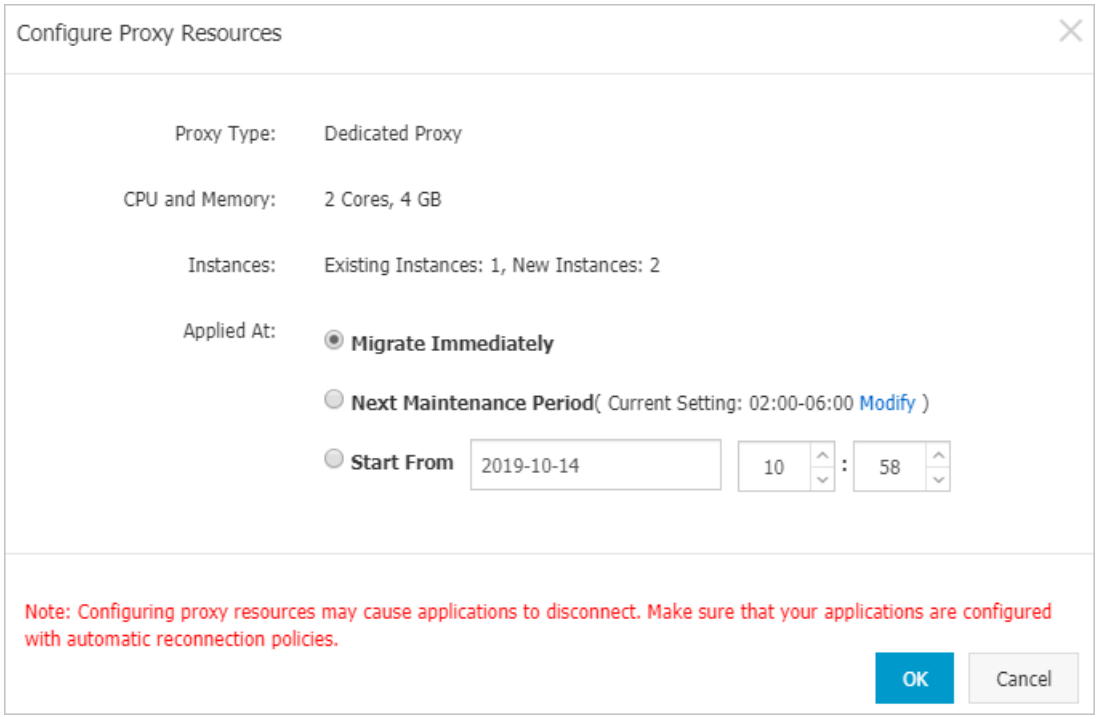
The adjustment causes a transient connection error on your application. Make sure that your application is configured to automatically reconnect to your RDS instance.

Procedure

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. In the Proxy Instance section of the Proxy Service tab, modify the number to the right of the **Instances Associated** parameter and then click **Apply** to the right of the Adjustment Plan parameter

> **Note** We recommend that you specify the number of dedicated proxies as the rounded-up integer of the total number of CPU cores for your RDS instance and its read-only RDS instances divided by 8. A maximum of 60 dedicated proxies are supported.
>
> For example, if your RDS instance has eight CPU cores and its read-only RDS instances have four CPU cores, the recommended number of dedicated proxies is 2 based on the following calculation: (8 + 4)/8 = 1.5 (rounded up to 2).

4. Set the **Applied At** parameter and click **OK**.



Related operations

Operation	Description
ModifyDBProxyInstance	Modifies the number of dedicated proxies on an ApsaraDB RDS instance.
DescribeDBProxy	Queries details about the dedicated proxies of an ApsaraDB RDS instance.

10.8. Other features

10.8.1. Upgrade the database proxy of an ApsaraDB RDS for MySQL instance from a shared proxy to a dedicated proxy

ApsaraDB RDS for MySQL provides the dedicated proxy feature. The dedicated proxy feature provides higher stability, scalability, and performance than the shared proxy feature. If the shared proxy feature is enabled for your primary RDS instance, you can upgrade the database proxy of the instance from a shared proxy to a dedicated proxy.

> **Note** Since April 1, 2021, Alibaba Cloud provides technical support only for the shared proxy feature of ApsaraDB RDS for MySQL and no longer updates or maintains the shared proxy feature. We recommend that you upgrade the database proxy of your primary RDS instance from a shared proxy to a dedicated proxy at the earliest opportunity. For more information, see [Notice] End of updates and maintenance for the shared proxy feature.

Upgrade promotions

You are provided a free trial period for the dedicated proxy feature. During the free trial period, you can use the default number of proxy instances that are provided free of charge. If you enable more proxy instances, you are charged for the additional proxy instances that you enable. The free trial period varies based on the billing method:

- If your primary RDS instance uses the pay-as-you-go billing method, the free trial period is one year.
- If your primary RDS instance uses the subscription billing method, the free trial period is at least one year and can vary based on the expiration time of the instance.
  - If you perform the upgrade more than one year before your primary RDS instance expires, the free trial period ends at the time when your primary RDS instance expires.

    > **Note** If you renew your primary RDS instance, the free trial period remains unchanged and does not extend to the new expiration time.

  - If you perform the upgrade less than one year before the expiration time of your primary RDS instance, the free trial period is one year.

Take note of the following points:

- The dedicated proxy of your primary RDS instance provides a default number of proxy instances. If you enable more proxy instances than the default number during the free trial period, you are charged for the additional proxy instances that you enable. For example, if the dedicated proxy provides six default proxy instances, you are not charged when the number of proxy instances that you enable does not exceed 6. However, if you enable seven proxy instances, you are charged for one proxy instance.
- If you disable the dedicated proxy feature for your primary RDS instance during the free trial period, the free trial period ends. You are charged for the dedicated proxy if you enable this feature again.

Prerequisites

- Your primary RDS instance and read-only RDS instances run one of the following MySQL versions and RDS editions:
  - MySQL 5.7 with a minor engine version of 20190925 or later on RDS High-availability Edition
  - MySQL 5.6 with a minor engine version of 20200229 or later on RDS High-availability Edition

> **Note**
>
> - If your primary RDS instance runs MySQL 5.6 on RDS Enterprise Edition, you cannot upgrade the database proxy of the instance from a shared proxy to a dedicated proxy.
> - If the "`current db not support db proxy`" error message appears during the upgrade, you must update the minor engine versions of your primary RDS instance and read-only RDS instances before you perform the upgrade. For more information, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.

- The shared proxy feature is enabled for your primary RDS instance. You can log on to the ApsaraDB RDS console and go to the **Database Proxy** page to check whether the shared proxy feature is enabled for your primary RDS instance.

Pricing

When you perform the upgrade, ApsaraDB RDS recommends a specific number of proxy instances based on the overall specifications of your primary RDS instance and read-only RDS instances. You are charged an hourly fee for the additional proxy instances that you enable. For more information, see Billing rules for dedicated proxy instances that are enabled on an ApsaraDB RDS for MySQL instance.

Background information

The read/write splitting feature of ApsaraDB RDS for MySQL is implemented based on the database proxy feature. For some existing RDS instances that run MySQL 5.6 or MySQL 5.7, the read/write splitting feature is implemented based on the shared proxy feature. However, the shared proxy feature cannot ensure service stability. We recommend that you upgrade the database proxies of these RDS instances from shared proxies to dedicated proxies. The dedicated proxy feature has the following advantages over the shared proxy feature:

- The dedicated proxy feature provides better stability and isolation.
- The dedicated proxy feature provides higher performance. If the instance configuration is not a bottleneck, each proxy instance can process up to 20,000 queries per second (QPS) in online transaction processing (OLTP) scenarios. This is verified by a test that is performed by using SysBench.
- The dedicated proxy feature supports scaling. To process more workloads, you can enable more proxy instances.
- The dedicated proxy feature supports performance monitoring. You can adjust the number of proxy instances based on the monitoring data and your business plan. An adjustment takes effect immediately after it is applied.
- A unified proxy endpoint is provided. This eliminates the need to modify the endpoint information on your application and reduces maintenance costs. The proxy endpoint remains valid until you release the proxy instances. For example, you may enable the read/write splitting feature during peak hours, and then release read-only RDS instances and disable the read/write splitting feature during off-peak hours. In these cases, the proxy endpoint remains connected, and you do not need to update the endpoint information on your application.
- A unified proxy endpoint is used to implement features such as read/write splitting, short-lived connection optimization, and transaction splitting.

For more information about the dedicated proxy feature, see What are database proxies?.

The following section describes how to upgrade the database proxy of your primary RDS instance from a shared proxy to a dedicated proxy.

Precautions

- After you enable the dedicated proxy feature, each connection is replicated to your primary RDS instance and read-only RDS instances in compliance with the 1:N connection model. We recommend that you specify the same connection specifications for these instances. If these instances have different connection specifications, the number of connections that are allowed is limited by the lowest connection specifications among these instances.
- If you create or restart a read-only RDS instance after you enable the dedicated proxy, only the requests over new connections are routed to the read-only RDS instance.
- The **max_prepared_stmt_count** parameter must be set to the same value for your primary RDS instance and read-only RDS instances.
- For more information, see Usage notes for database proxies.

Impacts

During the upgrade, the endpoints of your primary RDS instance and read-only RDS instances encounter a transient connection that lasts 30 seconds. The read/write splitting endpoint is also unavailable for 30 seconds.

Procedure

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. Click **Upgrade to Dedicated Proxy**.

> **Note** Wait until the upgrade is complete.

10.8.2. Upgrade the dedicated proxy version of an ApsaraDB RDS for MySQL instance

This topic describes how to upgrade the dedicated proxy version of an ApsaraDB RDS for MySQL instance.

Prerequisites

Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance

For more information about dedicated proxy versions, see Release notes of dedicated proxy versions.

Precautions

The upgrade causes a restart of the proxy instances that are enabled on the RDS instance. During the restart process, a 30-second transient connection occurs. The time when the proxy instances restart varies based on the value of the **Upgrade Time** parameter. You can select **Upgrade Immediate** or **Upgrade within maintenance period** for the parameter. We recommend that you perform the upgrade during off-peak hours. Otherwise, make sure that your application is configured to automatically reconnect to your database system.

Procedure

1.

2. In the left-side navigation pane, click **Database Proxy**.
3. Click **Upgrade Dedicated Proxy Version**.
4. In the dialog box that appears, set the **Upgrade Time** parameter and click **OK**.

Related operations

Operation	Description
UpgradeDBProxyInstanceKernelVersion	Upgrades the dedicated proxy version of an ApsaraDB RDS instance.

10.8.3. Disable the dedicated proxy of an ApsaraDB RDS for MySQL instance

This topic describes how to disable the dedicated proxy of an ApsaraDB RDS for MySQL instance.

Prerequisites

The dedicated proxy of your RDS instance is enabled. For more information, see Enable and configure the dedicated proxy feature for an ApsaraDB RDS for MySQL instance.

Procedure

Notice If the proxy terminal feature is enabled, it is disabled when the dedicated proxy is disabled. For more information, see Enable the proxy terminal feature for an ApsaraDB RDS for MySQL instance.

1.
2. In the left-side navigation pane, click **Database Proxy**.
3. In the upper-right corner of the Proxy Service tab, click **Disable Proxy Service**.
4. In the message that appears, click **OK**.

Related API operations

Operation	Description
ModifyDBProxy	Enables or disables the dedicated proxy of an RDS instance.
DescribeDBProxy	Queries details about the dedicated proxy of an RDS instance.

10.8.4. Execute hints on an ApsaraDB RDS for MySQL instance

This topic describes how to execute hints on an ApsaraDB RDS for MySQL instance.

Limits

You can execute hints only on an RDS instance that is connected by a read/write splitting endpoint. For more information, see What is read/write splitting?.

Usage

- If you use the MySQL command-line interface (CLI) to connect to your RDS instance, you must add the `-c` parameter to the hints that you want to execute. If you do not add this parameter to the hints, the MySQL CLI filters out the hints.
- You can use the `/*FORCE_MASTER*/` hint to specify to query data from the primary RDS instance. You can also use the `/*FORCE_SLAVE*/` hint to specify to query data from the secondary RDS instance.

  > **Note**
  > - Hints are not subject to consistency or transaction limits. Therefore, hints have the highest route priorities. Before you execute hints, you must evaluate whether the hints are suitable for your workloads.
  > - Hints cannot contain statements that are used to reconfigure environment variables. For example, the `/*FORCE_SLAVE*/ set names utf8;` command is not allowed. If you include these statements in hints, errors may occur in your subsequent workloads.

- You can run the `/*force_node='<The ID of an RDS instance>'*/` command to query data from a specified RDS instance. For example, if you specify the `/*force_node='rr-bpxxxxx'*/ show processlist;` command, the `SHOW PROCESSLIST` statement is executed only on the read-only RDS instance named rr-bpxxxxx. If the read-only RDS instance is faulty, the "`force hint server node is not found, please check.`" error message is returned.
- You can run the `/*force_proxy_internal*/set force_node = '<The ID of an RDS instance>';` command to always query data from the specified RDS instance. For example, after you run the `/*force_proxy_internal*/set force_node = 'rr-bpxxxxx';` command, all the subsequent commands are routed to the read-only RDS instance named rr-bpxxxxx. If the read-only RDS instance is faulty, the "`set force node 'rr-bpxxxxx' is not found, please check.`" error message is returned.

  > **Note** In most cases, we recommend that you do not use the `/*force_proxy_internal*/` syntax. This syntax specifies to route all the subsequent requests to the specified RDS instance. As a result, the read/write splitting feature becomes invalid.

11.Instance changes

11.1. ApsaraDB RDS for MySQL configuration items

This topic describes the items that you can configure for an ApsaraDB RDS for MySQL instance.

Configuration item	Description	References
Region	After your RDS instance is created, you cannot change the region where your RDS instance resides. If you want to deploy your database service in a different region, you can create an RDS instance in the region that you want. Then, you can migrate the data of your original RDS instance to the new RDS instance by using Data Transmission Service (DTS). After the data is migrated, you must update the endpoint configuration on your application and verify that your workloads run as expected on the new RDS instance. Then, you can release your original RDS instance. For more information, see Release or unsubscribe from an ApsaraDB RDS for MySQL instance.	Migrate data between ApsaraDB RDS for MySQL instances

Configuration item	Description	References
RDS Edition	Only the following downgrade and upgrade scenarios are supported: • If your RDS instance runs MySQL 5.6 on RDS Enterprise Edition, you can downgrade the RDS edition of the instance to High-availability Edition. • If your RDS instance runs MySQL 5.7 or MySQL 8.0 on RDS Basic Edition, you can downgrade the RDS edition of the instance to High-availability Edition. **Note** • If your RDS instance does not meet the preceding conditions, you cannot upgrade or downgrade the RDS edition of your RDS instance. • If none of the preceding downgrade and upgrade scenarios meets your business requirements, you can perform the following steps: i. Create an RDS instance. When you configure the parameters for the RDS instance, select the RDS edition that you want to use. For more information, see Create an ApsaraDB RDS for MySQL instance. ii. Migrate the data of your original RDS instance to the new RDS instance. For more information, see Migrate data between ApsaraDB RDS for MySQL instances. iii. Release the original RDS instance. For more information, see Release or unsubscribe from an ApsaraDB RDS for MySQL instance.	Change the specifications of an ApsaraDB RDS for MySQL instance Upgrade an ApsaraDB RDS for MySQL instance from Basic Edition to High-availability Edition
Instance type	You can change the instance type of your RDS instance regardless of the instance configuration. **Note** If the specifications of your RDS instance are outdated and cannot be directly changed, you can perform the following steps: 1. Create an RDS instance. When you configure the parameters for the RDS instance, select the instance type that you want to use. For more information, see Create an ApsaraDB RDS for MySQL instance. 2. Migrate the data of your original RDS instance to the new RDS instance. For more information, see Migrate data between ApsaraDB RDS for MySQL instances. 3. Release the original RDS instance. For more information, see Release or unsubscribe from an ApsaraDB RDS for MySQL instance.	Change the specifications of an ApsaraDB RDS for MySQL instance Enable the automatic scale-up feature for an ApsaraDB RDS for MySQL instance

Configuration item	Description	References
Storage Type	When you upgrade the RDS edition of your RDS instance to the High-availability Edition, you can change the storage type to local SSDs at the same time. This operation is supported only when your RDS instance runs RDS Basic Edition on MySQL 5.7 with standard SSDs.	Upgrade an ApsaraDB RDS for MySQL instance from Basic Edition to High-availability Edition
Storage capacity	You can increase the storage capacity of your RDS instance regardless of the instance configuration. If your RDS instance runs the RDS High-availability Edition with local SSDs, you can also decrease the storage capacity of your RDS instance. **Note** • You can decrease the storage capacity of your RDS instance only when your RDS instance runs the RDS High-availability Edition with local SSDs. • The storage capacity of a read-only RDS instance must be greater than or equal to the storage capacity of the primary RDS instance to which the read-only RDS instance is attached. • If your RDS instance is equipped with standard SSDs or enhanced SSDs (ESSDs) and does not run the RDS Basic Edition, you can increase the storage capacity of your RDS instance with no downtime. In most cases, no transient connections occur when you increase the storage capacity. • The new storage capacity of your RDS instance cannot exceed the maximum storage capacity that is supported by the selected instance type. If the maximum storage capacity that is supported by the instance type cannot meet your business requirements, you can upgrade the instance type of your RDS instance. For more information, see Primary ApsaraDB RDS instance types.	Change the specifications of an ApsaraDB RDS for MySQL instance Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance

Configuration item	Description	References
Zone	You can migrate your RDS instance across zones within the same region. After your RDS instance is migrated to a different zone, the attributes, configuration, and endpoints of your RDS instance remain unchanged. If your RDS instance runs MySQL 5.7 on RDS High-availability Edition, you must change the zone of your RDS instance when you upgrade the RDS edition to the Enterprise Edition. **Note** During the cross-zone migration process, ApsaraDB RDS migrates the data of your RDS instance from the original zone to the new zone that you specify. The amount of time that is required to migrate data increases with the amount of data that needs to be migrated.	Migrate an ApsaraDB RDS for MySQL instance across zones in the same region
Primary/secondary switchover	You can configure ApsaraDB RDS to automatically switch workloads over between your RDS instance and its secondary RDS instance. You can also manually switch workloads over between your RDS instance and its secondary RDS instance. After the switchover is complete, your RDS instance is demoted to run as the new secondary RDS instance.	Switch workloads over between primary and secondary ApsaraDB RDS for MySQL instances
Network Type	All RDS instances can be deployed in virtual private clouds (VPCs). Only the RDS instances that meet specific requirements can be deployed in the classic network. If your RDS instance supports both the VPC network type and the classic network type, you can switch the network type of your RDS instance between VPC and classic network.	Change the network type of an ApsaraDB RDS for MySQL instance
VPC and vSwitch	If your RDS instance meets specific requirements, you can change the VPC or vSwitch of your RDS instance.	切换专有网络VPC和虚拟交换机
Maintenance window	You can change the maintenance window of your RDS instance.	Set the maintenance window of an ApsaraDB RDS for MySQL instance
Data replication mode	You can change the mode based on which your RDS instance replicates data to its secondary RDS instance. This way, you can improve the availability of your database service.	Change the data replication mode of an ApsaraDB RDS for MySQL instance

Configuration item	Description	References
Instance parameter configuration	You can reconfigure some parameters of your RDS instance based on your business requirements.	For more information, see Modify the parameters of an ApsaraDB RDS for MySQL instance or Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances.
Database engine version	You can directly upgrade the database engine version of your RDS instance only from MySQL 5.5 to MySQL 5.6.	Upgrade the major engine version of an ApsaraDB RDS for MySQL instance
Billing method	You can change the billing method of your RDS instance between pay-as-you-go and subscription.	Change the billing method of an ApsaraDB RDS for MySQL instance from pay-as-you-go to subscription Change the billing method of an ApsaraDB RDS for MySQL instance from subscription to pay-as-you-go

11.2. Change the specifications of an ApsaraDB RDS for MySQL instance

This topic describes how to change the specifications of an ApsaraDB RDS for MySQL instance. The specifications include the RDS edition, instance type, and storage capacity.

For more information about the specific configuration items of an RDS instance, see ApsaraDB RDS for MySQL configuration items.

For more information about how to change the specifications of an RDS instance that runs a different database engine, see the following topics:

- Change the specifications of an ApsaraDB RDS for SQL Server instance
- Change the specifications of an ApsaraDB RDS for PostgreSQL instance
- Change the specifications of an ApsaraDB RDS for MariaDB TX instance

Prerequisites

- Your Alibaba Cloud account does not have unpaid renewal orders.
- The RDS instance is in the Running state.

Limits

- After you submit a specification change order, you cannot cancel the order. Therefore, before you submit a specification change order, we recommend that you evaluate whether the new specifications meet your business requirements.
- When you change the specifications of a read-only RDS instance, the primary RDS instance to which the read-only RDS instance is attached must be in the Running state.
- The storage capacity of a read-only RDS instance must be greater than or equal to the storage capacity of the primary RDS instance to which the read-only RDS instance is attached.
- Only the configuration items that are described in ApsaraDB RDS for MySQL configuration items can be changed.

Impacts

Storage type of the RDS instance	Configuration item	Impact
Local SSD	Instance type, RDS edition, and storage capacity.	If the host on which your RDS instance is deployed cannot provide sufficient resources, ApsaraDB RDS migrates the data of your RDS instance to a new RDS instance. The incremental data that is generated in your RDS instance during the migration process is synchronized to the new RDS instance. After the migration process is complete, ApsaraDB RDS switches your workloads over to the new RDS instance during the switching time that you specify. **Note** A specification change causes a transient connection that lasts approximately 30 seconds. We recommend that you change the specifications of your RDS instance during off-peak hours. In addition, make sure that your application is configured to automatically reconnect to your RDS instance. During transient connections, you cannot perform most of the operations that are related to databases, accounts, and network settings.

Storage type of the RDS instance	Configuration item	Impact
Standard SSD or enhanced SSD (ESSD).	Instance type and RDS edition	If the host on which your RDS instance is deployed cannot provide sufficient resources, the specification change is complete within minutes. The time that is required to change the specifications is not affected by the data volume of your RDS instance. **Note** A specification change causes a transient connection that lasts approximately 30 seconds. We recommend that you change the specifications of your RDS instance during off-peak hours. In addition, make sure that your application is configured to automatically reconnect to your RDS instance. During transient connections, you cannot perform most of the operations that are related to databases, accounts, and network settings.
	Storage capacity	You can expand the storage capacity of your RDS instance without the need to migrate data. When you change the storage capacity of your RDS instance, no transient connections occur.

Note

- After you change the specifications of your RDS instance, you do not need to manually restart the instance.
- After you change the specifications of your RDS instance, the ID and endpoints of the instance remain unchanged.
- For more information about storage types, see Storage types.

Billing

For more information, see Specification change fees.

Procedure

1.
2. In the **Configuration Information** section of the Basic Information page, click **Change Specifications**.
3. In the dialog box that appears, select a specification change method and click **Next step**. This step is required only when the RDS instance uses the subscription billing method.

> **Note** You can select one of the following specification change methods: **Upgrade** or **Downgrade**
>
> - After you change the specifications, the new specifications immediately take effect. These specification change methods are supported for both subscription RDS instances and pay-as-you-go RDS instances.
> - After you submit a specification change order, ApsaraDB RDS synchronizes the data of the RDS instance from the disk to a new RDS instance. Then, ApsaraDB RDS switches the information, such as the ID and endpoints, about the original RDS instance over to the new RDS instance based on the **specification change method that you select**. You can use the same ID and endpoints to connect to your RDS instance.

4. Change the specifications of the RDS instance.

> **Note**
>
> - You can change the instance type and storage capacity of the RDS instance regardless of the instance configuration. For more information, see Primary ApsaraDB RDS for MySQL instance types.
> - If the RDS instance runs MySQL 5.6 on RDS Enterprise Edition, you can downgrade the RDS edition of the RDS instance to RDS High-availability Edition.
> - If the RDS instance runs MySQL 5.7 or MySQL 8.0 on RDS Basic Edition, you can upgrade the RDS edition of the RDS instance to RDS High-availability Edition.
> - Make sure the new specifications that you specify for the RDS instance meet your business requirements.

5. Configure the **Switching Time** parameter.
   - **Switch Immediately After Data Migration**: After the data of the RDS instance is migrated to a new RDS instance, ApsaraDB RDS immediately switches your workloads over to the new RDS instance.
   - **Switch Within Maintenance Window**: ApsaraDB RDS switches your workloads over to the new RDS instance within the maintenance window that you specify. For more information, see Set the maintenance window of an ApsaraDB RDS for MySQL instance.

> **Note**
>
> - If you are using RDS Basic Edition, no secondary RDS instance is provided as a hot standby for the RDS instance. Therefore, your database service is unavailable during a specification change for a long period of time. We recommend that you change the specifications of the RDS instance during off-peak hours to prevent interruptions to your workloads.
> - If you are not using RDS Basic Edition, a transient connection that lasts approximately 30 seconds may occur during the specification change. The transient connection does not interrupt your workloads. However, we still recommend that you change the specifications of the RDS instance during off-peak hours or make sure that your application is configured to automatically reconnect to the RDS instance.
> - If you select **Switch Within Maintenance Window**, the RDS instance stays in the **Upgrading** state until the switchover is complete. During the switchover process, you cannot perform specific operations on the RDS instance. For example, you cannot upgrade or downgrade the RDS instance, upgrade the database engine version of the RDS instance, or migrate the RDS instance across zones.
> - The storage capacity expansion or change in the ESSD storage type does not interrupt your workloads and can immediately take effect. In this case, you do not need to select **Switch Within Maintenance Window**.

6. Read and select Terms of Service, click **Pay Now**, and then complete the payment.

> **Warning**
>
> - After you submit a specification change order, you cannot cancel the order. Therefore, before you submit a specification change order, we recommend that you evaluate whether the new specifications meet your business requirements.
> - After you submit a specification change order, do not perform DDL operations before the specification change is applied.

FAQ

- How do I change the storage type of my RDS instance between local SSDs, standard SSDs, and ESSDs?

  For more information, see How do I migrate an ApsaraDB RDS instance from standard or enhanced SSDs to local SSDs?

- When I expand the storage capacity of my RDS instance, what do I do if an error that indicates insufficient storage resources occurs?

  You can migrate your RDS instance to a different zone and then expand the storage capacity of your RDS instance again. For more information, see Migrate an ApsaraDB RDS for MySQL instance across zones in the same region. After you migrate your RDS instance to a different zone, the endpoints of your RDS instance remain unchanged. However, the IP addresses that are associated with the endpoints change. Make sure that your application is configured to automatically reconnect to your RDS instance.

- If I expand only the storage capacity of my RDS instance, does ApsaraDB RDS migrate the data of my RDS instance to a new RDS instance?

In most cases, if your RDS instance uses standard SSDs or ESSDs and does not run RDS Basic Edition, no transient connections occur when you expand the storage capacity. If your RDS instance uses local SSDs, ApsaraDB RDS migrates the data of your RDS instance based on the storage resources on the host on which your RDS instance is deployed:

- The host on which your RDS instance is deployed can provide sufficient storage. In this case, you can directly expand the storage capacity. The expansion process does not interrupt your workloads.
- The host on which your RDS instance is deployed cannot provide sufficient storage. In this case, ApsaraDB RDS creates a primary RDS instance and a secondary RDS instance on a different host that can provide sufficient storage. Then, ApsaraDB RDS synchronizes the data of your original RDS instance to the new primary and secondary RDS instances. The synchronization does not interrupt the workloads on your original RDS instance. However, after the synchronization is complete, a switchover of your workloads causes a transient connection that lasts approximately 30 seconds.

> **Note** You cannot obtain the amount of available storage in the host on which an RDS instance is deployed.

- When I upgrade my primary RDS instance, does ApsaraDB RDS automatically upgrade the read-only RDS instances that are attached to my primary RDS instance?

  No, when you upgrade your primary RDS instance, ApsaraDB RDS does not automatically upgrade the read-only RDS instances that are attached to your primary RDS instance. You must manually upgrade the read-only RDS instances.

- When I change the specifications of my RDS instance, are my online workloads interrupted?

  For more information, see Impacts.

- After I change the specifications of my RDS instance, do the endpoints of my RDS instance change?

  No, after you change the specifications of your RDS instance, the endpoints of your RDS instance remain unchanged. An example endpoint is rm-bpxxxxx.mysql.rds.aliyuncs.com. However, the IP addresses that are associated with the endpoints may change. We recommend that you add the endpoints to your application. Do not add the IP addresses to your application.

- What factors affect the period of time that is required to change the specifications of my RDS instance?

  For more information, see Which factors affect the time that is required to change the specifications of my ApsaraDB RDS for MySQL instance?

Related operations

Operation	Description
ModifyDBInstanceSpec	Changes the specifications of an ApsaraDB RDS instance.

11.3. Upgrade the storage type of an ApsaraDB RDS for MySQL instance from standard SSDs to ESSDs

Alibaba Cloud provides enhanced SSDs (ESSDs), which come in different performance levels (PLs). ESSDs of PL1 are designed based on the new-generation distributed block storage architecture. An ESSD of PL1 delivers higher IOPS, higher throughput, and more stable I/O performance than an standard SSD. However, the fee for an ESSD of PL1 is the same as the fee for a standard SSD in most Alibaba Cloud regions. You can upgrade the storage type of an ApsaraDB RDS for MySQL instance from standard SSDs to ESSDs of PL1 to increase cost-effectiveness.

Prerequisites

- The RDS instance is in the Running state.
- The RDS instance runs one of the following MySQL versions and RDS editions:
  - MySQL 8.0 on RDS High-availability Edition or RDS Basic Edition with standard SSDs
  - MySQL 5.7 on RDS High-availability Edition or RDS Basic Edition with standard SSDs
- The minor engine version of the RDS instance is 20201031 or later. This requirement must be met if the RDS instance runs RDS Basic Edition. For more information about how to upgrade the minor engine version of an RDS instance, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.

Performance comparison between ESSDs of PL1 and standard SSDs

An ESSD of PL1 is charged at a similar price but delivers significantly higher performance than a standard SSD. The following table describes the differences between ESSDs of PL1 and standard SSDs.

Comparison item	ESSD of PL1	Standard SSD
Maximum capacity per SSD (GiB)	20 to 32,768	32,768
Maximum IOPS	50,000	25,000
Maximum throughput (MB/s)	350	300
IOPS per SSD	min{1,800 + 50 × Capacity, 50,000}	min{1,800 + 30 × Capacity, 25,000}
Throughput per SSD (MB/s)	min{120 + 0.5 × Capacity, 350}	min{120 + 0.5 × Capacity, 300}
Average random write latency per connection (ms)	0.2	0.5 to 2

Billing rules

When you upgrade the storage type, you are charged a specific fee. The fee varies based on the region where the RDS instance resides. You can view the fee when you perform the upgrade.

Impacts

- When you upgrade the storage type, a transient connection that lasts approximately 30 seconds occurs. We recommend that you perform the upgrade during off-peak hours and make sure that your application is configured to automatically reconnect to the RDS instance.
- When the storage type is being upgraded, you cannot perform operations on the RDS instance. For example, you cannot upgrade or downgrade the RDS instance, upgrade the database engine version of the RDS instance, or migrate the RDS instance across zones.

Procedure

1.
2. In the **Basic Information** section of the page that appears, click the button for upgrading the storage type next to **Storage Type**.

   > **Note** If the button cannot be found, you must check whether the RDS instance meets all requirements that are specified in the "**Prerequisites**" section of this topic.

3. On the tab that appears, read and select the terms of service, click **Pay Now**, and then complete the payment.
   The status of the RDS instance changes to **Upgrading/Downgrading**. When the status of the RDS instance changes back to **Running**, the upgrade is complete.

FAQ

- Why am I unable to select the ESSD storage type for my RDS instance?

  If the zones in which your RDS instance resides cannot provide sufficient resources or do not support ESSDs, you cannot select the ESSD storage type for your RDS instance. In this case, you must update the minor engine version of your RDS instance and migrate your RDS instance to zones in which ESSDs are supported before you upgrade the storage type. For more information, see Update the minor engine version of an ApsaraDB RDS for MySQL instance and Migrate an ApsaraDB RDS for MySQL instance across zones in the same region.

- When I upgrade the storage type of my RDS instance, does ApsaraDB RDS automatically upgrade the storage types of the read-only RDS instances that are attached to my RDS instance?

  No, ApsaraDB RDS does not automatically upgrade the storage types of the read-only RDS instances. You must repeat the procedure in this topic to manually upgrade the storage type of each read-only RDS instance.

- When I change the specifications of my RDS instance, are my online workloads interrupted?

  For more information, see the "Impacts" section of this topic.

- Do the endpoints and IP addresses of my RDS instance change after I change the storage type?

  The endpoints of your RDS instance remain unchanged after you change the storage type. An example endpoint is rm-bpxxxxx.mysql.rds.aliyuncs.com. However, the IP addresses that are associated with the endpoints may change. We recommend that you add the endpoints rather than the IP addresses to your application.

- Which factors affect the amount of time that is required to change the storage type of my RDS instance?

  For more information, see Which factors affect the time that is required to change the specifications of my ApsaraDB RDS for MySQL instance?

Related operations

Operation	Description
ModifyDBInstanceSpec	Changes the specifications of an ApsaraDB RDS instance.

11.4. Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance

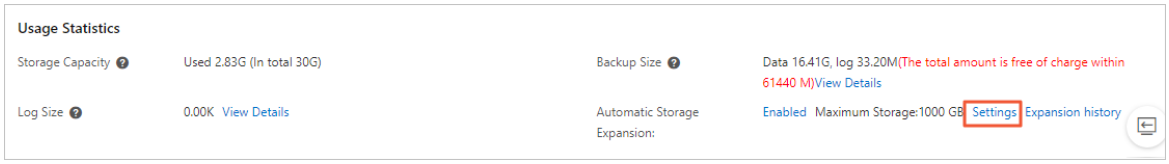
This topic describes how to configure automatic storage expansion for an ApsaraDB RDS for MySQL instance.

Prerequisites

- The RDS instance runs RDS High-availability Edition and uses standard SSDs or enhanced SSDs (ESSDs).
- The balance in your Alibaba Cloud account is sufficient for the expansion.
- Your RDS instance is in the Running state.

Procedure

1.
2. In the **Usage Statistics** section of the page that appears, click **Settings** to the right of **Automatic Storage Expansion**.



3. Configure the following parameters.

Parameter	Description
Automatic Resource Scalability	The switch that is used to enable or disable automatic storage expansion.

Parameter	Description
Available Storage Threshold<=	The threshold based on which ApsaraDB RDS triggers an automatic storage expansion. The threshold is expressed as a percentage. If the storage usage reaches the threshold, ApsaraDB RDS increases the storage capacity of the RDS instance. **Note** The maximum amount of storage that you can expand is the largest value among the following values: ○ 5 GB. ○ 15% of the current storage capacity of the RDS instance. ○ The value of the Disk Space metric over the previous 7 hours. You can obtain the value of this metric from the Standard Monitoring tab. For more information, see 查看监控信息.
Maximum Storage	The maximum storage capacity that is allowed for an automatic storage expansion. The value of this parameter must be greater than or equal to the current storage capacity of the RDS instance. ○ RDS instances that use ESSDs: 32,000 GB ○ RDS instances that use standard SSDs: 6,000 GB **Note** The standard SSD storage type is phased out. We recommend that you upgrade the storage type to ESSDs. For more information, see Upgrade the storage type of an ApsaraDB RDS for MySQL instance from standard SSDs to ESSDs.

4. Click **Confirm**.

Related operations

Operation	Description
ModifyDasInstanceConfig	Configures automatic storage expansion.

11.5. Enable the automatic scale-up feature for an ApsaraDB RDS for MySQL instance

The automatic scale-up feature of Database Autonomy Service (DAS) can automatically scale up your ApsaraDB RDS for MySQL instance based on your workloads to handle traffic spikes and ensure the stability of your RDS instance.

Prerequisites

- Your RDS instance runs RDS High-availability Edition with standard SSDs or enhanced SSDs (ESSDs).

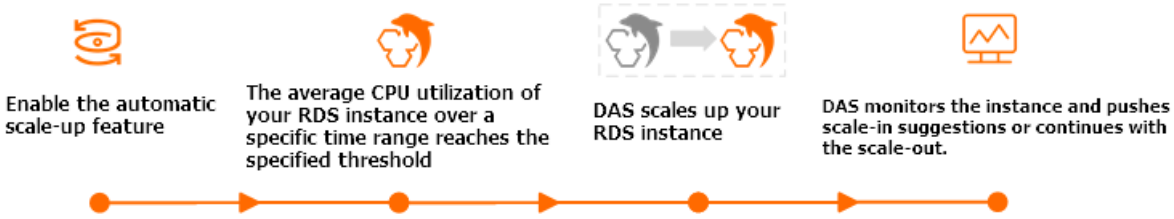
> **Note** The automatic scale-up feature is not supported for RDS instances whose instance types are phased out. If the instance type of your RDS instance is phased out, you must change the instance type of the instance before you can enable the automatic scale-up feature. For more information about how to change the instance type of an RDS instance, see Change the specifications of an ApsaraDB RDS for MySQL instance.

- DAS is activated within your Alibaba Cloud account. For more information, see Autonomy center.
- The balance in your Alibaba Cloud account is sufficient for scale-ups.

Billing rules

By default, the automatic scale-up feature is disabled and does not produce fees. If you enable the automatic scale-up feature, you are charged based on the new instance type that is selected during the automatic scale-up. The price for a scale-up varies based on the instance configuration, such as the region and the new instance type. For more information, visit the ApsaraDB RDS buy page.

Automatic scale-up process



After you enable the automatic scale-up feature for your RDS instance, DAS scales up the RDS instance based on the loads on the RDS instance if the average CPU utilization of the RDS instance over the specified observation window reaches **the specified threshold**. The specifications of your RDS instance after a scale-up cannot exceed the specifications that are specified by the **Upper Limit of Specifications** parameter. You can specify this parameter in the ApsaraDB RDS console. After your RDS instance is scaled up, DAS continues to monitor the CPU utilization of the RDS instance. If the CPU utilization over the specified observation window reaches the specified threshold again, DAS scales up your RDS instance again. This process continues until the specifications of your RDS instance reach the maximum specifications that are specified by the Upper Limit of Specifications parameter.

> **Note** After your RDS instance is scaled up, the RDS instance cannot be automatically scaled down. You can subscribe to scale-down suggestions and manually change the specifications of your RDS instance based on the scale-down suggestions. For more information, see Event subscription. If the average CPU utilization of your RDS instance remains lower than 30% over the specified **Observation Window**, ApsaraDB RDS pushes scale-down suggestions to you. You can manually change the specifications of your RDS instance based on the scale-down suggestions. For more information, see Change the specifications of an ApsaraDB RDS for MySQL instance.

Impacts

- When the new specifications are being applied to an RDS instance, a transient connection that lasts approximately 30 seconds may occur. Make sure that your application is configured to automatically reconnect to the RDS instance.
- If an RDS instance does not run the latest minor engine version, DAS updates the minor engine version

of the RDS instance to the latest version during a specification change. This ensures the performance and stability of the RDS instance.

- After you enable the automatic scale-up feature for an RDS instance, ApsaraDB RDS assigns the AliyunServiceRoleForDAS service-linked role to DAS. This way, DAS can access the resources that are related to ApsaraDB RDS.

Procedure

1.
2. In the **Configuration Information** section of the page that appears, click **Settings** to the right of **Automatic Scale-up**.
3. In the dialog box that appears, configure the following parameters.

Parameter	Description
Automatic Resource Elasticity	Specify whether to enable the automatic scale-up feature.
Observation Window	Select the time range during which you want to observe the CPU utilization of the RDS instance. Valid values: ◦ **20 minutes** ◦ **30 minutes** ◦ **40 minutes** ◦ **60 minutes** **Note** DAS periodically checks the CPU utilization of the RDS instance during the selected time range. If the CPU utilization reaches the value of the **CPU Trigger Threshold** parameter, an automatic scale-up is triggered.
CPU Trigger Threshold	Specify the average CPU utilization threshold based on which DAS scales up the RDS instance. Unit: percent (%). Valid values: 50 to 90. You can adjust the threshold at a step size of 10%.
Upper Limit of Specifications	Select the maximum specifications that are supported by the automatic scale-up feature. **Note** If the average CPU utilization of the RDS instance reaches the value of the **CPU Trigger Threshold** parameter over the specified observation window, DAS scales up your RDS instance based on your business requirements. If the average CPU utilization of the RDS instance reaches the value of the **CPU Trigger Threshold** parameter again over the specified observation window, DAS scales up the RDS instance again. This process continues until the specifications of the RDS instance reach the maximum specifications that are allowed.

4. Click **OK**.
   In the **Configuration Information** section of the page, the status of **Automatic Scale-up** changes to **Enabled**.

Related information

- Change the specifications of an ApsaraDB RDS for MySQL instance
- Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance

11.6. Switch workloads over between primary and secondary ApsaraDB RDS for MySQL instances

ApsaraDB RDS for MySQL provides the primary/secondary switchover feature to ensure high availability. If the primary RDS instance of your database system fails, ApsaraDB RDS automatically switches your workloads over from the primary RDS instance to the secondary RDS instance to ensure service availability. After the primary/secondary switchover is complete, the secondary RDS instance serves as the primary RDS instance. The endpoint that is used to connect to your database system remains unchanged. Your application can automatically connect to the new primary RDS instance by using the endpoint. You can also manually switch your workloads over between the primary RDS instance and the secondary RDS instance.

Prerequisites

The primary RDS instance runs one of the following RDS editions:

- High-availability Edition
- Enterprise Edition

> **Note** If you use an RDS instance that runs RDS Basic Edition, no secondary RDS instance is provided, and the primary/secondary switchover feature is not supported.

Context

- Automatic primary/secondary switchover: By default, the automatic primary/secondary switchover feature is enabled. If the primary RDS instance fails, ApsaraDB RDS automatically switches your workloads over to the secondary RDS instance. For more information about the causes of primary/secondary switchovers, see Reasons for primary/secondary switchovers.
- Manual primary/secondary switchover: You can manually switch your workloads over between the primary RDS instance and the secondary RDS instance even if the automatic primary/secondary switchover feature is enabled. You can perform manual primary/secondary switchovers for disaster recovery drills. You can also perform manual primary/secondary switchovers if you use the multi-zone deployment method and want to connect your application to the RDS instance in the zone that is closest to your application.

> **Note** Data is synchronized between the primary RDS instance and the secondary RDS instance in real time. You can access only the primary RDS instance. The secondary RDS instance runs only as a standby.

For more information about how to switch workloads over between the primary and secondary RDS instances that run different database engines, see the following topics:

- Switch workloads over between primary and secondary ApsaraDB RDS for SQL Server instances

- Switch workloads over between primary and secondary ApsaraDB RDS for PostgreSQL instances
- Switch workloads over between primary and secondary ApsaraDB RDS for MariaDB TX instances

Impacts

- Transient connections may occur during a primary/secondary switchover. Make sure that your application is configured to automatically reconnect to your database system.
- After a primary/secondary switchover, the read-only RDS instances that are attached to the primary RDS instance must re-establish the connections that are used to replicate data to and synchronize incremental data from the primary RDS instance. As a result, the data on the read-only RDS instances shows latencies of a few minutes.
- A primary/secondary switchover does not cause changes to the endpoints that are used to connect to your database system.

Perform a manual primary/secondary switchover

1.
2. In the left-side navigation pane, click **Service Availability**.
3. In the **Availability Information** section of the page that appears, click **Switch Primary/Secondary Instance**.
4. Specify the time at which you want to perform a primary/secondary switchover. Then, click **OK**.

> **Note** You cannot perform specific operations during a primary/secondary switchover. For example, you cannot manage databases and accounts or change the network type. We recommend that you select **Switch Within Maintenance Window**.

Disable the automatic primary/secondary switchover feature for a short period of time

By default, the automatic primary/secondary switchover feature is enabled. If the primary RDS instance fails, ApsaraDB RDS automatically switches your workloads over from the primary RDS instance to the secondary RDS instance. You can disable the automatic primary/secondary switchover feature in the following situations:

- A large-scale sales promotion during which you do not want a primary/secondary switchover to affect system availability
- An important application upgrade during which you do not want a primary/secondary switchover to cause unexpected issues
- A major event during which you do not want a primary/secondary switchover to affect system stability

1.
2. In the left-side navigation pane, click **Service Availability**.
3. In the **Availability Information** section of the page that appears, click **Configure Primary/Secondary Switchover**.

> **Note** If you cannot find **Configure Primary/Secondary Switchover**, you must check whether the primary RDS instance meets all prerequisites.

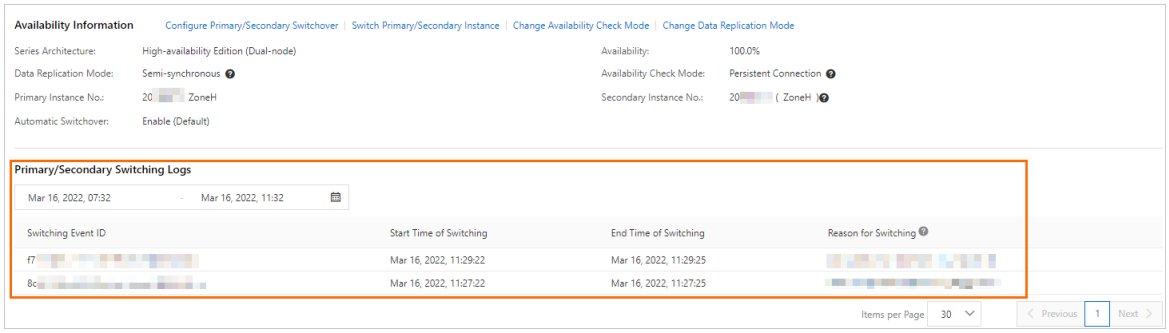
4. Select **Disable Temporarily**, configure the **Deadline** parameter, and then click **OK**.

> **Note**
>
> - When the date and time specified by the **Deadline** parameter arrives, the automatic primary/secondary switchover feature is enabled.
> - If you do not configure the Deadline parameter, the automatic primary/secondary switchover is disabled for one day. You can set the Deadline parameter to 23:59:59 seven days later at most.

After you disable the automatic primary/secondary switchover feature, you can go to the **Service Availability** page to check the deadline after which the automatic primary/secondary switchover feature can be automatically enabled.

View primary/secondary switchover logs

1.
2. In the left-side navigation pane, click **Service Availability**.
3. In the **Primary/Secondary Switching Logs** section of the page that appears, select a time range and view the primary/secondary switchover logs that are generated over the selected time range.



FAQ

- Can I access the secondary RDS instance of my database system?

  No, you cannot access the secondary RDS instance of your database system. You can access only the primary RDS instance of your database system. The secondary RDS instance runs only as a standby.

- Do I need to manually switch my workloads over from the secondary RDS instance to the primary RDS instance after a primary/secondary switchover?

  No, you do not need to manually switch your workloads over from the secondary RDS instance to the primary RDS instance after a primary/secondary switchover. The data in the primary RDS instance is the same as the data in the secondary RDS instance. After a primary/secondary switchover, the secondary RDS instance serves as the new primary RDS instance. No additional operations are required.

- Each time a primary/secondary switchover is performed, my RDS instance does not run as expected 10 minutes after the primary/secondary switchover is complete. What are the possible causes? How do I handle the issue?

If an exception on your RDS instance triggers a primary/secondary switchover to ensure high availability, your application may fail to identify and respond to the changes to the connections. If no timeout periods are specified for socket connections, your application waits for the database to return the results. In most cases, your application is disconnected after hundreds of seconds. During this period, some connections to the database cannot work as expected, and a large number of SQL statements fail to be executed. To avoid invalid connections, we recommend that you configure the **connectTimeout** and **socketTimeout** parameters to prevent your application from waiting for a long period of time due to network errors. This reduces the time required to recover from failures.

You must configure these parameters based on your workloads and usage modes. For online transactions, we recommend that you set **connectTimeout** to 1 to 2 seconds and **socketTimeout** to 60 to 90 seconds. This configuration is for reference only.

Related operations

Operation	Description
Switch services between a primary ApsaraDB for RDS instance and its secondary instance	Switches workloads over between primary and secondary ApsaraDB RDS instances.
Enable or disable automatic primary/secondary switchover	Enables or disables the automatic primary/secondary switchover feature for an ApsaraDB RDS instance.
Query settings of automatic primary/secondary switchover	Queries the settings of the automatic primary/secondary switchover feature for an ApsaraDB RDS instance.

11.7. Reasons for primary/secondary switchovers

This topic describes the reasons why a primary/secondary switchover is triggered for an ApsaraDB RDS for MySQL instance.

A primary/secondary switchover is triggered due to one of the following reasons:

- Vulnerabilities

  Alibaba Cloud detects vulnerabilities in each RDS instance. An RDS instance may fail to run as normal due to vulnerabilities. If this occurs, ApsaraDB RDS fixes the vulnerabilities in the secondary RDS instance of an RDS instance. Then, ApsaraDB RDS initiates a primary/secondary switchover during the specified maintenance window to switch your workloads over to the secondary RDS instance. For more information, see Set the maintenance window of an ApsaraDB RDS for MySQL instance. In most cases, if ApsaraDB RDS detects a high-risk vulnerability in an RDS instance, ApsaraDB RDS fixes the high-risk vulnerability at the earliest opportunity and triggers a primary/secondary switchover.

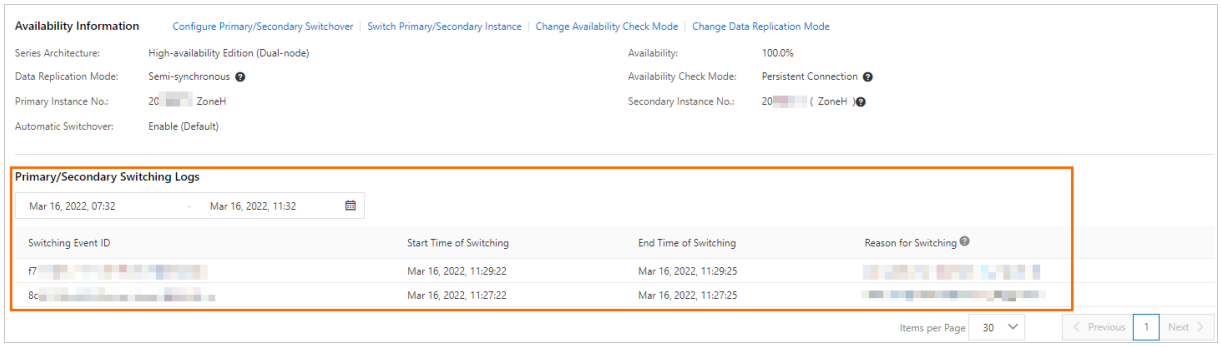
- Manual operations

  You or an authorized Alibaba Cloud technical expert manually performs a primary/secondary switchover.

- Instance failures

Alibaba Cloud detects failures in each RDS instance. An RDS instance may fail to run as normal due to failures. If this occurs, ApsaraDB RDS initiates a primary/secondary switchover to switch your workloads over to the secondary RDS instance of an RDS instance. This minimizes the impacts of the failures.

After a primary/secondary switchover is complete, the status of an RDS instance shows as **Running**. You do not need to perform other operations, and an RDS instance can run as normal. If you want to view the primary/secondary switchover logs, go to the **Primary/Secondary Switching Logs** section of the **Service Availability** page in the ApsaraDB RDS console.



11.8. Set the maintenance window of an ApsaraDB RDS for MySQL instance

This topic describes how to set the maintenance window of an ApsaraDB RDS for MySQL instance. After you set a maintenance window for your RDS instance, the backend system performs maintenance on your instance during the maintenance window. This ensures the stability of your RDS instance. The default maintenance window spans from 02:00:00 to 06:00:00. We recommend that you set the maintenance window to an off-peak hour based on your business requirements. This allows you to avoid interruptions to your workloads.

For more information about how to set the maintenance window of an RDS instance that runs another database engine, see the following topics:

- Set the maintenance window of an ApsaraDB RDS for SQL Server instance
- Set the maintenance window of an ApsaraDB RDS for PostgreSQL instance
- Set the maintenance window of an ApsaraDB RDS for MariaDB TX instance

Precautions

- Before the maintenance starts, ApsaraDB RDS sends emails to the contacts that are associated with your Alibaba Cloud account. We recommend that you check your email box on a regular basis to obtain up-to-date information.
- When the maintenance window arrives, your RDS instance enters the **Maintaining Instance** state. This ensures a smooth maintenance process. Database access and query operations such as performance monitoring are not affected while the instance is in this state. However, except for account and database management and IP address whitelist configuration, modification operations such as upgrade, downgrade, and restart are temporarily unavailable.
- During the maintenance window, one or two transient connections may occur. Make sure that your application is configured to automatically reconnect to your RDS instance.

Modify the maintenance window of a single RDS instance

1.
2. In the **Configuration Information** section, click **Configure** next to **Maintenance Window**.
3. Select an appropriate maintenance window and click OK to save the setting.

> **Note** The time zone of the maintenance window is the same as that of the computer that you use to log on to the ApsaraDB RDS console.

Modify the maintenance window of multiple RDS instances at a time

1.
2. Select the RDS instances whose maintenance window you want to modify and click **Modify Maintenance Window** blow the instance list.
3. In the **Modify Maintenance Window** dialog box, select an appropriate maintenance window and click **OK**.

> **Note** The time zone of the maintenance window is the same as that of the computer that you use to log on to the ApsaraDB RDS console.

Related operations

Operation	Description
Modify the maintenance time	Modifies the maintenance window of an ApsaraDB RDS instance.

11.9. Migrate an ApsaraDB RDS for MySQL instance across zones in the same region

Prerequisites

-
-
- Update the minor engine version of an ApsaraDB RDS for MySQL instance
- Regions and zones
- [Important] RDS network link upgrade
- Primary ApsaraDB RDS for MySQL instance types Change the specifications of an ApsaraDB RDS for MySQL instance
- Migrate an ApsaraDB RDS for PostgreSQL instance across zones in the same region
- Migrate an ApsaraDB RDS for SQL Server instance across zones in the same region
-
-

- Note
  -
  -
- Fix database shard connections
- What is DTS?

1.
2.
3. Note
   -
   - Migrate an ApsaraDB RDS for MySQL instance across zones in the same region

Operation	
Migrate an instance across zones	

11.10. Change the data replication mode of an ApsaraDB RDS for MySQL instance

This topic describes how to change the data replication mode of an ApsaraDB RDS for MySQL instance. The data replication mode specifies how data is replicated from the RDS instance to its secondary RDS instances. A suitable data replication mode increases the availability of your database service.

Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability Edition (with local SSDs)
- MySQL 5.7 on RDS High-availability Edition (with local SSDs)
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5

Data replication modes

- Synchronous

  - After an update that is initiated by your application is complete on the primary RDS instance, the related log is synchronized to all the secondary RDS instances. The transaction that is used to perform the update is considered committed only after at least one of the secondary RDS instances receives and stores the log.
  - In synchronous mode, data replication remains synchronous and cannot be demoted to the asynchronous mode.
  - Synchronous replication is supported only when your database system consists of three or more RDS instances. This means that only the RDS Enterprise Edition supports synchronous replication. In addition, if the primary RDS instance runs the RDS Enterprise Edition, you cannot change the data replication mode.

- Semi-synchronous

  After an update that is initiated by your application is complete on the primary RDS instance, the related log is synchronized to all the secondary RDS instances. The transaction that is used to perform the update is considered committed after the secondary RDS instances receive the log. Your database system does not need to wait for the secondary RDS instances to replay the log.

  If the secondary RDS instances are unavailable or a network interruption occurs between the primary and secondary RDS instances, the semi-synchronous mode is demoted to the asynchronous mode.

- Asynchronous

  When your application initiates a request to add, delete, or modify data, the primary RDS instance responds to your application immediately after it completes the requested operation. At the same time, the primary RDS instance asynchronously replicates data to the secondary RDS instances. In asynchronous mode, the unavailability of the secondary RDS instances does not interrupt the workloads on the primary RDS instance. However, the unavailability of the primary RDS instance may cause data inconsistencies between the primary and secondary RDS instances.

> **Note** In extreme circumstances, the High-availability Edition may not be able to prevent data losses. If you require a data security level of 100% and a recovery point objective (RPO) of 0, we recommend that you use the Enterprise Edition. For more information, see Enterprise Edition.

Data replication modes supported by various MySQL versions and RDS editions

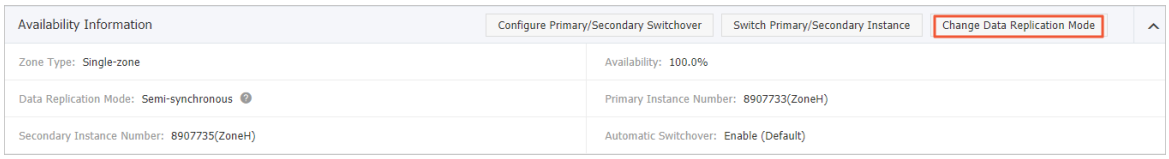
MySQL version	RDS edition	Replication mode
MySQL 8.0	Enterprise Edition	Synchronous
	High-availability Edition	• Semi-synchronous • Asynchronous
MySQL 5.7	Enterprise Edition	Synchronous
	High-availability Edition	• Semi-synchronous • Asynchronous
	Enterprise Edition	Synchronous

MySQL version	RDS edition	Replication mode
MySQL 5.6	High-availability Edition	• Semi-synchronous • Asynchronous
MySQL 5.5	High-availability Edition	• Semi-synchronous • Asynchronous

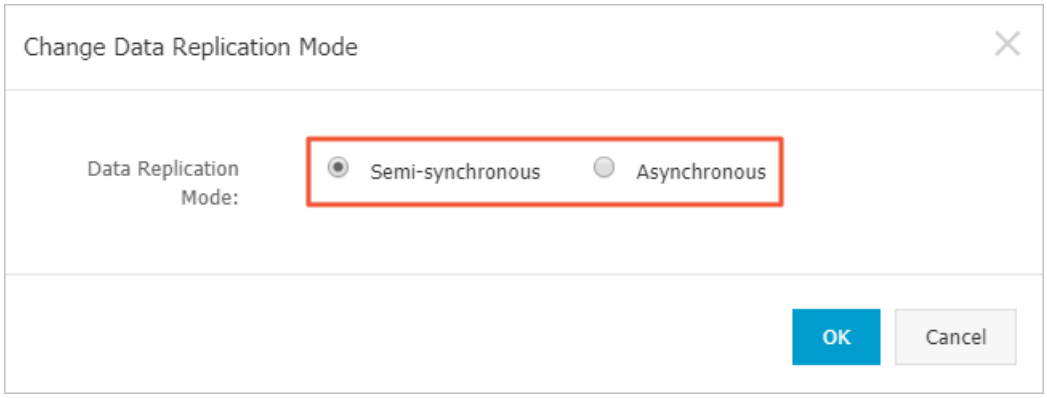
Note The RDS Basic Edition allows you to have only a single RDS instance in your database system and therefore does not require data replication.

Procedure

1.
2.
3.
4. In the left-side navigation pane, click **Service Availability**.
5. Click **Change Data Replication Mode**.



6. In the dialog box that appears, select a data replication mode and click **OK**.



FAQ

- Which data replication mode is recommended?

  You can select a data replication mode based on your business requirements. If you require strong data consistency to ensure finance-grade reliability, we recommend that you use the synchronous mode. The synchronous mode is supported only for the RDS Enterprise Edition. If you require quick responses, we recommend that you use the asynchronous mode. In other scenarios, you can use the semi-synchronous mode.

- Why am I unable to change the data replication mode of my RDS instance?

Different RDS instances support different data replication modes. For more information, see the "Data replication modes supported by various MySQL versions and RDS editions" section of this topic.

Related operations

Operation	Description
Change the high availability mode and data replication mode	Changes the data replication mode and high availability mode of an ApsaraDB RDS instance.

11.11. Change the billing method of an ApsaraDB RDS for MySQL instance from pay-as-you-go to subscription

This topic describes how to change the billing method of an ApsaraDB RDS for MySQL instance from pay-as-you-go to subscription.

Prerequisites

- Your RDS instance is not using a phased-out instance type. For more information, see Primary ApsaraDB RDS instance types. If your RDS instance uses a phased-out instance type, you must change the instance type before you change the billing method of your RDS instance from pay-as-you-go to subscription. For more information, see Change the specifications of an ApsaraDB RDS for MySQL instance.
- The billing method of your RDS instance is pay-as-you-go.
- Your RDS instance is in the Running state.
- Your RDS instance has no unpaid subscription orders.

Impacts

A billing method change for your RDS instance does not affect the workloads on your RDS instance.

Precautions

- If your RDS instance has an unpaid subscription order, the order becomes invalid when you change the instance type. In this case, you must cancel the order in the Billing Management console. Then, you can change the billing method of your RDS instance again.

Procedure

1. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Instances**. In the top navigation bar, select the region where your RDS instance resides.
2. Find your RDS instance and use one of the following methods to go to the **Switch to Subscription Billing** page:
   - Click **Switch to Subscription Billing** in the **Billing Method** column.
   - Click the ID of your RDS instance. In the **Status** section of the page that appears, click **Subscription Billing** on the right of **Billing Method**.
3. Configure the **Duration** parameter. Then, read and select Terms of Service.

4. Click **Pay Now**.

   > **Note** ApsaraDB RDS generates a subscription order. You must pay for the order. If the order is not paid or canceled, you cannot purchase an RDS instance or change the billing method of your RDS instance from pay-as-you-go to subscription. You can pay for or cancel the order in the Billing Management console.

5. Complete the payment.

Related operations

Operation	Description
Change the billing method	Changes the billing method of an ApsaraDB RDS instance.

11.12. Change the billing method of an ApsaraDB RDS for MySQL instance from subscription to pay-as-you-go

This topic describes how to change the billing method of an ApsaraDB RDS for MySQL instance from subscription to pay-as-you-go.

Prerequisites

- Your RDS instance uses the subscription billing method. For more information about the billing methods of ApsaraDB RDS, see Billable items, billing methods, and pricing.
- Your RDS instance is in the Running state.

  > **Note** If a subscription RDS instance is locked due to expiration, you must first renew the RDS instance. For more information about how to renew an RDS instance, see Manually renew an ApsaraDB RDS for MySQL instance.

- Your RDS instance does not use a phased-out instance type. For more information, see Primary ApsaraDB RDS instance types. If your RDS instance uses a phased-out instance type, you must change the instance type before you change the billing method of your RDS instance to pay-as-you-go.

Billing

After you change the billing method of your RDS instance to pay-as-you-go, a refund is returned based on the original payment method.

Refund = Fee actually paid - Fee for consumed resources

- The fee actually paid is the money that you paid and does not include the amount that is covered by coupons or vouchers.
- The fee for consumed resources is calculated based on the following formula:

  Fee for consumed resources = Daily fee × Consumed subscription duration × Discount for the consumed subscription duration

The daily fee is equal to the order-specific fee divided by 30.

> **Note** The consumed subscription duration is accurate to days. The part that is less than one day is counted as one day.

Description

A billing method change for your RDS instance does not affect the workloads on your RDS instance.

> **Note** If you want to use the RDS instance for a long period of time, we recommend that you use the subscription billing method because the subscription billing method is more cost-effective than the pay-as-you-go billing method and you are offered more discounts for longer subscription durations.

Procedure

1. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Instances**. In the top navigation bar, select the region where the RDS instance resides.
2. Find the RDS instance and use one of the following methods to go to the **Switch to Pay-as-you-go Billing** page:
   - Click **Switch to Pay-as-you-go Billing** in the **Billing Method** column.
   - Click the instance ID. In the **Status** section of the page that appears, click **Convert to Pay as you go** on the right of **Billing Method**.
3. Confirm the configuration of your RDS instance, read and select Terms of Service, click **Pay Now**, and then complete the payment.

Related operations

Operation	Description
Change the billing method	Changes the billing method of an ApsaraDB RDS instance.

12.Version upgrade

12.1. Upgrade an ApsaraDB RDS for MySQL instance from Basic Edition to High-availability Edition

This topic describes how to upgrade the RDS edition of an ApsaraDB RDS for MySQL instance that runs MySQL 5.7 or MySQL 8.0 from the Basic Edition to the High-availability Edition. The upgrade increases the reliability of your database service.

In the High-availability Edition, your database system consists of a primary RDS instance and a secondary RDS instance. These instances work in a high availability architecture. The High-availability Edition is suitable for more than 80% of business scenarios. For more information, see High-availability Edition.

For more information about the fee that you need to pay for the upgrade, see Specification change fees.

Additional information

If your RDS instance runs MySQL 5.7 on RDS Basic Edition with standard SSDs, you can change the storage type of the instance from standard SSDs to local SSDs when you upgrade the RDS edition of your RDS instance to the High-availability Edition.

Impacts

- The upgrade may trigger a data migration at the underlying layer. This data migration may require a few minutes. After the data migration finishes, a switchover of your workloads is triggered at the switching time that you specify. During the switchover, a network interruption that lasts about 30 seconds occurs. Make sure that your application is configured to automatically reconnect to your RDS instance.

  > **Note** The endpoints of your RDS instance remain unchanged after the upgrade.

- After the upgrade is complete, you cannot downgrade the RDS edition of your RDS instance to the Basic Edition.

Prerequisites

- Your RDS instance runs MySQL 5.7 or MySQL 8.0 on RDS Basic Edition. You can log on to the ApsaraDB RDS console and go to the **Basic Information** page of your RDS instance to view the RDS edition of your RDS instance.
- The minor engine version of your RDS instance must be 20201031 or later. For more information about how to update the minor engine version, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.

  > **Note** If your RDS instance runs MySQL 5.7 on RDS Basic Edition with standard SSDs and you upgrade your RDS instance to run MySQL 5.7 on RDS High-availability Edition with local SSDs, your RDS instance does not need to meet this prerequisite.

Procedure

1.
2. In the **Configuration Information** section of the Basic Information page, click **Change Specifications**.
3. In the dialog box that appears, select **Upgrade** and click **Next step**. This step is required only for subscription RDS instances.
4. Configure the following parameters.

Parameter	Description
Edition	Select **High-availability**.
Storage Type	Optional. Select Local SSD to change the storage type to local SSDs. **Note** This parameter is supported only by RDS instances that run MySQL 5.7 on RDS Basic Edition.
Switching Time	Specify the time at which a switchover of your workloads is triggered. Valid values: ○ Switch Immediately After Data Migration ○ Switch Within Maintenance Window

5. Read and select Terms of Service, click **Pay Now**, and then complete the payment.

Related operations

Operation	Description
Change the specifications of an ApsaraDB RDS instance	Changes the specifications of an ApsaraDB RDS instance.

12.2. Update the minor engine version of an ApsaraDB RDS for MySQL instance

ApsaraDB RDS for MySQL supports both automatic updates and manual updates to the minor engine version of an ApsaraDB RDS for MySQL instance. These updates improve performance, introduce new features, or fix known issues.

For more information about the minor engine versions of ApsaraDB RDS for MySQL, see AliSQL小版本 Release Notes.

For more information about how to update the minor engine version of an RDS instance that runs a different database engine, see the following topics:

- Update the minor engine version of an ApsaraDB RDS for SQL Server instance

- Update the minor engine version of an ApsaraDB RDS for PostgreSQL instance

Overview

- **Auto**: ApsaraDB RDS automatically updates the minor engine version of your RDS instance to the new minor engine version during the maintenance window that you specify. You can log on to the ApsaraDB RDS console and go to the **Basic Information** page to view the value of the **Minor Version Upgrade Mode** parameter. When a new minor engine version is released, Alibaba Cloud pushes notifications at irregular intervals. For more information, see Set the maintenance window of an ApsaraDB RDS for MySQL instance.
- **Manual**: When the lifecycle of the minor engine version that is run on your RDS instance ends, Alibaba Cloud pushes notifications to you at irregular intervals. The notifications remind you to update the minor engine version of your RDS instance to the latest stable minor engine version. In most cases, the lifecycle of a minor engine version spans one year. You can manually update the minor engine version on the **Basic Information** page in the ApsaraDB RDS console. For more information, see Configure the manual update to the minor engine version.

  > **Note** After the lifecycle of a minor engine version ends, the minor engine version is no longer available.

Precautions

- If the minor engine version is outdated, the workloads of your RDS instance may be interrupted. We recommend that you update the minor engine version of your RDS instance on a regular basis or after you receive an O&M notification from Alibaba Cloud.
- If a read-only RDS instance is attached to your RDS instance that uses local SSDs and the minor engine version of the read-only RDS instance is earlier than the new minor engine version, ApsaraDB RDS updates the minor engine version of the read-only RDS instance and then updates the minor engine version of your RDS instance.
- When you upgrade the major engine version or change the specifications of your RDS instance or Alibaba Cloud upgrades the hardware of your RDS database service, ApsaraDB RDS automatically updates the minor engine version of your RDS instance to the latest version. This applies if the minor engine version of your RDS instance is outdated or is no longer available.
- An update to the minor engine version triggers a restart of your RDS instance. During the restart process, a transient connection that lasts approximately 30 seconds may occur. The time when your RDS instance restarts varies based on the value of the **Upgrade Time** parameter. You can set this parameter to **Upgrade Immediately** or **Switch Within Maintenance Window**. We recommend that you update the minor engine version of your RDS instance during off-peak hours. Alternatively, make sure that your application is configured to automatically reconnect to your RDS instance.
- After you update the minor engine version of your RDS instance, you cannot roll the minor engine version of the RDS instance back to the previous version.
- After you update the minor engine version of your RDS instance that runs RDS Basic Edition, the **Backup Size** parameter on the **Basic Information** page in the ApsaraDB RDS console may be displayed as 0. After the next scheduled backup is complete, this error is automatically fixed.

Configure the mode to update the minor engine version

1.
2. In the **Configuration Information** section of the Basic Information page, click **Configure** to the right of **Minor Version Upgrade Mode**.

3. Select **Auto** or **Manual** and click **OK**.

Configure the manual update to the minor engine version

1.
2. In the **Configuration Information** section, click **Upgrade Kernel Version**.

   > **Note** If you cannot find Upgrade Kernel Version, the in-use minor engine version is the latest version.

3. In the dialog box that appears, configure the **Available Upgrade** and **Upgrade Time** parameters and click **OK**.

FAQ

- After I update the minor engine version of my RDS instance, why does the SELECT @@version statement still return the minor engine version that I used before the update?

  The SELECT @@version statement returns the engine version of MySQL. To view the engine version of AliSQL, you must run the `show variables like '%rds_release_date%'` command.

- Do I update the minor engine version of my RDS instance only to the next minor engine version?

  You can update the minor engine version of your RDS instance to any new minor engine version that is available in the ApsaraDB RDS console.

- Why does a transient connection occur during the update process? Does the update cause other serious risks?

  Before ApsaraDB RDS updates the minor engine version of your RDS instance, ApsaraDB RDS updates the minor engine version of the secondary RDS instance of your RDS instance. After the minor engine version of the secondary RDS instance is updated, ApsaraDB RDS switches your workloads over to the secondary RDS instance. During the switchover, a transient connection that lasts approximately 30 seconds occurs. The update does not cause other serious risks.

Related operations

Operation	Description
Update minor engine version	Updates the minor engine version of an ApsaraDB RDS instance.

12.3. Upgrade the major engine version of an ApsaraDB RDS for MySQL instance

ApsaraDB RDS allows you to upgrade the major engine version of an ApsaraDB RDS for MySQL instance in the ApsaraDB RDS console. You can also migrate the data of an ApsaraDB RDS for MySQL instance to a new RDS instance that uses the required major engine version to upgrade the major engine version. For example, you can migrate the data of an RDS instance for which the Transparent Data Encryption (TDE) feature is enabled.

Upgrade methods

- Upgrade the major engine version of an RDS instance in the ApsaraDB RDS console
- Upgrade the major engine version of an RDS instance in the Data Transmission Service (DTS) console

> **Note** For more information about how to upgrade the major engine version of an RDS instance that runs a different database engine, see the following topics:
>
> - Upgrade the major engine version of an ApsaraDB RDS for PostgreSQL instance
> - Upgrade the RDS edition of an ApsaraDB RDS for SQL Server instance from Basic Edition to High-availability Edition
> - Upgrade the major engine version of an ApsaraDB RDS for SQL Server instance with local SSDs from SQL Server 2008 R2 to SQL Server 2012 or 2016
> - Upgrade the major engine version of an ApsaraDB RDS for SQL Server instance from SQL Server 2012 to SQL Server 2016

Upgrade the major engine version of an RDS instance in the ApsaraDB RDS console

Limits

Category	Description
Instances	You can upgrade the major engine version of an RDS instance in the ApsaraDB RDS console only if the instance runs RDS High-availability Edition and uses local SSDs.
	If read-only RDS instances are attached to a primary RDS instance and the instance types of the read-only RDS instances are different, you cannot upgrade the major engine version of the RDS instance. We recommend that you release the read-only RDS instances, upgrade the major engine version of the primary RDS instance, and then re-create the read-only RDS instances. For more information, see Release or unsubscribe from an ApsaraDB RDS for MySQL instance and Create a read-only ApsaraDB RDS for MySQL instance.
	Before you upgrade the major engine version of an RDS instance, make sure that the RDS instance is in the **Running** state. If the RDS instance is in a different state, such as **Restarting** or **Creating Network Connection**, you must wait until the running task is complete before you can upgrade the major engine version.
	If an RDS instance runs RDS High-availability Edition, a major engine version upgrade is supported only when both the primary RDS instance and the secondary RDS instance run as expected and no replication latency exists between the primary RDS instance and the secondary RDS instance. You can check the Replication Thread Status of Secondary Instances and Replication Latency of Secondary Instances metrics on the **Monitoring and Alerts** page in the ApsaraDB RDS console.
	You can upgrade the major engine version of an RDS instance only to the next major engine version. For example, if you want to upgrade MySQL 5.6 to MySQL 8.0, you must first upgrade MySQL 5.6 to MySQL 5.7 and then to MySQL 8.0.

Category	Description
Upgrades	You cannot downgrade the major engine version of an RDS instance after the major engine version is upgraded. **Note** After you upgrade the major engine version of an RDS instance, you cannot use the data backup files that are generated before the upgrade to restore the data of the RDS instance. You can restore the data of the RDS instance only by using the data backup files that are generated after the upgrade.
Encryption	If the SSL encryption feature is enabled for an RDS instance, you must disable the feature before you can upgrade the major engine version of the RDS instance. For more information, see Disable SSL encryption.
	If the Transparent Data Encryption (TDE) feature is enabled for an RDS instance, you cannot upgrade the major engine version of the RDS instance in the ApsaraDB RDS console. For more information, see Upgrade the major engine version of an RDS instance in the Data Transmission Service (DTS) console.
Database access	If an event is created in a database on an RDS instance that runs MySQL 5.7, you cannot upgrade the major engine version of the RDS instance from MySQL 5.7 to MySQL 8.0. You must delete the event, upgrade the major engine version, and then re-create the event.
	If a stored procedure, trigger, view, or function in a database involves features that are not supported by MySQL 8.0, the major engine version upgrade fails. For more information, see Changes in MySQL 8.0.
	If more than 200,000 tables are created in databases on an RDS instance, the major engine version upgrade is not supported. You must delete the tables that you no longer require before a major engine version upgrade.
	If the storage engine of an RDS instance is MyISAM, MEMORY, TokuDB, Sphinx, or RocksDB, you must change the storage engine of the RDS instance to InnoDB before a major engine version upgrade.
Instance types	If an RDS instance uses a phased-out instance type, you must upgrade the instance type of the RDS instance before you upgrade the major engine version of the RDS instance. For more information, see Primary ApsaraDB RDS for MySQL instance types and Change the specifications of an ApsaraDB RDS for MySQL instance.

Notice ApsaraDB RDS first upgrades the secondary RDS instance, performs a primary/secondary switchover, and then upgrades the original primary RDS instance. During the upgrade, your database system may become unavailable for up to 5 minutes. We recommend that you perform a major engine version upgrade during off-peak hours.

Before you begin

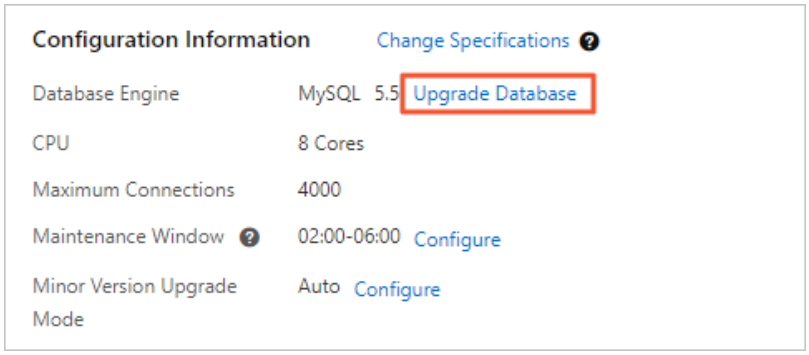
- Make sure that you understand the differences between the current major engine version and the major engine version to which you want to upgrade. We recommended that you create an RDS instance that runs MySQL 8.0 and test the syntax. This way, you can ensure that the features and

syntax that are used by your application are supported in the new major engine version. For more information about the differences between major engine versions, see the following topics:

- MySQL Release Notes
- Appendix 3: Differences between MySQL 5.7 and MySQL 8.0
- Appendix 4: Differences between MySQL 5.6 and MySQL 5.7

- Before you upgrade the major engine version of your RDS instance, we recommend that you understand the benefits of the major engine version to which you want to upgrade. For more information, see Appendix 1: Advantages of MySQL 8.0 over MySQL 5.7 or Appendix 2: Advantages of MySQL 5.7 over MySQL 5.6.
- We recommend that you clone the original RDS instance before a major engine upgrade and use the cloned RDS instance to test whether the new major engine version is compatible with your workloads. Make sure that the cloned RDS instance runs as expected before you upgrade the major engine version of the original RDS instance.
- Before a major engine version upgrade, check whether a full data backup file is generated over the last seven days. If no full data backup files are generated over the last seven days, perform a full data backup.
- When you upgrade the major engine version of an RDS instance, transient connections may occur. We recommend that you upgrade the major engine version during off-peak hours or make sure that your application is configured to automatically reconnect to the RDS instance.
- Make sure that the amount of available storage is sufficient before a major engine version upgrade.
- We recommend that you modify the retention policies for binary log files. You can increase the retention period and maximum storage usage of binary log files. For more information, see Retain the backup files of an ApsaraDB RDS for MySQL instance for a long period of time.

1.
2. In the **Configuration Information** section of the Basic Information page, click **Upgrade Database**.



> **Note** If Upgrade Database is not displayed, you must check whether the major engine version of the RDS instance meets the requirements.

3. In the dialog box that appears, select **Migrate Immediately** or **Switch Within Maintenance Window** and click **OK**.
   - **Migrate Immediately**: The upgrade is immediately started.
   - **Switch Within Maintenance Window**: The upgrade is started within the specified maintenance window. For more information, see Set the maintenance window of an ApsaraDB RDS for MySQL instance. You can also click **Change** on the right to change the maintenance

window.

Upgrade the major engine version of an RDS instance in the Data Transmission Service (DTS) console

If the major engine version of an RDS instance cannot be upgraded in the ApsaraDB RDS console, you can perform the following operations to upgrade the major engine version:

1. Create an RDS instance that runs the new major engine version. For more information, see Create an ApsaraDB RDS for MySQL instance.
2. Migrate the data of the original RDS instance to the new RDS instance. For more information, see Migrate data between ApsaraDB RDS for MySQL instances.
3. Release the original RDS instance. For more information, see Release or unsubscribe from an ApsaraDB RDS for MySQL instance.

For example, if you want to upgrade the major engine version of an RDS instance that runs MySQL 5.7 and has TDE enabled to MySQL 8.0, you must first create an RDS instance that runs MySQL 8.0 and migrate the data of the original RDS instance to the new RDS instance. After you verify that your workloads run as expected on the new RDS instance, you can release the original RDS instance.

> **Notice** You must verify that the new major engine version is compatible with your workloads after you upgrade your RDS instance by migrating data. You must also monitor the new RDS instance for a period of time. After you confirm that your workloads run as expected on the new RDS instance, you can release your original RDS instance.

Appendix 1: Advantages of MySQL 8.0 over MySQL 5.7

- The security of your database system is enhanced. You can manage accounts in a more flexible manner.
- You can create and manage resource groups.
- The features of the InnoDB storage engine are enhanced.
- New character sets, data types, and syntax are supported. Backup locks and optimizer_switch flags are introduced.
- JSON and XML expressions are enhanced.
- Optimizers are enhanced.
- Replication performance is enhanced.
- Multi-value indexes can be created. Derived condition pushdown is optimized.
- MySQL grant tables can be read.
- Resource allocation control is supported.

Appendix 2: Advantages of MySQL 5.7 over MySQL 5.6

- New features such as password management, account locking, and connection encryption are introduced. These new features help improve the security of your database system.
- Online DDL operations are supported. For example, you can use the RENAME INDEX clause to rename an index.
- The scalability of the InnoDB storage engine and the performance of temporary InnoDB tables are optimized to accelerate data loading.
- JSON expressions are supported.

- Index Condition Pushdown (ICP) is supported for partitioned tables, and spatial indexes are supported for InnoDB tables.
- Most of the used parsers, optimizers, and cost models are optimized to improve the maintainability, scalability, and performance of your database system.
- New character sets are supported. The character sets include the gb18030 character set that is defined in the Chinese National Standard GB 18030-2005: Information technology - Chinese coded character set.
- The ngram full-text parser plug-in is provided. The plug-in supports Chinese, Japanese, and Korean (CJK).
- Dump threads are optimized to reduce lock contention and increase throughput.
- The replication latency is significantly reduced.
- The sys system database is added to support multiple metrics. These metrics help reduce storage usage and simplify database management.

Appendix 3: Differences between MySQL 5.7 and MySQL 8.0

> **Note** The following table provides only the major differences between MySQL 5.7 and MySQL 8.0. For more information, see MySQL Release Notes.

Feature	MySQL 5.7	MySQL 8.0
GRANT...IDENTIFIED BY PASSWORD	Supported	Not supported
PASSWORD() function	Supported	Not supported
FLUSH QUERY CACHE and **RESET QUERY CACHE**	Supported	Not supported
Parameters for the **SQL_MODE** system variable: DB2, MAXDB, MSSQL, MYSQL323, MYSQL40, ORACLE, POSTGRESQL, NO_FIELD_OPTIONS, NO_KEY_OPTIONS, and NO_TABLE_OPTIONS	Supported	Not supported
GROUP BY for automatic sorting	Supported	Not supported
Syntax that contains the **EXTENDED** or **PARTITIONS** keyword	Supported	Not supported
Encryption functions such as ENCODE(), DECODE(), and ENCRYPT()	Supported	Not supported
Functions related to spatial analysis For more information, see Open source MySQL documentation.	Supported	Not supported
Functions that previously accepted either well-known binary (WKB) strings or geometry arguments but no longer accept geometry arguments	Supported	Not supported

Feature	MySQL 5.7	MySQL 8.0
Resolution of **\N** to **NULL**	Supported	Not supported
PROCEDURE ANALYSE() function	Supported	Not supported
Creation of partitioned tables by using the NDB storage engine	Supported	Not supported
Compression of temporary tables by using the InnoDB storage engine	Supported	Not supported
JSON_APPEND() function	Supported	Not supported
Placing of table partitions in shared tablespaces	Supported	Not supported
ALTER TABLE...UPGRADE PARTITIONING	Supported	Not supported

Appendix 4: Differences between MySQL 5.6 and MySQL 5.7

Note The following table provides only the major differences between MySQL 5.6 and MySQL 5.7. For more information, see MySQL Release Notes.

Feature	MySQL 5.6	MySQL 5.7
CREATE...AS SELECT in global transaction identifier (GTID) mode	Supported	Not supported
Usage of temporary tables when the GTIDs feature is enabled	Supported	Not supported
Configuration of partition keys in partitioned tables	Supported	Not supported
ENGINE_NO_CACHE	Supported	Not supported
Invisible indexes	Supported	Not supported
UPDATE non_affected_rows INSERT	Supported	Not supported
Proxy-related commands	Supported in SET command mode	Supported in Call Procedure mode
TokuDB, Sphinx, RocksDB, and MEMORY storage engines	Supported	Not supported
str_ord() function	Supported	Not supported
raiseerror() function	Supported	Not supported
OPTIMIZE TABLE table ASYNC	Supported	Not supported

Feature	MySQL 5.6	MySQL 5.7
ENGINE_NO_CACHE	Supported	Not supported
INFORMATION.TABLE_UTILIZATION table	Supported	Not supported
The requesting_thd_id and blocking_thd_id columns of the INFORMATION_SCHEMA.INNODB_LOCK_WAITS table	Supported	Not supported
INFORMATION_SCHEMA.INNODB_RSEG table	Supported	Not supported
INFORMATION_SCHEMA.INNODB_IO_STATUS table	Supported	Not supported
Column compression	Supported	Not supported
Query Plan Cache	Supported	Not supported
Limit and Union syntax	Parentheses () not required	Parentheses () required
SHOW FULL PROCESSLIST	The memory and query_memory columns are removed from the results that are returned by the SHOW FULL PROCESSLIST statement in MySQL 5.7.	
max_statement_time and max_execution_time	The max_statement_time parameter is removed from MySQL 5.7. The max_execution_time parameter is retained in MySQL 5.7.	
RDS_SQL_MAX_AFFECTED	The number of rows on which the **UPDATE** or **DELETE** statement is executed can no longer be specified by the **RDS_SQL_MAX_AFFECTED** variable in MySQL 5.7. The number of rows is specified by the rds_sql_max_affected_rows variable.	

Feature	MySQL 5.6	MySQL 5.7
Performance optimization and adjustment for concurrency control	The following parameters can no longer be used for concurrency control in MySQL 5.7: • innodb_adaptive_tickets_algo • innodb_min_concurrency_tickets • rds_threads_running_ctl_mode • rds_threads_running_high_watermark • rds_filter_key_cmp_in_order • rds_reset_all_filter • rds_sql_delete_filter • rds_sql_select_filter • rds_sql_update_filter • rds_strict_concurrency • rds_thread_extra_concurrency • rds_strict_trx_idle_timeout • rds_sql_buf_read_bandwidth • rds_sql_buf_read_threshold_bytes • rds_sql_buf_write_bandwidth • rds_sql_buf_write_threshold_bytes • rds_sql_max_iops	
Variables used to specify the number of connections	The following variables are deleted from MySQL 5.7: • extra_max_connections • rds_root_connections • rds_sysinfo_connections • rds_sysinfo_user_list	

Feature	MySQL 5.6	MySQL 5.7

Replication-related adjustments	• Compatibility adjustments in MySQL 5.7:
	  ○ Replication between GTIDs-enabled databases and GTIDs-disabled databases is no longer supported.
	  ○ The sql_slave_skip_counter parameter is not supported when the GTIDs feature is enabled.
	  ○ The **CREATE...SELECT** statement is no longer supported.
	• Adjustments to secondary RDS instances that run MySQL 5.7:
	  ○ The **SHOW SLAVE LAG** statement is no longer supported.
	  ○ The **SHOW SLAVE STATUS** statement no longer supports timeouts.
	  ○ The amount of information that is returned by the **SHOW SLAVE STATUS** statement is reduced.
	  ○ The sql_thread thread on a secondary RDS instance no longer supports timeouts.
	  ○ The sql_thread thread on a secondary RDS instance can no longer skip specific statements.
	• Adjustments to binary logs in MySQL 5.7:
	  ○ The transmission speed can no longer be adjusted.
	  ○ The rds_rpl_receive_buffer_difftime parameter is no longer supported.
	  ○ The rds_rpl_receive_buffer_size parameter is no longer supported.	

Log-related adjustments	The following adjustments are made to error logs in MySQL 5.7: • The IP address, user, I/O latency, and network latency are no longer recorded for the **SHUTDOWN** command. • Duplicate keys are no longer supported for table names.	

FAQ

Why does a transient connection occur when I upgrade the major engine version of an RDS instance? Does the upgrade cause other serious risks?

To ensure service stability, ApsaraDB RDS upgrades the major engine version of the secondary RDS instances. Then, ApsaraDB RDS switches your workloads to the secondary RDS instance before it upgrades the major engine version of the primary RDS instance. During the switchover, a transient connection occurs. The major engine upgrade does not cause other serious risks.

Does ApsaraDB RDS upgrade the major engine version of the primary RDS instance and the secondary RDS instance at the same time?

No, ApsaraDB RDS first upgrades the major engine version of the secondary RDS instance, switches your workloads to the secondary RDS instance, and then upgrades the major engine versions of the primary RDS instance.

Related operations

Operation	Description
Upgrade the major engine version of an ApsaraDB RDS for MySQL instance	Upgrades the major engine version of an ApsaraDB RDS instance.

13.Instance parameters

13.1. View the parameters of an ApsaraDB RDS for MySQL instance

This topic describes how to view the parameters of an ApsaraDB RDS for MySQL instance in the ApsaraDB RDS console or by using an SQL statement.

Method 1: View the parameters of an RDS instance in the ApsaraDB RDS console

1.
2. In the left-side navigation pane, click **Parameters**. On the Editable Parameters tab of the page that appears, view the value of each parameter.

> **Note** The ApsaraDB RDS console displays only the parameters that you can configure. For more information about how to view all parameters, see the "Method 2: View the parameters of an RDS instance by using an SQL statement" section of this topic.

Method 2: View the parameters of an RDS instance by using an SQL statement

1. Connect to the RDS instance. For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.
2. Execute the following statement to view all the parameters of the RDS instance.

```
SHOW VARIABLES;
```

You can also execute the `SHOW VARIABLES LIKE '<Parameter name>';` statement to view only a specific parameter.

> **Note** The `<Parameter name>` variable supports wildcards (%). You can insert a wildcard (%) into any part of the `<Parameter name>` variable. A wildcard (%) can represent any number of characters. Examples:
>
> - The `SHOW VARIABLES LIKE 'thread_cache%';` statement queries all parameters whose names start with `thread_cache`.
> - The `SHOW VARIABLES LIKE '%cache_size';` statement queries all parameters whose names end with `cache_size`.
> - The `SHOW VARIABLES LIKE 'thread%size';` statement queries all parameters whose names start with `thread` and end with `size`.
> - The `SHOW VARIABLES LIKE '%';` statement is equivalent to the `SHOW VARIABLES;` statement.

Related operations

Operation	Description
Query parameter configurations	Queries the parameters of an ApsaraDB RDS instance.
Modify parameters of an ApsaraDB for RDS instance	Reconfigures the parameters of an ApsaraDB RDS instance.
Query the parameter template of an ApsaraDB for RDS instance	Queries the parameter templates that are available for an ApsaraDB RDS instance.

Related information

- Modify the parameters of an ApsaraDB RDS for MySQL instance
- Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances
- Optimize parameters of an ApsaraDB RDS for MySQL instance

13.2. Modify the parameters of an ApsaraDB RDS for MySQL instance

This topic describes how to modify the parameters of an ApsaraDB RDS for MySQL instance in the ApsaraDB RDS console or by using the ApsaraDB RDS API. This topic also describes how to view the parameter modification history in the ApsaraDB RDS console.

For more information about how to view the parameter settings of an RDS instance, see View the parameters of an ApsaraDB RDS for MySQL instance.

Precautions

- To ensure the stability of your RDS instance, you can modify only the parameters that are displayed in the ApsaraDB RDS console.
- When you modify the parameters of your RDS instance, you can view the value range of each parameter in the **Value Range** column on the **Editable Parameters** tab of the Parameters page in the ApsaraDB RDS console.
- The new values of some parameters take effect only after you restart your RDS instance. For more information, view the **Force Restart** column on the **Editable Parameters** tab of the Parameters page in the ApsaraDB RDS console. We recommend that you modify parameters during off-peak hours and make sure that your application is configured to automatically reconnect to your RDS instance.
- If read-only RDS instances are attached to a primary RDS instance, the modifications to some parameters of the primary RDS instance are automatically synchronized to the read-only RDS instances. For more information, see Appendix: Parameters to be synchronized to read-only RDS instances.

Modify parameters

> **Note** If you want to modify multiple parameters at a time, we recommend that you use a parameter template. For more information, see Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances.

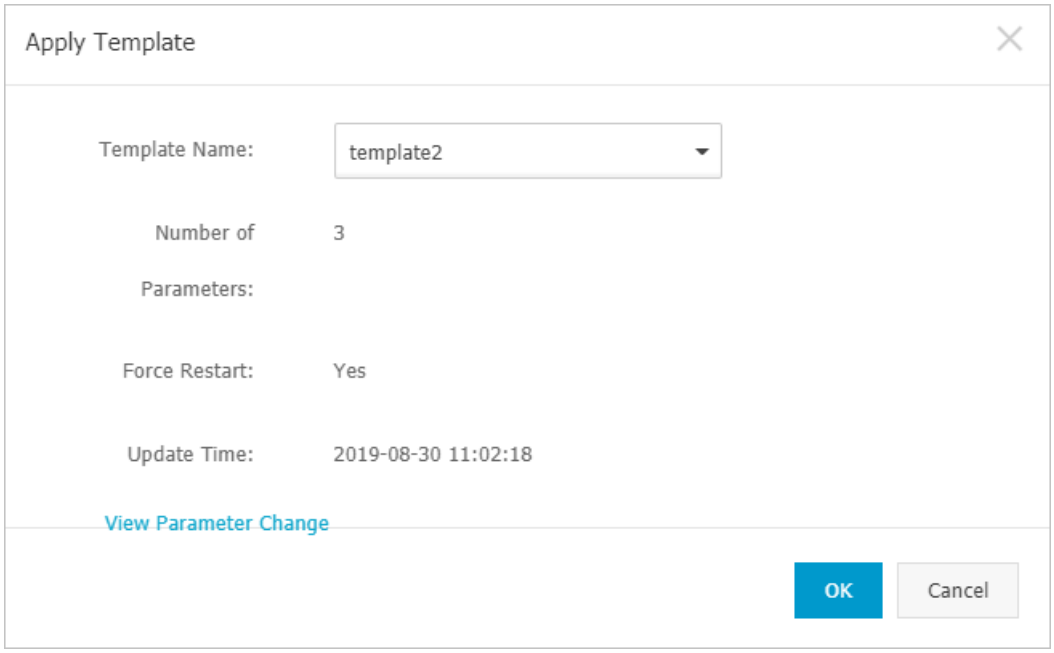
1.
2. In the left-side navigation pane, click **Parameters**.
3. Perform the following operations based on your business requirements:
   - Apply a parameter template to the RDS instance.

     > **Note** If the parameter template takes effect only after you restart the RDS instance, we recommend that you apply the parameter template during off-peak hours and make sure that your application is configured to automatically reconnect to the RDS instance.

     a. On the Editable Parameters tab, click **Apply Template**.

     b. In the Apply Template dialog box, select the parameter template that you want to use and click **OK**.

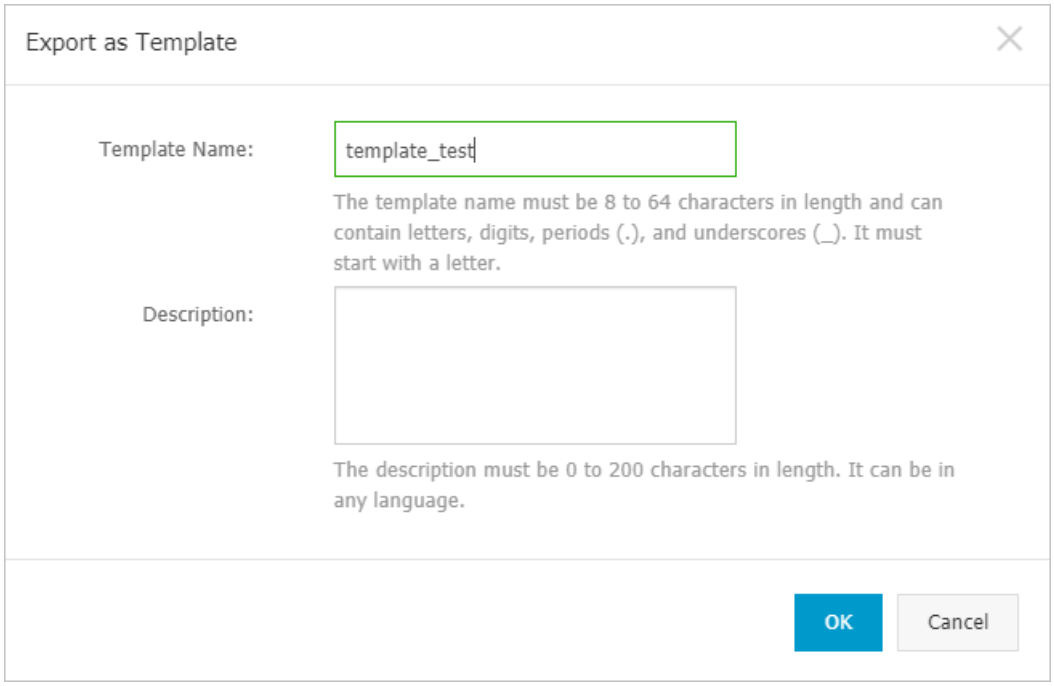
     > **Note**
     > - You can view the number of parameters in the parameter template and check whether you must restart the RDS instance for the new values of the parameters to take effect. You can also click **View Parameter Change** to view the changes in the values of the parameters in the parameter template.
     > - If you cannot find the parameter template that you want to use, you must check whether the parameter template resides in the same region as the RDS instance. If the parameter template and the RDS instance reside in different regions, you can replicate the parameter template to the region where the RDS instance resides. For more information, see Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances.

   - Export the parameter settings of the RDS instance as a parameter template in the region that you select.

     a. On the Editable Parameters tab, click **Export as Template**.

b. Configure the following parameters.



Parameter	Description
Template Name	The name of the parameter template. The name must be 8 to 64 characters in length and can contain letters, digits, periods (.), and underscores (_). The name must start with a letter.
Description	The description of the parameter template. The description can be up to 200 characters in length.

c. Click **OK**.

- Export the parameter settings of the RDS instance to your computer.

  On the Editable Parameters tab, click **Export Parameters**. The parameter settings of the RDS instance are exported to your computer as a TXT file.

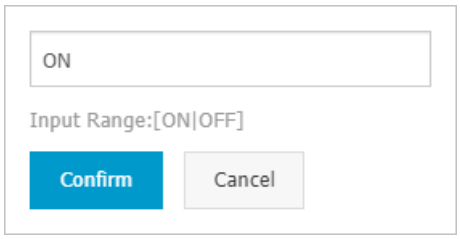
  a. Modify the parameters in the TXT file. Click **Import Parameters**. In the Import Parameters dialog box, copy the parameter settings from the TXT file and paste them to the dialog box.

  b. Click **OK**.

c. Click **Apply Changes**. In the dialog box that appears, select the time range at which you want the new parameter settings to take effect. You can select **Take Effect Immediately**, **Take Effect Within Maintenance Window**, or **Take Effect Within Specified Time Range**.

> **Note**
>
> - If the new parameter settings take effect only after you restart the RDS instance, ApsaraDB RDS prompts you to restart the RDS instance. We recommend that you restart the RDS instance during off-peak hours and make sure that your application is configured to automatically reconnect to the RDS instance.
> - Before the new parameter values take effect, you can click **Cancel Changes** to revoke the changes.
> - If ApsaraDB RDS prompts you that `the operation failed because a new parameter value is in an invalid format`, you must check whether the new parameter value is within the **Value Range** of the parameter.

- Modify a single parameter of the RDS instance.
  a. On the **Editable Parameters** tab, find the parameter that you want to modify and click the icon in the **Running Parameter Value** column.
  b. Enter a new value based on the value range that is displayed.



  c. Click **OK**.
  d. Click **Apply Changes**. In the dialog box that appears, select the time range at which you want the new parameter settings to take effect. You can select **Take Effect Immediately**, **Take Effect Within Maintenance Window**, or **Take Effect Within Specified Time Range**.

> **Note**
>
> - If the new parameter settings take effect only after you restart the RDS instance, ApsaraDB RDS prompts you to restart the RDS instance. We recommend that you restart the RDS instance during off-peak hours and make sure that your application is configured to automatically reconnect to the RDS instance.
> - Before the new parameter value takes effect, you can click **Cancel Changes** to revoke the change.

View the parameter modification history

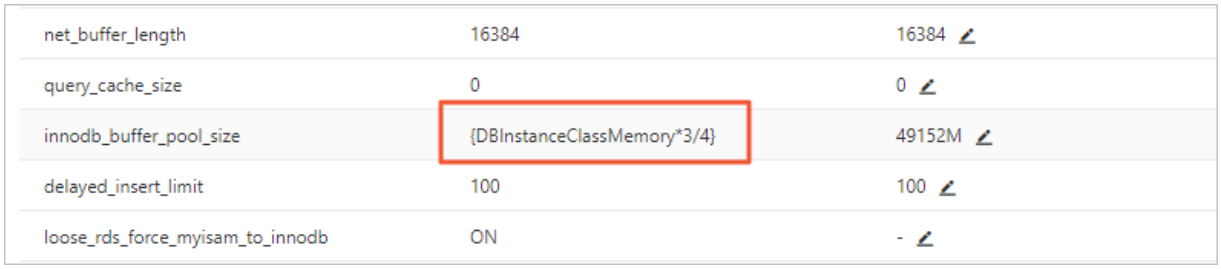
1.
2. In the left-side navigation pane, click **Parameters**.

3. On the page that appears, click the **Edit History** tab.
4. Select a time range and click **OK**.

Configure parameters by using expressions

You can set a parameter to an expression for an RDS instance. If you set an instance type-related parameter to an expression, the value of the parameter dynamically changes when the instance type changes. This ensures the stability of the RDS instance.

In the example shown in the following figure, the innodb_buffer_pool_size parameter is set to `{DBInstanceClassMemory*3/4}`. If the value of the DBInstanceClassMemory variable changes, the value of the innodb_buffer_pool_size parameter dynamically changes, and you do not need to manually change the value.



The following table describes the supported expression syntax.

Category	Description	Example

Category	Description	Example
Variables	• AllocatedStorage: specifies the storage capacity of an RDS instance. The value of this variable is an integer. • DBInstanceClassMemory: specifies the available memory capacity of an RDS instance. The value of this variable is calculated by deducting the memory that is occupied by the control processes on the instance from the memory capacity that is supported by the instance type. The value of this variable is an integer. For example, if the memory capacity that is supported by the instance type is 16 GB and the memory that is occupied by the control processes is 4 GB, the value of the DBInstanceClassMemory variable is 12 GB. • DBInstanceClassCPU: specifies the number of cores that are supported by the instance type. The value of this variable is an integer. • DBInstanceClassConnections: specifies the maximum number of connections that are supported by the instance type. The value of this variable is an integer. **Note** • For more information about the instance types and the storage capacity, memory capacity, number of cores, and maximum number of connections that are supported by each instance type, see Primary ApsaraDB RDS instance types. • For more information about how to configure the innodb_buffer_pool_size parameter of an RDS instance, see Background information. • The memory that is occupied by control processes is the smallest value between the values that are calculated by using the following formulas: **Memory capacity supported by the instance type × 0.65** and **(Memory capacity supported by the instance type/16384 + 1) × 2048**.	{DBInstanceClassMemory*3/4}

Category	Description	Example
Operators	• Expression syntax: An expression is enclosed by a pair of braces ({}). • Division operator (/): This operator is used to divide a number by another number. If the quotient is a decimal, only the integer part of the quotient is returned. The dividend and the divisor must be integers. For example, ApsaraDB RDS can process {DBInstanceClassMemory*3/4} but not {DBInstanceClassMemory*0.75}. • Multiplication operator (*): This operator is used to multiply a number by another number. If the product is a decimal, only the integer part of the product is returned. The two numbers must be integers.	
Functions	• The GREATEST() function returns the largest value among an array of integers or the largest value that is calculated by an array of expressions. • The LEAST() function returns the smallest value among an array of integers or the smallest value that is calculated by an array of expressions. • The SUM() function adds a specified integer or the value that is calculated by a specified expression.	LEAST({DBInstanceClassMemory/256},10485760)

Modify the parameters of multiple RDS instances at a time

1.
2. Select the RDS instances for which you want to modify parameters and click **Modify Parameters** below the instance list.

   > **Note** The RDS instances must run the same version and edition. Otherwise, the **Modify Parameters** button is dimmed.

3. In the **Parameter Settings** dialog box, click the icon on the right of the parameter that you want to modify.
4. Select or enter a new value in the **Current Value** column and click **OK**.
5. In the **Parameter Settings** dialog box, check the new value and click **OK**.

FAQ

- After I modify the parameters of my RDS instance, do the new values of the parameters immediately take effect? Do I need to restart my RDS instance?

  After you modify the parameters of your RDS instance, the new values of some parameters take effect in approximately 5 minutes even if you do not restart your RDS instance. However, the new values of some parameters take effect only after you restart your RDS instance. For more information, view the **Force Restart** column on the **Editable Parameters** tab of the Parameters page in the ApsaraDB RDS console.

- After I modify the parameters of my RDS instance, why do the new values of the parameters not take effect?

  After you modify the parameters of your RDS instance, you must click **Apply Changes** on the Editable Parameters tab on the Parameters page in the ApsaraDB RDS console to make the new values of the parameters take effect.

Appendix: Parameters to be synchronized to read-only RDS instances

If read-only RDS instances are attached to a primary RDS instance, the modifications to some parameters of the primary RDS instance are automatically synchronized to the read-only RDS instances. The following list provides these parameters.

- lower_case_table_names

  > **Note** If your RDS instance runs MySQL 8.0, you cannot change the value of this parameter.

- innodb_large_prefix

Related operations

Operation	Description
Modify parameters of an ApsaraDB for RDS instance	Modifies the parameters of an ApsaraDB RDS instance.
Query the parameter template of an ApsaraDB for RDS instance	Queries the parameter templates that are available for an ApsaraDB RDS instance.
Query parameter configurations	Queries the parameter settings of an ApsaraDB RDS instance.

Related information

- View the parameters of an ApsaraDB RDS for MySQL instance
- Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances
- Optimize parameters of an ApsaraDB RDS for MySQL instance

13.3. Change the size of the InnoDB buffer pool for an ApsaraDB RDS for MySQL instance

This topic describes how to configure the innodb_buffer_pool_size parameter of an ApsaraDB RDS for MySQL instance based on your business requirements to improve the performance of the instance.

Prerequisites

The RDS instance runs one of the following RDS editions:

- High-availability Edition

- Basic Edition

Background information

You can reconfigure the innodb_buffer_pool_size parameter to change the size of the InnoDB buffer pool for an RDS instance. The value of this parameter is calculated by using the following formula:

```
{DBInstanceClassMemory*X/Y}
```

Example:

```
{DBInstanceClassMemory*7/10}
```

> **Note**
> - DBInstanceClassMemory is a system variable, which specifies the memory capacity of the RDS instance.
> - X is the numerator, and Y is the denominator.
> - The size of the InnoDB buffer pool must be within the following range: [128 MB, DBInstanceClassMemory × 8/10]. The minimum size is 128 MB. The maximum size is 80% of the memory capacity that you purchased for the RDS instance.

The default size of the InnoDB buffer pool for an RDS instance is calculated based on the following rules:

- If the RDS instance is a dedicated instance that uses standard SSDs or enhanced SSDs (ESSDs), the default size of the InnoDB buffer pool is calculated by using the following formula: `Default size of the InnoDB buffer pool = (Purchased memory capacity of the RDS instance - Reserved memory capacity of the RDS instance) × 0.75`.

  > **Note** The reserved memory capacity of the RDS instance is calculated by using the following formula:
  >
  > ```
  > MIN{Purchased memory capacity of the RDS instance × 0.65, [(Purchased memory capacity of the RDS instance/16384) + 1] × 2048}
  > ```

- If the RDS instance is a general-purpose instance that uses standard SSDs or ESSDs or the RDS instance uses local SSDs, the default size of the InnoDB buffer pool is calculated by using the following formula: `Default size of the InnoDB buffer pool = Purchased memory capacity of the RDS instance × 0.75`.

> **Note** The default size of the InnoDB buffer pool is an integer multiple of 128. If the calculated result is not an integer multiple of 128, the result is rounded to the nearest integer that is a multiple of 128. For example, an RDS instance provides 1,024 MB of memory, the calculated result is 268, and the nearest integer that is a multiple of 128 is 256. In this case, the default size of the InnoDB buffer pool for the RDS instance is 256 MB.

The following table provides the default size and maximum size of the InnoDB buffer pool for various memory capacities.

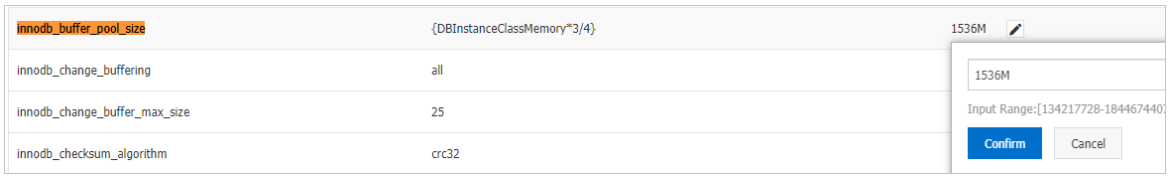
Memory capacity (Unit: MB)	Default buffer pool size (Unit: MB)	Maximum buffer pool size (Unit: MB)
1,024	256	256
2,048	512	512
4,096	1,536	1,536
8,192	4,608	4,608
16,384	12,288	12,288
24,576	18,432	19,456
32,768	24,576	25,600
49,152	36,864	38,912
65,536	49,152	52,224
98,304	73,728	77,824
131,072	98,304	104,448
196,608	147,456	156,672
229,376	172,032	183,296
262,144	196,608	208,896
393,216	294,912	314,368
491,520	368,640	393,216
786,432	589,824	628,736

The size of the InnoDB buffer pool must be an integer multiple of the result that is obtained by using the following formula: `Value of the innodb_buffer_pool_chunk_size parameter × Value of the innodb_buffer_pool_instances parameter`. If the size of the InnoDB buffer pool is not an integer multiple of the result that you obtain by using the formula, ApsaraDB RDS changes the size to an integer multiple of the result. For example, if the result that you obtain by using the formula is 1 GB and you set the `innodb_buffer_pool_size` parameter to 1.5 GB, ApsaraDB changes the value of the innodb_buffer_pool_size parameter to 2 GB.

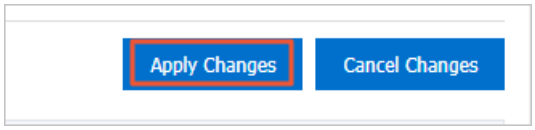
Change the size of the InnoDB buffer pool for a single RDS instance

1.
2. In the left-side navigation pane, click **Parameters**.
3. Find the **innodb_buffer_pool_size** parameter and click the icon. In the dialog box that appears, enter a new value and click **OK**.

> **Warning** After you change the value of the **innodb_buffer_pool_size** parameter for an RDS instance, the instance restarts. Proceed with caution.

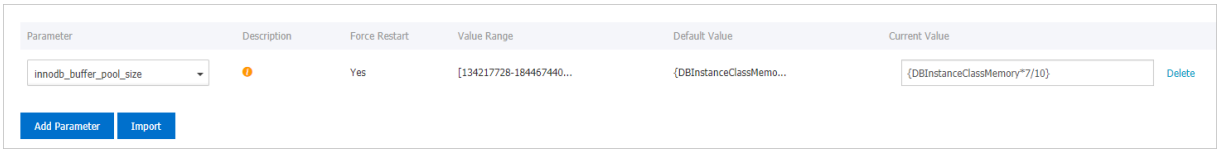


4. Click **Apply Changes** above the parameter list. In the message that appears, click **OK**. Then, wait for the RDS instance to restart.



Change the size of the InnoDB buffer pool for multiple RDS instances at a time

You can create a parameter template that contains the **innodb_buffer_pool_size** parameter. Then, you can apply the template to multiple RDS instances to change the size of the InnoDB buffer pool for these RDS instances at a time. If a parameter template that contains this parameter has been created, you need to only change the value of this parameter in the template. Then, you can apply the template to multiple RDS instances at a time. For more information, see Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances.



13.4. Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances

This topic describes how to use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances. ApsaraDB RDS for MySQL provides system parameter templates and custom parameter templates.

Prerequisites

The RDS instance runs one of the following MySQL versions:

- MySQL 8.0
- MySQL 5.7
- MySQL 5.6

Context

To ensure service availability, you can configure only some parameters in the ApsaraDB RDS console.

ApsaraDB RDS for MySQL provides various system parameter templates. You can also create custom parameter templates to meet specific business requirements such as the requirements for high performance.

> **Note** For more information about how to configure a single parameter, see Modify the parameters of an ApsaraDB RDS for MySQL instance.

Introduction to system templates

> **Note** ApsaraDB RDS for MySQL provides system templates for RDS instances that run RDS High-availability Edition or RDS Basic Edition. You can create custom parameter templates for RDS instances that run RDS Enterprise Edition. For more information, see Create a custom parameter template.

ApsaraDB RDS for MySQL provides the following system parameter templates for RDS instances that run RDS High-availability Edition or RDS Basic Edition:

- Default parameter template

  This parameter template provides the highest data security but requires a longer period of time to take effect. If you use this parameter template, data is replicated in semi-synchronous mode. The following parameter settings are fixed in this parameter template to ensure data security:

  - InnoDB
    - innodb_flush_log_at_trx_commit = 1
    - sync_binlog = 1
  - X-Engine (Only the default parameter template is provided.)

    sync_binlog = 1

- Asynchronous parameter template (phased-out)

  > **Note** For more information about how to change the data replication mode to asynchronous, see Change the data replication mode of an ApsaraDB RDS for MySQL instance.

- High-performance parameter template

  This parameter template provides average data security but can take effect within the shortest period of time. If you use this parameter template, data is replicated in asynchronous mode. The following parameter settings are fixed in this parameter template to ensure data security:

  - innodb_flush_log_at_trx_commit = 2
  - sync_binlog = 1000

> **Note** You cannot use a custom parameter template to change the values of the parameters that are included in the system parameter templates.

The following table describes the parameters in the system parameter templates.

Parameter	Value	Description
innodb_flush_log_at_trx_commit	1	When you commit a transaction, the system writes the transaction log record from the buffer to the log file and immediately synchronizes the log file to the disk.
	2	When you commit a transaction, the system writes the transaction log record from the buffer to the log file but does not immediately synchronize the log file to the disk. The log file is written to the disk once every second. If the system stops responding before a write operation is performed, the log records that are generated over the most recent second are lost.
sync_binlog	1	When you commit a transaction, the binary log file is written to the disk and the disk is immediately refreshed. The binary log file is not written to the buffer.
	1,000	The log records in the buffer are written to the disk and the disk is refreshed once every time when 1,000 log records are submitted to the buffer. This may result in data loss.

Apply a parameter template

1.
2. In the left-side navigation pane, click **Parameter Templates**.
3. On the **Custom Parameter Templates** or **System Parameter Templates** tab, find the parameter template that you want to apply and click **Apply to Instance** in the **Actions** column.
4. In the All Instances section of the panel that appears, select the RDS instances to which you want to apply the parameter template, click the icon to move the selected RDS instances to the Selected Instances section, and then view the changes to the parameter settings in the Parameter Comparison section.

   > **Note** Before you apply a parameter template to multiple RDS instances, you must verify that the parameter settings are suitable for all the RDS instances.

5. Click **OK**.

Create a custom parameter template

1.
2. In the left-side navigation pane, click **Parameter Templates**. On the **Parameter Templates** page, click **Create Parameter Template**.
3. Configure the following parameters.

Parameter	Description

Parameter	Description
Template Name	Enter the name of the parameter template. The name must be 8 to 64 characters in length and can contain letters, digits, periods (.), and underscores (_). The name must start with a letter.
Database Engine	Select the database engine that is run by the RDS instance. Set the value to MySQL.
Engine Version	Select the version of the database engine that is run by the RDS instance. The supported database engine versions are MySQL 5.6, MySQL 5.7, and MySQL 8.0.
Description	Enter a description for the parameter template. The description can be up to 200 characters in length.
Add Parameter	Click **Add Parameter** and select a parameter from the **Parameter** drop-down list. Then, you can configure the parameter. You can also view the value range and default value of the parameter. **Note** ◦ For more information about the available parameters, see the parameters on the **Editable Parameters** tab of the **Parameters** page. ◦ If you want to add another parameter, you must click **Add Parameter** again. ◦ To delete a parameter, you must click **Delete** to the right of the parameter.
Import	After you export a parameter template to your computer, you can edit the parameter template based on your business requirements. Then, you can click **Import** to copy the parameter settings from the parameter template to the RDS instance. For more information about how to export a parameter template, see Modify the parameters of an ApsaraDB RDS for MySQL instance.

4. Click **OK**.

Clone a parameter template

You can clone a parameter template from one region to another region.

1.
2. In the left-side navigation pane, click **Parameter Templates**.
3. Find the parameter template that you want to clone, click the ⋮ icon, and then select **Clone** in the **Actions** column.
4. Configure the following parameters.

Parameter	Description
Region	Specify the destination region to which you want to clone the parameter template.
Template Name	Enter the name of the parameter template. The name must be 8 to 64 characters in length and can contain letters, digits, periods (.), and underscores (_). The name must start with a letter.
Description	Enter a description for the parameter template. The description can be up to 200 characters in length.

5. Click **OK**.

Manage parameter templates

1.
2. In the left-side navigation pane, click **Parameter Templates**.
3. Manage the parameter templates in this region.

> **Note** You can perform the **View** and **Apply to Instance** operations on system templates.

View a parameter template.

Find the parameter template that you want to view and click **View** in the **Actions** column to view the basic information and parameter settings of the parameter template.

Modify a parameter template.

i. Find the parameter template that you want to modify, click the icon, and then select **Modify** in the **Actions** column. Alternatively, click **Edit Parameter Template** in the panel that appears after you click View in the Actions column of the parameter template. For more information, see Step 3 in the "Create a custom parameter template" section of this topic.

ii. Click **OK**.

Delete a parameter template.

Find the parameter template that you want to delete, click the icon, and then select **Delete** in the **Actions** column.

> **Note** When you delete a parameter template, the RDS instances to which the parameter template is applied are not affected.

Related operations

Operation	Description
Create a parameter template	Creates a parameter template.
Modify a parameter template	Modifies a parameter template.

Operation	Description
Copy a parameter template	Clones a parameter template from one region to another region.
Query parameter templates	Queries the parameter templates that are available in a region.
Query information of a parameter template	Queries the details about a parameter template.
Delete a parameter template of an ApsaraDB RDS instance	Deletes a parameter template from a region.

13.5. Optimize parameters of an ApsaraDB RDS for MySQL instance

You can modify parameter values of an ApsaraDB RDS for MySQL instance in the ApsaraDB RDS console. Improper values of key parameters may downgrade performance of an RDS instance or cause errors in your application. This topic provides optimization suggestions for key parameters.

back_log

- Applicable MySQL versions: 8.0, 5.7, 5.6, and 5.5.
- Default value: 3000.
- Whether to restart the instance after parameter modification: Yes.
- Function: The primary MySQL thread creates a new thread for each connection request that it processes. If frontend applications initiate a large number of short-lived connections when the primary thread creates a new thread, ApsaraDB RDS for MySQL restricts the short-lived connection requests to enter the queue based on the back_log parameter. When the number of waiting connection requests in the queue exceeds the value of the back_log parameter, ApsaraDB RDS for MySQL denies new connection requests. If you want ApsaraDB RDS for MySQL to process a large number of short-lived connections, increase the value of this parameter.
- Symptom: If the value of this parameter is too small, the application may encounter the following error:

  ```
  SQLSTATE[HY000] [2002] Connection timed out;
  ```

- Suggestion: Increase the value of this parameter.

innodb_autoinc_lock_mode

- Applicable MySQL versions: 8.0, 5.7, 5.6, and 5.5.
- Default value: 1.
- Whether to restart the instance after parameter modification: Yes.
- Function: In MySQL 5.1.22 and later, the innodb_autoinc_lock_mode parameter is used in InnoDB to control auto-increment locks. Valid values: 0, 1, or 2. Default value: 1. The default value indicates that InnoDB uses a lightweight mutex to obtain auto-increment locks, instead of table-level locks. However, the SQL statements that are used to load data (including the `INSERT ... SELECT` and `REPLACE ... SELECT`) use auto-increment table locks. If the application initiates a number of SQL statements that are concurrently executed to load data, a deadlock may occur.

- Symptom: If the SQL statements that are used to load data (including `INSERT ... SELECT` and `REPLACE ... SELECT`) use auto-increment table locks, the following deadlock occurs during concurrent data loading:

```
RECORD LOCKS space id xx page no xx n bits xx index PRIMARY of table xx.xx trx id xxx lock_mode X insert intention waiting. TABLE LOCK table xxx.xxx trx id xxxx lock mode AUTO-INC waiting;
```

- Suggestion: Change the value of this parameter to 2. This value indicates that all SQL statements that are used to load data in row mode use a lightweight mutex. This avoids the AUTO-INC deadlock and greatly improves performance of the `INSERT ... SELECT` statement.

> **Note** If you set the parameter to 2, you must set the format of binary logs to row.

query_cache_size

- Applicable MySQL versions: 5.7, 5.6, and 5.5.
- Default value: 3145728.
- Whether to restart the instance after parameter modification: No.
- Function: This parameter controls the memory capacity of the MySQL query cache. If you enable the MySQL query cache, the system locks the query cache before it performs a query. Then, the system checks for the query result in the cache. If the query result exists in the query cache, the system directly returns the result. Otherwise, it performs the query to obtain the result. The INSERT, UPDATE, and DELETE operations invalidate the query cache and cause changes in schemas or indexes. Frequent invalidation of the query cache brings heavy pressure on the RDS instance. If data on the RDS instance is not frequently updated, the query cache can greatly improve query efficiency. However, if the database processes a large number of write operations on a few tables, the lock mechanism of the query cache may cause frequent lock conflicts. Both the write and read requests on the locked table wait for the query cache to be unlocked. This reduces query efficiency of SELECT statements.
- Symptom: A large number of database connections are in the following states: `checking query cache for query`, `waiting for query cache lock`, and `storing result in query cache`.
- Suggestion: By default, ApsaraDB RDS disables query cache. If you have enabled query cache and encountered the preceding symptom, disable query cache.

net_write_timeout

- Applicable MySQL versions: 8.0, 5.7, 5.6, and 5.5.
- Default value: 60.
- Whether to restart the instance after parameter modification: No.
- Function: This parameter sets the timeout period that ApsaraDB RDS waits before it sends a block to a client.
- Symptom: If the parameter value is too small, the client may encounter the following error:

```
"the last packet successfully received from the server was milliseconds ago" or "the last packet sent successfully to the server was milliseconds ago"
```

- Suggestion: The default value of this parameter is 60s. If the value is too small, the client may be frequently disconnected from the RDS instance when the network is not stable or it takes a long time for the client to process each block. We recommend that you increase the value of this parameter.

tmp_table_size

- Applicable MySQL versions: 8.0, 5.7, 5.6, and 5.5.
- Default value: 2097152.
- Whether to restart the instance after parameter modification: No.
- Function: This parameter determines the maximum size of an internal temporary memory table. The size is assigned to each thread. The actual value is the smaller one between tmp_table_size and max_heap_table_size. If the size of the temporary memory table exceeds the limit, ApsaraDB RDS for MySQL automatically converts the table to a disk-based MyISAM table. When you optimize query statements, do not use internal temporary tables. If you have to use a temporary table, make sure that the temporary table is stored in the memory.
- Symptom: If you use a temporary table for complicated SQL statements that contain GROUP BY or DISTINCT clauses and cannot be optimized by using indexes, SQL execution takes a longer time.
- Suggestion: If the SQL statements contain a large number of GROUP BY or DISTINCT clauses and the instance has enough memory, increase the values of the tmp_table_size and max_heap_table_size parameters to improve query performance.

loose_rds_max_tmp_disk_space

- Applicable MySQL versions: 5.6 and 5.5.
- Default value: 10737418240.
- Whether to restart the instance after parameter modification: No.
- Function: This parameter controls the size of temporary files on the RDS instance.
- Symptom: If the size of temporary files exceeds the value of the loose_rds_max_tmp_disk_space parameter, the application may encounter the following error:

  ```
  The table '/home/mysql/dataxxx/tmp/#sql_2db3_1' is full
  ```

- Suggestion: Evaluate whether you can optimize the SQL statements that cause an increase of temporary files by using indexing or other methods. If your instance has enough space, increase the value of this parameter to ensure normal execution of SQL statements.

loose_tokudb_buffer_pool_ratio

- Applicable version: 5.6
- Default value: 0.
- Whether to restart the instance after parameter modification: Yes.
- Function: This parameter specifies the size of buffer memory that can be used by TokuDB tables. For example, if the innodb_buffer_pool_size parameter is set to 1000 MB and the tokudb_buffer_pool_ratio parameter to 50 (indicating 50%), the size of buffer memory that can be used by TokuDB tables is 500 MB.
- Suggestion: If the TokuDB engine is used on the RDS instance, increase the value of this parameter to improve access performance of TokuDB tables.

loose_max_statement_time

- Applicable MySQL version: 5.6
- Default value: 0.
- Whether to restart the instance after parameter modification: No.

- Function: This parameter sets a limit on how long a query can take before it times out. By default, the query time is not limited. If this parameter is configured and the query time exceeds the specified limit, the query fails.
- Symptom: If the query time exceeds the value of this parameter, the following error occurs:

```
ERROR 3006 (HY000): Query execution was interrupted, max_statement_time exceeded
```

- Suggestion: If you want to limit the time to execute SQL statements, set this parameter to a non-zero value. Unit: milliseconds.

loose_rds_threads_running_high_watermark

- Applicable MySQL versions: 5.6 and 5.5.
- Default value: 50000.
- Whether to restart the instance after parameter modification: No.
- Function: This parameter limits the number of concurrent queries. For example, if you set the rds_threads_running_high_watermark parameter to 100, 100 MySQL queries can be concurrently executed. Additional queries are denied.
- Suggestion: This parameter is used to handle burst requests and requests during peak hours to protect the RDS instance.

14.Backup

14.1. Overview

This topic provides an overview of the backup feature supported by ApsaraDB RDS for MySQL. This feature allows you to restore historical data with ease.

Methods

- Automatic backup: performed by the system on a regular basis. You can specify the time when automatic backups are performed. Automatic backup files contain all the data of an instance.
- Manual backup: manually initiated at any time. You can choose to back up the whole instance or specific databases or tables.

For more information, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.

Composition

ApsaraDB RDS provides data backup and log backup.

- **Data backup**: The system backs up data in the instance and generates backup sets. You can restore data to the time point when a backup set is created.
- **Log backup**: After log backup is enabled, you can restore data to **any point in time** within a specific time range based on data and log backup files.

  For example, if a data backup set of an instance is created at 00:00:01 on January 1, 2021 and log backup is enabled, you can restore data to any second since 00:00:01 on January 1, 2021.

Storage location

Data and log backup files of your ApsaraDB RDS instance are stored by Alibaba Cloud and do not consume the **storage capacity of your instance**.

Data and log backup files of an ApsaraDB RDS instance are stored in the same region, but not necessarily in the same zone as the instance. For information about how to store the backup files in another region, see Enable cross-region backups for an ApsaraDB RDS for MySQL instance.

> **Note**
>
> - You can read data from the storage space for backup, but cannot use the storage space. For information about how to download backup files from the storage space, see Download the backup files of an ApsaraDB RDS for MySQL instance.
> - Each ApsaraDB RDS instance is allocated a free quota for backup storage. If your backup storage usage exceeds the free quota, you are charged extra fees. For more information, see Backup storage pricing of an ApsaraDB RDS for MySQL instance.

Impact

- Backup operations of an ApsaraDB RDS instance on High-availability Edition or Enterprise Edition are performed on the secondary instance. Therefore, they do not occupy CPU resources of the primary instance or affect its performance.

> **Note** In rare cases when the secondary instance is unavailable, backups are performed on the primary instance.

- If an ApsaraDB RDS instance runs the RDS Basic Edition, it stands alone without a standby. All backups are performed on the instance. In this case, the performance of the instance decreases during a backup.

14.2. View and manage the size of backup files for an ApsaraDB RDS for MySQL instance

This topic describes how to view and manage the size of backup files for an ApsaraDB RDS for MySQL instance.

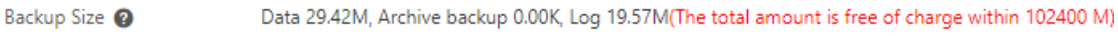
The size of backup files for your RDS instance is the total size of the data and log backup files that are generated on your RDS instance.

View the size of backup files

You can log on to the ApsaraDB RDS console and go to the **Basic Information** page for your RDS instance. In the lower-right corner of the page, you can view the size of backup files for your RDS instance.

In the following figure, the size of backup files includes 33.2 GB of data backup files and 20.19 MB of log backup files.

> **Note**
>
> - In the following figure, **Archive backup** indicates the size of the backup files that are retained for more than 730 days, and **Data** indicates the size of the data backup files that are not archived.
> - After you update the minor engine version of your RDS instance that runs RDS Basic Edition, the **Backup Size** parameter on the **Basic Information** page in the ApsaraDB RDS console may be displayed as 0. After the next scheduled backup is complete, this error is automatically fixed.



Additional information

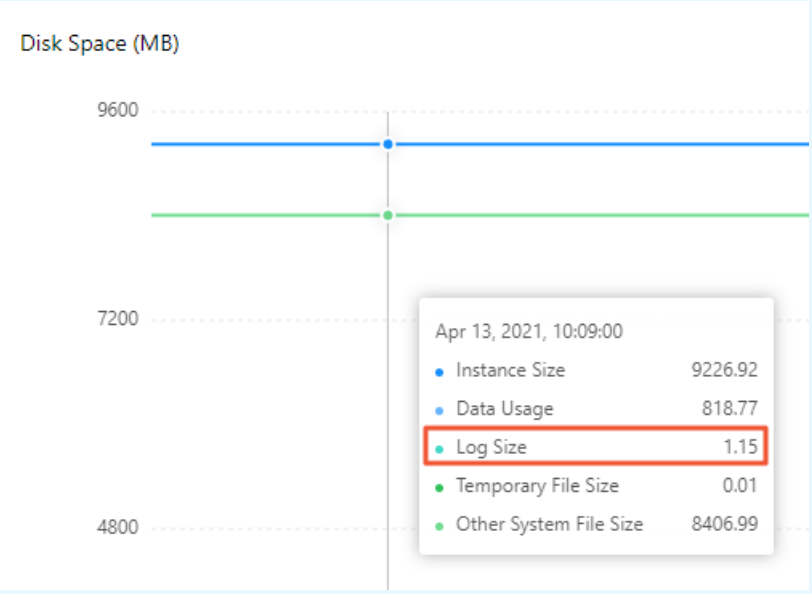
- Relationship between the size of backup files and the storage capacity
  - Data and log backup files consume the backup storage that is provided by Alibaba Cloud. **These files do not consume the storage capacity of your RDS instance.**

- Log files are categorized as local log files and log backup files.

Log file type	Description	Billing	Purpose
Local log files	Local log files refer to the raw log files that are generated on your RDS instance. Local log files are stored on your RDS instance.	Local log files do not incur charges. However, local log files consume the storage capacity of your RDS instance.	Local log files can be used to build the primary/secondary architecture.
Log backup files	After you enable the log backup feature, ApsaraDB RDS uploads the local log files of your RDS instance to the specified backup storage.	For more information, see Backup storage pricing of an ApsaraDB RDS for MySQL instance.	Log backup files are used to restore the data of your RDS instance to a specific point in time.

Note

- You can upload the local log files of your RDS instance to an Object Storage Service (OSS) bucket. This reduces the storage space that is consumed by local log files on your RDS instance. However, this does not reduce the size of log backup files. For more information, see Upload the binary log files of an ApsaraDB RDS for MySQL instance to an OSS bucket.
- You can log on to the ApsaraDB RDS console and go to the **Monitoring and Alerts** page for your RDS instance. On this page, you can view the storage space that is consumed by local log files on your RDS instance.



- Relationship between the size of backup files and the amount of data

  The size of backup files may not equal the amount of data on your RDS instance.

If your RDS instance uses standard or enhanced SSDs, ApsaraDB RDS takes snapshots to back up your RDS instance. **The size of a snapshot backup file may be significantly larger than the amount of data.** Therefore, the free quota for backup storage that is allocated to an RDS instance that uses standard or enhanced SSDs is four times the free quota for backup storage that is allocated to an RDS instance that uses local SSDs. For more information, see Backup storage pricing for an ApsaraDB RDS for MySQL instance.

> **Note** When ApsaraDB RDS calculates the size of a snapshot backup file, it counts in all the non-empty blocks to which data is written. If the write operations are scattered among blocks, none of the blocks is empty. For example, if 3 MB of data is written across two, three, or four blocks, none of these blocks is empty. In this case, ApsaraDB RDS counts in all these blocks to calculate the total size. As a result, the size of the snapshot backup file is larger than the actual amount of data.

- Relationship between the size of backup files and the instance architecture

  The size of backup files does not vary based on the instance architecture. For example, an RDS instance that runs the High-availability Edition has the same amount of data as an RDS instance that runs the Basic Edition. In this case, these RDS instances have the same size of backup files. The size of backup files does not increase even if you use an RDS edition that provides high availability.

Delete data backup files or reduce the data backup frequency

- **Delete data backup files.**
  i.
  ii. In the left-side navigation pane, click **Backup and Restoration**.
  iii. On the **Data Backup** tab, find the data backup file that you want to delete, and click **Delete** in the Actions column.

  > **Note** If you cannot find the **Delete** button, you can check whether the following requirements are met:
  >
  > - If the log backup feature is disabled, you can delete only the data backup files that are retained for more than seven days.
  > - If the log backup feature is enabled, you can delete only the data backup files that are retained for longer than the specified log backup retention period. For example, if you set the log backup retention period to seven days, you can delete the data backup files that are retained for more than seven days.

- **Configure a policy that allows ApsaraDB RDS to automatically delete data backup files.**
  i.
  ii. In the left-side navigation pane, click **Backup and Restoration**.
  iii. On the **Backup Settings** tab, click Edit. In the Backup Settings dialog box, reduce the data backup retention period.

  ApsaraDB RDS deletes the data backup files that are stored for longer than the specified data backup retention period. For example, if you set the data backup retention period of your RDS instance to 30 days, ApsaraDB RDS immediately deletes the data backup files that have been retained for more than 30 days on your RDS instance.

- **Reduce the data backup frequency.**

i.

ii. In the left-side navigation pane, click **Backup and Restoration**.

iii. On the **Backup Settings** tab, click Edit. In the Backup Settings dialog box, reduce the data backup frequency.

- **Delete or migrate the data that does not need to be backed up.**

Delete log backup files or reduce the log backup frequency

- **Reduce the log backup retention period.**

  i.

  ii. In the left-side navigation pane, click **Backup and Restoration**.

  iii. On the **Backup Settings** tab, click Edit. In the Backup Settings dialog box, reduce the log backup retention period.

  For example, if you set the log backup retention period of your RDS instance to 30 days, ApsaraDB RDS immediately deletes the log backup files that have been retained for more than 30 days on your RDS instance.

  > **Note** You can set the log backup retention period only when the log backup feature is enabled.

- **Disable the log backup feature.**

  i.

  ii. In the left-side navigation pane, click **Backup and Restoration**.

  iii. On the **Backup Settings** tab, click Edit. In the Backup Settings dialog box, disable the log backup feature.

  > **Note** After the log backup feature is disabled, ApsaraDB RDS immediately deletes all log backup files. You cannot restore the data of your RDS instance to a specific point in time.

- **Reduce unnecessary add, delete, and update operations, especially the update operations on large fields.**

  Add, delete, and update operations increase the total size of log backup files.

  > **Note** You can use the SQL Explorer feature to view the add, delete, update, and query operations that are performed on your RDS instance. For more information, see Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance.

14.3. Backup storage pricing of an ApsaraDB RDS for MySQL instance

This topic describes the backup storage pricing of an ApsaraDB RDS for MySQL instance.

Backup storage fee

If the total size of the backup files of your RDS instance does not exceed the free quota for backup

storage, no fees are charged. For more information, see View and manage the size of backup files for an ApsaraDB RDS for MySQL instance. If the total size exceeds the free quota, you are charged an hourly fee for the amount of excess backup storage that you use. The hourly fee is calculated by using the following formula: **Hourly fee for backup storage = (Total size of backup files - Free quota) × Unit price**.

- **Total size of backup files**: the total size of data backup files and log backup files. You can log on to the ApsaraDB RDS console, go to the Basic Information page of your RDS instance, and then view the free quota in the lower-right corner of the page.
- **Free quota**: a specific amount of storage that is allocated to store the backup files of your RDS instance. This amount of storage is free of charge. If your RDS instance uses standard SSDs or enhanced SSDs (ESSDs), the free quota is 200% of the storage capacity that you purchased for your RDS instance. If your RDS instance uses local SSDs, the free quota is 50% of the storage capacity that you purchased for your RDS instance. The free quota is measured in GB. You can only round the free quota up to the next integer. You cannot round the free quota down. You can log on to the ApsaraDB RDS console, go to the Basic Information page of your RDS instance, and then view the free quota in the lower-right corner of the page.
- **Unit price**: The unit price varies based on several factors. For more information, see the following table.

Storage type	Unit price for backup files that are stored for up to 730 days (USD/GB)	Unit price for archived backup files that are stored for more than 730 days (USD/GB)
Standard SSD or ESSD	0.00004	The unit price varies in different regions: ○ Japan (Tokyo), India (Mumbai), Singapore (Singapore), Australia (Sydney), Indonesia (Jakarta), and Philippines (Manila): 0.000034 ○ China (Hong Kong): 0.000031 ○ UAE (Dubai): 0.000028 ○ Germany (Frankfurt), UK (London), US (Silicon Valley), and US (Virginia): 0.000022 ○ Other regions: 0.000025
Local SSD	0.00020	

> **Note** The backup storage fee can be deducted from your Database Backup (DBS) storage plan. For more information, see Storage fees.

Usage notes

The backup storage fee varies based on the total size of backup files. Backup files do not consume the storage capacity of your RDS instance. Therefore, the backup storage fee does not vary based on the storage usage.

When you analyze the backup storage fee, you must check the total size of backup files. You do not need to check the storage usage.

Methods of reducing the backup storage fee

- Reduce the total size of backup files.

You can delete the backup files that are no longer required. You can also reduce the backup frequency. For more information, see View and manage the size of backup files for an ApsaraDB RDS for MySQL instance.

- Increase the free quota.

  You can expand the storage capacity of your RDS instance. For more information, see Change the specifications of an ApsaraDB RDS for MySQL instance.

  The free quota varies based on the storage capacity of your RDS instance. For example, if your RDS instance uses local SSDs and you expand the storage capacity of your RDS instance from 150 GB to 300 GB, the free quota is increased from 75 GB to 150 GB.

14.4. Perform backups

14.4.1. Methods of backing up and restoring an ApsaraDB RDS for MySQL instance

This topic describes the methods that you can use to back up and restore an ApsaraDB RDS for MySQL instance.

Scenario	Method	References
Perform backups	Perform scheduled backups.	Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance
	Perform incremental backups or log backups.	
	Back up individual databases and tables.	Back up the individual databases and tables of an ApsaraDB RDS for MySQL instance
	Perform cross-region backups.	Enable cross-region backups for an ApsaraDB RDS for MySQL instance
	Perform cross-account backups.	Back up and restore data across Alibaba Cloud accounts
Store backup files	Store backup files to your Object Storage Service (OSS) bucket.	Back up an ApsaraDB RDS for MySQL, PolarDB for MySQL, or self-managed MySQL instance by using logical backup
	Store backup files to a multi-level storage pool.	What is a storage pool?
Download backup files	Manually download backup files.	Download the backup files of an ApsaraDB RDS for MySQL instance
	Configure a policy based on which ApsaraDB RDS can automatically download backup files.	
	Encrypt backup files at rest.	Encrypt the backup files of an

Encrypt backup files Scenario	Method	References
	Encrypt backup files in transit by using SSL.	
Restore data	**Restore data in a few seconds.**	Create a sandbox instance for the emergency disaster recovery of an ApsaraDB RDS for MySQL database
	Query data from backup files at high speeds.	Query data from DBS-generated logical backup files
	Restore individual databases and tables.	Restore the individual databases and tables of an ApsaraDB RDS for MySQL instance Restore data by database or table
	Restore data to a specific point in time.	Restore the data of an ApsaraDB RDS for MySQL instance Native Flashback Restore a MySQL database from a logical backup
	Restore data to a new RDS instance.	
	Restore data to the original RDS instance.	
	Restore data to an existing RDS instance rather than the original RDS instance.	
	Restore data to an on-premises database.	

14.4.2. Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance

This topic describes how to enable the automatic backup feature for an ApsaraDB RDS for MySQL instance. ApsaraDB RDS for MySQL supports automatic backups and manual backups.

For more information about how to enable the automatic backup feature for an RDS instance that runs a different database engine, see the following topics:

- Back up an ApsaraDB RDS for SQL Server instance
- Back up an ApsaraDB RDS for PostgreSQL instance
- Automatically back up the data of an RDS MariaDB instance

> **Note** The default backup feature that is provided by ApsaraDB RDS stores backup files in the same region as your RDS instance. For more information about how to store backup files in a region that is different from the region of your RDS instance, see Enable cross-region backups for an ApsaraDB RDS for MySQL instance.

Introduction to backups

You can back up your RDS instance by using one of the following methods:

- ApsaraDB RDS: ApsaraDB RDS provides a default backup feature that supports full backups and incremental backups. Full backups are also called data backups, and incremental backups are also called log backups.
- Database Backup (DBS): DBS provides an advanced backup feature. This feature supports cross-account backups and allows DBS to create a backup within seconds.

For more information about the differences between the default backup feature of ApsaraDB RDS and the advanced backup feature of DBS, see ApsaraDB RDS-generated backups and DBS-generated backups.

Use ApsaraDB RDS to perform automatic backups

ApsaraDB RDS supports automatic data backups and automatic log backups. Data backups cannot be disabled. However, you can change the data backup frequency based on your business requirements.
Data backup frequency

Instance configuration	Minimum frequency	Maximum frequency
• RDS instances that run RDS High-availability Edition or RDS Enterprise Edition with local SSDs • RDS instances that run RDS Basic Edition with standard SSDs or enhanced SSDs (ESSDs)	Twice every week	Once every day
RDS instances that run RDS High-availability Edition with standard SSDs or ESSDs		Once every 15 minutes (You must enable the **Increase Snapshot Frequency** feature.)

Precautions

- Backup policies cannot be configured for read-only RDS instances.
- Do not execute DDL statements during a backup. These statements trigger locks on tables, and the backup may fail as a result of the locks.
- We recommend that you back up your RDS instance during off-peak hours.
- If the number of tables that are created on your RDS instance exceeds 600,000, backups cannot be created. In this case, we recommend that you shard the databases on your RDS instance.
- If the number of tables that are created on your RDS instance exceeds 50,000, you cannot restore individual databases or tables. For more information, see Restore the individual databases and tables of an ApsaraDB RDS for MySQL instance.

Procedure

1.
2. In the left-side navigation pane, click **Backup and Restoration**.
3. On the **Backup and Restoration** page, click the **Backup Settings** tab. In the Data Backup Settings section of the tab that appears, click **Edit**.
4. Configure the following parameters and click **OK**.

> **Note** If your RDS instance is equipped with standard SSDs or ESSDs, ApsaraDB RDS creates snapshots to back up the data of the RDS instance.

Data backup settings

Instance configuration	Parameter	Description
RDS instances with all types of configurations	Data Backup Retention (Days)/Retention Period	The number of days for which data backup files are retained. Default value: 7. Valid values: ◦ For RDS instances that are equipped with standard SSDs or ESSDs: 7 to 730. Note ■ If your RDS instance runs MySQL 5.7 on RDS Basic Edition, the value is fixed as 7 and cannot be changed. ■ If you enable the **Single-digit Second Backup** feature, the valid value of this parameter ranges from 7 to 730. ◦ For RDS instances that are equipped with local SSDs: 7 or more. The number of days must be less than 2 to the power of 31. ■ Data backup files that are retained for 730 days or less are considered regular backup files. ■ Data backup files that are retained for more than 730 days are converted into archived backup files. The cost of archived backup files is less than the cost of regular backup files. For more information, see Backup storage pricing of an ApsaraDB RDS for MySQL instance. Note If you set this parameter to a value greater than 730 or select **Long-term Retention**, the data backup files that are retained for more than 730 days are automatically converted into archived backup files. Therefore, you must configure the Archived Backup Retention parameter. For example, if you select Monthly and enter 2 in the unit field, ApsaraDB RDS retains the first two archived backup files that are generated each month.
	Backup Cycle	The cycle based on which data backups are created. You must select at least two days of the week.
	Backup Time	The hour at which a data backup is created. For example, you can select the 05:00-06:00 period. We recommend that you select an off-peak hour.

Instance configuration	Parameter	Description
RDS instances that are equipped with local SSDs	**Data Backup Retention Policy After Release**	Specifies whether to retain a specific number of data backup files after your RDS instance is released. Select **Latest** or **All**. **Note** • This policy is used to **permanently retain** data backup files. **You are not charged for the storage of the data backup files that are retained**. • This policy is supported for pay-as-you-go RDS instances and subscription RDS instances. • After your RDS instance is released, you can go to the Backup for Deleted Instances tab to download the data backup files of your RDS instance. Then, you can use the data backup files to restore the data of your RDS instance.
	Restore Individual Database /Table	Specifies whether to support the restoration of individual databases and tables. For more information, see Restore individual databases and tables of an ApsaraDB RDS for MySQL instance. The Restore Individual Database/Table feature is automatically enabled and cannot be disabled. **Note** • The Restore Individual Database/Table feature is supported only for RDS instances that run MySQL 8.0, MySQL 5.7, or MySQL 5.6 on RDS High-availability Edition with local SSDs. • After you enable the Restore Individual Database/Table feature, a new backup file format takes effect. For more information, see [Notice] New physical backup file format phased in for some ApsaraDB RDS for MySQL instances.

Instance configuration	Parameter	Description
	Restore Speed	The speed at which an individual database or table is restored. Valid values: ○ **Standard**: Databases and tables are restored at a regular speed. ○ **Fast**: Databases and tables are restored at a high speed. Fast restoration provides a restoration speed that is 50% to 95% faster than standard restoration. Fast restoration is in public review and is provided free of charge. For more information about the restoration of databases and tables, see Restore individual databases and tables of an ApsaraDB RDS for MySQL instance. **Note** The fast restoration of databases and tables is supported only in the China (Zhangjiakou) and China (Hohhot) regions.
RDS instances that are equipped with standard SSDs or ESSDs	**Increase Snapshot Frequency**	The frequency at which snapshot backups are created. This feature enables you to configure a shorter interval to create snapshots by using the Single-digit Second Backup feature, which increases the frequency of the snapshots. You can increase the frequency to up to once every 15 minutes. The snapshot retention policies vary based on the value of this parameter. ○ A value at the minute granularity: All snapshots that are completed within 1 hour are retained. For snapshots that are retained for more than 1 hour, ApsaraDB RDS deletes the snapshots except for the first snapshot after the hour. For snapshots that are retained for more than 24 hours, ApsaraDB RDS deletes the snapshots except for the first snapshot after 00:00 every day. ○ A value at the hour granularity: All snapshots that are completed within 24 hours are retained. For snapshots that are retained for more than 24 hours, ApsaraDB RDS deletes the snapshots except for the first snapshot after 00:00 every day. **Note** ○ The Increase Snapshot Frequency feature is supported only for RDS instances that run the RDS High-availability Edition with standard SSDs or ESSDs. ○ If you want to enable this feature, you must enable the **Single-digit Second Backup** feature. If the **Single-digit Second Backup** feature is disabled, ApsaraDB RDS automatically enables the **Single-digit Second Backup** feature when you enable the Increase Snapshot Frequency feature.

Instance configuration	Parameter	Description
	Single-digit Second Backup	Allows ApsaraDB RDS to create a backup within seconds. **Note** The Single-digit Second Backup feature is supported only for RDS instances that run RDS High-availability Edition with ESSDs.

Log backup settings

Parameter	Description
Log Backup	Specifies whether to enable the log backup feature. After this feature is enabled, you can restore the data of your RDS instance to a specified point in time. This feature is enabled by default. **Note** You cannot disable this feature for RDS instances that run MySQL 5.7 on RDS Basic Edition.
Log Retention Period (Days)	○ The valid value ranges from 7 to 730. The default value is 7. ○ The value of this parameter must be less than or equal to the value of the Data Backup Retention (Days) parameter. **Note** The value is fixed to 7 for RDS instances that run MySQL 5.7 on RDS Basic Edition.

Use DBS to perform automatic backups

1. Create a backup schedule. Make sure that you select the logical backup method.
2. Configure the backup schedule that you created.

For more information, see Back up an ApsaraDB RDS for MySQL, PolarDB for MySQL, or self-managed MySQL instance by using logical backup.

FAQ

1. When a backup is in progress, does the performance of my RDS instance decrease?
   - Backup operations of an ApsaraDB RDS instance on High-availability Edition or Enterprise Edition are performed on the secondary instance. Therefore, they do not occupy CPU resources of the primary instance or affect its performance.

     Note In rare cases when the secondary instance is unavailable, backups are performed on the primary instance.

- If an ApsaraDB RDS instance runs the RDS Basic Edition, it stands alone without a standby. All backups are performed on the instance. In this case, the performance of the instance decreases during a backup.

2. How do I query data from backup files at fast speeds?

   If you have full logical backup files, you can use DBS to query data from these files. This way, you do not need to restore the data of these files. For more information, see Overview.

3. Can I disable the data backup feature for my RDS instance?

   No, you cannot disable the data backup feature for your RDS instance. However, you can reduce the data backup frequency to as low as twice a week. The retention period for data backup files must be at least seven days.

4. Can I disable the log backup feature for my RDS instance?

   Yes, if your RDS instance does not run MySQL 5.7 on RDS Basic Edition, you can disable the log backup feature for your RDS instance in the ApsaraDB RDS console.

5. Why did my backup task fail?

   A possible cause is that you executed time-consuming DDL statements or UPDATE statements when your backup task was in progress. These statements triggered locks on tables, and your backup task failed as a result of the locks.

6. Why is the size of a snapshot backup file significantly larger than the amount of data on my RDS instance?

   Cloud Disk instances are backed up by using snapshots. The size of a **snapshot backup may be much larger than the size of** the data. Therefore, the free backup quota of a cloud disk instance is four times that of an instance with local disks.

   > **Note** When the snapshot backup size is calculated, the size of all non-empty blocks is calculated. If the writing time is scattered (for example, 3MB of data may occupy two, three, or even four blocks), this causes many non-empty blocks, and therefore leads to a large number of snapshot backups.

References

- Download the backup files of an ApsaraDB RDS for MySQL instance
- Restore the data of an ApsaraDB RDS for MySQL instance

Related operations

- ApsaraDB RDS
  - Modify backup settings
  - 查询备份设置
  - Query the data backup files
  - Query backup tasks
- DBS
  - Create a backup plan
  - Configure a backup schedule

14.4.3. Create a backup for an ApsaraDB RDS for MySQL instance

This topic describes how to create a backup that is immediately performed on an ApsaraDB RDS for MySQL instance.

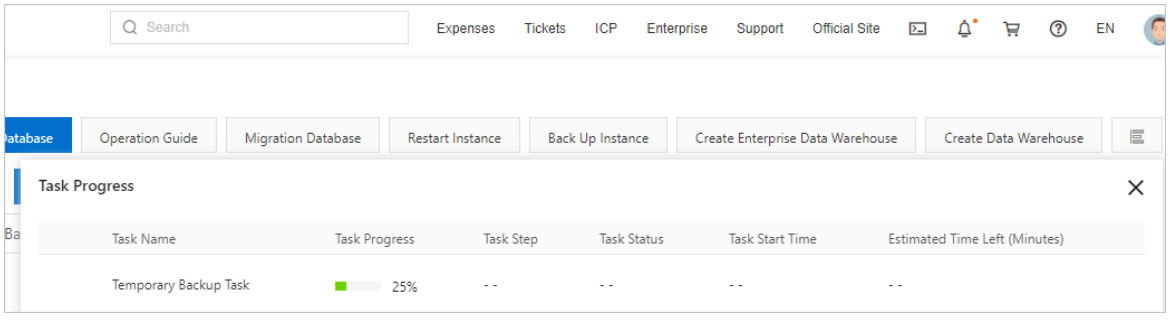
Procedure

1. In the upper-right corner of the page, click **Back Up Instance**.
2. Back up all databases or only the specified databases of your RDS instance. The supported backup methods vary based on the type of storage media that is used by your RDS instance. For more information, see the following table.

   > **Note** The data of physical backup files and snapshot backup files can be restored to a new RDS instance. The data of logical backup files cannot be restored to a new RDS instance.

Storage type	Backup all databases	Backup specified databases
Local SSDs	Use one of the following backup methods: ○ Select **Physical Backup** from the Select Backup Mode drop-down list. The speeds of physical backup and restoration are faster than the speeds of logical backup and restoration. ○ Select **Logical Backup** from the Select Backup Mode drop-down list. Then, set the Backup Policy parameter to **Instance Backup**.	Select **Logical Backup** from the Select Backup Mode drop-down list. Then, set the Backup Policy parameter to **Database/Table Backup**.
Standard SSDs or enhanced SSDs (ESSDs)	**Snapshot Backup**	Not supported.

3. In the upper-right corner of the page, click the **Task Progress** icon to view the progress of the backup task. Wait until the backup task is completed.



Related operations

Create data backup

14.4.4. Back up the individual databases and tables of an ApsaraDB RDS for MySQL instance

This topic describes how to back up the individual databases and tables of an ApsaraDB RDS for MySQL instance.

Configure ApsaraDB RDS to automatically back up individual databases and tables

For data security purposes, ApsaraDB RDS performs automatic backups to back up all the databases and tables of your RDS instance. These automatic backups do not support individual databases and tables.

- If you do not need to back up some databases and tables, we recommend that you delete these databases and tables or migrate them to your computer.
- If you want to reduce the size and storage cost of your backup files, see Backup storage pricing of an ApsaraDB RDS for MySQL instance.
- If you want to restore individual databases and tables in the event of exceptions, see the following table.

Instance configuration	Restoration method
MySQL 8.0, 5.7, or 5.6 on RDS High-availability Edition with local SSDs	◦ Log on to the ApsaraDB RDS console and set the Restore Individual Database/Table option to Enabled in the backup settings of your RDS instance. Then, you can use the newly generated backup files to restore individual databases and tables. For more information, see Use ApsaraDB RDS to perform automatic backups. ◦ On the **Backup and Restoration** page, click to back up individual databases and tables. For more information, see Restore individual databases and tables of an ApsaraDB RDS for MySQL instance.
Other configuration	Use the mysqldump plug-in to back up and restore your RDS instance. For more information, see How do I back up and restore the database of an ApsaraDB RDS for MySQL instance?

Back up individual databases and tables

> **Note** You can manually back up individual databases and tables only when your RDS instance uses local SSDs. For more information, see Storage types.

1.
2. In the upper-right corner of the page, click **Back Up Instance**.
3. In the dialog box that appears, set the **Select Backup Mode** parameter to **Logical Backup** and set the **Backup Policy** parameter to **Database/Table Backup**.
4. Select the databases that you want to back up in the left-side section, click the rightwards arrow to move the selected databases to the right-side section, and then click **OK**.

ApsaraDB RDS generates a logical backup task. You can view the progress of the logical backup task in the upper-right corner of the page.

You can restore the data of your RDS instance from the generated logical backup file. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance.

Related operations

Operation	Description
CreateBackup	Creates a data backup for an ApsaraDB RDS instance.

14.4.5. Enable cross-region backups for an ApsaraDB RDS for MySQL instance

This topic describes how to enable cross-region backups for an ApsaraDB RDS for MySQL instance. After you enable cross-region backups, the backup files of the original RDS instance are automatically replicated from the source region to a specified destination region. You can use the backup files in the destination region to manage and restore the data of the original RDS instance.

Context

You can use one of the following methods to enable cross-region backups:

- Configure cross-region backups in the ApsaraDB RDS console
- Enable cross-region backups by using DBS

If a cross-region backup is complete, you can restore the data of the original RDS instance from the generated cross-region backup file to an existing RDS instance or to a new RDS instance that resides in the destination region. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance across regions.

> **Note**
>
> - For more information about default backups, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.
> - For more information about how to enable cross-region backups for an ApsaraDB RDS for PostgreSQL instance, see Enable cross-region backups for an ApsaraDB RDS for PostgreSQL instance.
> - For more information about how to enable cross-region backups for an ApsaraDB RDS for SQL Server instance, see Enable cross-region backups for an ApsaraDB RDS for SQL Server instance.

Differences between cross-region backups and default backups

Item	Cross-region backup	Default backup
Default configuration	By default, cross-region backups are disabled. You must manually enable cross-region backups.	By default, default backups are enabled.
Storage	Cross-region backup files are stored in a region that is different from the region of the original RDS instance.	Default backup files are stored in the region where the original RDS instance resides.
Restoration	Data from cross-region backup files can be restored to the following RDS instances: • Original RDS instance • New RDS instance in the destination region • Existing RDS instance	Data from default backup files can be restored to the following RDS instances: • New RDS instance that resides in the same region as the original RDS instance • Original RDS instance
Retention period	After the original RDS instance is released, its cross-region backup files are still retained based on the cross-region backup retention period that you specify.	By default, after the original RDS instance is released, its default backup files are retained for seven days.

Prerequisites

Where to perform cross-region backups	Prerequisite
ApsaraDB RDS console	The original RDS instance runs one of the following MySQL versions and RDS editions: • MySQL 8.0, MySQL 5.7, or MySQL 5.6 on RDS High-availability Edition with local SSDs • MySQL 5.6 on RDS Enterprise Edition with local SSDs

DBS	• Database Backup (DBS) is activated, and a backup schedule is created. For more information, see Purchase a backup schedule.
	  ◦ The region that you specify on the DBS buy page is not the region where the original RDS instance resides.
	  ◦ The Backup Method parameter is set to Logical Backup for the backup schedule.
	• A public endpoint is allocated to the original RDS instance. For more information, see Apply for or release a public endpoint for an ApsaraDB RDS for MySQL instance.

Billing

Where to perform cross-region backups	Billing
ApsaraDB RDS console	You are charged for the storage and traffic that are consumed by the cross-region backup files. • Remote storage fee: USD 0.0002 per GB-hour. • Network traffic fee: For more information, see Billing overview.
DBS	If you want to store cross-region backup files in the built-in storage of DBS, you are charged for the storage that you use in DBS. For more information, see Billing overview.

Precautions

If you want to enable cross-region backups in the ApsaraDB RDS console, take note of the following information:

- You can restore data from cross-region backup files to the source region or the destination region. However, if Transparent Data Encryption (TDE) is enabled for the original RDS instance, you can restore data from cross-region backup files only to the source region. For more information, see Configure TDE for an ApsaraDB RDS for MySQL instance.
- Cross-region backups do not affect default backups. These two types of backups exist at the same time.
- After a default backup is complete, a cross-region backup is triggered. During the cross-region backup process, the original RDS instance dumps the generated default backup files to the destination region.
- After you enable cross-region backups, the original RDS instance checks whether valid data backup files are generated over the most recent 24 hours. If no valid data backup files are generated over the most recent 24 hours, the original RDS instance triggers a backup on its secondary RDS instance.
- After you enable cross-region log backups, the original RDS instance checks the valid data backup files that are generated over the most recent 24 hours.
  - If continuous binary log files are generated following the valid data backup files, the original RDS instance dumps the binary log files to the destination region.
  - If no continuous binary log files are generated following the valid data backup files, the original RDS instance triggers a backup on its secondary RDS instance.
- Cross-region backups are supported only in a few Alibaba Cloud regions due to network reasons. The following table lists the Alibaba Cloud regions in which cross-region backups are supported.

Source region	Destination region

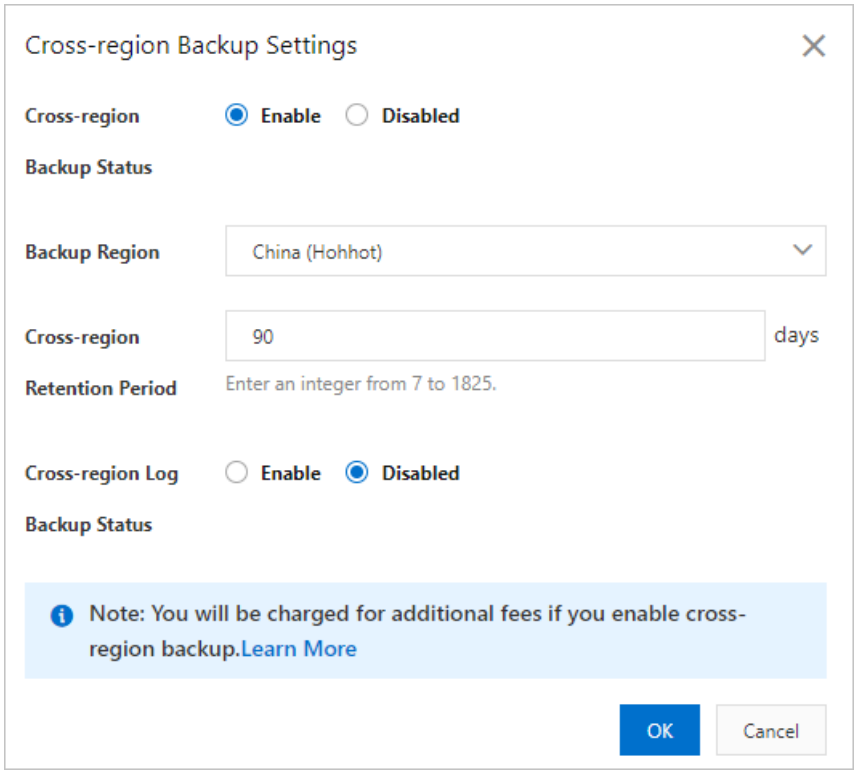
Source region	Destination region
China (Hangzhou), China (Shanghai), China (Qingdao), China (Beijing), China (Zhangjiakou), China (Hohhot), China (Shenzhen), China (Hong Kong), China (Ulanqab), China (Chengdu), China (Guangzhou), and China (Heyuan)	China (Hong Kong), China (Hangzhou), China (Shanghai), China (Qingdao), China (Shenzhen), China (Zhangjiakou), China (Hohhot), China (Beijing), China (Ulanqab), China (Chengdu), China (Guangzhou), and China (Heyuan) **Note** The backup files from a source region can be dumped to regions except the source region. The available destination regions may vary based on your network environment.
US (Silicon Valley)	US (Virginia)
US (Virginia)	US (Silicon Valley)
China East 1 Finance	China East 2 Finance and China South 1 Finance
China East 2 Finance	China East 1 Finance and China South 1 Finance
China South 1 Finance	China East 1 Finance and China East 2 Finance

Configure cross-region backups in the ApsaraDB RDS console

- **To enable cross-region backups for a single RDS instance, perform the following operations:**
  i. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Instances**. In the top navigation bar, select the region where the RDS instance resides.
  ii. Find the RDS instance for which you want to enable cross-region backups. In the Actions column, choose **More > Cross-region Backup Settings**.

  Note
  - You can also go to the **Backup and Restoration** page, click the **Backup Settings** tab, and then click **Edit** in the **Cross-region Backup Settings** section.
  - If the **Cross-region Backup Settings** tab cannot be found, you must check whether the RDS instance meets all prerequisites.

  iii. Configure the following parameters.



Parameter	Description
Cross-region Backup Status	Specify whether to enable or disable cross-region backups. Select **Enable**.
Backup Region	Select the destination region to which the backup files of the RDS instance are automatically replicated.
Cross-region Retention Period	Specify the retention period of cross-region backup files. Valid values: 7 to 1825. Unit: days. The longest cross-region backup retention period spans five years. **Note** After the RDS instance expires or is released, its cross-region backup files are still retained based on the cross-region backup retention period that you specify. You can log on to the ApsaraDB RDS console, click Backups in the left-side navigation pane, and then click the Cross-region Backup tab to view the cross-region backup files that are retained.
Cross-region Log Backup Status:	Specify whether to enable or disable cross-region log backups. After you enable cross-region log backups, the log backup files of the RDS instance are automatically replicated to a specified Object Storage Service (OSS) bucket in the destination region.

iv. Click **OK**.

- **To enable cross-region backups for multiple RDS instances at a time, perform the following operations:**

  i. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Backups**. In the top

navigation bar, select the region where the RDS instance resides.

ii. Click the **Cross-region Backup** tab. On the tab that appears, click the **Pending Instances** tab.

iii. Select the RDS instances for which you want to enable cross-region backups. Then, click **Backup Settings**.

> **Note** You can also click **Settings** in the Cross-region Backup Settings column of a single RDS instance to enable cross-region backups only for that RDS instance.

iv. Configure the following parameters.

Parameter	Description
Cross-region Backup Status	Specify whether to enable or disable cross-region backups. Select **Enable**.
Backup Region	Select the destination region to which the backup files of the RDS instance are automatically replicated.
Cross-region Retention Period	Specify the retention period of cross-region backup files. Valid values: 7 to 1825. Unit: days. The longest cross-region backup retention period spans five years. **Note** After the RDS instance expires or is released, its cross-region backup files are still retained based on the cross-region backup retention period that you specify. You can log on to the ApsaraDB RDS console, click Backups in the left-side navigation pane, and then click the Cross-region Backup tab to view the cross-region backup files that are retained.
Cross-region Log Backup Status:	Specify whether to enable or disable cross-region log backups. After you enable cross-region log backups, the log backup files of the RDS instance are automatically replicated to a specified Object Storage Service (OSS) bucket in the destination region.

v. Click **OK**.

- **To modify the cross-region backup settings of an RDS instance, perform the following operations:**

  i. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Backups**. In the top navigation bar, select the region where the RDS instance resides.

  ii. On the Backups page, click the **Cross-region Backup** tab. Click the **Backup Instances** tab and find the RDS instance whose cross-region backup settings you want to modify. Then, click **Settings** in the Cross-region Backup Settings column to modify the cross-region backup settings of the RDS instance.

  > **Note** If the RDS instance is released, you can modify only the cross-region backup retention period.

- **To disable cross-region backups for an RDS instance, perform the following operations:**

  If you no longer require cross-region backups, you can disable cross-region backups.

i. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Backups**. In the top navigation bar, select the region where the RDS instance resides.

ii. On the Backups page, click the **Cross-region Backup** tab. Click the **Backup Instances** tab and find the RDS instance for which you want to disable cross-region backups. Then, click **Settings** in the Cross-region Backup Settings column.

iii. In the dialog box that appears, set the **Cross-region Backup Status** parameter to **Disabled** and set the **Cross-region Retention Period** parameter to **7**.

> **Note** After you disable cross-region backups, no new cross-region backup files are generated. However, the existing cross-region backup files are still retained for at least seven days. You can set the cross-region backup retention period to seven days. After the seven-day retention period that you specify elapses, all existing cross-region backup files are automatically deleted. Then, you are no longer charged for the storage of cross-region backup files.

iv. Click **OK**.

- **To download the cross-region data backup files of an RDS instance, perform the following operations:**

  After a cross-region backup is complete, you can download the generated cross-region backup files in the ApsaraDB RDS console.

  i. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Backups**. In the top navigation bar, select the region where the RDS instance resides.

  ii. On the Backups page, click the **Cross-region Backup** tab. Click the **Backup Instances** tab and click the ID of the RDS instance for which you want to download cross-region backup files.

  iii. On the **Data Backup** tab or the **Log Backup** tabs, click **Download** in the Actions column to download the full data backup file or the incremental backup file.

  iv. Click **Download**.

  > **Note** If you download data backup files over an internal network, the traffic is free of charge. If you download data backup files the Internet, the traffic is charged. For more information, see Network traffic fees.

Enable cross-region backups for an RDS instance by using DBS

In this example, the source region is China (Hangzhou), and the destination region is China (Beijing).

To create a cross-region backup for an RDS instance, perform the following operations:

To download the data backup files of an RDS instance, perform the following operations:

1. Log on to the DBS console.
2. In the left-side navigation pane, click **Backup Schedules**. In the upper-left corner of the Backup Schedules page, select the region where you purchase the backup schedule. In this example, select the China (Beijing) region.
3. Click the Schedule Name and go to the detail page.
4. Multiple methods are provided to download data backup files on the detail page. For more information, see Overview.

FAQ

After I disable cross-region backups for my RDS instance, why am I still charged for the storage of cross-region backup files?

After you disable cross-region backups for your RDS instance, no new cross-region backup files are generated and you are no longer charged for the traffic that is consumed to transmit cross-region backup files. However, you are still charged for the storage of the existing cross-region backup files within the cross-region backup retention period that you specify. The existing cross-region backup files are retained for at least seven days. You can set the cross-region backup retention period to seven days. After the cross-region backup retention period that you specify elapses, all existing cross-region backup files are automatically deleted and you are no longer charged for the storage of cross-region backup files.

Related API operations

Operation	Description
Check cross-region backup	Checks whether an ApsaraDB RDS instance has a cross-region data backup file that can be used to restore data across regions.
Restore data to a new instance across regions	Restores the data of an ApsaraDB RDS instance to a new RDS instance that resides in a different region than the source region.
Modify cross-region backup settings	Modifies the cross-region backup settings of an ApsaraDB RDS instance.
Query cross-region backup settings	Queries the cross-region backup settings of an ApsaraDB RDS instance.
Query cross-region data backup files	Queries the cross-region data backup files of an ApsaraDB RDS instance.
Query cross-region log backup files	Queries the cross-region log backup files of an ApsaraDB RDS instance.
Query regions that support cross-region backup	Queries the available destination regions to which the cross-region backup files from a specified source region can be stored.
Query the time range to which you can restore data by using a cross-region backup set	Queries the restorable time range that is supported by a specified cross-region backup file.
Query ApsaraDB for RDS instances on which cross-region backup is enabled	Queries the ApsaraDB RDS instances for which cross-region backups are enabled in a specified region and the cross-region backup settings of these instances.

14.5. Manage backups

14.5.1. Retain the backup files of an ApsaraDB RDS for MySQL instance for a long period of time

This topic describes how to retain the backup files of an ApsaraDB RDS for MySQL instance for a long period of time. The long-term retention of backup files helps you protect your data assets.

Background information

When you use ApsaraDB RDS, you may encounter the following issues:

- Your data is unexpectedly deleted. However, you have backed up only the data that was generated over the most recent seven days. As a result, you cannot restore the data that was generated seven days ago.
- Your RDS instance is released by mistake or due to overdue payments. However, it has been a long period of time since your RDS instance was released. As a result, you cannot recover your data or backup files.
- You want to release your RDS instance but retain its backup files for future use.

We recommend that you follow the instructions provided in this topic to retain the backup files of your RDS instance. This ensures that you can restore your data if required.

Increase the backup retention period

You can increase the backup retention period. This allows you to restore earlier data.

> **Note** This operation increases the size of your backup files. This operation may also increase the storage cost of your RDS instance. For more information, see View and manage the size of backup files for an ApsaraDB RDS for MySQL instance and Backup storage pricing of an ApsaraDB RDS for MySQL instance.

1. Log on to the ApsaraDB RDS console and go to the **Backup and Restoration** page of your RDS instance. On the **Backup Settings** tab, click **Edit**.
2. Increase the data backup retention period.
   - Specify the number of days for which you want to retain data backup files. For example, you can increase the number from the default value 7 to 30.
   - If your RDS instance is equipped with local SSDs, you can select **Long-term Retention**. After you select Long-term Retention, you do not need to specify the data backup retention period. All data backup files are retained until your RDS instance is released.
3. Increase the log backup retention period.

Retain data backup files after instance release

In most cases, after your RDS instance expires or becomes overdue, ApsaraDB RDS retains your RDS instance and its backup files based on the retention period that you specify. After the retention period elapses, ApsaraDB RDS releases your RDS instance and deletes its backup files. For more information, see Unlock or rebuild an expired or overdue ApsaraDB RDS instance.

If you set the **Data Backup Retention Policy After Release** parameter to Latest or All, ApsaraDB RDS retains your RDS instance and its most recent or all data backup files regardless of whether your RDS instance is automatically or manually released.

> **Note** This feature is supported only when your RDS instance is equipped with local SSDs.

1. Log on to the ApsaraDB RDS console and go to the **Backup and Restoration** page of your RDS instance. On the **Backup Settings** tab, click **Edit**.
2. Select **Data Backup Retention Policy After Release**.

> **Note**

Download a backup file

You can download a backup file of your RDS instance to your computer or to your Elastic Compute Service (ECS) instance. The backup file can be stored on your computer or ECS instance for a long period of time. For more information, see What is ECS? and Perform a regular download.

14.5.2. Download the backup files of an ApsaraDB RDS for MySQL instance

This topic describes how to download the backup files of an ApsaraDB RDS for MySQL instance.

For more information about how to download the backup files of an RDS instance that runs a different database engine, see the following topics:

- Download the data backup files and log backup files of an ApsaraDB RDS for SQL Server instance
- Download the data backup files and log backup files of an ApsaraDB RDS for PostgreSQL instance
- Download the log backup files of an ApsaraDB RDS for MariaDB TX instance

Before you begin

Before you download the backup files of your RDS instance, you must select a download method based on your business scenario.

> **Note** The data of the backup files that you download cannot be directly restored to your RDS instance. For more information, see Overview of data restoration methods.

Business scenario	Recommended download method

Business scenario	Recommended download method
• Migrate the data of your RDS instance to a different RDS instance or to a self-managed MySQL instance. • Restore the data of your RDS instance from a backup file.	Use one of the following methods: • Use Data Transmission Service (DTS) to migrate the data of your RDS instance to the destination RDS instance or the destination self-managed MySQL instance. For more information, see Overview of data migration scenarios. • Use Data Management (DMS) to export the data of your RDS instance. Then, import the data into the destination RDS instance. For more information, see Export data. • Restore the data of your RDS instance from the backup file to a new RDS instance. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance, Restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL database, and Restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance.
• Query the data that exists in your RDS instance at a specific point in time. • Query the data of a backup file of your RDS instance.	Use one of the following methods: • Restore the backup file to a new pay-as-you-go RDS instance. After the query is complete, release the pay-as-you-go RDS instance. For more information about how to restore data, see Restore the data of an ApsaraDB RDS for MySQL instance. • Obtain the URL that you can use to download the backup file. For more information, see the "Perform a regular download" section of this topic. Download the backup file, restore the backup file to a self-managed MySQL instance, and then query the data from the self-managed MySQL instance. • If the backup file is generated by Database Backup (DBS), you do not need to restore the backup file. Directly query the data from the backup file. For more information, see Query data in a single backup set.
• Analyze the data of your RDS instance. • Archive the data of your RDS instance to an on-premises device.	For more information, see Perform an advanced download
• Save a backup file of your RDS instance to an on-premises device. • Download a backup file of your RDS instance and use the backup file for auditing.	Use one of the following methods: • Obtain the URL that you can use to download the backup file. For more information, see the "Perform a regular download" section of this topic. Then, download the backup file. • Use DMS to export the backup file from your RDS instance. For more information, see Export data. • Use the mysqldump plug-in to back up and restore the data of your RDS instance. For more information, see How do I back up and restore the data of an ApsaraDB RDS for MySQL instance?

Business scenario	Recommended download method
• Upload a backup file of your RDS instance to an Object Storage Service (OSS) bucket.	Backup files do not consume the storage capacity of your RDS instance. Backup files are stored in the provisioned backup storage and cannot be moved. Use one of the following methods to store a backup file of your RDS instance to an OSS bucket: • Use DBS to back up your RDS instance. Make sure that you select an OSS bucket as your storage. For more information, see Configure a backup schedule. • Obtain the URL that you can use to download the backup file. For more information, see the "Perform a regular download" section of this topic. Then, download the backup file and upload it to the specified OSS bucket.
• Retain the backup files of your RDS instance for a long period of time.	• For more information, see Retain the backup files of an ApsaraDB RDS for MySQL instance for a long period of time
• Configure DBS to automatically download the backup files of your RDS instance.	• Use DBS to perform logical backups and configure DBS to automatically download the backup files that are generated. For more information, see Back up an ApsaraDB RDS for MySQL, PolarDB for MySQL, or self-managed MySQL database and Configure automatic download of backup sets.

Backup storage fee

- **ApsaraDB RDS provides a free quota for backup downloads over the Internet. Starting from 00:00 on November 25, 2021, you are charged for the excess traffic that you consume to download backup files over the Internet.** For more information,see [Notice] Upgrade and commercial release of the backup download feature.
  - Downloads over an internal network: You are not charged for the traffic that you consume.
  - Downloads over the Internet: After the free quota is exhausted, you are charged a daily fee for the excess traffic that you consume. For more information, see Billing overview.
  - Free quota for backup downloads over the Internet: The free quota for each RDS instance is 500 GB per month.
- **You are charged if you convert the backup files of your RDS instances.** This applies only when you perform advanced downloads. The advanced download feature is in public preview. In the public preview phase, advanced downloads are free of charge.

> **Note**
>
> To view the volume of traffic that you consumed to download backup files over the Internet, you can log on to the ApsaraDB RDS console, find your RDS instance, go to the **Basic Information** page, and then view the **Backup Downloads** parameter in the **Usage Statistics** section of the page.

Perform an advanced download

You can convert a backup file of your RDS instance into a CSV file and download the file to an on-premises device. This process is called advanced download. If you want to analyze data or archive data to an on-premises device, you can perform advanced downloads.

Prerequisites

- Your RDS instance runs one of the following MySQL versions:
  - MySQL 8.0 with standard SSDs or enhanced SSDs (ESSDs)
  - MySQL 5.7 with standard SSDs or ESSDs
- Your RDS instance resides in one of the following regions: China (Zhangjiakou), China (Guangzhou), China (Beijing), China (Shanghai), China (Hangzhou), and China (Shenzhen).

  > **Note** The feature will be available in the other regions soon.

- The minor engine version of your RDS instance must be a later version than 20201031. For more information, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.
- The disk encryption feature is not enabled for your RDS instance. RDS instances for which disk encryption is enabled do not support advanced downloads.
- The Resource Access Management (RAM) user that you want to use to log on to your RDS instance is granted the permissions to download backup files. For more information about how to grant permissions to a RAM user, see Grant backup file download permissions to a RAM user with read-only permissions.

> **Note** The advanced download feature is in public preview. You are not charged if you convert the backup files of your RDS instance. For more information, see Backup storage fee. In the public preview phase, the versions of RDS instances and formats of downloaded files that are supported by the advanced download feature may be adjusted. You may also optimize and adjust the limits of the advanced download feature.

Limits

- The advanced download feature allows you to export most schemas. Some schemas cannot be exported.
  - You can export column information, primary key indexes, non-primary key indexes, unique indexes, partition table information, table engines, and table-level or database-level character sets and character collations.
  - You cannot export expression indexes, foreign keys, generated columns, hidden columns, views, functions, stored procedures, system variables, or triggers.
- The advanced download feature does not support fields of spatial data types. If your RDS instance contains fields of the following spatial data types, file conversions fail:

  GEOMETRY, POINT, LINESTRING, POLYGON, MULTIPOINT, MULTILINESTRING, MULTIPOLYGON, and GEOMETRYCOLLECTION
- The CSV files that you export from your RDS instance do not contain the following system libraries:

  `information_schema`, `mysql`, `performance_schema`, `sys`, and `__recycle_bin__`

Procedure

1.
2. In the left-side navigation pane, click **Backup and Restoration**.

3. On the **Data Backup** tab of the page that appears, find the data backup file that you want to download, and click **Download Instance Backup** in the **Actions** column.

   > **Note** By default, the ApsaraDB RDS console displays the backup files that were generated over the most recent eight days. If you want to view the backup files that were generated eight days ago, you must change the default time range.

4. In the **Select Download Mode** step of the wizard, click **Next**.

   > **Note** Only the **Download By Backup Set** option is provided.

5. In the **Select Download Content** step of the wizard, click **Next**.

   > **Note** Only the **Download Instances** option is provided.

6. In the **Select Download Destination and Format** step of the wizard, specify the **Download Destination** and **Download Format** parameters and click **Complete**.

   > **Note**
   > - **Download Destination**: Select the method that you want to use to download the data backup file. You can download backup files only by using URLs.
   > - **Download Format**: Select the format into which you want to convert the data backup file. You can convert data backup files only into CSV files.

7. On the **Backup Download** tab to which you are directed, view the **Status** of the data backup file. After the status changes from **Running** to **Finished**, download the data backup file by using the internal or external URL that is displayed in the **Download Destination** column.

   > **Note**
   > - For more information about how to download data backup files, see Appendix: Download commands.
   > - When you perform an advanced download, you are charged a file conversion fee and a traffic consumption fee. In the public preview phase, file conversions and downloads over internal networks are free of charge. Downloads over the Internet are charged. For more information about the fees that are charged to you, see Backup storage fee.

8. Restore the data of the data backup file to a self-managed MySQL instance.

   Limits

   When you restore the data of a CSV file that you downloaded by using the advanced download feature to a self-managed MySQL instance, take note of the following limits:

   - Fields of the following data types are not supported: BIT, BINARY, VARBINARY, TINYBLOB, BLOB, MEDIUMBLOB, and LONGBLOB.

> **Note** If the CSV file that you downloaded contains fields of the BINARY data type, the fields are stored as hexadecimal representations. When you import the data of the CSV file into the self-managed MySQL instance, the fields that are stored as hexadecimal representations are processed as strings. In this case, you must use the UNHEX function in the **load data local infile** command to convert the fields from hexadecimal strings into raw binary strings.

- Fields of the following spatial data types are not supported: GEOMETRY, POINT, LINESTRING, POLYGON, MULTIPOINT, MULTILINESTRING, MULTIPOLYGON, and GEOMETRYCOLLECTION.

Procedure

i. Download and decompress the data backup file. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL database.

ii. Click Download to download the restore_from_downloads.py script file.

iii. Grant execute permissions to the `restore_from_downloads.py` script file.

```
chmod +x ./restore_from_downloads.py
```

iv. On the server that runs a Linux operating system, run the following command to restore the data of the data backup file:

```
python ./restore_from_downloads.py <Save path of the decompressed data backup file>
<Host on which the self-managed MySQL instance is deployed> <Port that is used to c
onnect to the self-managed MySQL instance> <Username of the account that is used to
connect to the self-managed MySQL instance> <Password of the account that is used t
o connect to the self-managed MySQL instance>
```

> **Note**
>
> - If a database in the self-managed MySQL instance has the same name as the database whose data you want to import, the import fails.
> - If the username or password of the account that you want to use contains special characters, such as number signs (#) and spaces, you must enclose the username or password in double quotation marks (" "). For example, if the password is `#1234`, you must enter `"#1234"` in the preceding command.

Perform a regular download

Prerequisites

- Your RDS instance uses local SSDs.
- The RAM user that you want to use to log on to your RDS instance is granted the permissions to download backup files. For more information about how to grant permissions to a RAM user, see Grant backup file download permissions to a RAM user with read-only permissions.

Procedure

1.

2. In the left-side navigation pane, click **Backup and Restoration**.
3. Click the **Data Backup** tab or the **Log Backup** tab.
4. Select a time range. This step is required if you want to view the backup files that are generated eight days ago. The default time range spans the most recent eight days.
5. Find the backup file that you want to download. In the **Actions** column, click **Download Instance Backup** (on the Data Backup tab) or **Download** (on the Log Backup tab).
   - When you download a **data backup file**, copy the internal or external URL or click **Download** in the dialog box that appears to download the data backup file.

     > **Notice**
     > - If you use the internal URL to download the data backup file, make sure that the server to which you log on and the RDS instance reside in the same virtual private cloud (VPC). If the server and the RDS instance reside in different VPCs or if the server resides in the classic network while the RDS instance resides in a VPC, you cannot download the data backup file by using the internal URL on the server.
     > - **I have learnt the billing rules for backup file download.** is selected by default. If you use the external URL to download the data backup file, you are charged for the excess Internet traffic that you consume. For more information,see Backup storage fee.
     > - For more information about how to download data backup files by using **URLs**, see Appendix: Download commands.

   - When you download a **log backup file**, copy the internal or external URL or click **Download** in the dialog box that appears to download the backup file.
     - **Copy Internal URL**: If your Elastic Compute Service (ECS) instance can communicate with the RDS instance over an internal network, you can log on to your ECS instance and use the internal URL to download the log backup file. This method is faster and more secure.
     - **Copy Public URL**: If the RDS instance cannot be connected over an internal network, you can use the public URL to download the log backup file.

Notice

- If you want to use the log backup file to restore the data of the RDS instance to a self-managed MySQL instance, the point in time at which the used data backup file is generated must be within the restorable time range that is supported by the log backup file. In addition, the log backup file and the data backup file must have the same instance ID.
- If the RDS instance runs RDS High-availability Edition or RDS Enterprise Edition, both the primary RDS instance and the secondary RDS instance generate log backup files. You can view the IDs of the primary RDS instance and the secondary RDS instance on the **Service Availability** page.
- The log backup files that are described in this section are the log backup files that are stored on the RDS instance. For more information about how to obtain these log backup files, see How do I use the mysqlbinlog command to view the binary logs of an ApsaraDB RDS for MySQL instance?
- If you use the internal URL to download the data backup file, make sure that the server to which you log on and the RDS instance reside in the same VPC. If the server and the RDS instance reside in different VPCs or if the server resides in the classic network while the RDS instance resides in a VPC, you cannot download the data backup file by using the internal URL on the server.
- **I have learnt the billing rules for backup file download.** is selected by default. If you use the external URL to download the data backup file, you are charged for the excess Internet traffic that you consume. For more information,see Backup storage fee.

Appendix: Download commands

This section describes the commands that are commonly used to download backup files.

Note

- If the speed of a download is lower than 64 KB per second, the download may be interrupted. When you download a backup file, we recommend that you ensure optimal network status.
- If you want to download a backup file to a disk that is attached by using the ossfs plug-in, you must adjust the value of the **multipart_size** parameter for the ossfs plug-in. The maximum value of this parameter is 100 GB. If the size of the data backup file that you want to download exceeds 100 GB, the download fails. For more information about the ossfs plug-in and its parameter settings, see Overview and Common options.
- We recommend that you use the commands, such as wget and curl, that are provided in this topic to download a data backup file. If you use third-party tools to download a data backup file, the data backup file may be downloaded multiple times. As a result, the amount of data that is downloaded is greater than the size of the data backup file, and you may be charged for the traffic that is consumed to download the excess amount of data over the Internet.

◉ wget ○ curl

Command: nohup wget -c -t 0 "The URL that is used to download the backup file" -O The save path

and name of the downloaded backup file> The file to which the downloaded data is saved &

Example: nohup wget -c -t 0 "https://example.aliyundoc.com/examplebackup.qp.xb" -O /backup/examplebackup.qp.xb > /tmp/download.log &

The following information provides the meanings of the options in the preceding command:

- -t 0: enables an unlimited number of retries.
- -c: enables resumable uploads.
- -O: specifies the save path and file name of the backup file after the backup file is downloaded.
- nohup: prevents interruptions to the download and specifies that the process automatically exits after the download is complete. If you accidentally replicate data or disconnect your database client during the download, the download is interrupted.

Command: nohup curl -C - --retry 10 "The URL that is used to download the backup file" -o The user-defined name of the downloaded backup file> The file to which the downloaded data is saved &

Example: nohup curl -C --retry 10 "https://example.aliyundoc.com/examplebackup.qp.xb" -o backup.qp.xb > /tmp/download.log &

The following information provides the meanings of the options in the preceding command:

- --retry 10: enables up to 10 retries if the download fails.
- -C -: enables resumable uploads.
- -o: specifies the save path and file name of the backup file after the backup file is downloaded.
- nohup: prevents interruptions to the download and specifies that the process automatically exits after the download is complete. If you accidentally replicate data or disconnect your database client during the download, the download is interrupted.

FAQ

How do I use the data backup files and log backup files that I downloaded?

You can restore the data of the backup files that you downloaded by using the advanced download feature to self-managed MySQL instances. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL database or Restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance.

My RDS instance runs standard SSDs or ESSDs. What do I do if the "`ERROR 1148 (42000): The used command is not allowed with this MySQL version`" error message appears when I restore the data of a backup file that I downloaded by using the advanced download feature to a self-managed MySQL instance?

Run the `show variables like 'local_infile';` command on the self-managed MySQL instance. If the output is OFF, execute the SET statement to set the `global local_infile` parameter to 1 to enable file import. After the preceding operations are complete, run the import script again.

Related operations

Operation	Description
DescribeBackups	Queries the data backup files of an ApsaraDB RDS instance.

Operation	Description
DescribeBinlogFiles	Queries the log backup files of an ApsaraDB RDS instance.

14.5.3. Delete the backup files or reduce the backup frequency of an ApsaraDB RDS for MySQL instance

This topic describes how to delete the backup files or reduce the backup frequency of an ApsaraDB RDS for MySQL instance. For more information, see Backup storage pricing of an ApsaraDB RDS for MySQL instance.

Precautions

The data backup files and log backup files of your RDS instance consume the backup storage that is provided by Alibaba Cloud to the instance. These backup files do not consume the storage space of your RDS instance. For more information about how to release storage space, see My ApsaraDB RDS for MySQL instance automatically locks after its disk space is exhausted. What do I do?

Delete or reduce data backups

- **Manually delete data backup files.**
  i.
  ii. Go to the **Backup and Restoration** page.
  iii. On the **data backup** tab, click **delete** for the target backup.

  > **Note** If no **delete** button exists, the following conditions are not met:
  > - When log backup is disabled, you can only manually delete data backups that have been retained for more than seven days.
  > - When log backup is enabled, you can only manually delete data backups beyond the log backup retention period. For example, if the log backup retention period is seven days, you can delete data backups that have been retained for more than seven days.

- **automatically delete data backups (shorten the retention period of data backups)**
  i.
  ii. Go to the **Backup and Restoration** page.
  iii. On the **Backup settings** tab, open the Settings dialog box to shorten the retention period of data backup.

  PolarDB-X deletes the backup files that are stored longer than the retention period. For example, if your instance keeps backups for 31 days, and you change the retention period from 30 days, the backups that have been retained for 31 days are automatically deleted.

- **reduce the number of data backups**

i.

ii. Go to the **Backup and Restoration** page.

iii. On the **Backup settings** tab, open the Settings dialog box to decrease the cycle for data backup.

- **delete or migrate data that does not need backup**.

Delete or reduce log backups

- **Shorten log backup retention period**

  i.

  ii. Go to the **Backup and Restoration** page.

  iii. On the **Backup settings** tab, open the Settings dialog box to shorten the log backup retention period.

  For example, if you set the log backup retention period to 30 days, log backups that have been retained for more than 30 days are automatically deleted.

  > **Note** You can set the log backup retention period only when log backup is enabled.

- **disable log backup**

  If you disable it, all log backups are automatically deleted and the restore by time point feature is unavailable.

  i.

  ii. Go to the **Backup and Restoration** page.

  iii. On the **Backup settings** tab, open the Settings dialog box and disable log backup.

- **reduces unnecessary additions, deletions, and modifications, and updates on large fields**.

  Any addition, deletion, and modification of databases can increase the log backup size.

  > **Note** You can use the Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance function to view the add, delete, modify, and query records of the database.

Related operations

Operation	Description
DeleteBackup	Deletes one or more data backup files from an ApsaraDB RDS instance.
ModifyBackupPolicy	Modifies the automatic backup settings of an ApsaraDB RDS instance.

14.5.4. Query data from DBS-generated logical backup files

When you query data from the data backup files of an ApsaraDB RDS for MySQL instance, Alibaba Cloud creates an RDS instance, copies the data backup files to the new RDS instance, and then restores the data of the data backup files to the new RDS instance. This process is time-consuming. If the backup files are created by Database Backup (DBS), you can query data from the data backup files without the need to restore the data. This topic describes how to query data from the logical backup files that are created by DBS.

Prerequisites

- A logical backup is created by DBS. For more information, see Back up an ApsaraDB RDS for MySQL, PolarDB for MySQL, or self-managed MySQL instance by using logical backup.
- Data Lake Analytics (DLA) is activated. For more information, see Activate DLA.

Scenarios

DLA supports real-time queries from full backup files. This relieves the need to restore data and reduces costs.

Precautions

- DLA supports only the queries of data from the data backup files that are created by DBS for ApsaraDB RDS for MySQL instances.
- DLA supports only the queries of data from full backup files. DLA does not support the queries of data from incremental backup files.
- DLA must reside in the same region as the Object Storage Service (OSS) bucket that stores the data backup files of your RDS instance.
- DLA supports only the queries of data from logical backup files.

Procedure

1. Configure the root account, endpoint, and OSS access permissions on DLA.
   - For more information about how to configure the endpoint, see Create an endpoint.
   - To configure the OSS access permissions, perform the following operations:

     > **Note** If OSS access permissions are configured, you can skip these operations.

     a. Log on to the DLA console.
     b. In the left-side navigation pane, choose **Data Lake Management > Metadata management**.
     c. Click **Go to the Wizard** in **OSS data source**.
     d. Click **Click Here to Authorize** next to **Role Name OSS Access Authorization Role AliyunOpenAnalyticsAccessingOSSRole**.
2. Create a schema on DBS.
   i. Log on to the Database Backup console.
   ii. In the left-side navigation pane, click **Backup Schedules**.
   iii. Find the backup schedule that you want to use. Then, click the ID of the backup schedule in the **Schedule ID/Name** column or click **Manage** in the **Actions** column.
   iv. In the left-side navigation pane, choose **Backup Tasks > Full Data**.

v. Find the data backup file that you want to use. In the Actions column, click **Query Backup Set**. In the **Query Backup Data** message, click **OK**.

> **Note** After you click **OK**, DLA automatically creates a schema for the data backup file.

3. Query data from the full backup file that is created by DBS.
   i. Log on to the DLA console.
   ii. In the left-side navigation pane, choose **Serverless Presto > SQL access point**.
   iii. On the **SQL access point** page, click **Log on in DMS**.
   iv. In the Login instance dialog box, enter the information that is used to log on to your RDS instance and click **Login**.

> **Note** Data Management (DMS) automatically specifies the **Database Type**, **Instance Region**, and **Connection string address** parameters. You must confirm the settings of these parameters and enter the username and password that are used to log on to your RDS instance.

   v. Execute the following SQL statements on DLA and your RDS instance to check whether the data volume on DLA is the same as the data volume on your RDS instance:

```
select 'bill' as tableName ,count(id) as countNumber from `bill`
union ALL
select 'dim_code_desc' as tableName ,count(id) as countNumber from `dim_code_desc`
;
```

   vi. Execute the following SQL statement on DLA to run a multi-table join query:

```
select  t.* from dim_code_desc   as t1,  BILL t
where   t1.id= t.id
and t1.code_id like '9%';
```

   Run a multi-table join query on your RDS instance. Then, compare the query result from DLA and the query result from your RDS instance.

   Verify that the query result from DLA is the same as the query result from your RDS instance.

   In this example, if ApsaraDB RDS clones your RDS instance to create an RDS instance, restores data from a full backup file to the new RDS instance, configures an IP address whitelist, and then returns the data that you query, about 1 hour is required and the query process is complicated. The combination of DBS and DLA relieves the need to restore data. In addition, this combination allows you to check for and recover a small amount of data that is accidentally deleted.

14.6. Introduction to binary log files and log backup files of an ApsaraDB RDS for MySQL instance

This topic describes the binary log files and log backup files of an ApsaraDB RDS for MySQL instance.

Overview

- Binary log files are used to subscribe to data and build a primary/secondary architecture. The binary logging feature is enabled by default and cannot be disabled.
- Log backup files can be used to restore data to a specific point in time within the backup retention period that you specify.

Binary log files

Binary log files are generated on RDS instances in real time. The rules based on which binary log files are generated varies based on the RDS edition of your RDS instance:

- RDS Basic Edition or RDS High-availability Edition: When the size of a binary log file reaches or the amount of time during which binary log data continues to be written to a binary log file exceeds 6 hours, a new binary log file is generated. 512 MB
- RDS Enterprise Edition: When the size of a binary log file reaches , a new binary log file is generated. 512 MB

The binary logging feature is provided free of charge. However, binary log files consume the storage capacity of your RDS instance.

Operations:

- View binary log files: You cannot view binary log files in the ApsaraDB RDS console. To view binary log files, you must execute the `SHOW BINARY LOGS;` statement. For more information, see ApsaraDB RDS for MySQL remotely obtains and parses binary log files.
- View the total size of binary log files:
  i.
  ii. Log on to the ApsaraDB RDS console, find your RDS instance, and then click the instance ID. In the left-side navigation pane, click **Monitoring and Alerts**. On the Standard Monitoring tab of the page that appears, view the total size of binary log files.

  > **Note** By default, the total size of binary log files on the primary RDS instance is the same as the size of binary log files on the secondary RDS instance. The sizes may differ due to the following reasons: The primary RDS instance replicates data to the secondary RDS instance at a specific latency, the binary log files are dumped at a specific latency, Data Transmission Service (DTS) is migrating data from or to the primary RDS instance, or the secondary RDS instance is reading and applying binary log records.
- Delete binary log files: For more information, see Delete the binary log files of an ApsaraDB RDS for MySQL instance.

Log backup files

ApsaraDB RDS supports data backups and log backups. If you enable the log backup feature for your RDS instance, the binary log files of your RDS instance are uploaded to dedicated backup storage in real time. The binary log files in the dedicated backup storage are called log backup files. You can use log backup files to restore data to a specific point in time within the backup retention period that you specify.

Log backup files are stored in dedicated backup storage and do not consume the storage capacity of your RDS instance. The dedicated backup storage is inaccessible.

Pricing: The log backup feature is provided free of charge, and a free quota for backup storage is provided. If the total size of the **data backup files and log backup files** of your RDS instance exceeds the free quota, you are charged for your excess backup storage.

Operations:

- Enable or disable the log backup feature:
  i.
  ii. In the left-side navigation pane, click **Backup and Restoration**.
  iii. Click the **Backup Settings** tab. In the Local Log Backup Settings section of the tab that appears, click **Edit**.
  iv. Enable or disable the log backup feature. You can also specify the log backup retention period.

  > **Note** If your RDS instance runs MySQL 5.7 on RDS Basic Edition, you cannot disable the log backup feature for your RDS instance and the log backup retention period is fixed as seven days.

- View log backup files:
  i.
  ii. In the left-side navigation pane, click **Backup and Restoration**.
  iii. Click the **Log Backup** tab.

  If your RDS instance runs RDS High-availability Edition or RDS Enterprise Edition, log backup files are generated on both your RDS instance and its secondary RDS instance. You can check the instance ID of each log backup file to distinguish the log backup files that are generated by your RDS instance from the log backup files that are generated by the secondary RDS instance.

- View the total size of log backup files:
  i.
  ii. In the lower-right corner of the Basic Information page, view the total size of log backup files.

  > **Note** Log backup files are stored in dedicated backup storage and do not consume the storage capacity of your RDS instance.

- Delete log backup files. For more information, see Delete the backup files or reduce the backup frequency of an ApsaraDB RDS for MySQL instance.
- Download log backup files. For more information, see Download the backup files of an ApsaraDB RDS for MySQL instance.
- Restore data to a specific point in time. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance or Restore individual databases and tables of an ApsaraDB RDS for MySQL instance.

FAQ

FAQ about how to view log backup files

1. Why are no log backup files displayed on the Log Backup tab of my RDS instance?

This issue occurs due to one of the following reasons:

- The log backup feature is not enabled for your RDS instance.
- Your RDS instance is recently created, or the log backup feature is recently enabled for your RDS instance. In this case, log backup files have not been uploaded to the backup storage, and no log backup files are displayed on the Log Backup tab.

2. Why is the most recent log backup file not displayed on the Log Backup tab?

   Only the log backup files to which no new data is written are displayed on the Log Backup tab. Log backup files to which data is being written are not uploaded to the backup storage.

3. If I set the log backup retention period to seven days, can I obtain log backup files that are generated seven days ago?

   No, if you set the log backup retention period to seven days, you cannot obtain log backup files that are generated seven days ago. ApsaraDB RDS automatically deletes the log backup files that are stored for a longer period of time than the specified log backup retention period. We recommend that you specify a log backup retention period based on your business requirements.

FAQ about how to disable the log backup feature and delete log backup files

1. How do I delete log backup files after I disable the log backup feature?

   You do not need to delete log backup files. ApsaraDB RDS automatically deletes log backup files immediately after you disable the log backup feature.

2. Why am I still able to query log backup files by executing the `SHOW BINARY LOGS;` statement after I disable the log backup feature?

   The files that you query by executing this statement are binary log files rather than log backup files. For more information about how to delete binary log files, see Delete the binary log files of an ApsaraDB RDS for MySQL instance.

3. Why is no storage released after I disable the log backup feature?

   Log backup files do not consume the storage capacity of your RDS instance. Binary log files consume the storage capacity.

4. Why does the size of log backup files suddenly increase?

   If you frequently perform add, delete, and update operations on the data in your RDS instance, the size of log backup files increases. For more information about how to delete log backup files, see Delete the backup files or reduce the backup frequency of an ApsaraDB RDS for MySQL instance.

14.7. ApsaraDB RDS-generated backups and DBS-generated backups

ApsaraDB RDS for MySQL can generate backups, which are called default backups. If the default backups that are generated by ApsaraDB RDS for MySQL cannot meet your business requirements, you can use Database Backup (DBS) to create advanced backups.

Scenario	Item	ApsaraDB RDS-generated backup (physical backup or snapshot backup)	DBS-generated backup (logical backup)	References
Perform backups.	Perform scheduled backups.	✔ (Scheduled backups are automatically enabled and cannot be disabled.)	✔ (Scheduled backups must be manually enabled.)	Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance
	Perform incremental backups or log backups.	✔	✔	
	Backs up individual databases and tables.	• Local SSDs: ✔. Standard SSDs or enhanced SSDs (ESSDs): □ • Manually enabled: ✔. Enabled at a scheduled point in time: □ • Incremental backups: □	✔	Back up the individual databases and tables of an ApsaraDB RDS for MySQL instance
	Perform cross-region backups.	Local SSDs: ✔. Standard SSDs or ESSDs: □	✔	Enable cross-region backups for an ApsaraDB RDS for MySQL instance
Store backup files.	Store backup files to your Object Storage Service (OSS) buckets.	□	✔	Back up an ApsaraDB RDS for MySQL, PolarDB for MySQL, or self-managed MySQL instance by using logical backup
	Support multi-level storage pools.	□	✔	What is a storage pool?
Download backup files.	Manually download backup files.	Local SSDs: ✔. Standard SSDs or ESSDs: □	✔	Download the backup files of an ApsaraDB RDS for MySQL instance
	Automatically download backup files.	□	✔	

Scenario	Item	ApsaraDB RDS-generated backup (physical backup or snapshot backup)	DBS-generated backup (logical backup)	References
Encrypt backup files.	Encrypt backup files at rest.	✔	✔	Encrypt the backup files of an ApsaraDB RDS for MySQL instance
	Encrypt backup files in transit.	□	✔	
Restore data by using backup files.	**Restore data in single-digit seconds.**	□	Local SSDs: ✔	Create a sandbox instance for the emergency disaster recovery of an ApsaraDB RDS for MySQL database
	Query backup files at a high speed.	□	✔	Query data from DBS-generated logical backup files
	Restore individual databases and tables.	Local SSDs: ✔. Standard SSDs or ESSDs: □	✔	Restore individual databases and tables of an ApsaraDB RDS for MySQL instance Restore individual databases and tables by using DBS

Scenario	Item	ApsaraDB RDS-generated backup (physical backup or snapshot backup)	DBS-generated backup (logical backup)	References
	Restore data to an RDS instance by using backup files.	• Restore data to a new RDS instance by using backup files: ✔☺ • Restore data to the original RDS instance by using backup files: local SSDs: ✔☺. Standard SSDs or ESSDs: □ • Restore data to a different existing RDS instance by using backup files: □ • Restore data to a self-managed database by using backup files: □	✔☺	Restore the backup files of an ApsaraDB RDS for MySQL instance Restore the backup files by using DBS

15.Restoration

15.1. Overview of data restoration methods

This topic describes the methods that you can use to restore the data of an ApsaraDB RDS for MySQL instance.

Scenarios

Scenario	Method
Restore the data of an RDS instance that is accidentally released	• **Locked Instances**: Log on to the ApsaraDB RDS console and go to the Locked Instances page. In the top navigation bar, **select the region where the RDS instance resides**. If you can find the RDS instance on the Locked Instances page, you can rebuild the RDS instance. • **Backups**: If the RDS instance uses local SSDs and you specified the **Backup Retention Policy After Release** parameter on the Backup Settings tab to configure the RDS instance to retain backup files, log on to the ApsaraDB RDS console, go to the Backup for Deleted Instances tab of the Backups page, download the required backup files, and then restore the data of the RDS instance.
Restore the data that is accidentally deleted from an RDS instance	• Restore the data of an ApsaraDB RDS for MySQL instance • Restore individual databases and tables of an ApsaraDB RDS for MySQL instance • Use the data tracking feature of DMS to restore the data of an ApsaraDB RDS for MySQL instance

<table>
<tr><th colspan="2">Scenario</th><th>Method</th></tr>
<tr><td colspan="2">• View the data in the data backup files of an RDS instance
• View the data that exists at a specified point in time in an RDS instance</td><td>• You can use the native flashback feature to view the data that exists at a specified point in time in the RDS instance. For more information, see Native Flashback. This method requires a short period of time to restore data.
• If logical backups are created by Database Backup (DBS) for the RDS instance, query the data from the generated logical backup files. For more information, see Query data from DBS-generated logical backup files. This method requires a short period of time to restore data.
• Create a DBS sandbox instance to obtain the data backup files of the RDS instance and restore the data of the RDS instance from the data backup files. For more information, see Create a sandbox instance for the emergency disaster recovery of an ApsaraDB RDS for MySQL database. Then, view the data of the RDS instance. This method requires a short period of time to restore data.
• Restore the full data of the RDS instance or only the data from the specified individual databases and tables of the RDS instance. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance or Restore individual databases and tables of an ApsaraDB RDS for MySQL instance. Then, view the data of the RDS instance. Compared with the previous two methods, this method requires a longer period of time to restore data.</td></tr>
<tr><td rowspan="2">• Migrate the data of an</td><td>Migrate or restore the data of an RDS instance to the cloud</td><td>• Migrate the most recent data of the RDS instance. For more information, see the following topic:
Migrate data from a self-managed MySQL database to an ApsaraDB RDS for MySQL instance
• Restore the data that exists at a specified point in time
Native Flashback
• Migrate the historical data of the RDS instance:
◦ Restore the data from the backup files of the RDS instance to an on-premises database. Then, migrate the data from the on-premises database to an RDS instance. For more information, see Migrate data from a self-managed MySQL database to an ApsaraDB RDS for MySQL instance.
◦ Use DBS to create a logical backup. Then, restore the data from the generated logical backup file to an RDS instance on the cloud. For more information, see Restore a MySQL database from a logical backup.</td></tr>
<tr><td></td><td></td></tr>
</table>

Scenario	RDS instance	Method

• Restore the data from the backup files of an RDS instance to an on-premises database or to the cloud	Migrate or restore the data of an RDS instance to an on-premises database	• Migrate the most recent data of the RDS instance. For more information, see the following topic:
		  ○ Migrate data from an ApsaraDB RDS for MySQL instance to a self-managed MySQL database
		• Migrate the historical data of the RDS instance:
		  ○ If logical backups are created by DBS for the RDS instance, restore the data from the generated logical backup files to the on-premises database. For more information, see Restore a MySQL database from a logical backup.
		  ○ If you restore the data of the RDS instance from the backup files that are downloaded from the ApsaraDB RDS console, follow the instructions that are provided in Restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL database or Restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance.

	Migrate the data of an RDS instance to a different RDS instance	• Migrate the most recent data of the RDS instance:
		  ○ Migrate data between ApsaraDB RDS for MySQL instances
		• Migrate the historical data of the RDS instance:
		  ○ If logical backups are created by DBS for the RDS instance, restore the data from the generated logical backup files to the destination RDS instance. For more information, see Restore a MySQL database from a logical backup.
		  ○ If only default backups are created for the RDS instance, restore the full data of the RDS instance or only the data from the specified individual databases and tables of the RDS instance to the same RDS instance or to a new RDS instance. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance or Restore individual databases and tables of an ApsaraDB RDS for MySQL instance. Then, migrate the data to the destination RDS instance.

More scenarios

- Restore the data of an ApsaraDB RDS for MySQL instance across regions
- Use mysqldump to back up and restore the data of an RDS instance

15.2. Restore the data of an ApsaraDB RDS for MySQL instance

This topic describes how to restore the data of an ApsaraDB RDS for MySQL instance.

For more information about how to restore the data of an RDS instance that runs a different database engine, see the following topics:

- Restore the data of an ApsaraDB RDS for SQL Server instance
- Restore the data of an ApsaraDB RDS for PostgreSQL instance
- Restore the data of an ApsaraDB RDS for MariaDB TX instance

Restore data to a new RDS instance

You can restore data to a new RDS instance, verify the data on the new RDS instance, and then migrate the data from the new RDS instance back to the original RDS instance. This feature was previously known as instance cloning.

Description

- The new RDS instance must have the same whitelist settings, backup settings, and parameter settings as the original RDS instance.
- The data in the new RDS instance must be the same as the data in the specified data or log backup file of the original RDS instance.
- The information about the account that is used to create the specified data or log backup is replicated to the new RDS instance.

Time required for the restoration

For more information about the amount of time that is required for the restoration, see Appendix: Amount of time required for data restoration to a new RDS instance.

Billing

You are charged for the new RDS instance. We recommend that you create a new pay-as-you-go RDS instance. After the restoration is complete, you can release the new RDS instance.

Procedure

1.
2. In the left-side navigation pane, click **Backup and Restoration**.
3. Click **Restore Database (Previously Clone Database)**.
4. Configure the following parameters.

Parameter	Description
Billing Method	○ **Subscription**: A subscription instance is an instance for which you pay an upfront fee. For long-term use, the subscription billing method is more cost-effective than the pay-as-you-go billing method. You are offered lower prices for longer subscription periods. ○ **Pay-As-You-Go**: A pay-as-you-go instance is charged per hour based on your actual resource usage. For short-term use, we recommend that you select the pay-as-you-go billing method. If you no longer need a pay-as-you-go instance, you can release the instance to reduce costs.

Parameter	Description
Restore Mode	○ **By Time**: This mode allows you to restore data to a specific point in time within the log backup retention period that you specify. The time is accurate to the second. For more information about how to view or change the log backup retention period, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance. ○ **By Backup Set**: This mode allows you to restore data from a data backup file. You can restore data only from the data backup file that is generated from a physical backup. You cannot restore data from the data backup file that is generated from a logical backup. **Note** The **By Time** option is available only after the log backup feature is enabled.
Zone of Primary Node	A zone is an independent physical location within a region. The **Zone of Primary Node** parameter specifies the zone to which the primary RDS instance belongs. The **Zone of Secondary Node** parameter specifies the zone to which the secondary RDS instance belongs. You can select the **Single-zone Deployment** or **Multi-zone Development** method. ○ **Single-zone Deployment**: If you select this deployment method, the values of the **Zone of Primary Node** and **Zone of Secondary Node** parameters are the same. ○ **Multi-zone Development**: If you select this deployment method, the values of the **Zone of Primary Node** and **Zone of Secondary Node** parameters are different. We recommend that you select this deployment method to perform zone-disaster recovery. You must manually configure the **Zone of Primary Node** and **Zone of Secondary Node** parameters. **Note** ○ After the new RDS instance is created, you can view the information about the new RDS instance and its secondary RDS instance on the **Service Availability** page. ○ If you select RDS Basic Edition, the database system consists of only one primary RDS instance and supports only the single-zone deployment method.

Parameter	Description
Instance Type	○ **General-purpose**: A general-purpose instance exclusively occupies the allocated memory and I/O resources. However, it shares CPU and storage resources with the other general-purpose instances that are deployed on the same host. ○ **Dedicated**: You can select a dedicated instance type or a dedicated host instance type. A dedicated RDS instance exclusively occupies the allocated CPU, memory, storage, and I/O resources. Dedicated host instance types provide the highest specifications in the dedicated instance family. A dedicated host instance exclusively occupies all the CPU, memory, storage, and I/O resources on the physical host on which the instance is deployed. **Note** Each instance type supports a specific number of cores, memory capacity, maximum number of connections, and maximum IOPS. For more information, see Primary ApsaraDB RDS instance types.
Capacity	The maximum amount of storage that is provisioned to store data files, system files, binary log files, and transaction files in the new RDS instance. You can adjust the storage capacity at a step size of 5 GB.

5. Click **Next: Instance Configuration**.
6. Configure the following parameters.

Parameter	Description
Network Type	○ **Classic Network**: the traditional type of network. ○ **VPC**: the recommended type of network. A virtual private cloud (VPC) is an isolated virtual network that provides higher security and higher performance than the classic network. If you select the VPC network type, you must configure the **VPC** and **vSwitch of Primary Node** parameters. If you set the **Deployment Method** parameter in the previous step to **Multi-zone deployment**, you must also configure the **vSwitch of Secondary Node** parameter. **Note** The network type of the new RDS instance must be the same as the network type of the Elastic Compute Service (ECS) instance that you want to connect. If the new RDS instance and the ECS instance both reside in VPCs, these instances must reside in the same VPC. If the new RDS instance and the ECS instance reside in different VPCs, these instances cannot communicate over an internal network.

7. Click **Next: Confirm Order**.
8. Confirm the settings in the **Parameters** section, configure the **Purchase Plan** and **Duration** parameters, read and select Terms of Service, click **Pay Now**, and then complete the payment. You must configure the Duration parameter only when the new RDS instance uses the subscription billing method.

> **Note** If you select the subscription billing method for the new RDS instance, we recommend that you select **Auto-Renew Enabled** below the Duration parameter. This eliminates the need to renew the new RDS instance on a regular basis. This also helps prevent interruptions to your workloads on the new RDS instance if a payment is overdue.

What to do next

1. Log on to the new RDS instance and verify the data in the new RDS instance. For more information, see Connect to an ApsaraDB RDS for MySQL instance.
2. Optional. Migrate the data that you need from the new RDS instance back to the original RDS instance. For more information, see Migrate data between RDS instances.

> **Note** The migration of data from the new RDS instance to the original RDS instance does not interrupt the workloads that are run on the original RDS instance.

Restore data to the original RDS instance, a different existing RDS instance, or an on-premises database instance

Method 1

After you restore data to the new RDS instance, verify the data on the new RDS instance. Then, migrate the data from the new RDS instance to the original RDS instance or a different existing RDS instance. For more information, see Restore data to a new RDS instance.

Method 2

If a logical backup is created by Database Backup (DBS), restore the data of the data backup file that is generated to the original RDS instance or a different existing RDS instance.

For more information, see Restore a MySQL database from a logical backup.

Appendix: Amount of time required for data restoration to a new RDS instance

Factors

The restoration speed varies based on a number of factors, and the restoration may fail in a few circumstances. You may also need to manually troubleshoot the errors that occur due to the executions of SQL statements. The following factors affect the restoration speed:

- Volume of full data: A larger data volume indicates a lower restoration speed.
- Volume of incremental data: A larger data volume indicates a lower restoration speed.
- Large transactions: If the binary log files contain information about large transactions, the restoration speed is low.
- Hot data updates: If the binary log files contain information about hot data updates, the restoration speed is low.
- Foreign key constraints: Foreign key constraints increase the costs of verification and decrease the

restoration speed.

- Number of binary log records: If you restore data to a specific point in time, a larger number of binary log records indicates a lower restoration speed.
- Partitioned tables: If tables are partitioned, the restoration is not supported. This rule takes effect if you use MySQL 8.0.
- Storage type: The restoration speed is higher when you use standard SSDs or enhanced SSDs (ESSDs) than when you use local SSDs.
- Specifications: Higher specifications indicate a higher restoration speed.
- Database engine version: If the database engine version that you use supports parallel replication, the restoration speed is high. If the database engine version that you use does not support parallel replication, the restoration speed is low.

> **Note** The restoration may fail due to the following factors:
>
> - Binary logs are not parsed as expected because the new RDS instance runs an earlier database engine version than the original RDS instance.
> - Table names or column names contain Chinese characters or special characters.
> - Binary logs in the original RDS instance are deleted.
> - Tables that do not have primary keys cannot be restored because the implicit_primary_key parameter is set to off in the original RDS instance.

Amount of time required for data restoration

The following table lists the amount of time required to restore data to a new RDS instance that provides 2 cores and 4 GB of memory and runs RDS High-availability Edition with local SSDs.

Operation	Time required
Create an RDS instance	5 minutes
Configure the new RDS instance	15 minutes
Download the data backup file	400 GB per hour
Start the new instance.	5 minutes
Download the log backup file	400 GB per hour
Apply the log backup file	Dependent on the specific content of the log backup file

FAQ

- How do I restore one or more databases that I accidentally deleted?

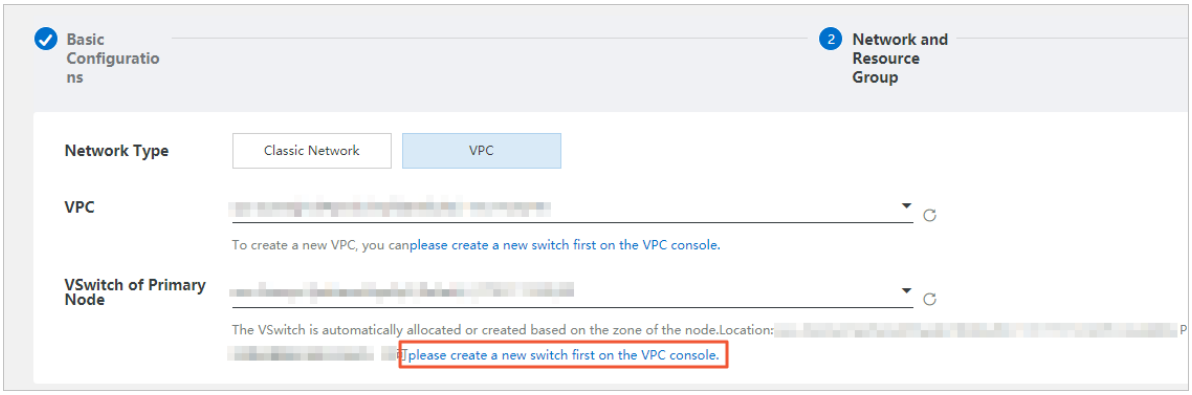
  ApsaraDB RDS allows you to restore the individual databases that you accidentally deleted. For more information, see Restore individual databases and tables of an ApsaraDB RDS for MySQL instance. If your RDS instance does not support the restoration of individual databases or tables, you can restore the data of the deleted databases to a new RDS instance, verify the data on the new RDS instance, and then migrate the data from the new RDS instance back to your original RDS instance.

- If my RDS instance does not have a data backup, can I restore data to a specific point in time?

No, if your RDS instance does not have a data backup, you cannot restore data to a specific point in time. To restore data to a specific point in time, you must find a full data backup that is completed before the specified point in time and restore the data of the data backup file that is generated from the full data backup. Then, you must restore the incremental data from the log backup file that is generated at the specified point in time.

- When I create an RDS instance to which I want to restore data, why am I unable to select a vSwitch from the vSwitch of Primary Node drop-down list?

  If no vSwitches are available in the zone that you specify in the Basic Configurations step, you cannot select a vSwitch from the vSwitch of Primary Node drop-down list in the Instance Configuration step. In this case, click **go to the VPC console**. In the VPC console, create a vSwitch. Then, you can select a vSwitch from the vSwitch of Primary Node drop-down list.



15.3. Restore individual databases and tables of an ApsaraDB RDS for MySQL instance

ApsaraDB RDS for MySQL supports database-level data restoration and table-level data restoration. If you accidentally delete a database or a table on your RDS instance, you can restore only the deleted database or table without the need to restore the data of the entire instance. This way, you can shorten the recovery time objective (RTO).

Introduction

When a physical backup is performed, ApsaraDB RDS for MySQL stores the data of your RDS instance at the database and table levels. Therefore, when you restore individual databases or tables of your RDS instance, ApsaraDB RDS for MySQL reads and restores the tables that you specify from the physical backup file. This significantly shortens the recovery time and increases the recovery speed.

ApsaraDB RDS for MySQL also provides the fast restoration feature for individual databases and tables. This feature uses sandbox instances to accelerate data restoration by approximately 50% to 95% compared with the standard individual database and table restoration feature. For more information, see Create a sandbox instance for the emergency disaster recovery of an ApsaraDB RDS for MySQL database. The fast restoration feature for individual databases and tables is in public preview and can be used free of charge. For more information about how to enable the fast restoration feature for individual databases and tables, see the "Appendix: Enable the fast restoration feature for individual databases and tables" section of this topic.

> **Note** The fast restoration feature for individual databases and tables is available only in the China (Zhangjiakou) and China (Hohhot) regions.

Billing rules

- The individual database and table restoration feature is enabled by default and is provided free of charge.
- The fast restoration feature for individual databases and tables is in public preview and can be used free of charge. After the fast restoration feature for individual databases and tables is commercially released, you are charged USD 0.24 per GB-month for using the feature. The following table describes the billing rules for the fast restoration feature for individual databases and tables.

Item	Public preview phase	Commercial use phase
Price	Free of charge	USD 0.24 per GB-month **Note** You can view the storage usage of your sandbox instance in the ApsaraDB RDS console. For more information, see Create a sandbox instance for the emergency disaster recovery of an ApsaraDB RDS for MySQL database.
Retention period of sandbox instances	1 day (The retention period cannot be adjusted in the public preview phase.)	1 day to 730 days
Functionality	Point-in-time restore supported	Point-in-time restore supported
Free trial	Not limited	7 days per RDS instance

Restore individual databases and tables of an RDS instance that uses local SSDs

Prerequisites

- The RDS instance runs MySQL 8.0, MySQL 5.7, or MySQL 5.6 on RDS High-availability Edition or runs MySQL 5.7 on RDS Enterprise Edition.
- The number of tables that are created in the RDS instance is less than 50,000.
- The individual database and table restoration feature is enabled for the RDS instance on the **Backup Settings** tab of the **Backup and Restoration** page in the ApsaraDB RDS console.

> **Note**
>
> - The individual database and table restoration feature is enabled for each new RDS instance and cannot be disabled.
> - The data that is generated before the individual database and table restoration feature is enabled cannot be restored. If you want to restore this data, we recommend that you restore all data of the RDS instance. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance.
> - After you enable the individual database and table restoration feature, the supported backup file format changes from TAR to xbstream to support the restoration of individual databases and tables. In addition, the feature cannot be disabled after it is enabled.

Impacts

If you restore individual databases and tables to the original RDS instance, a primary/secondary switchover is triggered during the restoration process. The switchover may cause a transient connection that lasts approximately 30 seconds. In this case, make sure that your application is configured to automatically reconnect to the RDS instance. If you restore individual databases and tables to a new RDS instance, no primary/secondary switchovers are triggered during the restoration process.

Procedure

1.
2. In the left-side navigation pane, click **Backup and Restoration**. On the page that appears, click **Restore Individual Database/Table**.

   > **Note** If the **Restore Individual Database/Table** button is not displayed, you must check that the RDS instance meets all prerequisites.

3. Configure the following parameters.

Parameter	Description
Restore To	• **Current Instance**: If you select this option, ApsaraDB RDS restores the data to the original RDS instance. • **New Instance**: If you select this option, ApsaraDB RDS restores the data to a new RDS instance.

Parameter	Description
Recovery Method	○ **By Backup Set**: If you select this option, you can restore the data from a data backup file. ○ **By Time**: If you select this option, you can restore the data to a point in time within the log backup retention period that you specify. For information about how to view or change the log backup retention period, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance. **Note** The **By Time** option is available only when the log backup feature is enabled.
Restore Mode	○ **Logical Restoration**: The restoration is slow. ○ **Physical Restoration**: The restoration is fast, but a primary/secondary switchover is triggered. In addition, all attached read-only RDS instances are restarted. If the RDS instance is being maintained, the amount of data that you want to restore is small, or the data replication to the attached read-only RDS instances is interrupted, the backend automatically selects **Logical Restoration**. **Note** This parameter is available only when read-only RDS instances are attached to the RDS instance.

4. Select the databases and tables that you want to restore. You can also specify the name of each restored database or table. Then, click **OK**.

   Note
   - You can select up to 50 databases or tables at a time.
   - If you restore the selected databases and tables to the original RDS instance, these databases and tables must have new names that are different from the original names. By default, ApsaraDB RDS adds _backup to the original names.
   - If you restore the selected databases and tables to the original RDS instance, you must make sure that the available storage of the RDS instance is sufficient to store the data from these databases and tables.

5. Select a billing method and configure the parameters that are required to create an RDS instance. This step is required only when you want to restore the data to a new RDS instance.
   - Subscription: A subscription instance is an instance for which you pay an upfront fee. For long-term use, the subscription billing method is more cost-effective than the pay-as-you-go billing method.
   - Pay-as-you-go: You are charged an hourly fee for a pay-as-you-go instance based on your actual resource usage. The pay-as-you-go billing method is suitable for short-term use. If you no longer need a pay-as-you-go instance, you can release the instance to reduce costs.

Parameter	Description
Availability Zone	The zone where the new RDS instance resides. Each zone is an independent physical location within a region. Zones in the same region do not have substantial differences. The new RDS instance can reside in the same zone or in a different zone than the Elastic Compute Service (ECS) instance that you want to connect. **Note** The new RDS instance must reside in the same region as the original RDS instance. You cannot change the region.
Instance Type	The instance type of the new RDS instance. Each instance type supports a different number of cores, memory capacity, maximum number of connections, and maximum IOPS. For more information, see Primary ApsaraDB RDS instance types.
Capacity	The maximum amount of storage that is provisioned to store data files, system files, binary log files, and transaction files in the new RDS instance.
Network Type	◦ **Classic Network**: the traditional type of network. ◦ **VPC**: the recommended type of network. A virtual private cloud (VPC) is an isolated virtual network that provides higher security and higher performance than the classic network.

Note You can also use one of the following methods to restore individual databases and tables:

- Use Database Backup (DBS) to create a logical backup. Then, restore data from the generated logical backup file. For more information, see Back up an ApsaraDB RDS for MySQL, PolarDB for MySQL, or self-managed MySQL database and Restore a MySQL database from a logical backup.
- Manually create a logical backup. Then, restore data from the generated logical backup file to a self-managed database. For more information, see Create a logical backup for an ApsaraDB RDS for MySQL instance and Restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance.
- Use the mysqldump plug-in to back up and restore the RDS instance. For more information, see Use mysqldump to back up and restore an ApsaraDB RDS for MySQL instance.

Restore individual databases and tables of an RDS instance that uses standard SSDs or ESSDs

1. Use DBS to create a logical backup. For more information, see Back up an ApsaraDB RDS for MySQL, PolarDB for MySQL, or self-managed MySQL database.
2. Restore data from the generated logical backup file. For more information, see Restore a MySQL database from a logical backup.

Note For more information about how to restore individual databases and tables, see Use mysqldump to back up and restore an ApsaraDB RDS for MySQL instance.

Appendix: Enable the fast restoration feature for individual databases and tables

Prerequisites

- The RDS instance runs MySQL 8.0, MySQL 5.7, or MySQL 5.6 on RDS High-availability Edition with local SSDs.

  > **Note** The fast restoration feature for individual databases and tables runs at a lower speed in RDS instances that run MySQL 8.0 than in RDS instances that run a different MySQL version due to specific database engine issues.

- The RDS instance runs the InnoDB storage engine.
- The sandbox feature is not enabled for the RDS instance. If the sandbox feature is enabled, you must disable it before you enable the fast individual database and table restoration feature. For more information, see Create a sandbox instance for the emergency disaster recovery of an ApsaraDB RDS for MySQL database.

1.
2. In the left-side navigation pane, click **Backup and Restoration**.

   > **Note** If the **Enable Fast Restoration** dialog box appears, you can click **Enable** to enable the fast individual database and table restoration feature. If the dialog box does not appear, you can proceed with the subsequent steps.

3. Click the **Backup Settings** tab. In the **Backup Settings** section of the tab, click **Edit**.
4. In the **Backup Settings** dialog box, set **Restore Speed** to **Fast** and click Save.

   > **Note** If the **Restore Individual Database/Table** switch is turned off, the **Restore Speed** parameter is not displayed. To enable the individual database and table restoration feature, you must turn on the **Restore Individual Database/Table** switch.
   >
   > - The **Restore Individual Database/Table** feature is enabled for each new RDS instance and cannot be disabled.
   > - The data that is generated before you turn on the **Restore Individual Database/Table** switch cannot be restored. If you want to restore this data, we recommend that you restore all data of the RDS instance. For more information, see Restore the data of an ApsaraDB RDS for MySQL instance.
   > - After you turn on the **Restore Individual Database/Table** switch, the supported backup file format is changed from tar to xbstream.
   > - The **Restore Individual Database/Table** switch cannot be turned off after it is turned on.

Related operations

Operation	Description

Operation	Description
RestoreTable	Restores the specified individual databases and tables of an ApsaraDB RDS instance to the original RDS instance.
CloneDBInstance	Restores the specified individual databases and tables of an ApsaraDB RDS instance to a new RDS instance.
DescribeLocalAvailableRecoveryTime	Queries the time range within which you can restore data by using a backup file of an ApsaraDB RDS instance.

15.4. Use the data tracking feature of DMS to restore the data of an ApsaraDB RDS for MySQL instance

Context

If the data of an ApsaraDB RDS for MySQL instance does not meet your expectations due to accidental operations, such as accidental update, delete, and write operations, you can use the data tracking feature of DMS to restore the data of the RDS instance. This feature provides a method for you to restore data in a more efficient manner compared with other alternative restoration methods. For more information about the alternative restoration methods, see Restore the data of an ApsaraDB RDS for MySQL instance and Restore individual databases and tables of an ApsaraDB RDS for MySQL instance.

Differences between the data tracking feature and other alternative restoration methods

Restoration method	Workflow	Fee	Speed	Restorable time range

Restoration method	Workflow	Fee	Speed	Restorable time range
Data tracking	Use the data tracking feature of DMS to identify all updates that are made over the specified time range, generate statements that are used to roll the updates back, and aggregate the generated statements into a script. Then, submit a ticket to run the script in the RDS instance. For more information, see Change regular data.	• If an RDS instance is managed in Flexible Management mode, the data tracking feature is free of charge. • If an RDS instance is managed in Stable Change mode or Secure Collaborate mode, the data tracking feature is charged. For more information, see Pricing.	Fast	The restorable time range varies based on the control mode and the binary log retention period. • If an RDS instance is managed in Flexible Management mode, the restorable time range spans up to 1 hour. • If an RDS instance is managed in Stable Change mode or Secure Collaborate mode, the restorable time range varies based on the following factors: ◦ If log backups are not enabled, the restorable time range varies based on the binary log retention period. The restorable time range spans up to 168 hours. For more information, see Upload the binary log files of an ApsaraDB RDS for MySQL instance to an OSS bucket. ◦ If log backups are enabled, the restorable time range varies based on the log backup retention period. The restorable time range spans up to 730 days. For more information, see Enable automatic backups for an ApsaraDB RDS for MySQL instance.

Restoration method	Workflow	Fee	Speed	Restorable time range
Restore the data of an ApsaraDB RDS for MySQL instance	Restore all data of the original RDS instance to a new RDS instance, verify the data on the new RDS instance, and then migrate the data from the new RDS instance back to the original RDS instance, an existing RDS instance, or an on-premises database instance.	• You are charged for the new RDS instance. For more information about the price, visit the ApsaraDB RDS buy page. • You are charged for your backup storage usage that exceeds the provided free quota. For more information, see Backup storage pricing for an ApsaraDB RDS for MySQL instance. • You are charged for the traffic that is consumed to migrate the data of the RDS instance over the Internet. For more information, see Pricing.	Slow	The restorable time range varies based on the log backup retention period and the data backup retention period. The restorable time range spans up to 730 days. For more information about how to specify these retention periods, see Enable automatic backups for an ApsaraDB RDS for MySQL instance.
Restore individual databases and tables of an ApsaraDB RDS for MySQL instance	Enable the Restore Individual Database/Table feature. Then, restore the data of the specified individual databases and tables to a new RDS instance or an existing RDS instance. If you restore the data to an existing RDS instance, a primary/secondary switchover is triggered.	• If you restore the data to a new RDS instance, you are charged for the new RDS instance. For more information about the price, visit the ApsaraDB RDS buy page. • You are charged for your backup storage usage that exceeds the provided free quota. For more information, see Backup storage pricing for an ApsaraDB RDS for MySQL instance.	Slow	The restorable time range varies based on the log backup retention period and the point in time at which the Restore Individual Database/Table feature is enabled. The log backup retention period spans up to 730 days. For more information, see Back up the individual databases and tables of an ApsaraDB RDS for MySQL instance.

Prerequisites

- The RDS instance runs MySQL 5.6 or a later version.
- If the RDS instance is managed in Flexible Management mode or Stable Change mode, you have logged on to the RDS instance. If the RDS instance is managed in Security Collaboration mode, you do not need to log on to the RDS instance, but you must have obtained the permissions on the databases whose data you want to restore. For more information, see Control modes.

What's next

15.5. Create a sandbox instance for the emergency disaster recovery of an ApsaraDB RDS for MySQL database

provides the sandbox feature. This feature allows the system to automatically obtain backup sets, generate snapshots, and rapidly create sandbox instances for emergency disaster recovery.

Context

DBS provides a complete sandbox solution for MySQL databases based on copy data management (CDM). You can use the sandbox feature of DBS to rapidly create a sandbox instance to restore data in a backup set. Read and write operations are performed within sandbox instances and do not affect source databases. You can perform a variety of operations within sandbox instances, such as data restoration, recovery drill, development and testing, query and analysis, and emergency disaster recovery. For more information, see Overview.

Billing

- ApsaraDB RDS provides the physical backup feature for ApsaraDB RDS instances and charges you backup fees. For more information, see Backup storage pricing of an ApsaraDB RDS for MySQL instance.
-
-

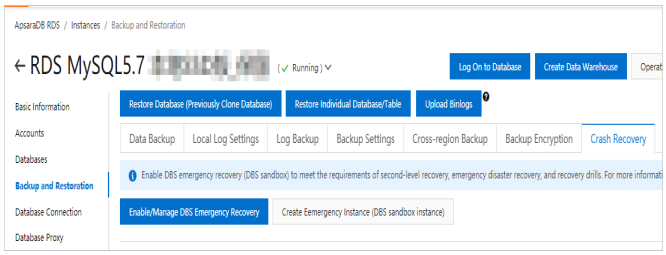
Prerequisites

- An ApsaraDB RDS for MySQL database with local SSDs is created. At least one physical backup is completed in the ApsaraDB RDS console. For more information, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.
- The instance resides in one of the following regions: China (Beijing), China (Shanghai), China (Hangzhou), China (Shenzhen), China (Zhangjiakou), China (Chengdu), China (Hohhot), China (Ulanqab), China (Heyuan), China (Hong Kong), Indonesia (Jakarta), Malaysia (Kuala Lumpur), Australia (Sydney), Germany (Frankfurt), and China East 1 Finance.
- Transparent Data Encryption (TDE) is not enabled for the ApsaraDB RDS for MySQL instance.

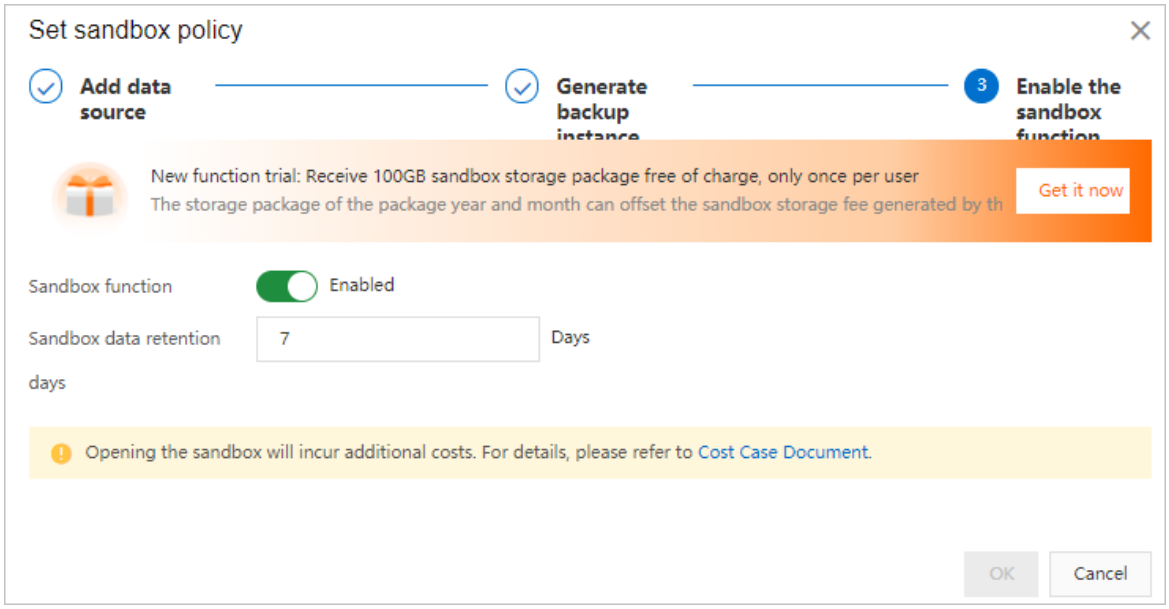
Enable the sandbox feature

After you enable the sandbox feature, DBS automatically synchronizes the data to be restored to sandbox storage. Then, you can create a sandbox instance for emergency disaster recovery.

1.
2. In the left-side navigation pane, click **Backup and Restoration**.
3. On the **Backup and Restoration** page, click **Crash Recovery**.



4. Click **Enable DBS Sandbox**.
5. In the **Set Sandbox Policy** dialog box, configure the parameters.

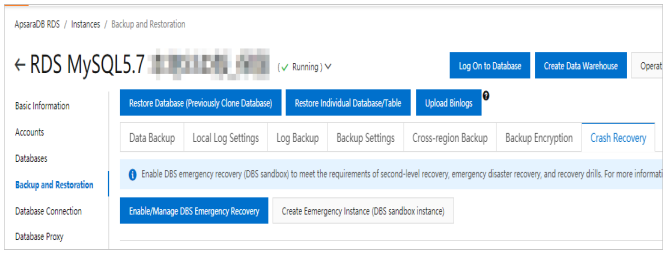


   - **Sandbox Feature**: on. The **Sandbox Feature** switch specifies whether to enable the sandbox feature.
   - **Sandbox Data Retention Period**: the number of days for which DBS retains the data that is used to create sandbox instances. Default value: 7. By default, DBS retains data for the last seven days. You can restore data to a point in time within the retention period.
6. Click **OK**.
   The sandbox feature is enabled. If this feature is enabled for the first time, DBS stores the latest full backup set of the ApsaraDB RDS for MySQL database to sandbox storage and generates snapshots that can be used to create sandbox instances. This process may take 5 to 20 minutes based on the backup data size.

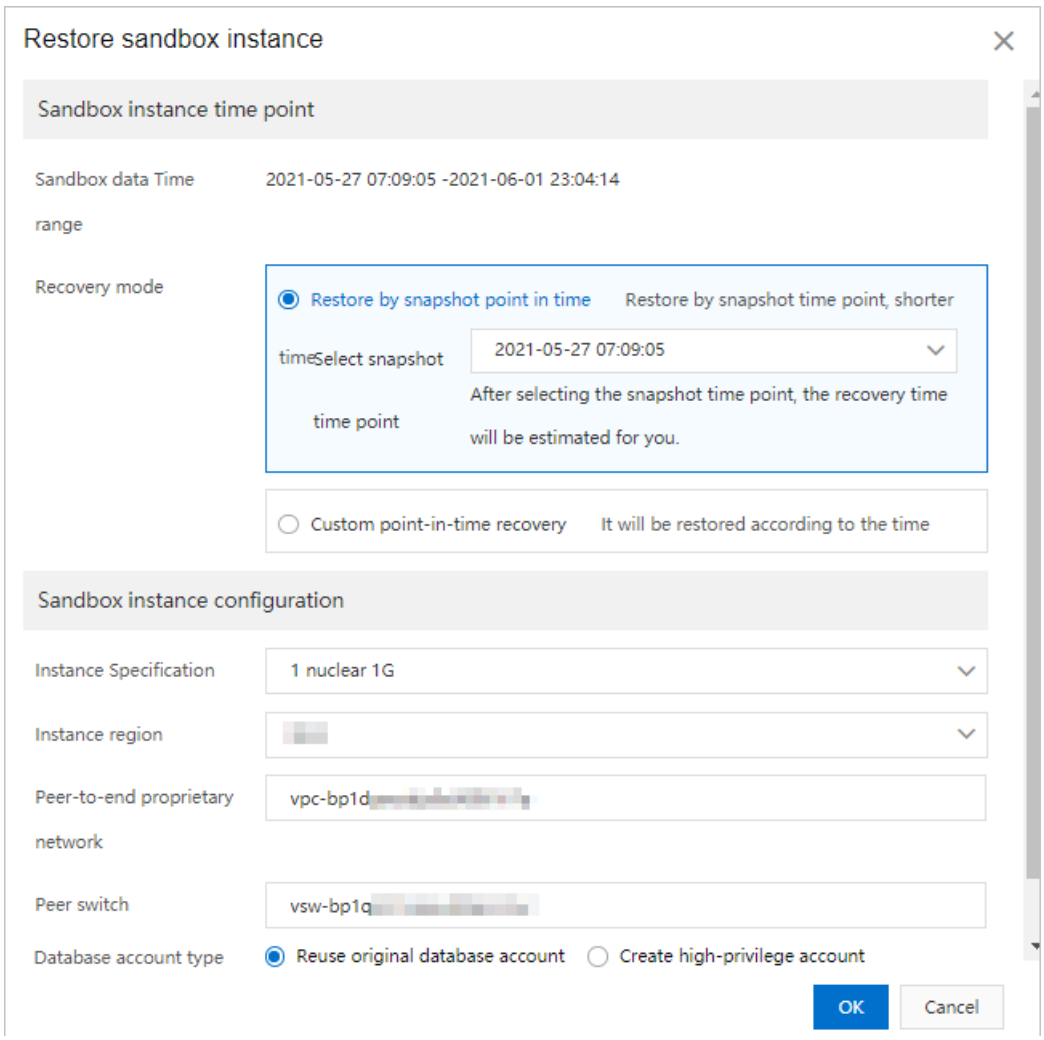
   After the available time range is displayed to the right of the **Available Time Range to Recover** parameter, you can create sandbox instances.

Create a sandbox instance for emergency disaster recovery

1.
2. In the left-side navigation pane, click **Backup and Restoration**.
3. On the **Backup and Restoration** page, click **Crash Recovery**.



4. Click **Recover Sandbox Instance**.
5. In the **Recover Sandbox Instance** dialog box, set the parameters that are described in the following table.



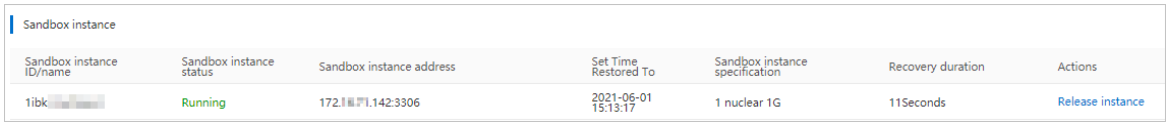
Type	Configuration item	Description
	Sandbox Data Time Range	The time range available to restore the source database. The value of this parameter depends on the value of the **Sandbox Data Retention Period** parameter that you specify for the sandbox feature. For example, if the **Sandbox Data Retention Period** parameter is set to 7, the available time range is the last seven days.

Type	Configuration item	Description
Sandbox Instance Point in Time	Recovery Method	The following two methods are supported to restore data: ○ **Recovery by Snapshot Point in Time**: You can select only the point in time at which a snapshot is created. This restoration method takes less time to complete. We recommend that you select this method. ○ **Recovery by Custom Point in Time**: You can select a specific point in time within the backup set retention period. The time that the restoration takes to complete varies based on the backup size.
Sandbox Instance Configurations	Specifications	The following eight specifications of the sandbox instance to be created are supported: ○ 1 CPU core and 1 GB of memory ○ 1 CPU core and 2 GB of memory ○ 2 CPU cores and 4 GB of memory ○ 2 CPU cores and 8 GB of memory ○ 4 CPU cores and 8 GB of memory ○ 4 CPU cores and 16 GB of memory ○ 8 CPU cores and 16 GB of memory ○ 8 CPU cores and 32 GB of memory The specifications of sandbox instances do not significantly affect restoration speed. However, sandbox instances with higher specifications provide better performance. For more information, see Sandbox instance fees.
	Region	The region where the sandbox instance resides.
	Peer VPC	The virtual private cloud (VPC) within which the sandbox instance is created. If you want to connect to the sandbox instance by using your Elastic Compute Service (ECS) instance, you must set this parameter to the VPC where the ECS instance resides. For more information about how to create a VPC, see Create a VPC or VSwitch.
	Peer vSwitch	The vSwitch to which the sandbox instance is connected. The sandbox instance can be accessed by using the CIDR block specified for the vSwitch.

Type	Configuration item	Description
	Database Account Type	The following options are supported: ○ **Use Original Database Account**: uses the account of the source database. ○ **Create Privileged Account**: creates a privileged account in the sandbox instance. The privileged account has permissions on all databases in the sandbox instance. You must enter a new database account and a password. The account of the source database is retained in the sandbox instance. **Note** A privileged account is created only in the specific sandbox instance. If the account name of the source database is the same as the name of the privileged account, the account of the source database is overwritten by the privileged account in the sandbox instance.

6. Click **OK**.
   DBS performs a precheck and creates the sandbox instance. When the status of the sandbox instance changes to running, you can connect to the sandbox instance.

   The account, the password, and all configurations of the sandbox instance are the same as those of the source database. If you select **Create Privileged Account** in Step 5, you can also use the created privileged account to connect to the sandbox instance.



What to do next

You can obtain the endpoint of the sandbox instance in the ApsaraDB RDS console and connect to the endpoint by using the corresponding client. You can also connect to the sandbox instance by using Data Management (DMS) or ECS. For more information, see Use DMS to access sandbox instances or Use ECS to access sandbox instances.

Note The client, such as an ECS instance, that you use to connect to the sandbox instance must be deployed within the same VPC as the sandbox instance.

15.6. Restore the data of an ApsaraDB RDS for MySQL instance across regions

This topic describes how to restore the data of an ApsaraDB RDS for MySQL instance from a cross-region backup file to an existing or new RDS instance. The existing or new RDS instance must reside in the region to which the original RDS instance belongs or in the region where the cross-region backup file is stored.

Prerequisites

The cross-region backup feature is enabled. For more information, see Back up an ApsaraDB RDS for MySQL instance across regions.

> **Note**
>
> - For more information about how to restore the data of an ApsaraDB RDS for SQL Server instance cross regions, see Restore the data of an ApsaraDB RDS for SQL Server instance across regions.
> - For more information about how to restore the data of an ApsaraDB RDS for PostgreSQL instance across regions, see Restore the data of an ApsaraDB RDS for PostgreSQL instance across regions.

Precautions

Before you can connect to the new RDS instance to which you want to restore data, you may need to reset the password of the logged-on account. This applies if the original RDS instance has the database proxy feature enabled and does not have a privileged account.

Restore data to a new RDS instance

1. Log on to the RDS management console , in the left-side navigation pane, click **Backups** , and then select a region above.
2. On the **Backup Instances** tab of the **Cross-region Backup** tab, find your RDS instance and click the ID of the instance. On the page that appears, find the backup file that you want to use, and click **Restore** in the **Actions** column.
3. On the **Data Backup** tab, find the data backup file that you want to use, and click **Restore** in the Actions column.
4. Select **Restore to New Instance** and click **OK**.
5. On the **Restore Database** page, click the **Subscription** or **Pay-As-You-Go** tab and configure the following parameters.

Parameter	Description
Restore Mode	○ **By Backup Set**: allows you to restore the data of your RDS instance from a data backup file. ○ **By Time**: allows you to restore the data of your RDS instance to a specific point in time. The point in time must be within the specified log backup retention period.
Backup Set	The data backup file from which you want to restore the data of your RDS instance. This parameter appears only when you set the **Restore Mode** parameter to **By Backup Set**.

Parameter	Description
Restore Point	The point in time to which you want to restore the data of your RDS instance. This parameter appears only when you set the **Restore Mode** parameter to **By Time**. **Note** Both local and cross-region log backup files can be used to restore the data of your RDS instance to a specific point in time.
Region	The region to which the new RDS instance belongs.
Zone	The zone where the new RDS instance resides. Each zone is an independent physical location within a region. Zones in the same region provide the same services. You can create the new RDS instance in the same zone as the Elastic Compute Service (ECS) instance to which you want to connect. You can also create the new RDS instance in a different zone than the ECS instance to which you want to connect.
CPU and Memory	The specifications of the new RDS instance. Each instance type supports a specific number of CPU cores, memory capacity, maximum number of connections, and maximum input/output operations per second (IOPS). For more information, see Primary ApsaraDB RDS instance types.
Capacity	The storage capacity that the new RDS instance has available to store data files, system files, archived log files, and transaction files.
Network Type	◦ **Classic Network**: the traditional type of network. ◦ **VPC**: the recommended type of network. A virtual private cloud (VPC) is an isolated virtual network that provides higher security and higher performance than the classic network. If you select the VPC network type, you must also select a vSwitch that is associated with the specified VPC.

Note The settings of some parameters cannot be modified. These parameters include Database Engine, Version, and Edition. The same settings of these parameters must be specified for both your RDS instance and the new RDS instance.

6. Specify the **Duration** and **Quantity** parameters. Then, click **Buy Now**. You must specify the Duration parameter when the new RDS instance is billed on a subscription basis.
7. On the **Order Confirmation** page, read and select Terms of Service, Service Level Agreement, and Terms of Use. Then, click Pay Now and complete the payment.

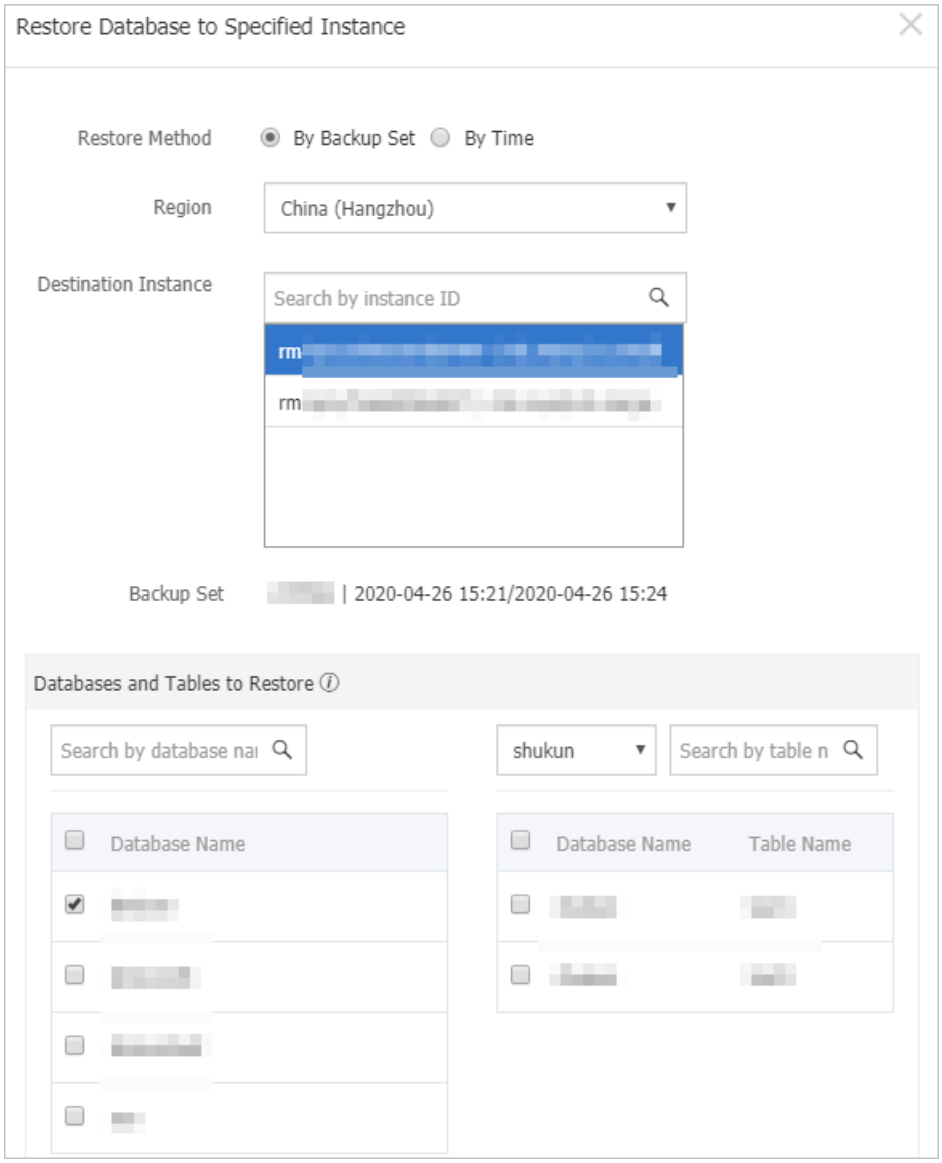
Restore data to an existing RDS instance

Note Make sure that the individual database/table backup feature is enabled and at least one individual database and table backup is created on the existing RDS instance.

1. Log on to the RDS management console , in the left-side navigation pane, click **Backups** , and then select a region above.
2. On the **Backup Instances** tab of the **Cross-region Backup** tab, find your RDS instance and click

the ID of the instance. On the page that appears, find the backup file that you want to use, and click **Restore** in the **Actions** column.

3. Select **Restore to Existing Instance** and click **OK**.
4. Configure the following parameters.



Parameter	Description
Restore Mode	○ **By Backup Set**: allows you to restore the data of your RDS instance from a data backup file. ○ **By Time**: allows you to restore the data of your RDS instance to a specific point in time. The point in time must be within the specified log backup retention period.
Region	The region to which the existing RDS instance belongs.

Parameter	Description
Destination Instance	The existing RDS instance to which you want to restore the data of your RDS instance.
Databases and Tables to Restore	Select the databases and tables that you want to restore.
Selected Databases and Tables	The new names of the databases and tables on the existing RDS instance. If you do not specify the new names, the `_backup` suffix is added to the original names.

5. Click **OK**.

References

After you create an RDS instance, you must configure IP address whitelists or security groups and create accounts. For more information, see Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance and Create accounts and databases for an ApsaraDB RDS for MySQL instance. If you want to connect to the RDS instance over the Internet, you must also apply for a public endpoint. For more information, see Apply for or release a public endpoint on an ApsaraDB RDS for MySQL instance. After you complete these operations, you can connect to the RDS instance. For more information, see Connect to an ApsaraDB RDS for MySQL instance.

Related operations

Operation	Description
Check cross-region backup	Checks whether an ApsaraDB RDS instance has a cross-region data backup file that can be used to restore data across regions.
Restore data to a new instance across regions	Restores the data of an ApsaraDB RDS instance to a new RDS instance that resides in a different region than the original RDS instance.
Modify cross-region backup settings	Modifies the cross-region backup settings of an ApsaraDB RDS instance.
Query cross-region backup settings	Queries the cross-region backup settings of an ApsaraDB RDS instance.
Query cross-region data backup files	Queries the cross-region data backup files of an ApsaraDB RDS instance.
Query cross-region log backup files	Queries the cross-region log backup files of an ApsaraDB RDS instance.
Query regions that support cross-region backup	Queries the regions to which the cross-region backup files from the current region can be restored.
Query the time range to which you can restore data by using a cross-region backup set	Queries the restorable time range that is supported by a cross-region backup file.

Operation	Description
Query ApsaraDB for RDS instances on which cross-region backup is enabled	Queries the ApsaraDB RDS instances for which the cross-region backup feature is enabled in a region and the cross-region backup settings of these instances.

15.7. Restore from backup files

15.7.1. Restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL database

This topic describes how to restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL database.

> **Note**
> - For information about how to restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL database, see Restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance.
> - Due to software limits, you can restore the data of an ApsaraDB RDS for MySQL instance from a data backup file only to a self-managed MySQL database that runs in a Linux operating system. For more information about how to restore the data of an ApsaraDB RDS for MySQL instance from a data backup file to a self-managed MySQL database that runs in a Windows operating system, see Use mysqldump to migrate data from a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance.

Step 1: Set up the environment

1. Make sure that the RDS instance runs MySQL 8.0, MySQL 5.7, MySQL 5.6, or MySQL 5.5 on RDS High-availability Edition with local SSDs.

   > **Note** You can download the physical backup files of the RDS instance only when the RDS instance meets this requirement. For more information about how to restore the data of an RDS instance that runs RDS Basic Edition, see the "FAQ" section of this topic.

2. Make sure that the tables in the RDS instance are not encrypted by Transparent Data Encryption (TDE). If tables in the RDS instance are encrypted by TDE, errors occur during the restoration process. We recommend that you decrypt the encrypted tables before you start a restoration task. For more information, see Decrypt a table.
3. Obtain a computer or a server that runs a 64-bit Linux operating system. Make sure that the MySQL service is installed on the computer or the server. In addition, make sure that the computer or the server runs the same MySQL version as the RDS instance.

> **Note** Make sure that no other services are running on top of the MySQL service.

4. Install Percona XtraBackup on the computer or the server.
   - If the RDS instance runs MySQL 5.7, MySQL 5.6, or MySQL 5.5, install Percona XtraBackup 2.4 on the computer or the server.
   - If the RDS instance runs MySQL 8.0, install Percona XtraBackup 8.0 on the computer or the server.
5. Install qPress on the computer or the server. qPress is an extraction tool.

```
wget "http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/attach/183466/cn_zh/1608
011575185/qpress-11-linux-x64.tar"
tar xvf qpress-11-linux-x64.tar
chmod 775 qpress
cp qpress /usr/bin
```

Step 2: Download the physical backup file that you want to use

1.
2. Open the **Backup and Restoration** page and click the **Data Backup** tab.
3. Select a time range. This step is required if you want to view the backup files that were generated eight days ago. The default time range spans the most recent eight days.
4. Find the physical backup file that you want to use. In the **Actions** column, click **Download Instance Backup**.

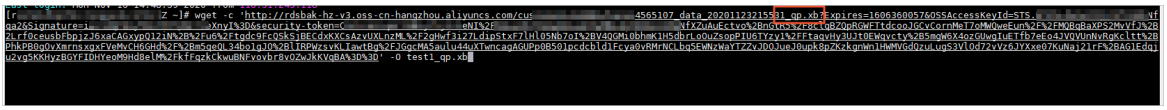
   Download Instance Backup may not be displayed due to the following reasons:
5. In the dialog box that appears, copy the URL that you can use to download the physical backup file.

> **Note**
>
> - A free quota for backup downloads over the Internet is provided. If the amount of traffic that you consume to download backup files over the Internet exceeds the free quota, you are charged for the excess traffic that you consume. For more information, see Download the backup files of an ApsaraDB RDS for MySQL instance.
> - If your Elastic Compute Service (ECS) instance resides in the same virtual private cloud (VPC) as the RDS instance, you can use the internal URL to download the logical backup file. This download method is faster and more stable.

6. Run the following command on the computer or the server to download the physical backup file:

```
wget -c 'http://...' -O test1_qp.xb
```

> **Note**
>
> - You must replace *http://...* with the URL that you can use to download the physical backup file.
> - *test1_qp.xb* is the name that is used for the physical backup file after the physical backup file is downloaded. You can change the file name based on your business requirements. However, you must make sure that the extension of the file name remains unchanged.



Step 3: Decompress the physical backup file that you downloaded and restore data from the file that is generated from the decompression

1. Create a directory that is used to store the file that is generated from the decompression on the computer or the server. For example, you can create a directory named */home/mysql/data*.

   ```
   mkdir /home/mysql/data
   ```

2. Decompress the physical backup file. The command that is used to decompress the physical backup file varies based on the extension of the file name.

Extension	Command used for decompression
.tar.gz	`tar -izxvf test1.tar.gz -C /home/mysql/data`
.xb.gz	`gzip -d -c test1.xb.gz | xbstream -x -v -C /home/mysql/data`

_qp.xb	`## Unpack the physical backup file.`
	`cat test1_qp.xb | xbstream -x -v -C /home/mysql/data`
	`## Decompress the physical backup file.`
	`### If the RDS instance runs MySQL 5.6 or MySQL 5.7, run the following command:`
	`innobackupex --decompress --remove-original /home/mysql/data`
	`### If the RDS instance runs MySQL 8.0, run the following command:`
	`xtrabackup --decompress --remove-original --target-dir=/home/mysql/data`

_xb.qp	`qpress -do  test1_xb.qp  | xbstream -x -v -C /home/mysql/data`

> **Note** You can replace *test1* and */home/mysql/data* in the preceding commands with the actual names and save paths of your physical backup file.

3. Run the following command to query the file that is generated from the decompression:

```
ls -l /home/mysql/data
```

The system returns the following information, in which the information in blue indicates the databases whose data is contained in the physical backup file.

```
[root@testcentos ~]# ls -l /home/mysql/data
total 204844
-rw-r--r-- 1 root root       297 Apr 28 21:13 backup-my.cnf
-rw-rw---- 1 root root 209715200 Apr 28 21:04 ibdata1
drwxr-xr-x 2 root root      4096 Apr 28 22:01 mysql
drwxr-xr-x 2 root root      4096 Apr 28 22:01 performance_schema
drwxr-xr-x 2 root root      4096 Apr 28 22:01 test
drwxr-xr-x 2 root root      4096 Apr 28 22:01 xiangluo
```

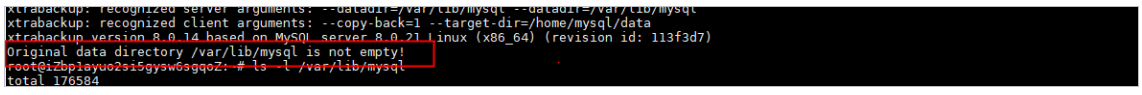
4. Restore the data of the file that is generated from the decompression to the self-managed MySQL database.

```
## MySQL 5.6/5.7
innobackupex --defaults-file=/home/mysql/data/backup-my.cnf --apply-log /home/mysql/dat
a
## MySQL 8.0
xtrabackup --prepare --target-dir=/home/mysql/data
xtrabackup --datadir=/var/lib/mysql --copy-back --target-dir=/home/mysql/data
```

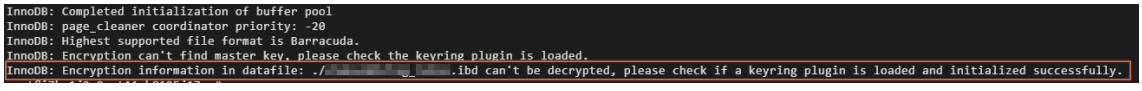
- If the system displays the following or similar information, the data is restored to the self-managed MySQL database.

```
InnoDB: Shutdown completed; log sequence number 1635350
150428 22:08:40  innobackupex: completed OK!
[root@testcentos ~]#
```

- If the system returns the following error, run the `rm -rf /var/lib/mysql` command to delete all files from the directory. Then, run the `chown -R mysql:mysql /var/lib/mysql` command to modify the permissions on the directory.



- If the system returns the following error, check whether the tables in the RDS instance are encrypted by TDE. For more information, see the "Step 1: Set up the environment" section of this topic.



> **Note** The Percona XtraBackup version that you use must match the MySQL version of the RDS instance:
>
> - If the RDS instance runs MySQL 5.7, MySQL 5.6, or an earlier version, you must install Percona XtraBackup 2.4. For more information, see the Percona XtraBackup 2.4 documentation.
> - If the RDS instance runs MySQL 8.0, you must install Percona XtraBackup 8.0. For more information, see the Percona XtraBackup 8.0 documentation.

Step 4: Run MySQL

1. Modify the backup-my.cnf file to ensure version compatibility.
   i. Run the following command to open the backup-my.cnf file in text mode:

   ```
   vi /home/mysql/data/backup-my.cnf
   ```

   ii. Add the following parameter setting to the backup-my.cnf file:

   ```
   lower_case_table_names=1
   ```

   iii. Comment out the following parameters, which are not supported by the self-managed MySQL database:

   ```
   #innodb_log_checksum_algorithm
   #innodb_fast_checksum
   #innodb_log_block_size
   #innodb_doublewrite_file
   #innodb_encrypt_algorithm
   #rds_encrypt_data
   #redo_log_version
   #master_key_id
   #server_uuid
   ```

   > **Note** The MyISAM storage engine is incompatible with the InnoDB storage engine that is used by ApsaraDB RDS. If the self-managed MySQL database runs MyISAM, you must comment out the following parameters and add the skip-grant-tables parameter:
   >
   > ```
   > #innodb_log_checksum_algorithm=strict_crc32
   > #redo_log_version=1
   > skip-grant-tables
   > ```

   iv. Press **Esc**, enter `:wq`, and then press Enter to save the backup-my.cnf file.
2. Run the following command to change the owner of the backup-my.cnf file to a user who has the permissions to manage the self-managed MySQL database:

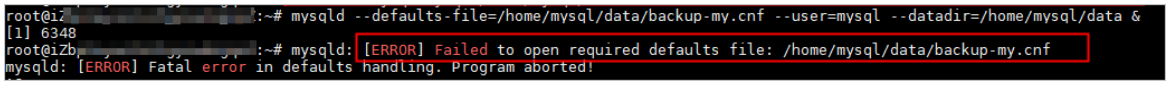
   ```
   chown -R mysql:mysql /home/mysql/data
   ```

3. Run the following command to run MySQL:

   ```
   mysqld --defaults-file=/home/mysql/data/backup-my.cnf --user=mysql --datadir=/home/mysql/data &
   ```

> **Note**
>
> - The following issues related to the password of the root user may occur:
>   - If the RDS instance runs MySQL 5.5 or MySQL 5.6, you must reset the password of the root user of the RDS instance. For more information, see the open source MySQL documentation.
>   - If the RDS instance runs MySQL 5.7 or MySQL 8.0, the password of the root user of the RDS instance is the same as the password of the root user of the self-managed MySQL database.
> - If an error is reported when you start MySQL, you can try to resolve the error by changing the storage engine. For more information, see the "FAQ" section of this topic.

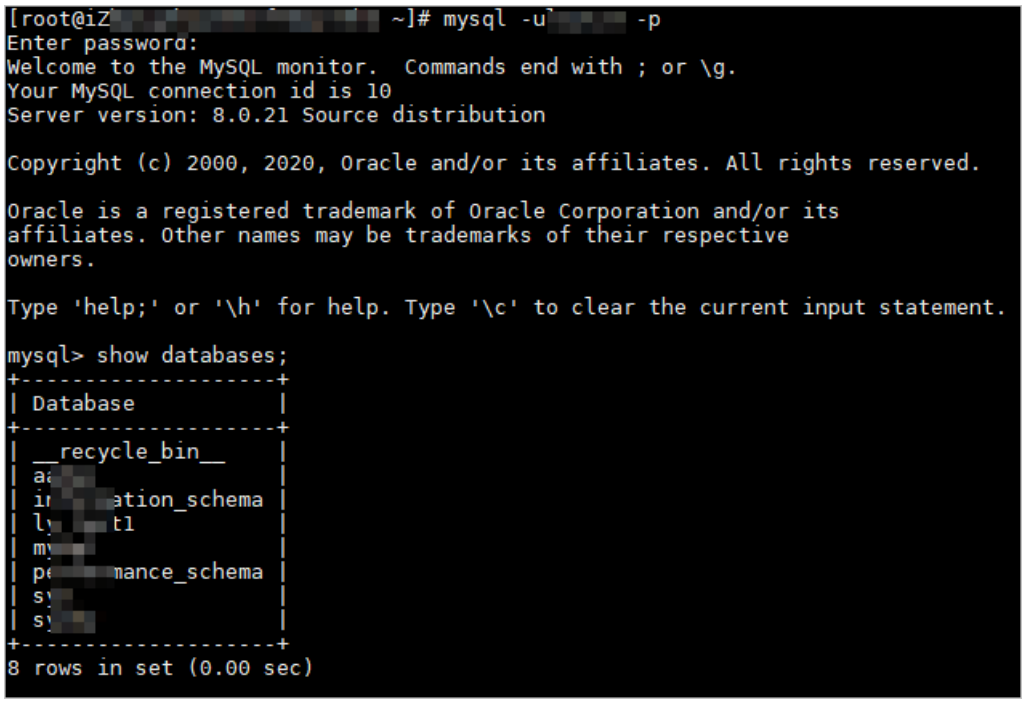
AppArmor is a built-in security program that is provided in an Ubuntu operating system. If you use an Ubuntu operating system, the system may report the error that is shown in the following figure. In this case, you must run the `apt install -y apparmor-utils` and `aa-complain /usr/sbin/mysqld` commands to modify the settings of the AppArmor security program.



4. Run the following command to log on to the self-managed MySQL database and verify that MySQL is running:

```
mysql -u<The username of the account that is used to connect to the RDS instance> -p<The password of the preceding account>
```

You can run the `show databases;` command to view the self-managed MySQL database and check whether the restoration task is successful.



FAQ

- How do I restore the data of my RDS instance over a specified time range to a self-managed MySQL database?

  You can download the log backup file that is generated over the specified time range in the ApsaraDB RDS console. Then, you can use the log backup file to restore the data of your RDS instance to a self-managed MySQL database. For more information, see Download the backup files of an ApsaraDB RDS for MySQL instance.

- After a restoration task is completed, what do I do if the "`error 1105 Unknown error`" message is displayed when I use the self-managed MySQL database?

  Execute the following SQL statements to convert the storage engine:

  ```
  use mysql;
  alter table proc engine=myisam;
  alter table event engine=myisam;
  alter table func engine=myisam;
  ```

- In addition to using the data backup files that I downloaded, what other methods can I use to restore the data of my RDS instance to a self-managed MySQL database?

  You can use Data Transmission Service (DTS) to migrate the data of your RDS instance to a self-managed MySQL database. For more information, see Migrate data from an ApsaraDB RDS for MySQL instance to a self-managed MySQL database.

- When I download a data backup file, why does the system report errors?

  If you run the following command to download the data backup file, check that the public URL is enclosed in a pair of single quotation marks ('): `wget -c '<The public URL that you can use to download the data backup file>' -O <The name that you want to use for the data backup file after the file is downloaded>.tar.gz`. The single quotation marks (') are used by the system to identify the public URL.

- When I decompress the data backup file that I downloaded, what do I do if the system reports errors?

  i. Check whether the data backup file is a physical backup file.

  ii. Check whether the data backup file is saved based on a valid file extension. Valid file extensions are .tar.gz, .xb.gz, and _qp.xb.

  iii. Check whether you ran a valid command that is supported for the format of the data backup file. For more information, see the "Step 3: Decompress the physical backup file that you downloaded and restore data from the file that is generated from the decompression" section of this topic.

- How do I restore or migrate the data of my RDS instance if my RDS instance runs RDS Basic Edition?

  RDS instances that run RDS Basic Edition support only snapshot backups. If your RDS instance runs RDS Basic Edition, use one of the following two methods to restore or migrate the data:

  - Use mysqldump to migrate the data of your RDS instance to a self-managed MySQL instance. For more information, see Use mysqldump to migrate data from a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance.
  - Use DTS to export the data of your RDS instance to your computer.

- Can I restore the data of my RDS instance from a data backup file that I downloaded to another RDS instance?

This operation is not supported by ApsaraDB RDS. We recommend that you use DTS to migrate the data of your RDS instance to another RDS instance.For more information, see Migrate data between RDS instances.

References

- Restore the data of an ApsaraDB RDS for MySQL instance
- Restore individual databases and tables of an ApsaraDB RDS for MySQL instance
- Restore the data of an ApsaraDB RDS for MySQL instance across regions

15.7.2. Restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance

This topic describes how to restore the data of an ApsaraDB RDS for MySQL instance from a logical backup file to a self-managed MySQL instance by using the mysqldump plug-in of MySQL.

Prerequisites

- The RDS instance runs one of the following MySQL versions and RDS editions:
  - MySQL 8.0 on RDS High-availability Edition (with local SSDs)
  - MySQL 5.7 on RDS High-availability Edition (with local SSDs)
  - MySQL 5.6
  - MySQL 5.5
- A logical backup is complete on the RDS instance. For more information, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.
- Tables in the RDS instance are not encrypted by using Transparent Data Encryption (TDE). If tables are encrypted by using TDE, errors occur during the restoration process. Before you restore the data of the RDS instance from a logical backup file, you must decrypt the encrypted tables. For more information, see Decrypt a table.

> **Note**
>
> - For more information about how to restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL instance, see Restore the data of an ApsaraDB RDS for MySQL instance from a physical backup file to a self-managed MySQL database.
> - For more information about how to back up an ApsaraDB RDS for MySQL instance, see Back up an ApsaraDB RDS instance.

Runtime environment

The self-managed instance is installed in a 64-bit Linux operating system and runs the same MySQL version as the RDS instance. In this topic, Linux 7 and MySQL 5.7 are used as examples.

Procedure

1.
2. In the left-side navigation pane, click **Backup and Restoration**.
3. On the Data Backup tab of the page, select a time range and click **OK**
4. Find the logical backup file that you want to download. Then, click **Download Instance Backup** in the Actions column.

   > **Note**
   > - Logical backups must be manually created. For more information, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.
   > - If **Download Instance Backup** cannot be found, you can check whether the MySQL version of the RDS instance supports the downloads of logical backup files. For more information, see Download the backup files of an ApsaraDB RDS for MySQL instance.

5. In the **Download Instance Backup Set** dialog box, click to the right of **Copy Public URL**.

   > **Note**
   > - A free quota for backup downloads over the Internet is provided. If the amount of traffic that you consume to download backup files over the Internet exceeds the free quota, you are charged for the excess traffic that you consume. For more information, see Download the backup files of an ApsaraDB RDS for MySQL instance.
   > - If your Elastic Compute Service (ECS) instance resides in the same virtual private cloud (VPC) as the RDS instance, you can use the internal URL to download the logical backup file. This download method is faster and more stable.

6. Log on to the Linux operating system on which the self-managed instance runs. Then, run the following command to download the logical backup file:

   ```
   wget -c '<The public URL from which you can download the logical backup file>' -O <The name that you want to specify as the name of the downloaded logical backup file>.tar
   ```

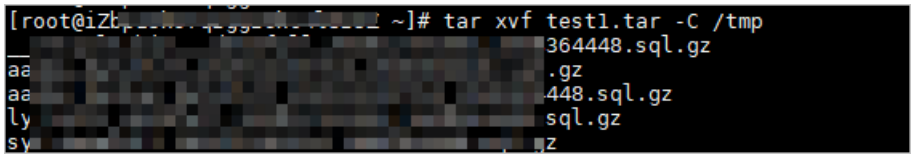
   > **Note**
   > - The -c option enables the resumable download feature.
   > - The -O option specifies to save the downloaded logical backup file based on the specified file name.

7. Run the following command to decompress the downloaded logical backup file, which includes the compressed files of the default system databases and the compressed files of the databases that you created:

   ```
   tar xvf <The name of the downloaded logical backup file>.tar -C /tmp
   ```

   Example:

   ```
   tar xvf hins123456.tar -C /tmp
   ```



> **Note** If an error message similar to "`This does not look like a tar archive`" is displayed, you must check whether the file that you downloaded is **a logical backup file of the RDS instance**.

8. Run the following command to decompress the compressed file of the database that you want to restore (the name of the compressed file is suffixed by `.sql.gz`):

```
gzip -d /tmp/The name of the compressed file of the database that you want to restore
```

Example:

```
gzip -d /tmp/testdata_datafull_202012101615_160xxxxxx.sql.gz
```

> **Note** The .sql file that is generated during the decompression process will be imported in Step 10.

9. Run the following commands to log on to the self-managed instance and create an empty database:

```
mysql -u root -p<The password that is used to log on to the self-managed instance>
create database <The name of the empty database>;
exit
```

10. Run the following command to import the .sql file into the empty database:

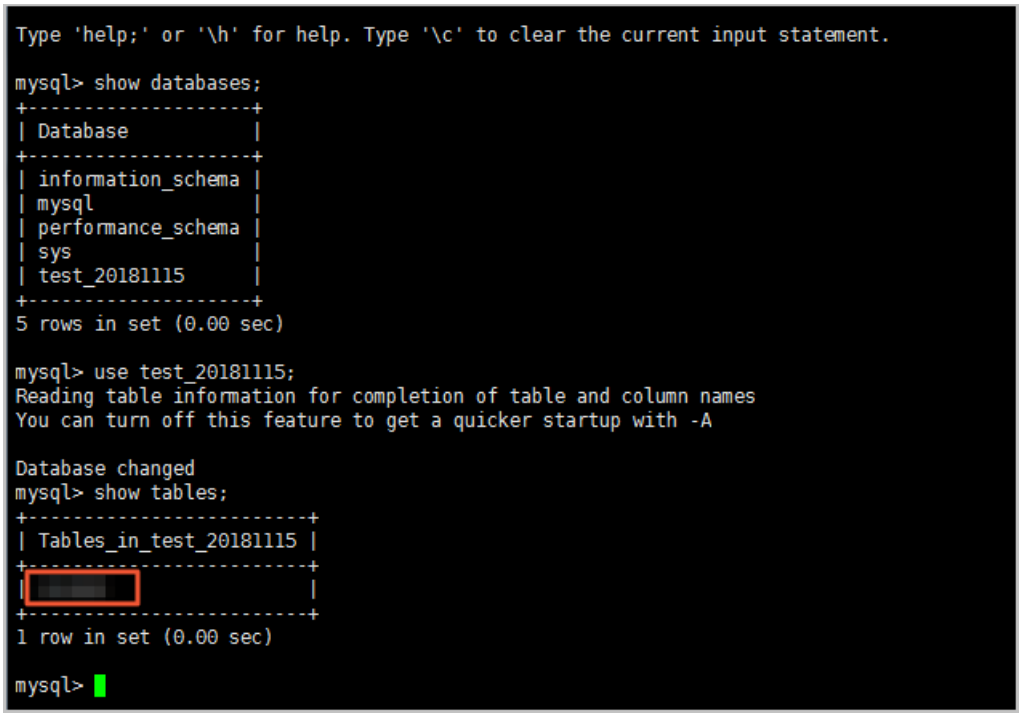
```
mysql -u root -p <The name of the empty database> < /tmp/The name of the decompressed f
ile that is generated in Step 8
```

Example:

```
mysql -u root -p testdb < /tmp/testdata_datafull_202012101615_160xxxxxx.sql
```

> **Note**
>
> - After the preceding command is successfully run, the system displays a message that prompts you to enter a password. Enter the password and press Enter.
> - If the "Can't find master key from keyring" error message is displayed, check whether the RDS instance meets all prerequisites.

11. Log on to the empty database. Then, check for data in the database. If you can find data in the database, the data of the RDS instance is successfully restored to the self-managed instance.



FAQ

- Why does my RDS instance not have logical backup files?

  By default, ApsaraDB RDS creates physical backups. You must manually create logical backups if required. For more information, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance.

- When I download a logical backup file, why is the value in the **Backup Set Restore Point** column displayed as 0 for the file?

  ApsaraDB RDS for MySQL allows you to restore data to a specific point in time by using a physical backup file and a log backup file. The ApsaraDB RDS console provides the **Backup Set Restore Point** column. In this column, you can view the timestamp of each physical backup file. Logical backup files cannot be used to restore data to a specific point in time. The value in the Backup Set Restore Point column is 0 for all logical backup files.

- What do I do if the "`ERROR 1840 (HY000) at line 24: @@GLOBAL.GTID_PURGED can only be set when @@GLOBAL.GTID_EXECUTED is empty.`" error message is displayed?

  This issue occurs due to GTIDs. You can use the following methods to resolve the issue:

  - Enable the GTID feature. Then, repeat the steps in the "Procedure" section of this topic to restore the data.
  - Do not enable the GTID feature. Comment out the GTID_PURGED parameter in the `.sql` file that you want to import. Then, repeat the steps in the "Procedure" section of this topic to restore the data.
  - Check that the synchronous replication mode is disabled. Log on to the self-managed instance, run the `reset master` command, and then repeat the steps in the "Procedure" section of this topic to restore the data.

- What do I do if the "`ERROR 3546 (HY000) at line 26: @@GLOBAL.GTID_PURGED cannot be changed: the added gtid set must not overlap with @@GLOBAL.GTID_EXECUTED`" error message is displayed?

The imported `.sql` file contains specific GTIDs that can be found in the self-managed instance. In this case, log on to the self-managed instance, run the `reset master` command, and then repeat the steps in the "Procedure" section of this topic to restore the data.

```
mysql> reset master;
Query OK, 0 rows affected (0.01 sec)
```

- After the data is restored to the self-managed instance, why is the data not automatically synchronized to the secondary instance of the self-managed instance?

  Check whether you can find the "`SESSION.SQL_LOG_BIN= 0`" setting at the end of the imported `.sql` file. If the setting is specified, the data that is restored to the self-managed instance cannot be automatically synchronized to the secondary instance of the self-managed instance.

```
SET @MYSQLDUMP_TEMP_LOG_BIN = @@SESSION.SQL_LOG_BIN;
SET @@SESSION.SQL_LOG_BIN= 0;
```

15.7.3. Restore the data of an ApsaraDB RDS for MySQL instance to a self-managed MySQL instance by using a CSV file

ApsaraDB RDS for MySQL allows you to export the backup data of an ApsaraDB RDS for MySQL instance that uses standard SSDs or enhanced SSDs (ESSDs) as a CSV file. You can use the CSV file to restore the data of the RDS instance to a self-managed MySQL instance.

Prerequisites

- The local_infile parameter is enabled for the self-managed MySQL instance.

  > **Note** You can execute the following SQL statements to check the status of local_infile or enable local_infile:
  >
  > - Check whether local_infile is enabled: `SHOW GLOBAL VARIABLES LIKE 'local_infile;`
  > - Enable local_infile: `SET GLOBAL local_infile=1;`

- Transparent Data Encryption (TDE) is disabled for the RDS instance. If tables are encrypted by using TDE, errors occur when you restore the data of the RDS instance. Before you restore the data of the RDS instance by using a CSV file, you must decrypt the encrypted tables. For more information, see Decrypt a table.

Limits

When you restore the data to a self-managed MySQL instance by using the CSV file that you downloaded, take note of the following limits:

- Fields of the following data types are not supported: BIT, BINARY, VARBINARY, TINYBLOB, BLOB, MEDIUMBLOB, and LONGBLOB.

> **Note** If the CSV file that you downloaded contains fields of the BINARY data type, the fields are stored as hexadecimal representations. When you import the CSV file to the self-managed MySQL instance, the fields that are stored as hexadecimal representations are processed as strings. In this case, you must call the UNHEX function in the **LOAD DATA LOCAL INFILE** command to convert the fields from hexadecimal strings to raw binary strings.

- Fields of the following spatial data types are not supported: GEOMETRY, POINT, LINESTRING, POLYGON, MULTIPOINT, MULTILINESTRING, MULTIPOLYGON, and GEOMETRYCOLLECTION.
- We recommend that you restore the data of an RDS instance to a self-managed MySQL instance that runs the same database engine version as the RDS instance. If the database engine versions of the self-managed MySQL instance and the RDS instance are different, incompatibility issues may occur and cause the restoration to fail.

Procedure

In this example, an Ubuntu 20.04 LTS operating system is used. If you use other operating systems, you must use the corresponding commands.

1. Use the advanced download feature to convert the backup file into a CSV file and download the CSV file to your computer as a package. For more information, see Perform an advanced download.
2. Decompress the downloaded package.

   The decompression command is in the following format: `tar -izxvf <Name of the downloaded package>.tar.gz -C <Path to store the file that is obtained from the downloaded package>`.

   Example:

   ```
   tar -izxvf test1.tar.gz -C /home/mysql/data
   ```

3. Download the Python script.
4. Run the following command to grant permissions on the Python script `restore_from_downloads.py`:

   ```
   chmod +x ./restore_from_downloads.py
   ```

5. Restore data to the destination database by using the CSV file.

   The restoration command is in the following format: `python ./restore_from_downloads.py <Path of the CSV file> <Host on which the self-managed MySQL instance is deployed> <Port that is used to connect to the self-managed MySQL instance> <Username of the account that is used to connect to the self-managed MySQL instance> <Password of the account that is used to connect to the self-managed MySQL instance>`.

   Example:

   ```
   python ./restore_from_downloads.py /home/mysql/data/ 127.0.0.1 3306 zhtxxxxx "#txxxxx"
   ```

> **Note**
>
> - If the `Command 'python' not found` error message is displayed, you must check the version of Python that is installed in your system and the command that is used to run the Python script. For example, the command may be `python3 ./restore_from_downloads.py ~/mysql/data/ 127.0.0.1 3306 zhtxxxxx "#txxxxx"` .
> - If the self-managed MySQL instance contains a database that has the same name as the database whose data you want to import, the import fails.
> - If the username or password of the account that you want to use contains special characters, such as number signs (#) and spaces, you must enclose the username or password in double quotation marks (" "). For example, if the password is `#1234` , you must enter `"#1234"` in the preceding command.

15.7.4. Migrate the data of a self-managed MySQL instance to the cloud

This topic describes how to migrate the data of a self-managed MySQL instance to an ApsaraDB RDS for MySQL instance that runs the same MySQL version as the self-managed MySQL instance. You can perform a full backup on the self-managed MySQL instance, upload the full backup file to an Object Storage Service (OSS) bucket, and then restore the data of the full backup file to the destination RDS instance.

Prerequisites

- The self-managed MySQL instance runs MySQL 5.7 or MySQL 8.0.
- A full backup of the self-managed MySQL instance is complete. For more information, see Migrate the data of a self-managed MySQL 5.7 or MySQL 8.0 instance to an ApsaraDB RDS for MySQL instance.
- You have an Alibaba Cloud account.

Billing

After you import a full backup file into an RDS instance, you are not charged for the storage of the file within 24 hours. If the file is stored for longer than 24 hours, you are charged storage fees. For more information, see Storage fees.

> **Note** The preceding billing rules take effect on July 15, 2021.

Import the full backup file into the destination RDS instance

1. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**.
2. On the **Next** tab, click **3. Import Data**. In the Import Backup wizard, read the messages that are displayed and click Next until you enter the Data import step.

> **Note** The Import Backup wizard walks you through the migration process.
>
> - **1. Back Up Source Database**: Perform a full backup on the self-managed MySQL instance. For more information, see Migrate the data of a self-managed MySQL 5.7 or MySQL 8.0 instance to the cloud.
> - **2. Upload Backup Files to OSS**: Upload the full backup file to an OSS bucket. For more information, see Migrate the data of a self-managed MySQL 5.7 or MySQL 8.0 instance to the cloud.

3. Configure the following parameters and click **OK**.

Parameter	Description
Region	Select the region that you specified in Step 1. The selected region must be the region to which the OSS bucket that stores the full backup file belongs.
OSS Bucket	Select the OSS bucket that stores the full backup file of the self-managed MySQL instance. For more information, see Upload objects.
OSS file name	Select the full backup file that is stored as an object in the OSS bucket. You can enter the name of the full backup file in the **OSS file name** field to search for the file. This search feature supports fuzzy match and exact match.
Note	Enter a description that helps you identify the full backup file.
Zone	Select the zone to which the OSS bucket that stores the full backup file belongs. After you select a zone, ApsaraDB RDS creates a snapshot in the zone in single-digit seconds. This greatly reduces the time that is required to import the full backup file into the OSS bucket. **Note** After the full backup file is imported into ApsaraDB RDS, you can restore the data of the full backup file to a new RDS instance. The new RDS instance resides in the zone that you select.
Storage Capacity	Specify the amount of storage space that is required for the full backup file. Valid values: 20 to 6000: Unit: GB. **Note** By default, the amount of storage space required is 3 times the size of the full backup file. If the file size multiplied by 3 does not exceed 20 GB, this parameter defaults to 20.

> **Note** If ApsaraDB RDS is not authorized to access OSS, click **Authorize** in the lower part of the **3. Import Data** step. In the lower-left corner of the page that appears, click **Confirm Authorization Policy**.

4. After ApsaraDB RDS creates a task to check the full backup file, wait until the **Status** of the task changes from **Verifying** to **Completed**. You can view the status of the task on the **User Backups** tab.

> **Note** The time that is required to complete the task varies based on the status of the self-managed MySQL instance during the full backup. For example, if a large number of redo log records are generated from a large number of write operations or large transactions are run during the full backup, the time that is required to complete the task increases.

Restore the data of the full backup file to a new RDS instance

After the full backup file is imported into ApsaraDB RDS, you can restore the data of the full backup file to a new RDS instance. For more information, see the "Import the full backup file into the destination RDS instance" section in this topic.

> **Note** The full backup file is retained three days by default. After the retention period elapses, ApsaraDB RDS automatically deletes the full backup file. We recommended that you restore the data of the full backup file to a new RDS instance within the retention period. You can adjust the retention period. For more information, see Set the retention period of the full backup file.

1. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**.
2. Find the full backup file. Then, click **Restore** in the **Actions** column to the right of the **Backup ID/Name** column.
3. Configure the following parameters and click **Next: Instance Configuration.**

Parameter	Description
Zone of Primary Node	The zone to which the primary RDS instance belongs. **Note** If you did not select a zone for the OSS bucket that stores the full backup file when you import the file, this parameter is displayed. If you selected a zone for the OSS bucket that stores the full backup file when you import the file, this parameter is not displayed.
Storage type	○ **ESSD PL1**: An enhanced SSD (ESSD) of performance level 1 (PL1) is a regular ESSD. ○ **Standard SSD**: A standard SSD is an elastic block storage device that is designed based on the distributed storage architecture of Alibaba Cloud. You can store data on standard SSDs to separate computing from storage. **Note** For more information about storage types, see Storage types.

Parameter	Description
Instance Type	**General-purpose**: specifies the general-purpose instance family. A general-purpose instance exclusively occupies the allocated memory and I/O resources. However, it shares CPU and storage resources with the other general-purpose instances that are deployed on the same server. **Note** Each instance type supports a specific number of cores, memory capacity, maximum number of connections, and maximum IOPS. For more information, see Primary ApsaraDB RDS instance types.
Capacity	The storage capacity that is used to store data files, system files, binary log files, and transaction files in the RDS instance. You can adjust the storage capacity at a step size of 5 GB.

4. Configure the following parameters and click **Next: Confirm Order**.

Parameter	Description
Network Type	The network type of the RDS instance. Select **VPC**. A virtual private cloud (VPC) is an isolated network that provides higher security and better performance than the classic network. If you select the VPC network type, you must also specify the **VPC** and **vSwitch of Primary Node** parameters. **Note** The RDS instance and the Elastic Compute Service (ECS) instance that you want to connect must reside in the same VPC. Otherwise, the RDS instance and the ECS instance cannot communicate over an internal network.
Parameter Template	The parameter template that is used by the RDS instance. You can select a system parameter template or a custom parameter template. For more information, see Use a parameter template to configure the parameters of ApsaraDB RDS for MySQL instances.
Time Zone	The time zone of the RDS instance.
Table Name Case Sensitivity	Specifies whether table names in the RDS instance are case-sensitive. If table names in the self-managed MySQL instance are case-sensitive, we recommend that you select **Case-sensitive** to facilitate data migration between the RDS instance and the self-managed MySQL instance.

5. Confirm the configuration of the RDS instance in the **Parameters** section, configure the **Purchase Plan** parameter, read and select Terms of Service, and then click **Pay Now** to complete the payment.

Note ApsaraDB RDS requires 1 minute to 5 minutes to create the RDS instance. Wait until the instance is created.

Set the retention period of the full backup file

By default, the full backup file is retained for three days after it is imported into ApsaraDB RDS. For more information, see the "Import the full backup file into the destination RDS instance" section in this topic. You can adjust the retention period based on your business requirements.

> **Note** If you no longer need a full backup file, you can delete the file. For more information, see the "Delete the full backup file" section in this topic.

1. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**.
2. Click **Set Retention Period** in the **Expiration Time** column to the right of the **Backup ID/Name** column.
3. In the dialog box that appears, select a retention period from the drop-down list provided by ApsaraDB RDS. Alternatively, select **Custom Retention Period**, and then enter a retention period or click the up and down arrows to adjust the retention period.

   > **Note** **Expiration Time** shows the expiration time of the full backup file. If the expiration time exceeds 2099, it is displayed as **Permanent**.

4. Click **OK**.

Add tags to the full backup file

After the full backup file is imported into ApsaraDB RDS, you can add tags to the file.

1. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**.
2. Click **+Add Tag** in the **Tags** column to the right of the **Backup ID/Name** column.
3. Click **Create Tag**, enter the **Key** and **Value** of the tag, click **OK** to the right of the field to create the tag, and then click **OK** in the lower-right corner of the dialog box to add the tag.

   > **Note** If you have created a tag, click **Select Tag** to add the tag to the full backup file.

4. If you want to change the tag of the full backup file, move the pointer over the tag and click **Edit**. In the tooltip that appears, repeat Step 3 to create or select a tag.

Check the information about the binary log data of the full backup file

If data changes are made in the self-managed MySQL instance during the full backup, the generated full backup file contains the information about binary log data. You can use the binary log data to restore the incremental data. For more information, see Migrate the data of a self-managed MySQL 5.7 or MySQL 8.0 instance to the cloud.

1. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**.
2. Click **View Details** in the **Actions** column to the right of the **Backup ID/Name** column.
3. In the message that appears, check the information about the binary log data.

> **Note** **Binlog Information** includes the following information:
>
> - **Master_Log_File:**: the name of the first binary log file that shows incremental data.
> - **Master_Log_Position:**: the first log record that shows incremental data in the first binary log file showing incremental data.

Delete the full backup file

If you no longer need the full backup file, you can delete the file to save costs.

1. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**.
2. Click **Delete** in the **Actions** column to the right of the **Backup ID/Name** column.
3. In the message that appears, click **OK**.

Other features

- **Filter Columns**: You can click the icon to show or hide columns on the **User Backups** tab. By default, the tab shows all columns.
  i. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**. On the page that appears, click the **User Backups** tab.
  ii. In the upper-right corner of the tab, click the  icon. In the dialog box that appears, select the columns that you want to display or hide, and click the  or  icon to move the columns.

  > **Note** The columns in the left-side list are hidden, and the columns in the right-side list are displayed.

  iii. Click **OK**.
- **Export Instance List** : You can click the icon to export the backup information as a CSV file.
  i. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups**. On the page that appears, click the **User Backups** tab.
  ii. In the upper-right corner of the tab, click the  icon to export the backup information.
- **Refresh**: You can click the icon to refresh the User Backups tab.
  i. Log on to the ApsaraDB RDS console. In the top navigation bar, select a region. In the left-side navigation pane, click **Backups** and then click **User Backups** tab.
  ii. Click the  icon to refresh the User Backups tab.

Related operations

- Import a full backup file
- Modify a full backup file

- Query full backup files
- Delete a full backup file

Related information

- Migrate the data of a self-managed MySQL 5.7 or MySQL 8.0 instance to an ApsaraDB RDS for MySQL instance

16.Read-only instances

16.1. Overview of read-only ApsaraDB RDS for MySQL instances

This topic provides an overview of read-only ApsaraDB RDS for MySQL instances. If your database system receives a small number of write requests but a large number of read requests, a single primary RDS instance may be overwhelmed by the read requests and your workloads may be interrupted. To offload read requests from the primary RDS instance, you can create one or more read-only RDS instances. Read-only RDS instances help increase the read capability of your database system and the throughput of your application.

For more information about read-only RDS instances that run a different database engine, see the following topics:

- Overview of read-only ApsaraDB RDS for SQL Server instances
- Overview of read-only ApsaraDB RDS for PostgreSQL instances

Background information

When a read-only RDS instance is being created, ApsaraDB RDS replicates data from the secondary RDS instance to the read-only RDS instance. Each read-only RDS instance has the same data as the primary RDS instance. After the data on the primary RDS instance is updated, ApsaraDB RDS immediately replicates the data updates to all the read-only RDS instances. You can specify a replication latency on each read-only RDS instance. For more information, see Set the data replication latency of a read-only ApsaraDB RDS for MySQL instance.

> **Note** Each read-only RDS instance runs in the high-availability architecture that allows the read-only RDS instance to have a secondary RDS instance as a standby.

Prerequisites

The primary RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability Edition or RDS Enterprise Edition
- MySQL 5.7 on RDS High-availability Edition or RDS Enterprise Edition
- MySQL 5.6

> **Note** If the primary RDS instance runs MySQL 5.7 on RDS Enterprise Edition but you cannot create read-only RDS instances, you must submit a .

Billing

Read-only RDS instances support both the pay-as-you-go billing method and the subscription billing method. For more information about the prices of read-only RDS instances, see Read-only ApsaraDB RDS instance types.

Precautions

- If the primary RDS instance uses the subscription billing method and you want to create a

subscription read-only RDS instance, you can configure the read-only RDS instance to have the same subscription period as the primary RDS instance.

- If the primary RDS instance is locked due to expiration, you can still access the read-only RDS instances of the primary RDS instance. However, the statuses of the read-only RDS instances change to **Running (Primary Instance Locked)**.
- After the primary RDS instance is released, the subscription read-only RDS instances of the primary RDS instance are automatically refunded and released. However, the pay-as-you-go read-only RDS instances of the primary RDS instance are directly released.

Usage notes

- Read-only RDS instances support both the pay-as-you-go billing method and the subscription billing method. The pay-as-you-go billing method is flexible and allows you to release your read-only RDS instances when you no longer need the instances. The subscription billing method is cost-effective for long-term commitments.
- Read-only RDS instances reside in the same region as the primary RDS instance, but optionally in different zones.
- The specifications of read-only RDS instances can differ from the specifications of the primary RDS instance. You can change the specifications of read-only RDS instances at any time. We recommend that the specifications of read-only RDS instances be higher than or equal to the specifications of the primary RDS instance. If the specifications of a read-only RDS instance are lower than the specifications of the primary RDS instance, the read-only RDS instance may encounter issues such as high latency or heavy load.
- The network types of read-only RDS instances can differ from the network type of the primary RDS instance. For more information, see Change the network type of an ApsaraDB RDS for MySQL instance.
- The databases and accounts on read-only RDS instances are synchronized from the primary RDS instance. You do not need to manage databases or accounts on read-only RDS instances.
- When you create a read-only RDS instance, ApsaraDB RDS replicates the IP address whitelists of the primary RDS instance to the read-only RDS instance. However, the IP address whitelists of the read-only RDS instance are independent of the IP address whitelists of the primary RDS instance. For more information about how to modify the IP address whitelists of a read-only RDS instance, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.
- Read-only RDS instances support monitoring and alerting. You can monitor near 20 metrics, such as disk usage, IOPS, number of connections, CPU utilization, and network traffic.

Limits

- A maximum of 10 read-only RDS instances can be created.

  > **Note** If you want to create more than 10 read-only RDS instances, you must submit a .

- You cannot configure backup policies or manually create backups for read-only RDS instances. These operations are performed on primary RDS instances.
- The following limits are imposed on the data restoration of read-only RDS instances:
  - You cannot create a temporary RDS instance that is used to restore the data of a read-only RDS instance from a backup file or to a specific point in time. In addition, you cannot overwrite the data of a read-only RDS instance by using a data backup file.
  - After a read-only RDS instance is created, you cannot overwrite the data of its primary RDS instance by using a data backup file.

- You cannot migrate data to read-only RDS instances.
- You cannot create or delete databases on read-only RDS instances.
- You cannot create or delete accounts, grant permissions to accounts, or change the passwords of accounts on read-only RDS instances.

Create a read-only RDS instance

Create a read-only ApsaraDB RDS for MySQL instance

FAQ

- After I create accounts on my primary RDS instance, can I manage the accounts on the read-only RDS instances of my primary RDS instance?

  No, although the accounts created on your primary RDS instance are synchronized to the read-only RDS instances, you cannot manage the accounts on the read-only RDS instances. The accounts have only the read permissions on the read-only RDS instances.

- Can I pause the billing for my read-only RDS instances? And can I set the read weights of my read-only RDS instances to 0 to stop the billing?

  No, you cannot pause the billing for your read-only RDS instances. If you no longer need your read-only RDS instances, we recommend that you immediately release the instances. For more information, see Release or unsubscribe from an ApsaraDB RDS for MySQL instance.

16.2. Create a read-only ApsaraDB RDS for MySQL instance

This topic describes how to create a read-only ApsaraDB RDS for MySQL instance. Read-only RDS instances help increase the read capability of your database system and the throughput of your application. Each read-only RDS instance is a replica of the primary RDS instance. This indicates that each read-only RDS contains the same data as the primary RDS instance. Data updates on the primary RDS instance are automatically synchronized to each read-only RDS instance.

For more information about how to create a read-only RDS instance that runs a different database engine, see the following topics:

- Create a read-only ApsaraDB RDS for SQL Server instance
- Create a read-only ApsaraDB RDS for PostgreSQL instance

For more information about read-only RDS instances, see Overview of read-only ApsaraDB RDS for MySQL instances.

Prerequisites

The primary RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability Edition or RDS Enterprise Edition
- MySQL 5.7 on RDS High-availability Edition or RDS Enterprise Edition
- MySQL 5.6

> **Note** If the primary RDS instance runs MySQL 5.7 on RDS Enterprise Edition but you cannot create read-only RDS instances, you must submit a .

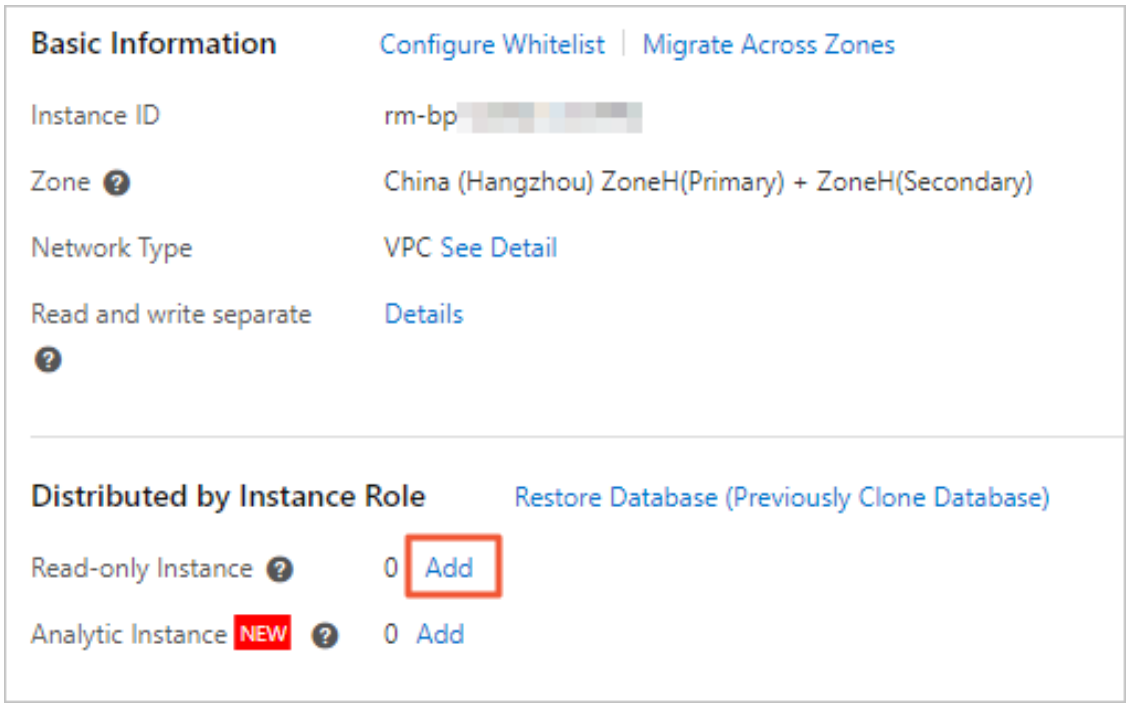
Precautions

- You can create read-only RDS instances for the primary RDS instance. You cannot convert existing RDS instances to read-only RDS instances.
- When a read-only RDS instance is being created, ApsaraDB RDS replicates data from the secondary RDS instance to the read-only RDS instance. This prevents interruptions to your workloads on the primary RDS instance.
- After the primary RDS instance is released, the subscription read-only RDS instances of the primary RDS instance are automatically refunded and released, and the pay-as-you-go read-only RDS instances of the primary RDS instance are directly released.
- A read-only RDS instance does not inherit the parameter settings of the primary RDS instance. ApsaraDB RDS generates default parameter settings for each read-only RDS instance. You can modify the parameter settings of a read-only RDS instance in the ApsaraDB RDS console.
- You can create up to 10 read-only RDS instances for a primary RDS instance. If you want to create more than 10 read-only RDS instances, you must submit a .
- Read-only RDS instances support both the pay-as-you-go billing method and the subscription billing method. For more information about the prices of read-only RDS instances, see Read-only ApsaraDB RDS instance types.
- When you create a read-only RDS instance, you must select a virtual private cloud (VPC) that meets the specified requirements. For more information, see FAQ.

Create a read-only RDS instance

1.
2. In the **Distributed by Instance Role** section of the Basic Information page, click **Add** to the right of **Read-only Instance**.

> **Note** If you are using the original ApsaraDB RDS console, click **Create Read-only Instance** in the Distributed by Instance Role section of the Basic Information page.



3. Configure the following parameters.

Parameter	Description
Billing Method	○ **Subscription**: A subscription instance is an instance for which you pay an upfront fee. For long-term use, the subscription billing method is more cost-effective than the pay-as-you-go billing method. You are offered lower prices for longer subscription periods. ○ **Pay-As-You-Go**: A pay-as-you-go instance is charged per hour based on your actual resource usage. For short-term use, we recommend that you select the pay-as-you-go billing method. If you no longer need a pay-as-you-go instance, you can release the instance to reduce costs.
Zone	A zone is an independent geographical location in a region. Zones in the same region do not have substantive differences.
Instance Type	○ **General-purpose (Entry-level)**: allows you to select a general-purpose instance type. A general-purpose RDS instance exclusively occupies the allocated memory and I/O resources, but shares CPU and storage resources with the other general-purpose RDS instances that are deployed on the same server. ○ **Dedicated**: allows you to select a dedicated instance type or a dedicated host instance type. A dedicated RDS instance exclusively occupies the allocated CPU, memory, storage, and I/O resources. Dedicated host instance types provide the highest specifications in the dedicated instance family. A dedicated host RDS instance occupies all CPU, memory, storage, and I/O resources on the physical host where the RDS instance is deployed. **Note** Each instance type supports a specific number of cores, memory capacity, maximum number of connections, and maximum IOPS. For information about how to connect to a host, see Primary ApsaraDB RDS instance types.
Capacity	The maximum amount of storage capacity that is provisioned to store data files, system files, binary log files, and transaction files in the read-only RDS instance. You can adjust the storage capacity at a step size of 5 GB. **Note** The storage capacity of the read-only RDS instance must be equal to or greater than the storage capacity of the primary RDS instance to which the read-only RDS instance is attached.

4. Click **Next: Instance Configuration** and configure the following parameters.

Parameter	Description
Network Type	○ **Classic Network**: the traditional type of network. ○ **VPC**: the recommended type of network. A VPC is an isolated virtual network that provides higher security and higher performance than the classic network. If you select the VPC network type, you must also specify the **VPC** and **vSwitch of Primary Node** parameters. **Note** The network type of the RDS instance must be the same as the network type of the Elastic Compute Service (ECS) instance that you want to connect. If the RDS instance and the ECS instance both reside in VPCs, these instances must reside in the same VPC. If the RDS instance and the ECS instance reside in different VPCs, these instances cannot communicate over an internal network.
Resource Group	The resource group to which the read-only RDS instance belongs.

5. Click **Next: Confirm Order**.
6. Confirm the settings in the **Parameters** section, specify the **Purchase Plan** and **Duration** parameters, read and select Terms of Service, click **Pay Now**, and then complete the payment. You must specify the Duration parameter only when the read-only RDS instance uses the subscription billing method.

> **Note**
> - If the primary RDS instance is charged based on the subscription billing method and you want to purchase a subscription read-only RDS instance, you can select **Consistent with Primary Instance** in the **Duration** section. This ensures that the read-only RDS instance has the same subscription period as the primary RDS instance.
> - If the read-only RDS instance that you create is charged based on the subscription billing method, we recommend that you select **Auto-Renew Enabled**. This reliefs the need to renew the read-only RDS instance on a regular basis and prevents interruptions to your workloads on the read-only RDS instance due to overdue payments.
> - If the primary RDS instance supports the dedicated proxy feature, you can select **MySQL Dedicated Proxy Service (Paid Service)** in the Confirm Order step for the instance. For more information, see What are database proxies?

View a read-only RDS instance

To view a read-only RDS instance on the Instances page, perform the following steps:

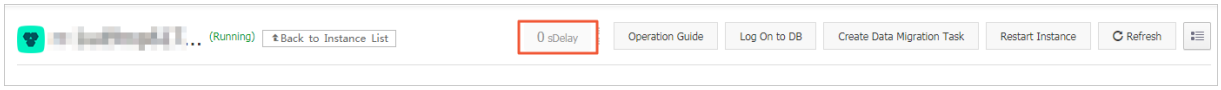
1. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Instances**. In the top navigation bar, select the region where the read-only RDS instance resides.
2. Find the read-only RDS instance and click the instance ID.

To view a read-only RDS instance on the Basic Information page of the primary RDS instance, perform the following steps:

1. Log on to the ApsaraDB RDS console. In the left-side navigation pane, click **Instances**. In the top navigation bar, select the region where the primary RDS instance resides.
2. Find the primary RDS instance and click the ID of the instance.
3. In the Distributed by Instance Role section of the **Basic Information** page, move the pointer over the number of read-only RDS instances and click the ID of the read-only RDS instance that you want to view.

View the latency of data replication to a read-only RDS instance

A read-only RDS instance may synchronize data from the primary RDS instance at a specific latency. You can go to the Basic Information page of a read-only RDS instance to view the latency of data replication to the instance.



FAQ

- When I create a read-only RDS instance, why am I unable to select a specific zone?

  The zone that you select does not have available resources. Select another zone and try again. This does not affect the read-only RDS instance.

- When I create a read-only RDS instance, can I select a VPC that is different from the VPC of the primary RDS instance?
  - If the primary RDS instance resides in the classic network, you can select the classic network or a VPC. The selected VPC can be different from the VPC of the primary RDS instance.

- If the primary RDS instance resides in a VPC:
  - You can select the classic network.
  - If local SSDs are used, you can select a VPC. The selected VPC can be different from the VPC of the primary RDS instance.
  - If standard SSDs or enhanced SSDs (ESSDs) are used, you can select only the VPC of the primary RDS instance.

- When I create a read-only RDS instance, ApsaraDB RDS replicates data from the secondary RDS instance to the read-only RDS instance. This prevents interruptions to your workloads that are run on the primary RDS instance. What is a secondary RDS instance?

  If you are using the RDS High-availability Edition, your database system consists of a primary RDS instance and a secondary RDS instance. These instances work in the classic high-availability architecture. If the primary RDS instance becomes faulty, your database system fails over to the secondary RDS instance.

Related operations

Operation	Description
Create a read-only instance	Creates a read-only ApsaraDB RDS instance.

16.3. Set the data replication latency of a read-only ApsaraDB RDS for MySQL instance

This topic describes how to set the latency at which a read-only ApsaraDB RDS for MySQL instance synchronizes data from its primary RDS instance.

Prerequisites

The primary RDS instance to which the read-only RDS instance belongs does not run RDS Enterprise Edition.

Precautions

After you set the data replication latency of a read-only RDS instance, you cannot add the instance to the read/write splitting link. Before you can add the instance to the read/write splitting link, you must set the data replication latency of the instance to 0. For more information, see Enable the read/write splitting feature for an ApsaraDB RDS for MySQL instance (shared proxy).

Procedure

1. Go to the **Basic Information** page of the read-only RDS instance.
   i.
   ii. Find the read-only RDS instance and click its ID.
2. In the left-side navigation pane, click **Service Availability**.
3. Click **Set Delayed Replication**.

4. In the dialog box that appears, set the data replication latency and click **OK**.

> **Note** The data replication latency is measured in seconds. The default value is 0. If you set the data replication latency to 0 seconds, the primary RDS instance sends operation logs to the read-only RDS instance after the logged operations are complete. After the read-only RDS instance receives the operation logs, it immediately performs the logged operations.

17.Disaster recovery instances

17.1. Create a disaster recovery ApsaraDB RDS for MySQL instance

This topic describes how to create a disaster recovery ApsaraDB RDS for MySQL instance for a primary ApsaraDB RDS for MySQL instance. A disaster recovery RDS instance resides in a different region than the region where its primary RDS instance resides. If your primary RDS instance is used for business scenarios in which high data reliability or compliance with financial regulations is required, you can create a disaster recovery RDS instance to increase data reliability.

Prerequisites

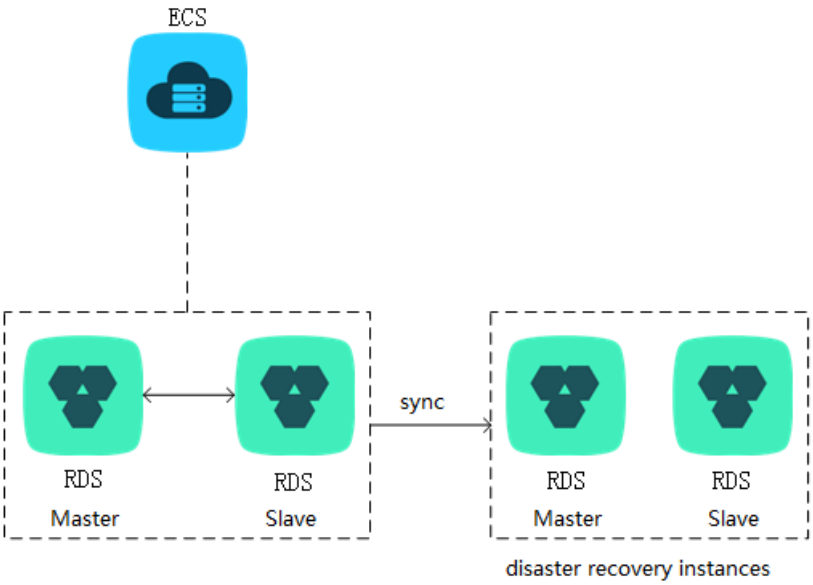
- Your primary RDS instance runs one of the following MySQL versions and RDS editions:
  - MySQL 8.0 on RDS High-availability Edition or RDS Enterprise Edition
  - MySQL 5.7 on RDS High-availability Edition or RDS Enterprise Edition
  - MySQL 5.6
- Your primary RDS instance resides in the classic network.
- Your primary RDS instance resides in one of the following regions: China (Hangzhou), China (Shanghai), China (Qingdao), China (Beijing), China (Shenzhen), China (Hong Kong), Singapore (Singapore), and US (Virginia).

Context

A primary RDS instance and its disaster recovery RDS instance synchronize data with each other in real time by using Data Transmission Service (DTS). The primary RDS instance and the disaster recovery RDS instance are configured as a high-availability architecture. If the primary RDS instance and the secondary RDS instance are inaccessible due to unexpected exceptions such as natural disasters, the database system fails over to the disaster recovery RDS instance. In this case, the disaster recovery RDS instance is promoted to run as the new primary RDS instance. After you update the endpoint information on the application that is connected to the database system, the application immediately connects to the new primary RDS instance.

In the DTS console, you can specify the synchronization settings for a disaster recovery RDS instance. For example, you can change the objects that you want to synchronize, specify the synchronization speed, and configure the synchronization link to report alerts at a specified latency. For more information, see What is DTS?

The following figure shows the topology of a database system that contains a disaster recovery RDS instance.



A disaster recovery RDS instance has the following characteristics:

- A disaster recovery RDS instance is connected over an independent endpoint. You can configure an application to connect to the endpoint of a disaster recovery RDS instance.
- A disaster recovery RDS instance runs in a high-availability architecture.
- You are charged for a disaster recovery RDS instance based on the pay-as-you-go billing method.
- You can configure IP address whitelists and manage accounts on a disaster recovery RDS instance.

Billing

- By default, a disaster recovery RDS instance has the same configuration as its primary RDS instance and uses the **pay-as-you-go** billing method. For more information, visit the ApsaraDB RDS buy page.
- By default, the DTS data synchronization link between a disaster recovery RDS instance and its primary RDS instance uses the **small** specification and the **pay-as-you-go** billing method. For more information, visit the DTS Price Calculator.

Limits

- A disaster recovery RDS instance does not support backup and restoration, data migration, database management, public endpoints, or endpoint modifications.
- A disaster recovery RDS instance does not synchronize database deletion operations from its primary RDS instance. After you delete a database from the primary RDS instance, you must log on to the disaster recovery RDS instance and then execute SQL statements to delete the database.

Procedure

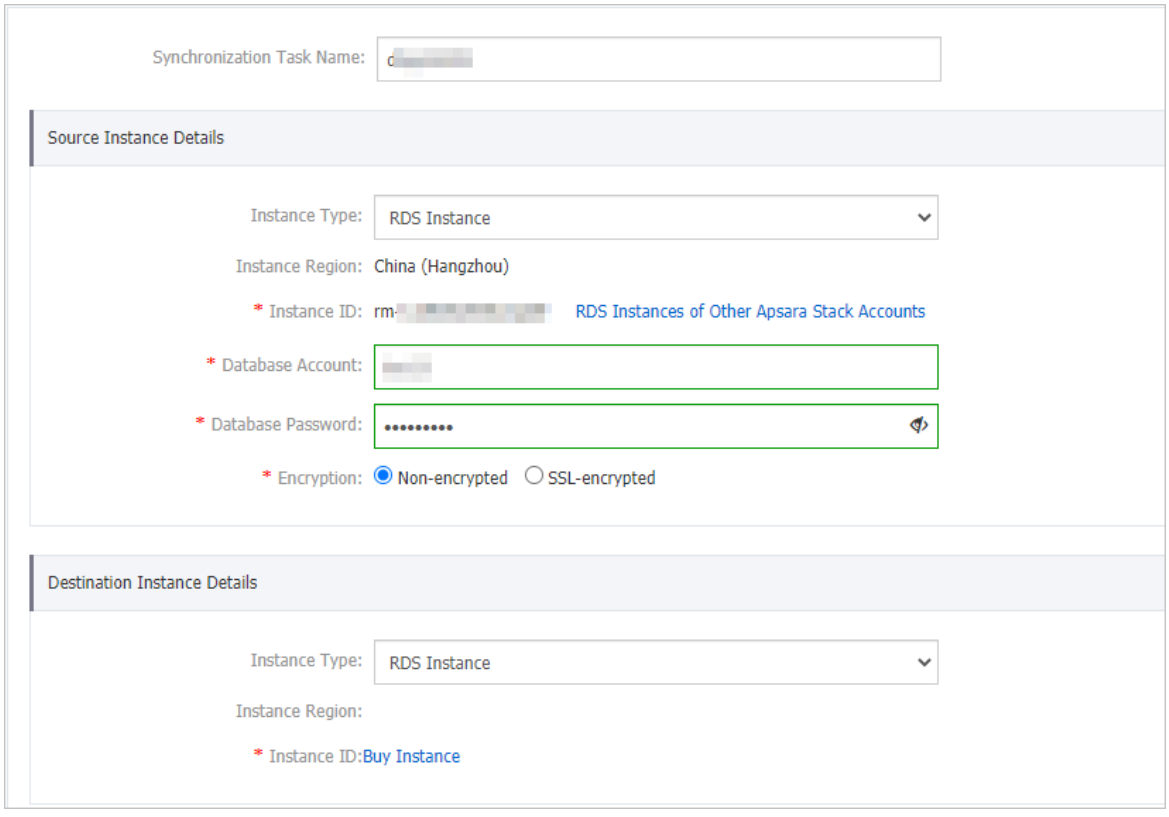
1.
2. In the **Distributed by Instance Role** section of the Basic Information page, click **Add** to the right of **DR Instance**. If you are using the original ApsaraDB RDS console, click **Add Guard** in the Distributed by Instance Role section of the Basic Information page.

> **Note** If the preceding entry points cannot be found, you must check whether the primary RDS instance meets all prerequisites that are specified in this topic.

3. In the **Create Data Synchronization Task** wizard, configure the Database Account and Database Password parameters.

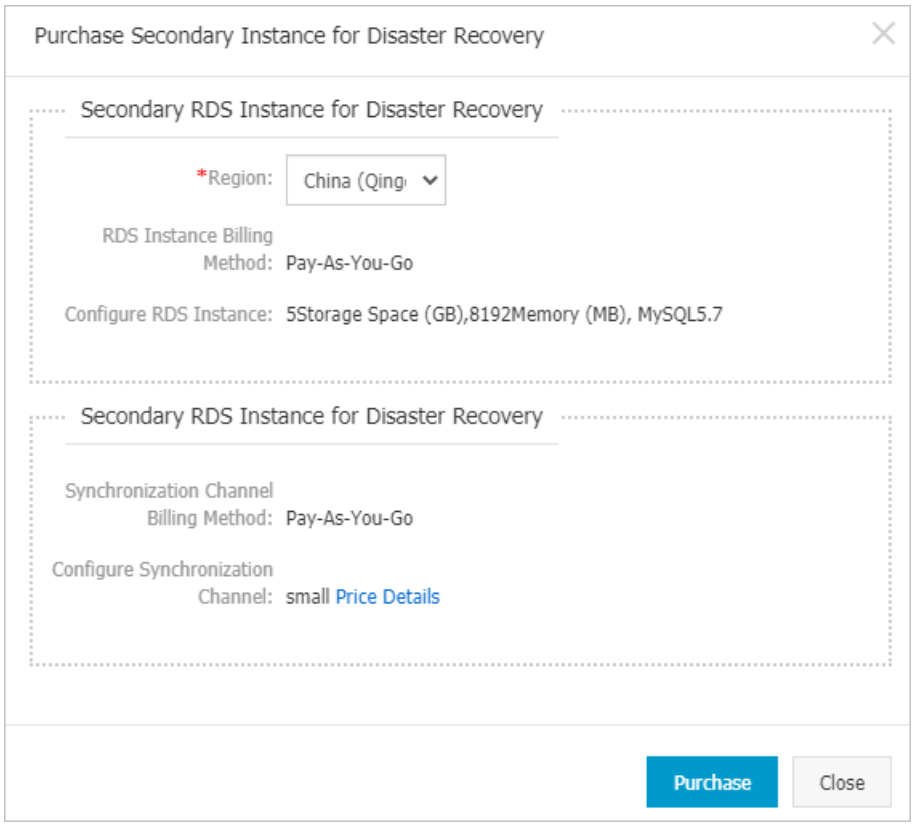
> **Note**
> - The account must have the REPLICATION SLAVE permission, the REPLICATION CLIENT permission, and the SELECT permission on all objects that you want to synchronize.
> - If you are using MySQL 5.6, you do not need to set the Database Account parameter or the Database Password parameter. You can skip this step.

4. Click **Buy Instance**.



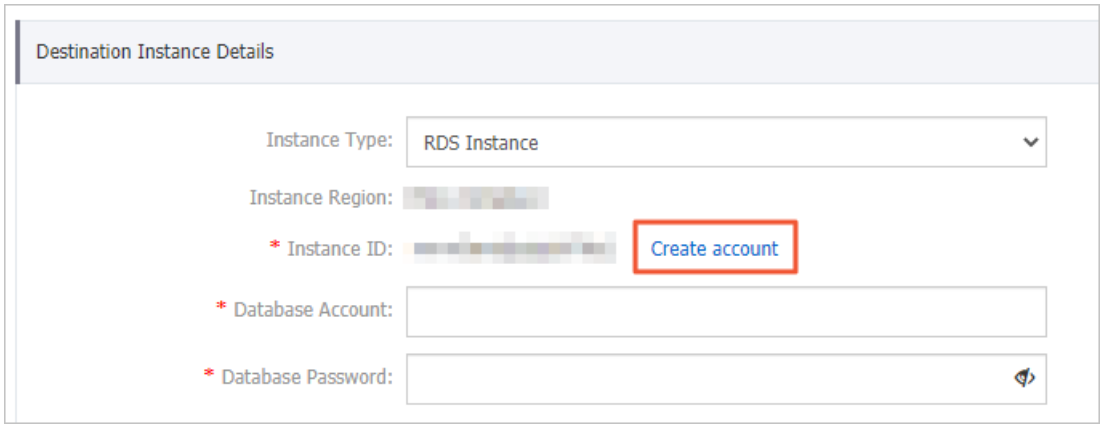
5. In the **Purchase Secondary Instance for Disaster Recovery** dialog box, select a region and click **Purchase**.

> **Note**
>
> - You can specify only the region where the disaster recovery RDS instance resides. The disaster recovery RDS instance supports only the pay-as-you-go billing method. All the other settings of the disaster recovery RDS instance are the same as the settings of the primary RDS instance. After the disaster recovery RDS instance is created, you can change its specifications in the ApsaraDB RDS console.
> - ApsaraDB RDS requires a few minutes to create the disaster recovery RDS instance. Do not close the dialog box until the disaster recovery RDS instance is created. If you close the dialog box before the disaster recovery RDS instance is created, the disaster recovery RDS instance may fail to be created.
> - The billing method of the DTS synchronization link defaults to **pay-as-you-go**. When you create the DTS synchronization link, you cannot select the subscription billing method for the DTS synchronization link. However, after the DTS synchronization link is created, you can change the billing method of the DTS synchronization link to **subscription** to reduce costs. For more information, see Switch the billing method from pay-as-you-go to subscription.



6. After the disaster recovery RDS instance is created, click **Create account** next to Instance ID in the Destination Instance Details section to create a privileged account that is used to synchronize data.

> **Note** If you are using MySQL 5.6, a privileged account is automatically created. Therefore, you can skip this step.

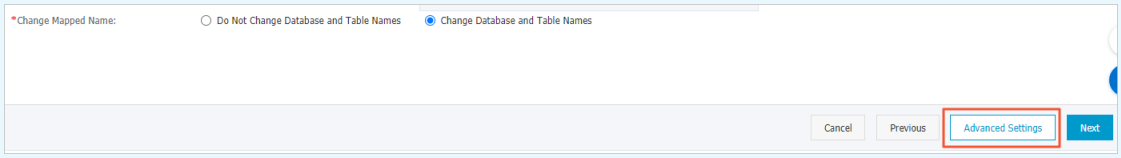


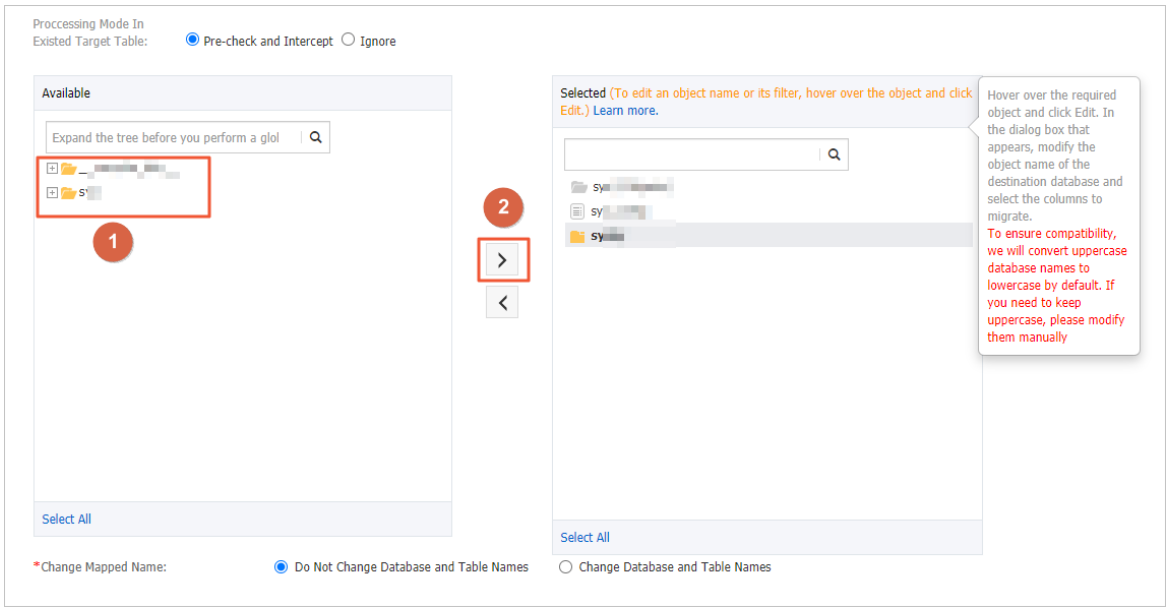
7. In the **Create Data Synchronization Task** wizard, configure the Database Account and Database Password parameters in the Destination Instance Details section. Then, click **Set Whitelist and Next**.

> **Note** If you are using MySQL 5.6, click **Set Whitelist and Next**. Wait until the account is created. Then, click **Next**.

8. In the **Available** section, select the objects that you want to synchronize. Click the **>** icon to move the selected objects to the **Selected** section. Then, click **Next**.

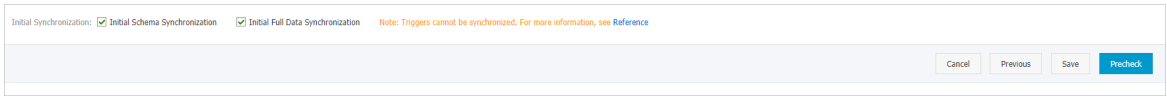
> **Note** If you want to change the names of multiple tables at a time, you can select the tables, select **Change Database and Table Names**, and then click **Advanced Settings**.





9. Set the **Initial Synchronization** parameter and click **Precheck**.

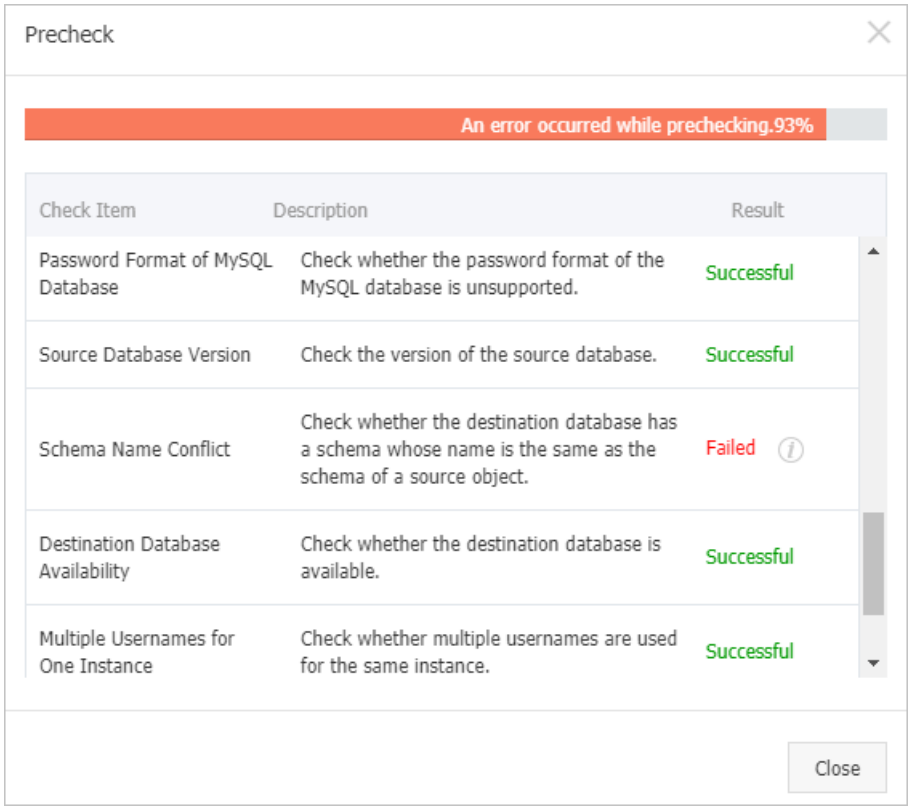
> **Note** During the **Initial Synchronization** process, DTS synchronizes the schemas and data of the selected objects from the primary RDS instance to the disaster recovery RDS instance. Later, DTS synchronizes the incremental data of the primary RDS instance to the disaster recovery RDS instance based on the schemas and the data. You can select the Initial Schema Synchronization option or the Initial Full Data Synchronization option. If you synchronize data for the first time, you must select both options.



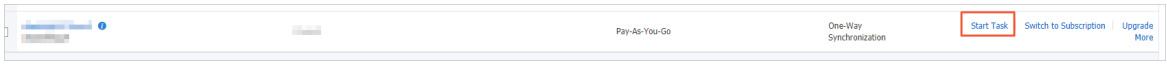
10. View the check items. This step is required only when the precheck fails. If the precheck is successful, go to Step 14.

    Click the

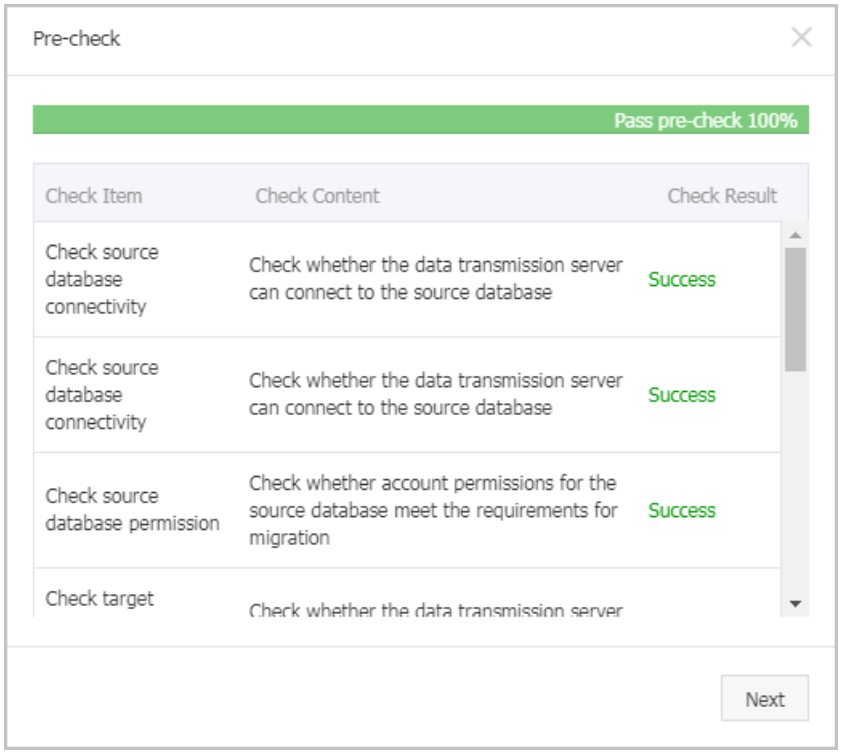
    icon next to **Failed** for each failed check item to view the details about the failure. Then, troubleshoot the issues that cause the failure.



11. On the **Synchronization Tasks** page, find the created synchronization task and click **Start Task**.



12. After you verify that the precheck is successful, click **Close**. The synchronization task starts.



13. On the **Synchronization Tasks** page, view and manage the synchronization task that is created. For example, you can change the objects that you want to synchronize, specify the monitoring and alerting settings, and change the synchronization speed. For more information, see What is DTS?

> **Note** To ensure that the data on the disaster recovery RDS instance is up-to-date, do not pause the synchronization task.

FAQ

- What benefits does the disaster recovery RDS instance of my database system bring?

  If the primary RDS instance and the secondary RDS instance are inaccessible due to unexpected exceptions such as natural disasters, your database system fails over to the disaster recovery RDS instance. In this case, the disaster recovery RDS instance is promoted to run as the new primary RDS instance. After you update the endpoint information on the connected application, the application immediately connects to the new primary RDS instance.

  > **Note** Data that is written to the new primary RDS instance cannot be synchronized to the original primary RDS instance.

- Can I select the subscription billing method for disaster recovery RDS instances?

  No, disaster recovery RDS instances support only the pay-as-you-go billing method.

- Why do I find an account named dtssyncwriter that I did not create?

  If you are using MySQL 5.6, an account named dtssyncwriter is automatically created when you create a disaster recovery RDS instance. The dtssyncwriter account is used by DTS to synchronize data. Do not modify or delete this account. If you modify or delete this account, synchronization errors occur.

17.2. Switch an ApsaraDB RDS for MySQL instance from the disaster recovery role to the primary role

This topic describes how to switch an ApsaraDB RDS for MySQL instance from the disaster recovery role to the primary role. The disaster recovery RDS instance resides in a different region than the primary RDS instance. If both the primary and secondary RDS instances are unavailable due to natural disasters, you can switch the disaster recovery RDS instance to the primary role. Then, you must immediately update the endpoint configuration on your application. This minimizes the downtime of your database system.

Prerequisites

A disaster recovery RDS instance is created. For more information, see Create a disaster recovery ApsaraDB RDS for MySQL instance.

Precautions

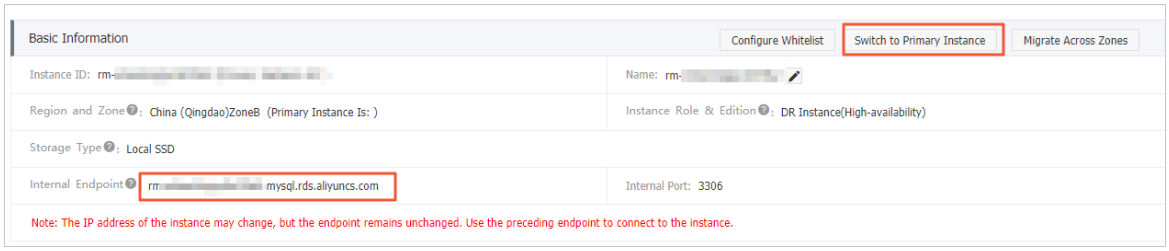
After the disaster recovery RDS instance is switched to the primary role, you cannot switch the instance back to the disaster recovery role.

Procedure

1. Log on to the ApsaraDB RDS console.
2. In the left-side navigation pane, click **Instances**. In the top navigation bar, select the region where the disaster recovery RDS instance resides.

   > **Note** The primary and disaster recovery RDS instances reside in different regions. You must select the region where the disaster recovery RDS instance resides.

3. Find the disaster recovery RDS instance and click its ID.
4. Click **Switch to Primary Instance**. In the message that appears, click **OK**.



> **Note** After the switchover is complete, you must update the endpoint configuration on your application. This update requires you to replace the endpoint of the original primary RDS instance with the endpoint of the new primary RDS instance.

17.3. Billing cases for disaster recovery instances

This topic describes how to calculate fees incurred by disaster recovery instances.

You use an ApsaraDB RDS for MySQL 5.6 High-availability Edition instance and select the subscription billing method. The following table describes the billing items.

> **Note** The prices provided in this topic are for reference only. If you want to know further details about the actual prices, go to the ApsaraDB for RDS console.

Billing item	Description
Instance	The subscription fee incurred for an ApsaraDB for RDS instance. The subscription fee for a year is USD 2,400.
Storage capacity	You are charged for the storage space of the RDS instance. The billing method is subscription. It is the same as the billing method of the instance. The price is USD 0.13/GB/month. The total cost of 2,000 GB of storage space is calculated as 2,000 x 0.13 x 12=USD 3,120.

If you do not select other billing items, the total cost is calculated as 2,400+3,120=USD 5,520 per year.

In this scenario, the business has high requirements for data reliability. Geo-disaster recovery instances must be deployed to enhance data reliability. We recommend that you use the disaster recovery instance of ApsaraDB for RDS. This can be directly created based on the original ApsaraDB RDS for MySQL instance.

> **Note** For more information about how to create a disaster recovery instance, see Create a disaster recovery ApsaraDB RDS for MySQL instance.

After you purchase a disaster recovery instance, you must pay for the billing items described in the following table.

> **Note** The prices provided in this topic are for reference only. If you want to know further details about the actual prices, go to the ApsaraDB for RDS console.

Billing item	Description
Disaster recovery instance	The pay-as-you-go fee cost by an ApsaraDB for RDS disaster recovery instance. The specifications are the same as those for the primary instance. The pay-as-you-go price is USD 0.35/hour. The cost of a year is 0.35 x 24 x 365≈USD 3,000.
Storage capacity	You are charged for the storage space of the disaster recovery instance. The billing method is pay-as-you-go. It is the same as the billing method of the disaster recovery instance. The price is USD 0.0003/GB/hour. The total cost of 2,000 GB of storage space is calculated as 2,000 x 0.0003 x 24 x 365≈USD 5,200.

Billing item	Description
DTS synchronization	The data of the primary instance is synchronized to the disaster recovery instance through Data Transmission Service (DTS). Therefore, you are charged based on the cost of DTS synchronization. The billing method is pay-as-you-go. The price is USD 0.349/hour. The cost of a year is 0.349 x 24 x 365≈USD3,000. **Note** On the **Basic Information** page of the disaster recovery instance, you can click **View Synchronous Job Details** in the upper-left corner of the Status section to go to the DTS console and view the synchronization task details.

When you use a disaster recovery instance, the total cost of a year is calculated as 3,000+5,200+3,000=USD 11,200. Compared with user-created disaster recovery instances, ApsaraDB for RDS disaster recovery instances provide high stability and low cost for hardware and network traffic.

18.Performance optimization and diagnosis

18.1. Troubleshoot slow SQL statements on an ApsaraDB RDS for MySQL instance

This topic describes how to troubleshoot the issues that cause slow SQL statements on an ApsaraDB RDS for MySQL instance. If your business scenario remains unchanged, the architecture design and the index design affect the query performance of your RDS instance. If these designs are suitable, the query performance of your RDS instance increases. If these designs are unsuitable, some SQL statements may be executed at low speeds.

Troubleshoot slow SQL statements caused by SQL exceptions

- Cause and symptom

  SQL exceptions may occur due to various issues, such as an unsuitable schema design, missing indexes, and an excessively large number of rows that need to be scanned.

  You can log on to the ApsaraDB RDS console and choose **SQL Explorer and Audit** in the left-side navigation pane. Then, you can view information about the executed SQL statements. The information includes the time that is required to execute each slow SQL statement and the number of times that each slow SQL statement is executed.

- Solution

  Optimize the SQL statements that you want to execute based on your business scenario. For more information, see SQL optimization.

Troubleshoot slow SQL statements caused by instance limits

- Cause and symptom

  In most cases, your RDS instance reaches its maximum performance due to the following reasons:

  - Your workloads continue to increase. However, the storage capacity is not scaled.
  - The physical host on which your RDS instance resides ages. This decreases the performance of your RDS instance.
  - The amount of data continues to increase, and the data structure changes. As a result, the speed at which ApsaraDB RDS executes some SQL statements becomes low.

  You can log on to the ApsaraDB RDS console and choose **Monitoring and Alerts** in the left-side navigation pane. Then, on the **Standard Monitoring** tab, you can click **Resource Monitoring** to view the resource usage of your RDS instance. If the values of all resource usage metrics are close to 100%, your RDS instance has reached its maximum performance.

- Solution

We recommend that you use SysBench to benchmark the maximum performance of your RDS instance. In most cases, the queries per second (QPS) and the transactions per second (TPS) do not exceed the maximum performance even if you run complex queries. For more information, see Test guidelines.

If your RDS instance has reached its maximum performance, we recommend that you upgrade the instance. For more information, see Change the specifications of an ApsaraDB RDS for MySQL instance.

Troubleshoot slow SQL statements caused by version upgrades

- Cause and symptom

  When you upgrade your RDS instance, the query plans for SQL statements may change. The following join types that are supported by query plans are sorted in descending order based on efficiency: system, const, eq_ref, ref, fulltext, ref_or_null, index_merge, unique_subquery, index_subquery, range, index, and all. For more information, see the official MySQL documentation.

  If your application frequently resends query requests that specify range and index joins but ApsaraDB RDS processes these query requests at low speeds, a number of SQL statements are parallelized. In this case, your application releases threads at low speeds. As a result, the connections in the connection pool are depleted. This affects all the workloads on your RDS instance.

  You can log on to the ApsaraDB RDS console and choose **Monitoring and Alerts** in the left-side navigation pane. Then, on the **Standard Monitoring** tab, you can click **Resource Monitoring** to view the connections to your RDS instance.

- Solution

  Analyze the index usage and the number of rows that need to be scanned. The analysis is based on the specified query plans. Then, estimate query efficiency, reconstruct SQL statements, and adjust indexes based on the analysis results. This allows you to improve query efficiency. For more information, see SQL optimization.

Troubleshoot slow SQL statements caused by unsuitable parameter settings

- Cause and symptom

  If the settings of the **innodb_buffer_pool_instances** and **join_buffer_size** parameters are unsuitable, ApsaraDB RDS executes SQL statements at low speeds.

  You can log on to the ApsaraDB RDS console and choose **Parameters** in the left-side navigation pane. Then, on the **Edit History** tab, you can view the reconfiguration histories of these parameters.

- Solution

  Reconfigure these parameters based on your business scenario.

Troubleshoot slow SQL statements caused by the expiration of cached entries

- Cause and symptom

  The system cache can bear a large number of queries. However, Alibaba Cloud does not guarantee a cache hit ratio of 100%. If the cached entries expire, a large number of queries are routed to your RDS instance. This decreases the query performance of your RDS instance.

You can log on to the ApsaraDB RDS console and choose **Monitoring and Alerts** in the left-side navigation pane. Then, on the **Standard Monitoring** tab, you can click **Resource Monitoring** to view the cache hit ratio, QPS, and TPS of your RDS instance.

- Solution

  Use the thread pool, fast query cache, and automatic SQL throttling features to increase the query performance of your RDS instance. For more information, see Thread Pool, Fast query cache, and Automatic SQL throttling.

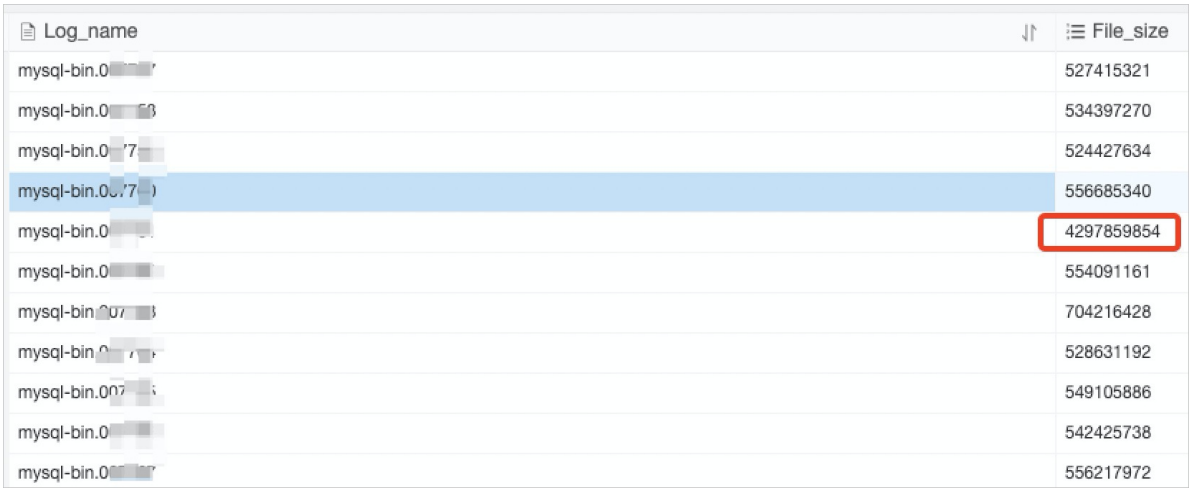
Troubleshoot slow SQL statements caused by batch operations

- Cause and symptom

  If a large number of operations are performed to import, delete, and query data, ApsaraDB RDS executes SQL statements at low speeds.

  You can identify slow SQL statements based on disk usage, SQL logs, or slow query statistics. For example, you can check the size of each binary log file. In normal cases, the size per binary log file is 500 MB. If the size of a binary log file exceeds 500 MB, you can check whether exceptions occur.

  You can log on to the ApsaraDB RDS console and choose **Monitoring and Alerts** in the left-side navigation pane. Then, on the **Standard Monitoring** tab, you can click **Resource Monitoring** to view the disk usage and input/output operations per second (IOPS) of your RDS instance. On this tab, you can also click **Engine Monitoring** to view the TPS of your RDS instance.

- Solution

  Perform batch operations during off-peak hours. Otherwise, split each batch operation into multiple requests and separately submit these requests.

Troubleshoot slow SQL statements caused by unclosed transactions

- Cause and symptom

  If a task suddenly slows down, but the CPU utilization and IOPS usage are normal and the number of active sessions continues to increase, some transactions are not closed.

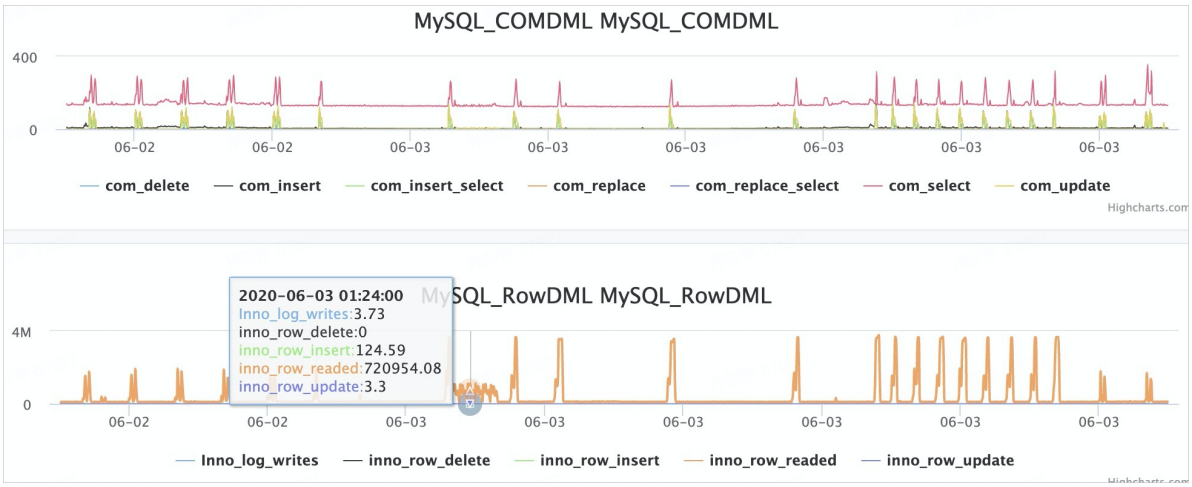
- Solution

  Check for locks that cause conflicts between transactions. Then, terminate the SQL statements that are included in the transactions.

Troubleshoot slow SQL statements caused by scheduled tasks

- Cause and symptom

  If the loads on your RDS instance change regularly over time, scheduled tasks may be configured.



- Solution

  Adjust the time when scheduled tasks are run. We recommend that you run scheduled tasks during off-peak hours.

Summary

Use the recommended features to troubleshoot the issues that cause slow SQL statements. For more information, see the following topics:

- View the resource, engine, and deployment metrics of an ApsaraDB RDS for MySQL instance
- View the slow log details of an ApsaraDB RDS for MySQL instance
- Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance
- DAS overview

18.2. Troubleshoot memory consumption issues on an ApsaraDB RDS for MySQL instance

This topic describes how to troubleshoot memory consumption issues on an ApsaraDB RDS for MySQL instance.

Context

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

View the memory consumption

The ApsaraDB RDS console provides various methods that can be used to view the memory consumption:

- Monitoring and alerting

  In the left-side navigation pane, click **Monitoring and Alerts**. On the page that appears, click the **Standard Monitoring** tab. On the Standard Monitoring tab, click **Resource Monitoring** to view the memory usage, and click **Engine Monitoring** to view the buffer pool hit ratio.

- Autonomy service

  In the left-side navigation pane, choose **Autonomy Service > Dashboard**. On the page that appears, click the **Performance Trends** tab. Then, you can view the **MySQL CPU Utilization/Memory Usage** metric and the **InnoDB Buffer Pool Hit Ratio** metric.

You can also use the PERFORMANCE_SCHEMA storage engine to configure memory instruments. This allows you to aggregate memory usage statistics into memory summary tables. For more information, see How MySQL Uses Memory.

- If you want to enable memory monitoring when your RDS instance is starting, add the `performance_schema = on` setting to the my.cnf file. Then, restart your RDS instance. The setting immediately takes effect.
- If you want to enable memory monitoring when your RDS instance is running, run the following command:

```
update performance_schema.setup_instruments set enabled = 'yes' where name like 'memory%'
;
```

The following tables provide the memory consumption from various dimensions:

- The memory_summary_by_account_by_event_name table provides the events and event names that match a specified account. The account is specified by the combination of a user and a host.
- The memory_summary_by_host_by_event_name table provides the events and event names that match a specified host.
- The memory_summary_by_thread_by_event_name provides the events and event names that match a specified thread.
- The memory_summary_by_user_by_event_name table provides the events and event names that match a specified user.
- The memory_summary_global_by_event_name table provides the events that match a specified event name.

Common reasons for excessively high memory usage

In normal cases, the InnoDB buffer pool consumes the most memory. The maximum memory that can be consumed by the buffer pool varies based on the parameter settings of the buffer pool. In addition, most of the memory is dynamically allocated and adjusted when the requests are being processed. The memory consumption includes the memory that is consumed by in-memory temporary tables, prefetch caches, table caches, hash indexes, and row lock objects. For more information about the memory usage and parameter limits, see How MySQL Uses Memory.

Multi-statement queries

MySQL allows you to combine multiple SQL statements into a single query. These SQL statements are separated by semicolons (;) in the query and are sent to MySQL at a time. MySQL processes these SQL statements one by one. However, some memory is released only after all these SQL statements are executed.

If a large number of SQL statements are sent at a time, the accumulative memory that is consumed by various objects to execute these SQL statements significantly increases. The increase can reach up to a few hundred MB. This may exhaust the available memory for the MySQL process.

In normal cases, if a large number of SQL statements are sent at a time, you can detect a sudden increase in network traffic by using the monitoring and alerting feature or the SQL Explorer feature. We recommend that you do not run multi-statement queries.

Buffer pool issues

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

The default size of the InnoDB buffer pool is set to 75% of the memory capacity that is provided by your RDS instance.

The following common issues that are related to the buffer pool may occur:

- Data pages are not sufficiently pre-warmed. This increases query latencies. If you restart your RDS instance, read cold data, or suffer a low buffer pool hit ratio, this issue may occur. Before you upgrade your RDS instance or launch a sales promotion, we recommend that you sufficiently pre-warm data pages.
- Excessive dirty pages are accumulated. For example, a dirty page has not been updated for a long period of time. In this case, if the difference between the earliest and current log sequence numbers (LSNs) of the dirty page exceeds 76%, a user thread is triggered to synchronously update the dirty page. This significantly decreases the performance of your RDS instance. To fix this issue, you can balance the write loads, prevent excessively high throughput for write operations, reconfigure the parameters that specify how to update dirty pages, and upgrade your RDS instance.
- Your RDS instance provides a large memory capacity. However, the innodb_buffer_pool_instances parameter of your RDS instance is set to a small value. In this case, if the QPS is high, the competition for locks in the buffer pool is fierce. We recommend that you set the innodb_buffer_pool_instances parameter to 8, 16, or a larger value.

Temporary table issues

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

Other issues

If an excessively large number of tables are created on your RDS instance or the QPS is high, the table cache may consume a specific amount of memory. We recommend that you do not create a large number of tables. Otherwise, we recommend that you do not set the table_open_cache parameter to a large value.

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

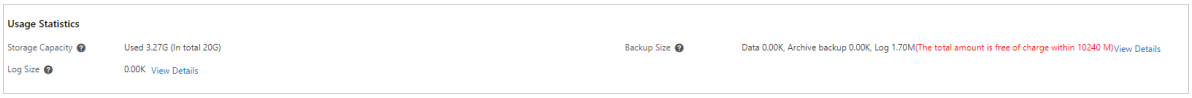
The memory usage of your RDS instance may increase due to a number of other issues. If the memory usage abnormally increases or the memory is exhausted, see How MySQL Uses Memory or submit a .

18.3. Troubleshoot storage issues on an ApsaraDB RDS for MySQL instance

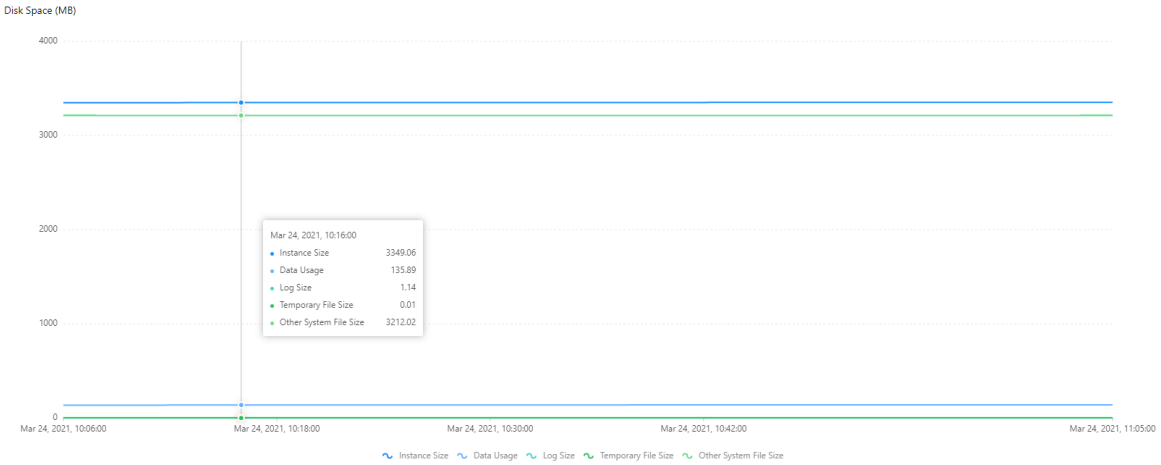
This topic describes how to troubleshoot storage issues on an ApsaraDB RDS for MySQL instance. Storage usage is an important metric that is used to measure the performance of your RDS instance. If the available storage space is insufficient, your RDS instance may encounter serious issues. For example, data writes or backups fail, and the time that is required for a storage expansion task is abnormally long.

View the storage usage

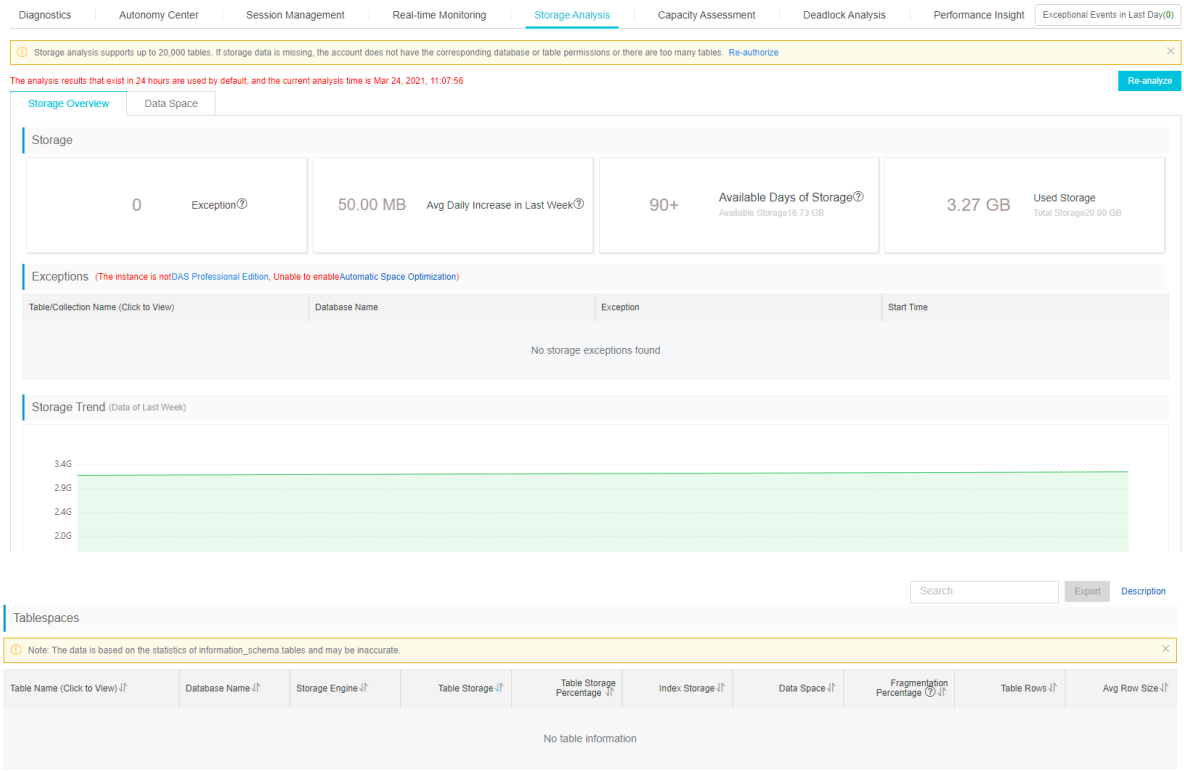
- Log on to the ApsaraDB RDS console and go to the **Basic Information** page. In the Usage Statistics section of the page, view the overall storage usage of your RDS instance. The Usage Statistics section does not provide the current or historical storage usage for various data types.



- Log on to the ApsaraDB RDS console and go to the **Monitoring and Alerts** page. On the **Standard Monitoring** tab of the page, click **Resource Monitoring**. Then, view the current and historical storage usage for various data types.



- Log on to the ApsaraDB RDS console. In the left-side navigation pane, choose **Autonomy Service > Diagnostics**. On the page that appears, click the **Storage Analysis** tab. Then, view further details about the storage usage of your RDS instance. These details include the storage usage comparison between data and logs, the storage usage trend, the storage usage for top databases, and the storage usage for top tablespaces.



> **Note**
>
> - In the **Tablespaces** section, you can view the data storage usage, index storage usage, and available storage space of a specific tablespace. The available storage space is the amount of unused storage space that is allocated to the tablespace.
> - The storage usage is obtained from the statistics of INFORMATION_SCHEMA tables and may be inaccurate.

- Log on to a specified database. Then, run the `show table status like '<The name of a table>';` command to view the storage usage for the specified table.

Troubleshoot out-of-space conditions caused by excessive indexes

- Symptom

  In most cases, a table contains primary key indexes and secondary indexes. More secondary indexes indicate higher storage usage for the table.

- Solution

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

Troubleshoot out-of-space conditions caused by large fields

- Symptom

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

- Solution

  Compress data before you insert the data into the table.

Troubleshoot out-of-space conditions caused by excessive idle tablespaces

- Symptom

  The fragmentation ratio of an InnoDB table is high. This results in an excessive number of idle tablespaces. InnoDB manages tablespaces by page. If some records of a full page are deleted and no new records are inserted into the positions from which these records are deleted, a large number of tablespaces are idle.

- Solution

  Run the `show table status like '<The name of the table>';` command to query the idle tablespaces that store the data of the table. If an excessively large number of tablespaces are idle, run the `optimize table <The name of the table>;` command to manage the tablespaces.

Troubleshoot out-of-space conditions caused by excessively large temporary tables

- Symptom
  - When you perform a semi-join, distinct, or sort operation on a table, a temporary table is created. The sort operation does not use an index. If the temporary table contains an excessive amount of data, the storage usage for the temporary table may be excessively high.
  - When you perform data definition language (DDL) statements to rebuild tablespaces, the temporary table that is generated from an index-based sort operation is large. This applies if the tablespaces are used to store the data of a large table. If your RDS instance runs MySQL 5.6 or MySQL 5.7, you cannot immediately add fields. Some DDL statements can be executed only on new tables. If you send requests to execute these DDL statements on a table, ApsaraDB creates a new table and executes these DDL statements on the new table. The new table is a replica of the original table. Therefore, you can find two file replicas when these DDL statements are being executed. After these DDL statements are executed, the original table is deleted.
- Solution
  - View the plans based on which the DDL statements are executed. This allows you to check whether the **Using Temporary** field is specified.
  - Before you execute DDL statements on large tables, check whether your RDS instance provides sufficient storage space. If the available storage space is insufficient, scale the storage capacity of your RDS instance. For more information, see Change the specifications of an ApsaraDB RDS for MySQL instance.

Optimize the storage usage

- Enable the automatic space debris recycling feature. This feature allows your primary RDS instance to automatically execute OPTIMIZE TABLE statements. These statements are used to recycle fragments from tablespaces. For more information, see Automatic tablespace fragment recycling.
- Use standard or enhanced SSDs. These types of SSDs provide a larger storage capacity than local SSDs. For more information, see Storage types.
- Use the X-Engine storage engine. X-Engine supports a high compression ratio. For more information, see X-Engine overview.
- Use PolarDB. PolarDB is based on a distributed storage system. It provides a large storage capacity that can be automatically scaled. The combination of PolarDB Archive Database and X-Engine significantly reduces the storage usage for various data types. For more information, see PolarDB

overview.

- If your RDS instance is equipped with standard or enhanced SSDs, enable the automatic storage expansion feature. This feature prevents your RDS instance from being locked due to insufficient storage space. For more information, see Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance.
- Use AnalyticDB for MySQL. For more information, see What is AnalyticDB for MySQL?

18.4. Troubleshoot high I/O on an ApsaraDB RDS for MySQL instance

This topic describes how to troubleshoot the issues that cause high I/O on an ApsaraDB RDS for MySQL instance. The I/O performance of your RDS instance varies based on three factors. These factors are the storage media, the database engine architecture, and the SQL statements that are executed to scan or modify a specific amount of data.

Storage media

ApsaraDB RDS for MySQL supports the following types of storage media:

- Local SSD

  Local SSDs deliver the lowest I/O latency among the supported types of storage media. However, the storage capacity that is provided by local SSDs is limited. If the local SSDs that are configured for your RDS instance cannot accommodate an increasing amount of data, you must migrate the instance to another suitable host. The migration is time-consuming. In addition, the migration triggers a switchover of your workloads, and the switchover causes a transient connection error.

- Cloud SSDs

  Cloud SSDs include standard SSDs and enhanced SSDs. Cloud SSDs use a distributed storage mechanism and deliver higher cost-effectiveness and larger storage capacity than local SSDs. In addition, cloud SSDs allow you to scale the storage capacity without the need to migrate data.

For more information about the different types of storage media, see Storage types.

Troubleshoot high I/O caused by high throughput

- Symptom

  If your application frequently initiates requests to update, delete, and insert data on tables, the I/O of your RDS instance significantly increases due to the data reads and the flushes of dirty pages. This applies if the tables contain a large number of indexes or large fields.

  You can log on to the ApsaraDB RDS console and choose **Autonomy Service > Dashboard** in the left-side navigation pane. Then, on the **Performance Trends** tab, you can view the read and write loads on your RDS instance.

- Solution

  We recommend that you reduce the read and write frequency, upgrade your RDS instance, or optimize the settings of the parameters that are used to flush dirty pages. The following parameters are used to flush dirty pages:

  - innodb_max_dirty_pages_pct: the percentage of dirty pages that are allowed in the buffer pool. Default value: 75.

- innodb_max_dirty_pages_pct_lwm: the low water mark for the percentage of dirty pages that are allowed in the buffer pool. If the percentage of dirty pages in the buffer pool exceeds the low water mark, ApsaraDB RDS flushes dirty pages to the disk. This ensures a proper percentage of dirty pages in the buffer pool. The default value 0 specifies to disable the low water mark.

  > **Note** The value of the innodb_max_dirty_pages_pct_lwm parameter must be less than or equal to the value of the innodb_max_dirty_pages_pct parameter. Otherwise, ApsaraDB RDS sets the innodb_max_dirty_pages_pct_lwm parameter to the value of the innodb_max_dirty_pages_pct parameter.

- innodb_io_capacity: the maximum number of I/O operations that are allowed by InnoDB per second for each background task. The value of this parameter affects the speed at which ApsaraDB RDS flushes dirty pages to the disk. The value of this parameter also affects the speed at which ApsaraDB RDS writes data to the buffer pool. The default value of this parameter is 20000.
- innodb_io_capacity_max: the maximum number of I/O operations that are allowed by InnoDB per second for each background task. This parameter takes effect only when the flushes of dirty pages are outdated. The value of this parameter is greater than the value of the innodb_io_capacity parameter. The default value of the innodb_io_capacity_max parameter is 40000.

Troubleshoot high I/O caused by temporary tables

- Symptom

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

  You can log on to the ApsaraDB RDS console and choose **Autonomy Service > Dashboard** in the left-side navigation pane. Then, on the **Performance Trends** tab, you can view the size of the tmp or other directory for your RDS instance.

- Solution

  We recommend that you optimize the SQL statements that you want to execute. This allows you to prevent slow SQL statements. The autonomy service of ApsaraDB RDS supports SQL optimization. For more information, see SQL optimization.

Troubleshoot high I/O caused by cold data reads

- Symptom

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

  You can log on to the ApsaraDB RDS console and choose **Autonomy Service > Dashboard** in the left-side navigation pane. Then, on the **Performance Trends** tab, you can view the buffer pool hit ratio of your RDS instance.

- Solution

  Redesign the cache policy based on your business scenario. Otherwise, upgrade your RDS instance.

Troubleshoot high I/O caused by DDL statements

- Symptom

  If your application initiates data definition language (DDL) statements, ApsaraDB RDS may rebuild the tablespaces of your RDS instance. During the rebuild process, ApsaraDB RDS scans each row of each table in the tablespaces, creates indexes that are used to sort data, and flushes the dirty pages generated from new tables. All these operations significantly increase the I/O of your RDS instance. If your application initiates requests to delete large tables, the I/O of your RDS instance may also increase.

  You can log on to the ApsaraDB RDS console and choose **Monitoring and Alerts** in the left-side navigation pane. Then, on the **Standard Monitoring** tab, you can click **Resource Monitoring** to view the disk usage and input/output operations per second (IOPS) of your RDS instance.

- Solution

  Use the Purge Large File Asynchronously feature to delete large files. This feature is provided by AliSQL. AliSQL is a MySQL branch that is developed by Alibaba Cloud. For more information, see Purge Large File Asynchronously.

Troubleshoot high I/O caused by binary log writes from large transactions

- Symptom

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

- Solution

  We recommend that you split each large transaction that you want to run. This allows you to reduce the flushes of dirty pages to the disk.

Appendix: Introduction to the InnoDB I/O system

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

- Operations to read data pages

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

- Operations to write data pages

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

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

18.5. Troubleshoot excessive active threads on an ApsaraDB RDS for MySQL instance

This topic describes how to troubleshoot the issues that cause excessive active threads on an ApsaraDB RDS for MySQL instance.

Context

The number of active threads or the number of active connections is an important metric that is used to measure the loads on your RDS instance. In most cases, the number is less than 10 if your RDS instance is healthy, and may increase to 20 to 30 if your RDS instance provides high specifications and high queries per second (QPS). If the number exceeds 100, the speed at which your RDS instance responds to queries is low due to an SQL query pile-up. In severe cases, your RDS instance does not respond and stops processing SQL queries.

View the number of active threads

The ApsaraDB RDS console provides various methods that can be used to view the number of active threads:

- Monitoring and alerting

  In the left-side navigation pane, click **Monitoring and Alerts**. On the page that appears, click the **Standard Monitoring** tab. On the Standard Monitoring tab, click **Engine Monitoring**. Then, you can view the number of active threads.

- Autonomy service

  In the left-side navigation pane, choose **Autonomy Service > Dashboard**. On the page that appears, click the **Performance Trends** tab. Then, you can view the number of active threads. If the number is excessively high, some sessions are blocked.

Troubleshoot piled-up slow SQL queries

- Symptom

  If the number of active threads is excessively high, you can execute the `SHOW PROCESSLIST` statement to check for slow SQL queries. If a large number of SQL queries require ApsaraDB RDS to scan an excessively large number of rows, the number of active threads may increase.

  In the left-side navigation pane, choose **Autonomy Service > Diagnostics**. On the page that appears, click the **Session Management** tab. Then, you can view the SQL queries that are in progress.

- Solution

  Enable the SQL throttling feature or terminate sessions. This mitigates the impact of slow SQL queries. For more information, see SQL throttling.

Troubleshoot table cache issues

- Symptom

If your RDS instance implements excessively high QPS or processes a large number of tables, a large number of SQL queries switch to the `Opening table` state due to an insufficient table cache size.

- Solution

  Increase the values of the table_open_cache and table_open_cache_instances parameters. The reconfiguration of the table_open_cache parameter does not require a restart of your RDS instance. However, the reconfiguration of the table_open_cache_instances parameter requires a restart of your RDS instance.

Troubleshoot metadata locking issues

- Symptom

  In the Prepare and Commit phases, data definition language (DDL) statements need to acquire metadata locks on tables. If the tables are involved in uncommitted transactions or slow SQL queries, these DDL statements are blocked. This in turn blocks more SQL queries. All the blocked SQL queries switch to the `Waiting for table metadata lock` state. As a result, the number of active threads increases.

- Solution

  Abort all the uncommitted transactions, slow SQL queries, and ongoing DDL statements.

Troubleshoot row lock conflicts

- Symptom

  If the values of the **Innodb_row_lock_waits** and **Innodb_row_lock_time** metrics are abnormally large, row lock conflicts may occur.

  In the left-side navigation pane, choose **Autonomy Service > Dashboard**. On the page that appears, click the **Performance Trends** tab. Then, you can view the metrics in the **RowLock** section.

- Solution

  Execute the `SHOW ENGINE INNODB STATUS` statement to check whether a large number of sessions are in the `Lock wait` state. If a large number of sessions are in the Lock wait state, severe row lock conflicts may occur. In this case, mitigate row lock conflicts by using all the suggested methods. For example, you can optimize hot data updates, reduce transaction sizes, and reduce the time that is required to commit transactions.

18.6. DAS overview

Database Autonomy Service (DAS) is a cloud service that uses machine learning and expert experience to automate perception, healing, optimization, operations and maintenance (O&M), and security for databases. It simplifies database management and eliminates service failures that may be caused by manual operations. This allows you to ensure the stability, security, and efficiency of your database service.

Features

In ApsaraDB RDS for MySQL, DAS provides the following features:

- Diagnostics

  You can diagnose your RDS instance and view the diagnostic results.

- Autonomy center

You can specify a time range and view events such as exception, optimization, and auto scaling over the specified time range. If DAS detects exceptions on crucial metrics, it diagnoses the related sessions, SQL statements, and storage capacity to identify possible causes. DAS also provides optimization or mitigation suggestions. After you confirm the suggestions, DAS implements the suggestions to resolve the exceptions.

- Session management

  You can view sessions, collect session statistics, analyze SQL statements, and optimize the execution of SQL statements.

- Real-time monitoring

  You can view the real-time monitoring information of your RDS instance. The monitoring information includes the queries per second (QPS), transactions per second (TPS), number of connections, and network traffic.

- Storage analysis

  You can view the storage usage, trend, exceptions, tablespaces, and data spaces of your RDS instance.

- Capacity assessment

  You can view the capacity suggestions, performance capacity, storage usage, and remaining time of your RDS instance. In addition, this feature can use machine learning and capacity algorithms to predict storage usage.

- Deadlock analysis
- Performance insight

  You can evaluate the loads of your RDS instance and locate the root causes of performance issues. This allows you to improve the stability of your RDS instance.

- Dashboard

  You can view and compare performance trends, customize monitoring dashboards, view exception diagnostic results, and view instance topologies.

- Slow query logs

  You can view the trends and statistics of slow queries.

- Full SQL statistics

  You can view and compare the number of execution times, execution duration, and execution duration distribution of each SQL statement over a specific time range. This allows you to identify problem SQL statements.

- Report

  This feature supports automatic and manual diagnostics of your RDS instance. In addition, you can view the diagnostic results, such as the instance health, alerts, and slow query logs.

18.7. Diagnostics

18.7.1. Diagnostics

In ApsaraDB RDS for MySQL, DAS provides the diagnostics feature. This feature diagnoses your ApsaraDB RDS for MySQL instance and visualizes the results.

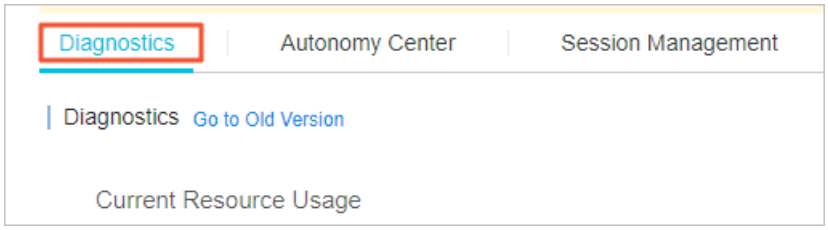
Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Open the Diagnostics page

1.
2. In the left-side navigation pane, choose **Database Autonomy Service (CloudDBA) > Diagnostics**.
3. Click the **Diagnostics** tab.



> **Note** For more information, see Diagnostics.

18.7.2. Autonomy center

In ApsaraDB RDS for MySQL, DAS provides the autonomy center feature. If DAS detects an exception on core metrics, it automatically diagnoses sessions, SQL statements, and database capacity to identify possible causes. DAS also provides optimization and mitigation suggestions. However, DAS does not implement the suggestions until you grant it permissions.

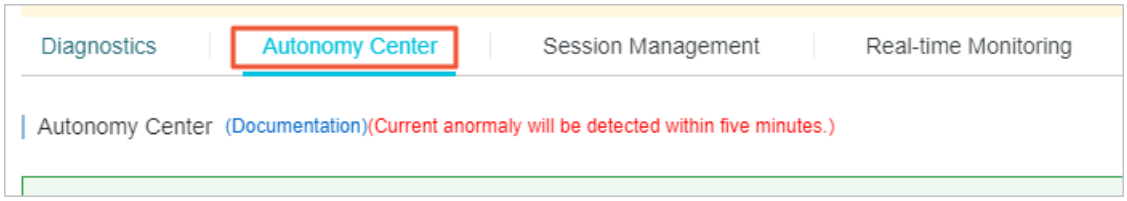
Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Open the Autonomy Center page

1.
2. In the left-side navigation pane, choose **Database Autonomy Service (CloudDBA) > Diagnostics**.
3. Click the **Autonomy Center** tab.



Note For more information, see Autonomy center.

18.7.3. Session management

Database Autonomy Service (DAS) provides the session management feature. This feature allows you to view and manage the sessions of an ApsaraDB RDS for MySQL instance.

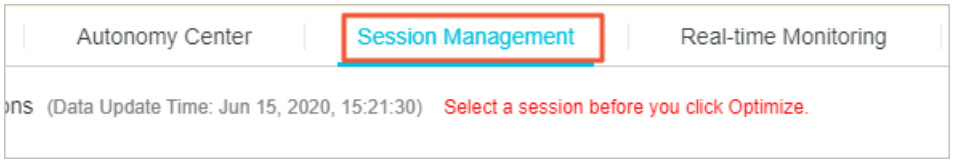
Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Navigate to the Session Management tab

1.
2. In the left-side navigation pane, choose **Database Autonomy Service (CloudDBA) > Diagnostics**.
3. Click the **Session Management** tab.



Note For more information, see Session management.

18.7.4. Real-time monitoring

This topic describes the real-time monitoring feature. Database Autonomy Service (DAS) provides the real-time monitoring feature for ApsaraDB RDS for MySQL instances. This feature allows you to view the real-time performance of your RDS instance.

Prerequisites

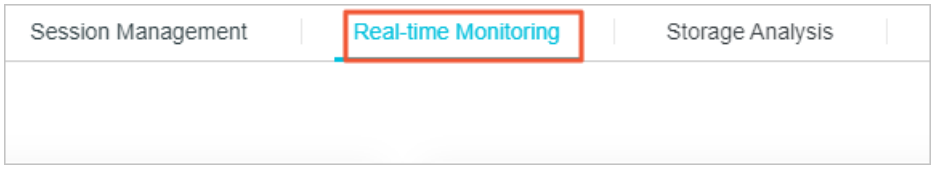
Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition

- MySQL 5.5 on RDS High-availability Edition

Open the Real-time Monitoring tab

1.
2. In the left-side navigation pane, choose **Database Autonomy Service (CloudDBA) > Diagnostics**.
3. Click the **Real-time Monitoring** tab.



> **Note** For more information, see Real-time monitoring.

18.7.5. Storage analysis

Database Autonomy Service (DAS) provides the storage analysis feature. This feature helps you identify and resolve storage exceptions at the earliest opportunity to ensure the stability of an ApsaraDB RDS for MySQL instance.

Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Context

The storage analysis feature allows you to view the disk usage of your RDS instance, the number of remaining days for which disk space is available, and the storage usage, fragmentation, and exception diagnostic results of a table.

Navigate to the Storage Analysis tab

1.
2. In the left-side navigation pane, choose **Database Autonomy Service (CloudDBA) > Diagnostics**.
3. Click the **Storage Analysis** tab.

> **Note** For more information, see Storage analysis.

18.7.6. Capacity assessment

In ApsaraDB RDS for MySQL, DAS provides the capacity assessment feature. This feature allows you to view capacity suggestions, performance capacity, storage usage, and the service time of available storage. It also uses machine learning and capacity algorithms to forecast storage usage.

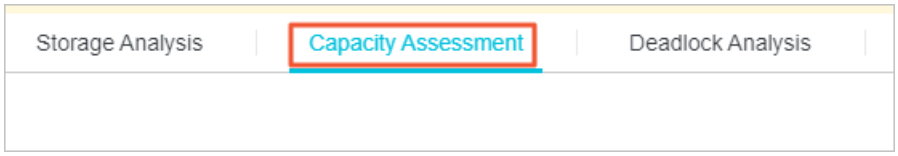
Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Open the Capacity Assessment page

1.
2. In the left-side navigation pane, choose **Database Autonomy Service (CloudDBA) > Diagnostics**.
3. Click the **Capacity Assessment** tab.



> **Note** For more information, see Capacity assessment.

18.7.7. Deadlock analysis

In ApsaraDB RDS for MySQL, DAS provides the deadlock analysis feature. This feature allows you to view and analyze the last deadlock in a database.

Prerequisites

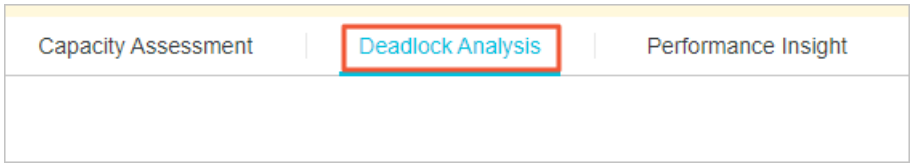
Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Open the Deadlock Analysis page

1.
2. In the left-side navigation pane, choose **Database Autonomy Service (CloudDBA) > Diagnostics**.

3. Click the **Deadlock Analysis** tab.



> **Note** For more information, see Deadlock analysis.

18.7.8. Performance insight

This topic describes the performance insight feature of Database Autonomy Service (DAS) in ApsaraDB RDS. This feature offers instance-level load monitoring, association analysis, and performance optimization. It helps you evaluate the loads on your RDS instance and troubleshoot performance issues to make your RDS instance more stable.

Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Open the Performance Insight page

1.
2. In the left-side navigation pane, choose **Autonomy Service > Diagnostics**.
3. On the page that appears, click the **Performance Insight** tab.

> **Note** For more information, see Performance insight.

18.8. Dashboard

DAS provides the dashboard function. This function allows you to view performance trends in multiple ways. In addition to the performance trend interval, DAS also supports performance trend comparison and custom performance trend viewing.

Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Go to the dashboard page

1.
2. In the left-side navigation pane, choose **Autonomy Service > Dashboard**.

> **Note** For more information, see Performance trends.

18.9. Slow query logs

In ApsaraDB RDS for MySQL, Database Autonomy Service (DAS) provides the slow log analysis feature. This feature allows you to view the trends and execution details of slow queries that are run on an ApsaraDB RDS for MySQL instance and obtain the optimization suggestions for the RDS instance.

Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Open the Slow Query Logs page

1.
2. In the left-side navigation pane, choose **Autonomy Service > Slow Query Logs**.

> **Note** For more information, see Slow query log analysis.

18.10. Use the SQL Explorer and Audit feature on an ApsaraDB RDS for MySQL instance

This topic describes how to use the SQL Explorer and Audit feature on an ApsaraDB RDS for MySQL instance. This feature is developed based on the full request feature and the SQL Audit feature. In addition, this feature is integrated with the following four features: Search, SQL Explorer, Security Audit, and Traffic Playback and Stress Test. This feature helps you obtain information about the SQL statements that are executed. You can use the information to troubleshoot various performance issues and identify the sources of high risks.

Prerequisites

- Database Autonomy Service (DAS) Professional Edition is purchased within your Alibaba Cloud account. For more information about how to purchase DAS Professional Edition, see Purchase DAS Professional Edition.
-
-

Billing

For more information, see Usage notes on DAS Professional Edition.

> **Note** After the SQL Explorer and Audit feature is enabled, the fees that you must pay for the original SQL Explorer feature are billed to DAS Professional Edition. The fees are no longer billed to your RDS instance. For more information, see Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance.

Introduction

- The Search feature is used to query and export the SQL statements that are executed and the information about the SQL statements. The information includes the database, status, and execution duration of each SQL statement. For more information, see the "Use the Search feature" section of this topic.
- The SQL Explorer feature is used to diagnose the health statuses of SQL statements, troubleshoot performance issues, and analyze business traffic. For more information, see the "Use the SQL Explorer feature" section of this topic.
- The Security Audit feature is used to identify risks, such as high-risk SQL statements, SQL injection attacks, and new access sources. For more information, see the "Use the SQL audit feature" section of this topic.
- The Traffic Playback and Stress Test feature supports traffic playback and stress testing. You can use this feature to check whether you need to upgrade your RDS instance to handle traffic spikes during peak hours. For more information, see the "Use the Traffic Playback and Stress Test feature" section of this topic.

Enable the SQL Explorer and Audit feature

1.
2. In the left-side navigation pane, choose **Autonomy Service > SQL Explorer and Audit**.
3. On the page that appears, click **Enable**. In the dialog box that appears, click **Enable Professional Edition**.

   > **Note** If the SQL Explorer feature is enabled for the RDS instance, you can directly click **Enable Professional Edition** in the dialog box to purchase DAS Professional Edition. For more information about the SQL Explorer feature, see Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance.

Use the Search feature

1.
2. In the left-side navigation pane, choose **Autonomy Service > SQL Explorer and Audit**.
3. On the page that appears, click the **Search** tab and specify query criteria.

   > **Note** If you want to specify multiple query criteria to narrow the search scope, you can click **Enable Advanced Query**.

4. Click **Query**. Then, you can view the search results in the **Logs** section.

> **Note** You can click **Export** in the upper-right corner of the Logs section, specify **Exported Fields**, specify **Export Time Range**, and then click **OK** to export the log data that is displayed.

Use the SQL Explorer feature

1.
2. In the left-side navigation pane, choose **Autonomy Service > SQL Explorer and Audit**.
3. On the page that appears, click the **SQL Explorer** tab. Then, you can view the health statuses of the SQL statements that are executed and can troubleshoot performance issues based on the obtained information. For more information about the **SQL Explorer** feature, see SQL Explorer.

Use the SQL Audit feature

1.
2. In the left-side navigation pane, choose **Autonomy Service > SQL Explorer and Audit**.
3. On the page that appears, click the **Security Audit** tab. Then, you can identify high-risk SQL statements, SQL injection attacks, and new access sources. For more information about the **Security Audit** feature, see Security audit.

Use the Traffic Playback and Stress Test feature

1.
2. In the left-side navigation pane, choose **Autonomy Service > SQL Explorer and Audit**.
3. On the page that appears, click the **Traffic Playback and Stress Test** tab. Then, you can view the information about traffic and stress testing.

18.11. Report

DAS provides the RDS for MySQL report function, allowing you to create and view diagnostic reports.

Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Go to the reports page

1.
2. In the left-side navigation pane, choose **Autonomy Service > Report**.

> **Note** For more information, see Diagnostic Report.

18.12. Use the inspection and scoring feature

Database Autonomy Service (DAS) provides the inspection and scoring feature for ApsaraDB RDS for MySQL. You can use the feature to inspect and score all RDS instances on a daily basis. DAS allows you to specify RDS instances and inspection periods and manually initiate inspection and scoring. This helps you understand the status of ApsaraDB RDS for MySQL instances.

Prerequisites

DAS provides the inspection and scoring feature only for the following types of RDS instances:

- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

> **Note** DAS provides the inspection and scoring feature for ApsaraDB RDS for MySQL from May 20, 2022.

Procedure

1. Log on to the ApsaraDB RDS console.
2. In the left-side navigation pane, click **Performance Center**.
3. On the **Performance Center** page, click the **Inspection and Scoring** tab.
4. On the **Inspection and Scoring** tab, perform the following operations based on your business requirements:
   - Enable automatic inspection and scoring

     Perform the following operations the first time that you enable the feature: Click **Enable**. In the **Configure Inspection and Scoring** dialog box, configure the Select Engine parameter and click **OK**.

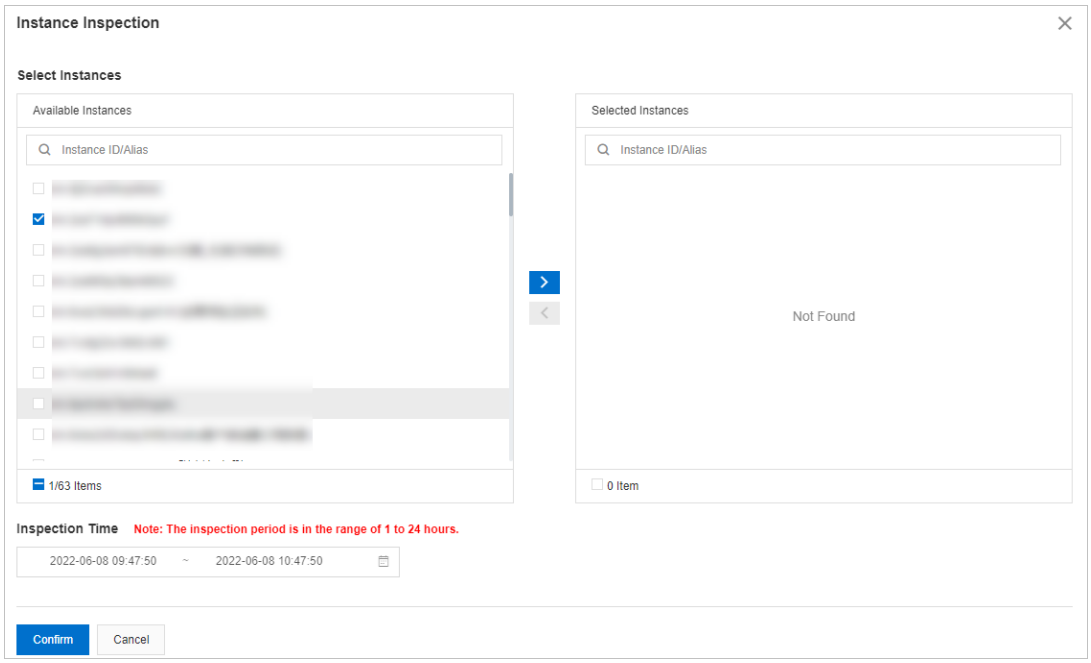
     > **Note** After you enable this feature, the system scores each connected RDS instance once every day.

   - Manually initiate inspection and scoring
     a. On the Inspection and Scoring tab, click **Start Inspection**.
     b. In the **Select Instances** section of the **Instance Inspection** dialog box, select one or more RDS instances in the Available Instances section and click the > icon to add the selected instances to the Selected Instances section.

c. In the **Inspection Time** section, specify a time range.

> **Note** The start time of an inspection cannot be later than the current time. The minimum interval between the start time and the end time is 1 hour, and the maximum interval is 24 hours.

d. Click **Confirm**.



- View the inspection and scoring results

  After you enable the inspection and scoring feature, you can search for an RDS instance by name or alias to view the inspection and scoring results within a specific time period.

  - You can click **Delete** in the **Actions** column of the required RDS instance to delete the results.
  - You can click **Deduction Details** in the **Actions** column of the required RDS instance to view the deduction details in the **Deduction Details** dialog box.
  - You can click **Report** in the **Actions** column of the required RDS instance to view the scoring report details in the **Scoring Report** dialog box.
  - You can click the ID of the required RDS instance to view the performance trend of the RDS instance on the **Performance Trends** tab. For more information, see Performance trends.
  - You can click the  icon to download the inspection and scoring results to your computer.

Scoring rules

Deduction item	Description	Sub-item	Condition	Deducted point

Deduction item	Description	Sub-item	Condition	Deducted point
CPU Utilization (cpuUsage)	The average daily CPU utilization. If an RDS instance has multiple CPU cores, the system calculates the CPU utilization of each core and then calculates the average CPU utilization of all cores.	Warning	70% ≤ cpuUsage ≤ 80%	1 + (cpuUsage - 0.7) × 20
		Critical	cpuUsage ≥ 80%	min[3 + (cpuUsage - 0.8) × 30,10]
Memory Usage (memUsage)	The average daily memory usage.	Warning	80% ≤ memUsage ≤ 90%	1 - (memUsage - 0.8) × 20
		Critical	memUsage > 0.9	min[(memUsage - 0.9) × 50,10]
Space Usage	The storage usage is calculated by using the following formula: Storage usage = Average used storage during a day/Total storage × 100%	Available days	availableDays ≤ 30	15 - availableDays/3
		Number of large tables	bigTableCount > 0	min(bigTableCount,15)
Connection Usage (connectionRate)	The connection usage is calculated by using the following formula: Connection usage = Average number of connections during a day/Maximum number of connections allowed at a point in time × 100%	Warning	70% ≤ connectionRate ≤ 80%	1
		Critical	connectionRate > 80%	3

Deduction item	Description	Sub-item	Condition	Deducted point
IOPS Usage (iopsUsage)	The IOPS usage is calculated by using the following formula: IOPS usage = Average IOPS during a day/Maximum IOPS allowed × 100%	Warning	70% < iopsUsage < 90%	3
		Critical	iopsUsage > 90%	5
Active Sessions (threadRunning)	The number of active sessions generated in a day.	Warning	threadRunning > min(2 × cpuCores + 8,64)	3
		Critical	threadRunning > min(4 × cpuCores + 8,96)	9
Slow SQL Statements (slowSqlCount)	The number of SQL statements that cause slow queries in a day. Note You can click the number to view the five slow SQL statements that are most frequently executed.	Detected	0 < slowSqlCount < 100	1 + (slowSqlCount - 10)/30
		Warning	100 ≤ slowSqlCount < 500	4 + (slowSqlCount - 100)/30
		Critical	slowSqlCount ≥ 500	min[18 + (slowSqlCount-50)/30,30)]
SQL security audit (sqlInjectionCount)	The number of high-risk SQL statements and the number of SQL injection attacks.	High-risk SQL statements	riskSqlCount > 0	min(riskSqlCount,5)
		SQL injection attacks	sqlInjectionCount > 0	Min(sqlInjectionCount,5)
Deadlock	Indicates whether a deadlock occurs during a day.	Detected	Detected	3

18.13. Use the monitoring dashboard feature

Database Autonomy Service (DAS) provides the monitoring dashboard feature for ApsaraDB RDS for MySQL. DAS allows you to specify RDS instances and metrics to monitor and compare the metrics of the RDS instances. You can also configure metric linkage. This helps you understand the status of ApsaraDB RDS for MySQL instances.

Prerequisites

DAS provides the monitoring dashboard feature only for the following types of RDS instances:

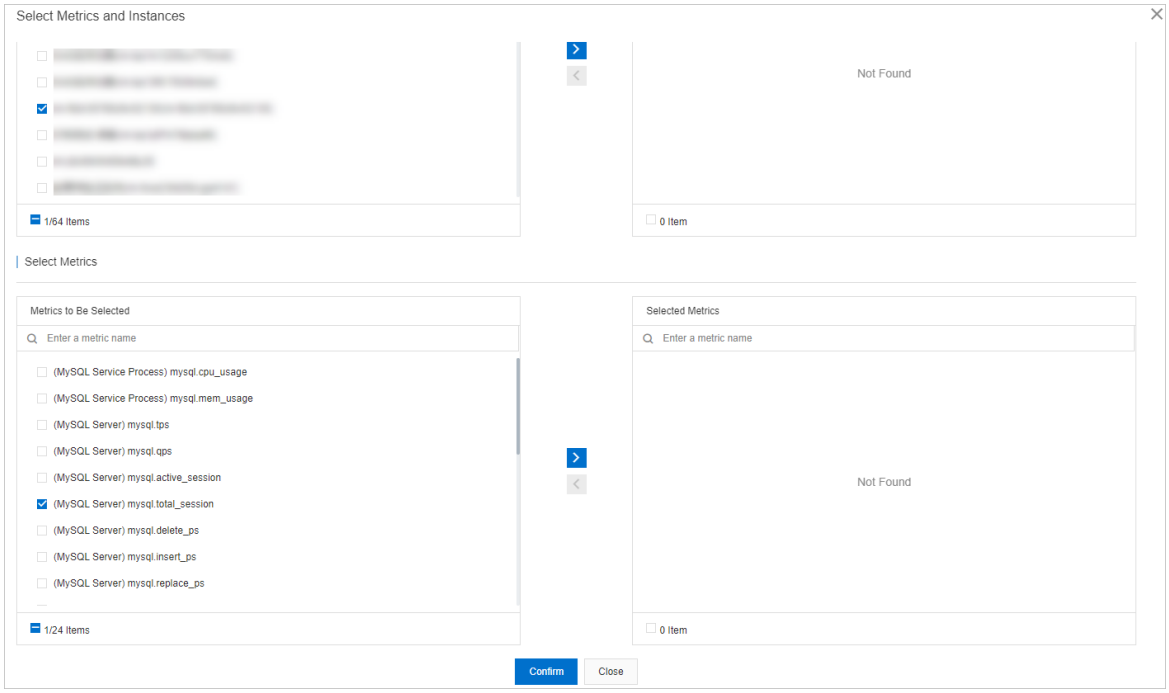
- MySQL 8.0 on RDS High-availability or Enterprise Edition
- MySQL 5.7 on RDS High-availability or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

> **Note** DAS provides the monitoring dashboard feature for ApsaraDB RDS for MySQL from May 20, 2022.

Create a monitoring dashboard

1. Log on to the ApsaraDB RDS console.
2. In the left-side navigation pane, click **Performance Center**.
3. On the **Performance Center** page, click the **Monitoring Dashboard** tab.
4. Click the tab for the database engine. Then, click **Add Monitoring Dashboard**.
5. In the dialog box that appears, configure the **Dashboard Name** parameter and click **OK**.
6. Click **Select Instances and Metrics**. In the dialog box that appears, select the RDS instances and the metrics that you want to monitor. Then, click the > icon to add the selected RDS instances to the Selected Instances section and the selected metrics to the Selected Metrics section.

   > **Note** For more information about the metrics, see Metrics.



7. Click **Confirm**.

> **Note** To modify the RDS instances or metrics in the monitoring dashboard, click **Add Instances and Metrics**.

View the metric trends of an RDS instance in the monitoring dashboard

1. Log on to the ApsaraDB RDS console.
2. In the left-side navigation pane, click **Performance Center**.
3. On the **Performance Center** page, click the **Monitoring Dashboard** tab.
4. Click the tab for the database engine, select the monitoring dashboard that you want to view, and then specify a time range to view the trend charts of the metrics during the specified time range.

> **Note** When you specify a time range, the end time must be later than the start time, and the interval between the start time and the end time cannot exceed seven days.

   - You can configure the **Instance filtering** parameter to filter for multiple RDS instances and then view and compare the metrics of the RDS instances.
   - You can turn on **Auto Refresh (Every 5 Seconds)** for the system to refresh the trend charts of the metrics every 5 seconds.
   - You can turn on **Linkage Chart** to view the values of different metrics at the same point in time.
   - You can configure the **Chart Layout** parameter to specify the number of trend charts of metrics in each row.
   - You can click **Add Instances and Metrics** to modify the RDS instances or metrics in the dashboard.
   - You can click **Details** in the trend chart of a metric to expand the chart. You can also change the time range to view the changes in the trend of the metric at the specified time range.

- You can click **Delete** in the trend chart of a metric to delete the chart from the dashboard.

Metrics

Category	Metric	Description
MySQL server	tps	The transactions per second (TPS).
	qps	The queries per second (QPS).
	active_session	The number of active sessions.
	total_session	The total number of sessions.
	delete_ps	The average number of times that DELETE statements are executed per second.
	insert_ps	The average number of times that INSERT statements are executed per second.
	replace_ps	The average number of times that REPLACE statements are executed per second.
	update_ps	The average number of times that UPDATE statements are executed per second.
	select_ps	The average number of times that SELECT statements are executed per second.
	bytes_received	The average number of bytes that are received from all clients per second.
	bytes_sent	The average number of bytes that are sent to all clients per second.
	iops_usage	The IOPS usage.
InnoDB storage engine	innodb_bp_hit	The read hit ratio of the InnoDB buffer pool.
	innodb_bp_dirty_pct	The ratio of dirty pages in the InnoDB buffer pool.
	innodb_bp_usage_pct	The utilization of the InnoDB buffer pool.
	innodb_data_written	The average number of bytes that are written to the InnoDB table per second.
	innodb_data_read	The average number of bytes that are read from the InnoDB table per second.
	innodb_rows_deleted	The average number of rows that are deleted from the InnoDB table per second.

Category	Metric	Description
	innodb_rows_read	The average number of rows that are read from the InnoDB table per second.
	innodb_rows_inserted	The average number of rows that are inserted into the InnoDB table per second.
	innodb_rows_updated	The average number of rows that are updated in the InnoDB table per second.
MySQL processes	cpu_usage	The CPU utilization of MySQL processes. The maximum value of this metric is 100% for ApsaraDB RDS instances.
	mem_usage	The memory usage of the ApsaraDB RDS for MySQL instance in the operating system.
	iops	The IOPS of the ApsaraDB RDS for MySQL instance.

Note You can click the icon on the right of a metric in a dashboard to view the description of the metric.

19.Monitoring and alerts

19.1. Set the monitoring frequency of an ApsaraDB RDS for MySQL instance

This topic describes how to set the monitoring frequency of an ApsaraDB RDS for MySQL instance.

Context

ApsaraDB RDS for MySQL provides the following three monitoring frequencies:

- Every 5 seconds
- Every 60 seconds
- Every 300 seconds

> **Note** ApsaraDB RDS allows you to query the monitoring data from the most recent 30 days. You may not be able to query the monitoring data over a time range of more than 30 days.

The following table describes the supported monitoring frequencies.

Instance configuration	Every 5 seconds	Every 60 Seconds	Every 300 Seconds
RDS High-availability Edition or RDS Enterprise Edition with a memory capacity **less than** 8 GB	Not supported	Supported free of charge	Supported free of charge (This is the default monitoring frequency.)
RDS High-availability Edition or RDS Enterprise Edition with a memory capacity **greater than or equal to** 8 GB	Supported with fees required	Supported free of charge (This is the default monitoring frequency.)	Supported free of charge
RDS Basic Edition	Not supported	Not supported	Supported free of charge (This is the default monitoring frequency.)

> **Note** If your RDS instance uses standard SSDs or enhanced SSDs (ESSDs), the monitoring frequency is fixed at once every 60 seconds. Changes to the monitoring frequency do not take effect.

For more information about how to set the monitoring frequency of an RDS instance that runs another database engine, see the following topics:

- Set the monitoring frequency of an ApsaraDB RDS for SQL Server instance
- Set the monitoring frequency of an ApsaraDB RDS for PostgreSQL instance

Billing

The monitoring frequencies that are supported with fees required are charged at an hourly rate based on the pay-as-you-go billing method. For example, if your RDS instance uses the Every 5 Seconds monitoring frequency, you are charged an hourly rate of USD 0.012.

Procedure

1.
2. In the left-side navigation pane, click **Monitoring and Alerts**.
3. Click the **Standard monitoring** tab.

   > **Note** For more information about the supported metrics, see 查看监控信息.

4. Click **Set Monitoring Frequency**.
5. In the **Set Monitoring Frequency** dialog box, select a monitoring frequency and click **OK**.

   > **Note** The Every 60 Seconds and Every 300 Seconds monitoring frequencies are supported free of charge. However, the Every 5 Seconds monitoring frequency charges you additional fees. For more information, visit the "Billing" section of this topic.

   If your RDS instance does not support the Every 5 Seconds monitoring frequency, a message appears in the **Set Monitoring Frequency** dialog box.

Related API operations

Operation	Description
Query the monitoring frequency of an ApsaraDB RDS instance	Queries the monitoring frequency of an ApsaraDB RDS instance.

19.2. Configure an alert rule for an ApsaraDB RDS for MySQL instance

This topic describes how to configure an alert rule for an ApsaraDB RDS for MySQL instance.

Context

The monitoring and alerting feature of ApsaraDB RDS is implemented by using Cloud Monitor. Cloud Monitor allows you to configure metrics and alert rules. You can also associate alert groups with metrics. If a metric meets the conditions that are specified in an alert rule, alerts are sent as emails to all the contacts in the alert group that is associated with the metric.

For more information about how to configure alert rules for RDS instances that run other database engines, see the following topics:

- Configure an alert rule for an ApsaraDB RDS for SQL Server instance
- Configure an alert rule for an ApsaraDB RDS for PostgreSQL instance
- Configure an alert rule for an ApsaraDB RDS for MariaDB TX instance

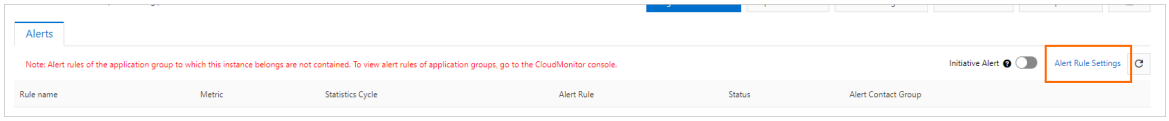
Enable the initiative alert feature

The initiative alert feature allows you to establish an alert system for multiple metrics in RDS. An alert notification is sent if an exception of a key metric occurs. You can then handle the exception at the earliest opportunity. For more information, see Enable the initiative alert feature.

1.
2. In the left-side navigation pane, click **Monitoring and Alerts**.
3. Click the **Alerts** tab.
4. In the right-side section of the page, turn on the **Initiative Alert** switch.

Create an alert rule

1.
2. In the left-side navigation pane, click **Monitoring and Alerts**.
3. Click the **Alerts** tab.
4. Click **Set Alert Rule** to go to the Cloud Monitor console.

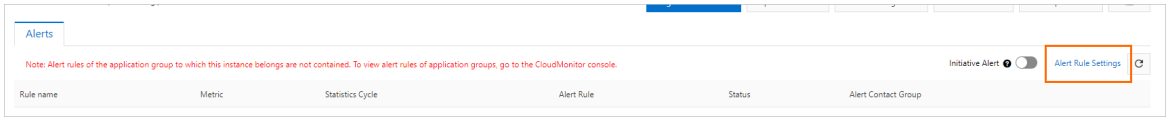


5. Create an alert group. For more information, see Create an alert contact or alert contact group.
6. Create an alert rule. For more information, see Create an alert rule.

> **Note** You can also configure Cloud Monitor to automatically monitor resources based on tags. For more information, see Monitor resources based on tags.

Manage an alert rule

1.
2. In the left-side navigation pane, click **Monitoring and Alerts**.
3. Click the **Alerts** tab.
4. Click **Set Alert Rule** to go to the Cloud Monitor console.



5. On the **Alert Rules** page, find the alert rule that you want to manage, and select one of the following operations in the Actions column:
   - View: View details about the alert rule.
   - Alert Logs: View the alerts that were triggered by the alert rule over a specific time range.
   - Modify: Modify the alert rule. For more information, see Create an alert rule.
   - Disable: Disable the alert rule. After you disable the alert rule, no alerts are triggered even if the metric meets the conditions that are specified in the alert rule.
   - Delete: Delete the alert rule. After you delete the alert rule, the alert rule cannot be restored. You can only re-create the alert rule if necessary.

20.Account

20.1. Create an account on an ApsaraDB RDS for MySQL instance

This topic describes how to create an account that is used to manage the databases of an ApsaraDB RDS for MySQL instance.

Prerequisites

Create an ApsaraDB RDS for MySQL instance

> **Note** You can create Resource Access Management (RAM) users within your Alibaba Cloud account and grant the permissions on specific RDS instances to the RAM users. For more information, see Create a RAM user.

Account types

ApsaraDB RDS for MySQL supports two types of accounts: privileged accounts and standard accounts. You can manage all the accounts and databases of your RDS instance by using the ApsaraDB RDS console. For more information about the permissions that can be granted to each type of account, see Account permissions.

> **Note** After an account is created, you cannot change the type of the account. However, you can delete the account. Then, you can create an account that has the same username as the deleted account. For more information, see Delete a standard account from an ApsaraDB RDS for MySQL instance.

Account type	Description
Privileged account	• You can create and manage privileged accounts in the ApsaraDB RDS console or by using the ApsaraDB RDS API. • Only one privileged account is allowed per RDS instance. A privileged account has the permissions to manage all the databases and standard accounts of the RDS instance on which the privileged account is created. • A privileged account allows you to manage more permissions at fine-grained levels based on your business requirements. For example, you can grant each standard account the permissions to query specific tables from the RDS instance on which the privileged account is created. • A privileged account has the permissions on all the databases of the RDS instance on which the privileged account is created. • A privileged account has the permissions to disconnect all the standard accounts of the RDS instance on which the privileged account is created.

Account type	Description
Standard account	• You can create and manage standard accounts by using the ApsaraDB RDS console, ApsaraDB RDS API, or SQL statements. • More than one standard account is allowed per RDS instance. The maximum number of standard accounts that are allowed varies based on the minor engine version that is used. • By default, a standard account can be used only to log on to one or more databases on which the account has permissions. You must manually grant specific permissions to each standard account. For more information, see Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance. • A standard account does not have the permissions to create, manage, or disconnect other accounts of the RDS instance on which the standard account is created.

Account type	Maximum number of databases	Maximum number of tables	Maximum number of accounts
Privileged account	Unlimited	< 200,000	Varies based on the minor engine version.
Standard account	500	< 200,000	Varies based on the minor engine version.

Note After a privileged account is created, the maximum number of databases that can be created by using standard accounts is unlimited.

Create a privileged account

1.
2. In the left-side navigation pane, click **Accounts**.
3. Click **Create Account**.
4. Configure the following parameters.

Parameter	Description
Database Account	Enter a username for the account. The username must meet the following requirements: ◦ If your RDS instance runs MySQL 5.6, the username must be 2 to 16 characters in length. If your RDS instance runs MySQL 8.0 or MySQL 5.7, the username must be 2 to 32 characters in length. ◦ The username must start with a lowercase letter and end with a lowercase letter or digit. ◦ The username can contain lowercase letters, digits, and underscores (_). ◦ The username cannot be the same as the username of an existing account.

Parameter	Description
Account Type	Select **Privileged Account**.
Password	Enter a password for the account. The password must meet the following requirements: ○ The password must be 8 to 32 characters in length. ○ The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. ○ The password can contain the following special characters: ! @ # $ % ^ & * ( ) _ + - = **Note** If your RDS instance runs MySQL 5.7, you can configure a custom password policy for your instance. For more information, see Configure a custom password policy for an ApsaraDB RDS for MySQL instance.
Confirm Password	Enter the password for the account again.
Description	Enter a description that is used to identify the account. The description can be up to 256 characters in length.

5. Click **Create**.

Reset the permissions of a privileged account

If the privileged account of your RDS instance encounters exceptions, for example, the permissions are accidentally revoked, you can perform the following steps to reset the permissions:

1.
2. In the left-side navigation pane, click **Accounts**.
3. Find the account whose Account Type is **Privileged Account**. Then, click **Reset Permissions** in the Actions column.
4. In the dialog box that appears, enter the password of the privileged account and click **OK**.

Create a standard account

1.
2. In the left-side navigation pane, click **Accounts**.
3. Click **Create Account**.
4. Configure the following parameters.

Parameter	Description

Parameter	Description
Database Account	Enter a username for the account. The username must meet the following requirements: ◦ If your RDS instance runs MySQL 5.6, the username must be 2 to 16 characters in length. If your RDS instance runs MySQL 8.0 or MySQL 5.7, the username must be 2 to 32 characters in length. ◦ The username must start with a lowercase letter and end with a lowercase letter or digit. ◦ The username can contain lowercase letters, digits, and underscores (_). ◦ The username cannot be the same as the username of an existing account.
Account Type	Select **Standard Account**.
Authorized Databases	Specify the authorized databases of the account. You can specify one or more authorized databases. You can leave this parameter unspecified. In this case, you can grant the permissions on specific databases to the account after the account is created. i. In the Unauthorized Databases section, select one or more databases. Then, click the **>** icon to move the selected databases to the Authorized Databases section. ii. In the Authorized Databases section, select the **Read/Write (DDL + DML)**, **Read-only**, **DDL Only**, or **DML Only** permissions for each authorized database. If you want to grant the same permissions on more than one authorized database at a time, select the authorized databases and click the Set All to button in the upper-right corner of the Authorized Database section. For example, you can click the button to grant the **Read/Write (DDL + DML)** permissions on the selected authorized databases. **Note** For more information, see Account permissions.
Password	Enter a password for the account. The password must meet the following requirements: ◦ The password must be 8 to 32 characters in length. ◦ The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. ◦ The password can contain the following special characters: ! @ # $ % ^ & * ( ) _ + - = **Note** If your RDS instance runs MySQL 5.7, you can configure a custom password policy for your instance. For more information, see Configure a custom password policy for an ApsaraDB RDS for MySQL instance.
Confirm Password	Enter the password for the account again.

Parameter	Description
Description	Enter a description that is used to identify the account. The description can be up to 256 characters in length.

5. Click **OK**.

FAQ

- Can I configure an account to have only the permissions to access my RDS instance over an internal network?

  Yes, you can use SQL statements to specify the source IP address from which an account can access your RDS instance. For more information, see Authorize an account to access its authorized databases from specified IP addresses in an ApsaraDB RDS for MySQL instance. However, this operation is not supported in the ApsaraDB RDS console.

- Can I configure the permissions of an account at finer-grained levels, such as the table level?

  Yes, you can use SQL statements to manage the permissions of an account at finer-grained levels. For more information, see Authorize accounts to manage tables, views, and fields. However, this operation is not supported in the ApsaraDB RDS console.

Related operations

Operation	Description
CreateAccount	Creates an account that is used to manage the databases of an ApsaraDB RDS instance.

20.2. Configure a custom password policy for an ApsaraDB RDS for MySQL instance

This topic describes how to configure a custom password policy for an ApsaraDB RDS for MySQL instance. You can use custom password policies to ensure the security of your RDS instance.

Prerequisites

- Your RDS instance runs one of the following database engine versions and RDS editions:
  - MySQL 5.7 on RDS Basic Edition
  - MySQL 5.7 on RDS High-availability Edition
- The minor engine version of your RDS instance is updated to the latest version. For more information, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.

Precautions

When you configure or modify a custom password policy in the ApsaraDB RDS console, the custom password policy cannot take precedence over the following default password policy:

- The password must be 8 to 32 characters in length.
- The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters.
- The password can contain any of the following special characters: ! @ # $ % ^ & * ( ) _ + - =

Introduction

If your RDS instance runs MySQL 5.7, you can use the `validate_password` plug-in to configure a custom password policy that is used to check password complexity. A custom password policy contains the following password complexity rules:

- Whether the password can be the same as the username
- The length of the password
- The number of uppercase letters and lowercase letters in the password
- The number of digits in the password
- The number of special characters in the password
- The strength of the password

Step 1: Install the validate_password plug-in

1. Connect to your RDS instance. For more information, see Connect to an ApsaraDB RDS for MySQL instance.

   > **Note** You must use the privileged account of your RDS instance to connect to your RDS instance. For more information, see Create a privileged account.

2. Execute the following statement in the SQL window to install the `validate_password` plug-in:

   ```
   INSTALL PLUGIN validate_password SONAME 'validate_password.so';
   ```

3. Execute the following statement in the SQL window to check whether the validate_password plug-in is installed:

   ```
   SHOW GLOBAL VARIABLES LIKE 'validate_password%';
   ```

   If information similar to the following figure is returned, the validate_password plug-in is installed.

Variable_name	Value
validate_password_dictionary_file	
validate_password_length	8
validate_password_mixed_case_count	1
validate_password_number_count	1
validate_password_policy	MEDIUM
validate_password_special_char_count	1

   > **Note** You can configure custom password policies only when your RDS instance runs MySQL 5.7 on RDS Basic Edition or RDS High-availability Edition. If your RDS instance runs a different database engine version or a different RDS edition, you can install the validate_password plug-in, but you cannot use the plug-in to configure custom password policies.

Step 2: Configure a custom password policy

1.
2. In the left-side navigation pane, click **Parameters**.
3. Configure the parameters whose names are prefixed by loose_validate_password. For more information, see Modify the parameters of an ApsaraDB RDS for MySQL instance.

> **Note** Before you configure the parameters, you must install the validate_password plug-in. If the plug-in is not installed, the new parameter settings do not take effect. For more information, see Step 1: Install the validate_password plug-in.

Parameter	Description
loose_validate_password_check_user_name	Specifies whether the password can be the same as the username. Valid values: ◦ **ON**: The password can be the same as the username. ◦ **OFF**: The password cannot be the same as the username. Default value: **OFF**.
loose_validate_password_policy	The strength of the password. Valid values: ◦ **0**: The strength of the password is low. ApsaraDB RDS checks only the length of the password. ◦ **1**: The strength of the password is medium. In addition to the length of the password, ApsaraDB RDS checks the number of digits, number of uppercase letters and lowercase letters, and number of special characters in the password. ◦ **2**: The strength of the password is high. ApsaraDB RDS checks the length and dictionary file of the password. In addition, ApsaraDB RDS checks the number of digits, number of uppercase letters and lowercase letters, and number of special characters in the password. > **Note** The dictionary file cannot be specified. This indicates that the value **1** and the value **2** specify the same password strength. Default value: **1**.
loose_validate_password_length	The length of the password. Valid values: **0 to 256**. Default value: **8**. > **Note** The default password policy that is applied in the ApsaraDB RDS console requires a password length of at least eight characters. The length of each password must be greater than or equal to eight characters even if you set this parameter to 5 in the ApsaraDB RDS console. However, you can execute the `SET PASSWORD` statement to change the password length to 5 characters.

Parameter	Description
loose_validate_password_number_count	The number of digits in the password. Valid values: **0 to 256**. Default value: **1**.
loose_validate_password_mixed_case_count	The number of uppercase letters and lowercase letters in the password. Valid values: **0 to 256**. Default value: **1**.
loose_validate_password_special_char_count	The number of special characters in the password. Valid values: **0 to 256**. Default value: **1**.

Note For more information, see Password Validation Plugin Options and Variables.

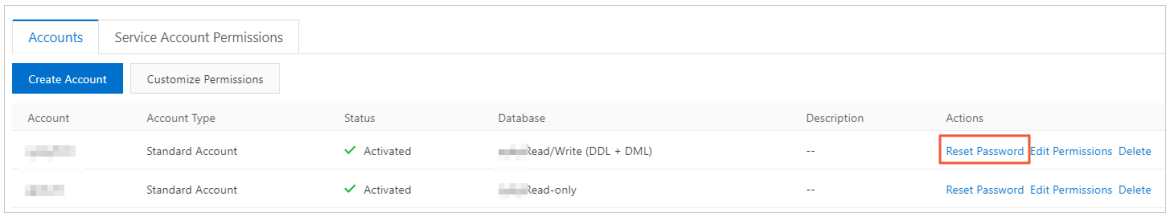
20.3. Reset the password of an account on an ApsaraDB RDS for MySQL instance

This topic describes how to reset the password of an account on an ApsaraDB RDS for MySQL instance. If the password of an account is lost, you can reset the password by using the ApsaraDB RDS console.

Procedure

Note For data security purposes, we recommend that you change the password of each account on a regular basis.

1.
2. In the left-side navigation pane, click **Accounts**.
3. Find the account whose password you want to reset, and click **Reset Password** in the Actions column.



4. In the dialog box that appears, specify a new password, confirm the new password, and then click **Create**.

> **Note** The password must meet the following requirements:
>
> - The password must be 8 to 32 characters in length.
> - The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters.
> - The password can contain any of the following characters:
>
> ! @ # $ % ^ & * ( ) _ + - =

Related operations

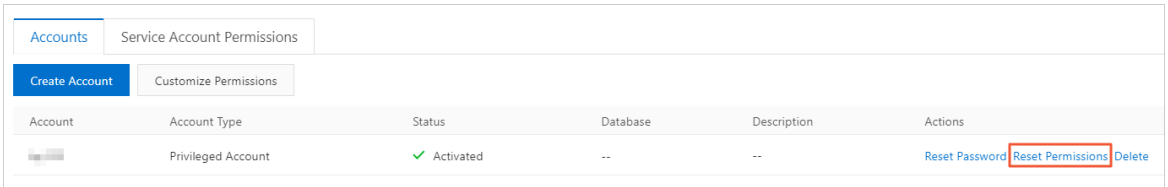
Operation	Description
ResetAccountPassword	Resets the password of an account on an ApsaraDB RDS instance.

20.4. Reset the permissions of the privileged account for an ApsaraDB RDS for MySQL instance

If the permissions of the privileged account for an ApsaraDB RDS for MySQL instance are abnormal, you can enter the password of the privileged account to reset the permissions. For example, you can reset the permissions if the permissions are accidentally revoked.

Procedure

1.
2. In the left-side navigation pane, click **Accounts**.
3. Find the **Privileged Account** and in the Actions column click **Reset Permissions**.



4. Enter the new password of the privileged account, confirm the new password, and then click **OK**.

20.5. Authorize the service account of an ApsaraDB RDS for MySQL instance

When you seek help from Alibaba Cloud technical support to locate problems that occurred on your ApsaraDB RDS for MySQL instance, you may need to grant permissions to a service account. The service account is used by Alibaba Cloud technical support to perform operations on the databases of your RDS instance. After the specified expiration time elapses, ApsaraDB RDS deletes the service account.

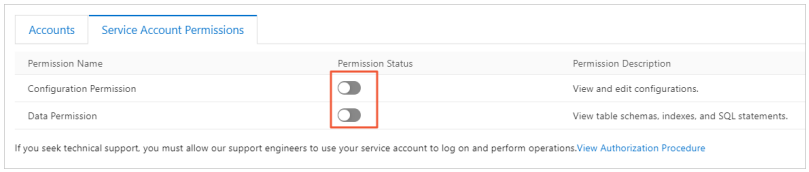
Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

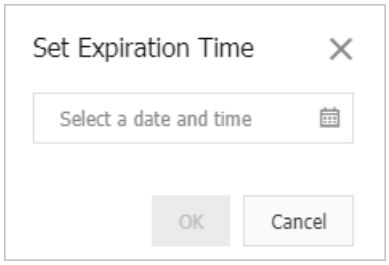
- MySQL 8.0 on RDS High-availability Edition (with local SSDs) or Enterprise Edition
- MySQL 5.7 on RDS High-availability Edition (with local SSDs) or Enterprise Edition
- MySQL 5.6 on RDS High-availability Edition
- MySQL 5.5 on RDS High-availability Edition

Procedure

1.
2. In the left-side navigation pane, click **Accounts**
3. On the **Service Account Permissions** tab, find the permission that you want to grant to the service account, and turn on the switch in the **Permission Status** column.
   - For problems that are related to IP address whitelists or database parameters, you can grant only the **Configuration Permission** to the service account.
   - For database performance problems that are caused by applications, you must grant the **Data Permission** to the service account.



4. In the dialog box that appears, specify the expiration time of the service account and click **OK**.



What to do next

After you grant permissions to the service account, you can revoke the permissions or change the expiration time on the **Service Account Permissions** tab at any time.

20.6. Delete a standard account from an ApsaraDB RDS for MySQL instance

This topic describes how to delete a standard account from an ApsaraDB RDS for MySQL instance by using the ApsaraDB RDS console or an SQL statement.

Use the console to delete a standard account

1.
2. In the left-side navigation pane, click **Accounts**.

3. Find the standard account that you want to delete and in the **Actions** column click **Delete**.
4. In the message that appears, click **OK**.

Execute an SQL statement to delete an standard account

This function is supported only for specific RDS instance configurations.

1. Use Data Management (DMS) to log on to the RDS instance. For more information, see Use DMS to log on to an ApsaraDB RDS for MySQL instance.
2. In the top navigation bar, choose **SQL Operations > SQL Window**.
3. Execute the following statement:

```
DROP USER 'username'@'localhost';
```

4. Click **Execute**.

Related operations

Operation	Description
DeleteAccount	Deletes an account from an ApsaraDB RDS instance.

20.7. Account permission

20.7.1. Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance

This topic describes how to modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance. The permissions of a privileged account can only be reset to the default settings but cannot be modified.

> **Note** You can use the Account Authorization and Management feature of Alibaba Cloud Data Management (DMS) to define permission combinations. You can also use this feature to manage the permissions on specific tables. For information, see Manage user permissions on MySQL databases.

Method 1: Modify the permissions of a standard account in the ApsaraDB RDS console

1.
2. In the left-side navigation pane, click **Accounts**.
3. Find the standard account whose permissions you want to modify. Then, click **Edit Permissions** in the Actions column.
4. In the **Modify Account Permissions** panel, modify the permissions of the standard account.
   - If you want to add or remove an authorized database, select the database and click the > or < icon.

- If you want to modify the permissions on an authorized database, select the database. Then, select the **Read/Write (DDL + DML)**, **Read-only**, **DDL Only**, or **DML Only** permissions in the **Authorized Databases** section.

  > **Note** You can use SQL statements to modify permissions at higher levels of granularity. For more information, see Account permissions.

5. Click **OK**.

Method 2: Modify the permissions of a standard account in the DMS console

You can modify the permissions of a standard account in the DMS console. For more information, see Manage user permissions on MySQL databases

Method 3: Modify the permissions of a standard account by using SQL statements

Prerequisites

A privileged account is created for the RDS instance and is used to modify the permissions.

1. Use a database client or the CLI to connect to the RDS instance. For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.
2. Execute the GRANT statement to grant permissions to the standard account.

> **Note**
> - For more information about the GRANT statement, see official MySQL documentation.
> - For more information about permissions that can be granted, see Account permissions.

FAQ

Why am I unable to create a database on an ApsaraDB RDS for MySQL instance by using a standard account? What do I do if the `ERROR 1044 (42000): Access denied for user 'xxxx'@'%' to database 'xxxx'` error message is displayed when I create a database on an ApsaraDB RDS for MySQL instance by using a standard account?

By default, a standard account for an ApsaraDB RDS for MySQL instance has only the permissions to log on to databases. If you want to use the standard account to create a database, you must first use a privileged account to grant the CREATE permission to the standard account. Example statement:

```
GRANT CREATE ON *.* TO '<Name of the standard account>'@'%';
```

20.7.2. Account permissions

This topic provides an overview of the permissions that ApsaraDB RDS for MySQL provides for both privileged and standard accounts.

Accounts and permissions

Account type	Permission	Operation				
Privileged accounts	N/A	SELECT	INSERT	UPDATE	DELETE	CREATE
		DROP	RELOAD	PROCESS	REFERENCES	INDEX
		ALTER	CREATE TEMPORARY TABLES	LOCK TABLES	EXECUTE	REPLICATION SLAVE
		REPLICATION CLIENT	CREATE VIEW	SHOW VIEW	CREATE ROUTINE	ALTER ROUTINE
		CREATE USER	EVENT	TRIGGER		
Standard accounts	**Read-only**	SELECT	LOCK TABLES	SHOW VIEW	PROCESS	REPLICATION SLAVE
		REPLICATION CLIENT				
	Read/write	SELECT	INSERT	UPDATE	DELETE	CREATE
		DROP	REFERENCES	INDEX	ALTER	CREATE TEMPORARY TABLES
		LOCK TABLES	EXECUTE	CREATE VIEW	SHOW VIEW	CREATE ROUTINE
		ALTER ROUTINE	EVENT	TRIGGER	PROCESS	REPLICATION SLAVE
		REPLICATION CLIENT				
	DDL-only	CREATE	DROP	INDEX	ALTER	CREATE TEMPORARY TABLES
		LOCK TABLES	CREATE VIEW	SHOW VIEW	CREATE ROUTINE	ALTER ROUTINE
		PROCESS	REPLICATION SLAVE	REPLICATION CLIENT		
		SELECT	INSERT	UPDATE	DELETE	CREATE TEMPORARY TABLES

Account type	Permission	Operation				
	DML-only	LOCK TABLES	EXECUTE	SHOW VIEW	EVENT	TRIGGER
		PROCESS	REPLICATION SLAVE	REPLICATION CLIENT		

20.8. Authorize an account to access its authorized databases from specified IP addresses in an ApsaraDB RDS for MySQL instance

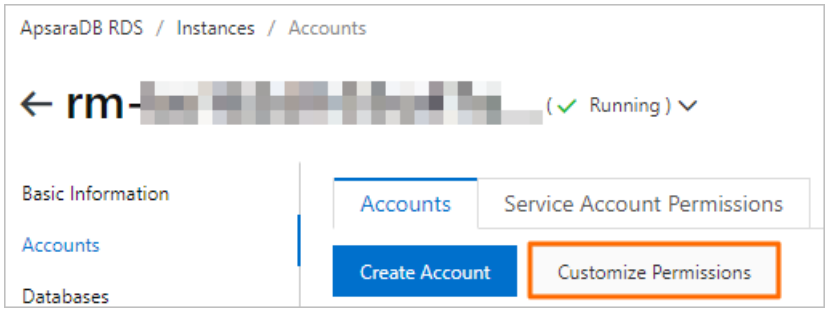
This topic describes how to authorize an account to access its authorized databases from specified IP addresses in an ApsaraDB RDS for MySQL instance. The IP address whitelists of an RDS instance take effect on all accounts that are created on the RDS instance. You cannot use IP address whitelists to restrict the IP addresses from which each account can access its authorized databases. If you use only IP address whitelists to control access to an RDS instance, the RDS instance may be exposed to security risks.

Prerequisites

A privileged account is created. For more information, see Create an account on an ApsaraDB RDS for MySQL instance.

Use DMS to authorize an account to access its authorized databases from specified IP addresses

1.
2. In the left-side navigation pane, click **Accounts**. On the **Accounts** page, click **Customize Permissions** to go to the Data Management (DMS) console.



3.

4. Click **Create User** in the upper-left corner of the page. Alternatively, click **Edit** in the Actions column for the account.
5. On the **Basic settings** tab, configure the Host parameter.

> **Note**
>
> - The Host parameter specifies the IP address from which the account can access its authorized databases. You can specify more than one IP address. Multiple IP addresses must be separated by commas (,). If you do not specify this parameter, the account is not authorized to access its authorized databases from specified IP addresses. The default value of this parameter is `%`.
> - The IP addresses that are specified by the Host parameter must be added to an IP address whitelist of the RDS instance. For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.
> - DMS allows you to grant more permissions to accounts. For more information, see Manage user permissions on MySQL databases.

6.
7.

Use SQL statements to authorize an account to access its authorized databases from specified IP addresses

1. Connect to the RDS instance on which you want to create an account. For more information, see Connect to an ApsaraDB RDS for MySQL instance.
2. Execute SQL statements to create an account on the RDS instance and authorize the account to access its authorized databases from specified IP addresses. You cannot view the authorized databases of the created account in the ApsaraDB RDS console.

   In the following example, you create an account named test001 and authorize the account to access the rds001 database from the 42.120.XX.XX IP address.

```
CREATE USER `test001`@`42.120.XX.XX`IDENTIFIED BY 'passwd';
GRANT PROCESS, REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'test001'@'42.120.XX.XX'
;
GRANT ALL PRIVILEGES ON `rds001`.* TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`help_topic` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`func` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`time_zone` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`slow_log` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`time_zone_transition` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`event` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`proc` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`help_category` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`help_relation` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`help_keyword` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`general_log` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`time_zone_leap_second` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`time_zone_transition_type` TO 'test001'@'42.120.XX.XX';
GRANT SELECT ON `mysql`.`time_zone_name` TO 'test001'@'42.120.XX.XX';
```

> **Note**
>
> - If you change the IP address from *42.120.XX.XX* to *%*, the created account is similar to an account that is created in the ApsaraDB RDS console. You can view the authorized database of the created account in the ApsaraDB RDS console.
> - You can execute the following statement to change the IP address to 42.121.XX.XXX:
>
>   ```
>   RENAME USER `test001`@`42.120.XX.XX` TO `test001`@`42.121.XX.XX`;
>   ```

20.9. Authorize accounts to manage tables, views, and fields

This topic describes how to execute SQL statements for authorizing accounts to manage tables, views, or fields in a database of an ApsaraDB for RDS instance. If you are using your Alibaba Cloud account, you have the permissions to manage all types of data in your ApsaraDB for RDS instance.

Prerequisites

You have created a privileged account in your ApsaraDB for RDS instance. For more information, see Create accounts and databases for an RDS for MySQL instance.

Procedure

1. Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.
2. Execute SQL statements to create an account and authorize it to manage tables, views, and fields in the target database.

   > **Note** The created account does not have permissions to view its authorized databases in the ApsaraDB for RDS console.

   - Create an account and authorize it to manage a table in the target database.

```
CREATE USER `<The username of the account you want to create>`@`%`IDENTIFIED BY '<The
password of the account you want to create>';
GRANT PROCESS, REPLICATION SLAVE, REPLICATION CLIENT ON *. * TO '<The username of the
created account>'@'%';
GRANT ALL PRIVILEGES ON `<The name of the target database>`. `<The name of the table
you want to create with the created account>`  TO '<The username of the created accou
nt>'@'%';
GRANT SELECT ON `mysql`.`help_topic` TO '<The username of the created account>'@'%';
GRANT SELECT ON `mysql`.`func` TO '<The username of the created account>'@'%';
GRANT SELECT ON `mysql`.`time_zone` TO '<The username of the created account>'@'%';
GRANT SELECT ON `mysql`.`slow_log` TO '<The username of the created account>'@'%';
GRANT SELECT ON `mysql`.`time_zone_transition` TO '<The username of the created accou
nt>'@'%';
GRANT SELECT ON `mysql`.`event` TO '<The username of the created account>'@'%';
GRANT SELECT ON `mysql`.`proc` TO '<The username of the created account>'@'%';
GRANT SELECT ON `mysql`.`help_category` TO '<The username of the created account>'@'%
';
GRANT SELECT ON `mysql`.`help_relation` TO '<The username of the created account>'@'%
';
GRANT SELECT ON `mysql`.`help_keyword` TO '<The username of the created account>'@'%'
;
GRANT SELECT ON `mysql`.`general_log` TO '<The username of the created account>'@'%';
GRANT SELECT ON `mysql`.`time_zone_leap_second` TO '<The username of the created acco
unt>'@'%';
GRANT SELECT ON `mysql`.`time_zone_transition_type` TO '<The username of the created
account>'@'%';
GRANT SELECT ON `mysql`.`time_zone_name` TO '<The username of the created account>'@'
%';
```

Example:

To create an account named test01 and authorize it to manage the test100 table in the rds001 database, execute the following SQL statements:

```
CREATE USER `test01`@`%`IDENTIFIED BY 'passwd';
GRANT PROCESS, REPLICATION SLAVE, REPLICATION CLIENT ON *. * TO 'test01'@'%';
GRANT ALL PRIVILEGES ON `rds001`.`test100`  TO 'test01'@'%';
GRANT SELECT ON `mysql`.`help_topic` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`func` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`time_zone` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`slow_log` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`time_zone_transition` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`event` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`proc` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`help_category` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`help_relation` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`help_keyword` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`general_log` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`time_zone_leap_second` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`time_zone_transition_type` TO 'test01'@'%';
GRANT SELECT ON `mysql`.`time_zone_name` TO 'test01'@'%';
```

> **Note** If you change *test100* in the third line of code to the *wildcard, you will give the test01 user the permissions to view its Authorized Databases in the ApsaraDB for RDS console.

- Authorize the created account to query a view from the target database.

```
grant select on <The name of the target database>. <The name of the view you want to query with the created account> to <The username of the created account>;
```

Example:

To authorize the test01 user to query the view_test1 view from the rds001 database, execute the following SQL statement:

```
grant select on rds001.view_test1 to test01;
```

- Authorize the created account to update or query a field name in a table from the target database.

```
grant update (<The field name you want to update with the created account>) on table <The name of the table where the field name you want to update resides> to <The username of the created account>;     -----Authorize the created account to update a field name in a table from the target database.
grant select (<The field name you want to query with the created account>) on table <The name of the table where the field name you want to query resides> to <The username of the created account>;     -----Authorize the created account to query a field name in a table from the target database.
```

Example:

To authorize the test01 user to update the testid field in the testtable table, execute the following SQL statement:

```
grant update (testid) on table testtable to test01;
```

In addition to executing SQL statements, you can choose **Tools > User Management** from the top navigation bar to change the permissions of an account.

20.10. System accounts of an ApsaraDB RDS for MySQL instance

This topic describes the system accounts that are provided in an ApsaraDB RDS for MySQL instance. In most cases, you do not need to consider the permissions and authorized operations of these system accounts.

Account	Description
root (aliyun_root in MySQL 5.7 and later versions)	The local O&M account that you can use to manage the RDS instance. For example, you can use this account to reconfigure the parameters that are related to the database engine and query the status of the RDS instance.

Account	Description
• aurora • rds_service	The account that you can use to remotely manage the RDS instance. If the RDS instance is faulty, you can provide this account to an Alibaba Cloud engineer. The engineer can use this account to log on to and manage the RDS instance. For example, the engineer can perform a primary/secondary switchover and monitor the RDS instance.
aurora_proxy	The account that you can use to forward connections after you enable the database proxy feature.
replicator	The account that you can use to replicate data from the RDS instance to its secondary RDS instance. This account is available only in RDS High-availability Edition.

Note All IP addresses of the preceding system accounts are internal IP addresses. You can run the `SELECT user();` command to view the current logon account and its IP address. Example:

```
'aurora_proxy'@'%';
'replicator'@'11.195.143.24';
'replicator'@'11.196.207.107';
'replicator'@'11.195.208.36';
'replicator'@'11.199.40.156';
'aliyun_root'@'127.0.0.1';
```

In the preceding return results, the 11 CIDR block of the replicator system account is a private CIDR block of Alibaba Cloud.

21.Database

21.1. Create a database on an ApsaraDB RDS for MySQL instance

This topic describes how to create a database on an ApsaraDB RDS for MySQL instance.

Prerequisites

An RDS instance is created. For more information, see Create an ApsaraDB RDS for MySQL instance.

Terms

- Instance: a virtualized database server, on which you can create and manage a number of databases.
- Database: a set of organized data that can be shared by a number of users. A database provides the minimal redundancy and is independent of applications. You can consider a database to be a warehouse that is used to store data.
- Character set: a collection of letters, special characters, and encoding rules that are used in a database.

Maximum number of databases and maximum number of tables

Account type	Maximum number of databases	Maximum number of tables
Privileged account	Unlimited	< 200,000
Standard account	500	< 200,000

Procedure

1.
2. In the left-side navigation pane, click **Databases**.
3. Click **Create Database**.
4. Configure the following parameters.

Parameter	Description
Database Name	◦ The name of the database must be 2 to 64 characters in length. ◦ The name of the database must start with a lowercase letter and end with a lowercase letter or digit. ◦ The name of the database can contain lowercase letters, digits, underscores (_), and hyphens (-). ◦ The name of the database must be unique within your RDS instance.

Parameter	Description
Supported Character Set	Specify the character set that is supported by the database. you can select utf8, gbk, latin1, or utf8mb4.
Authorized Account	Specify the account that is authorized to access the database. You can leave this parameter empty. In this case, you can specify the authorized account of the database after the database is created. For more information, see Modify the permissions of a standard account on an ApsaraDB RDS for MySQL instance. **Note** The **Authorized Account** drop-down list displays only the standard accounts that are created on your RDS instance. The privileged account has all permissions on all databases and does not require authorization.
Account type	Specify the permissions that you want to grant on the database. You can select the **Read/Write**, **Read-only**, **DDL Only**, or **DML Only** permissions. **Note** This parameter is available only when the **Authorized Account** parameter is set.
Description	Enter a description that is used to identify the database. The description can contain up to 256 characters.

5. Click **Create**.

What to do next

Connect to your RDS instance. For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.

Related operations

Operation	Description
CreateDatabase	Creates a database on an ApsaraDB RDS instance.

21.2. Delete a database from an ApsaraDB RDS for MySQL instance

This topic describes how to delete a database from an ApsaraDB RDS for MySQL instance. You can delete a database by using the ApsaraDB RDS console or an SQL statement.

Delete a database by using the ApsaraDB RDS console

1.
2. In the left-side navigation pane, click **Databases**.
3. In the **Actions** column click **Delete**.

4. In the message that appears, click **OK**.

Delete a database by using an SQL statement

1. Connect to the RDS instance to which the database belongs. For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.
2. Execute the following statement to delete the database:

```
drop database <database name>;
```

Related operations

Operation	Description
DeleteDatabase	Deletes a database from an ApsaraDB RDS instance.

22.Data security

22.1. Change the network isolation mode of an ApsaraDB RDS for MySQL instance to the enhanced whitelist mode

This topic describes how to change the network isolation mode of an ApsaraDB RDS for MySQL instance from the standard whitelist mode to the enhanced whitelist mode. An enhanced IP address whitelist can contain only the IP addresses from the classic network or virtual private clouds (VPCs).

Prerequisites

Your RDS instance uses local SSDs.

> **Note** The **enhanced whitelist mode** is no longer supported for new RDS instances. You can use only the **standard whitelist mode** for new RDS instances.

Context

RDS instances support the following network isolation modes:

- Standard whitelist mode

  A standard IP address whitelist can contain IP addresses from both the classic network and VPCs.

- Enhanced whitelist mode

  An enhanced IP address whitelist can contain only the IP addresses from the classic network or VPCs. When you create an enhanced IP address whitelist, you must specify the network type of the enhanced IP address whitelist.

Changes after you switch to the enhanced whitelist mode

- If your RDS instance resides in a VPC, an IP address whitelist of the VPC network type is automatically created. The new IP address whitelist contains all IP addresses and CIDR blocks that are replicated from the original IP address whitelists.
- If your RDS instance resides in the classic network, an IP address whitelist of the classic network type is automatically created. The new IP address whitelist contains all IP addresses and CIDR blocks that are replicated from the original IP address whitelists.
- If your RDS instance runs in hybrid access mode, the following two IP address whitelists are created: an IP address whitelist of the VPC network type and an IP address whitelist of the classic network type. The two IP address whitelists contain all IP addresses and CIDR blocks that are replicated from the original IP address whitelists. For more information, see Configure the hybrid access solution for an ApsaraDB RDS for MySQL instance.

Note

- After you switch to the enhanced whitelist mode, the Elastic Compute Service (ECS) instance groups that you configured remain unchanged. For more information, see Configure an IP address whitelist for an ApsaraDB RDS for PostgreSQL instance.
- ApsaraDB RDS requires approximately 3 minutes to switch your RDS instance to the enhanced whitelist mode. Your application remains connected to your RDS instance during the 3-minute period.

Precautions

- After you switch to the enhanced whitelist mode, you cannot roll the instance back to the standard whitelist mode.
- In enhanced whitelist mode, an IP address whitelist of the classic network type can also be used to allow access over the Internet. If you want to access your RDS instance from a host over the Internet, you must add the public IP address of the host to an IP address whitelist of the classic network type.

Procedure

1.
2. In the left-side navigation pane, click **Data Security**.
3. On the **Whitelist Settings** tab, click **Switch to Enhanced Whitelist (Recommended)**.
4. In the message that appears, click **Confirm**.

FAQ

- My RDS instance runs in enhanced whitelist mode. If I want to access my RDS instance from a host over the Internet, how do I determine the IP address whitelist to which I must add the public IP address of the host?

  If you want to access your RDS instance from a host over the Internet, you must add the public IP address of the host to an IP address whitelist of the classic network type.

- What advantages does the enhanced whitelist mode have over the standard whitelist mode?

  The enhanced whitelist mode allows you to distinguish between the IP addresses from the classic network and the IP addresses from VPCs. If you add an IP address to an IP address whitelist of the VPC network type, the IP address is granted access to your RDS instance only within the specified VPC. However, the IP address is not granted access to your RDS instance over the Internet. This increases the security of your RDS instance.

22.2. Set the whitelist

22.2.1. Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance

This topic describes how to configure an IP address whitelist for an ApsaraDB RDS for MySQL instance. After an RDS instance is created, you must configure IP address whitelists for the RDS instance. A device can access the RDS instance only after you add the IP address of the device to an IP address whitelist of the RDS instance .

Prerequisites

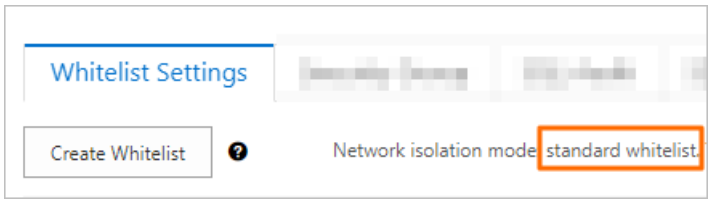
An ApsaraDB RDS for MySQL instance is created. For more information, see Create an ApsaraDB RDS for MySQL instance.

Procedure

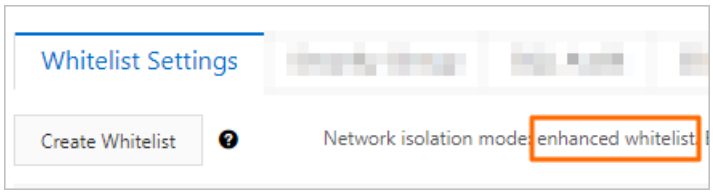
1.
2. In the left-side navigation pane, click **Data Security**.
3. View the network isolation mode of the RDS instance.

> **Note** Existing RDS instances may run in enhanced whitelist mode. All new RDS instances run in standard whitelist mode.

Standard whitelist mode

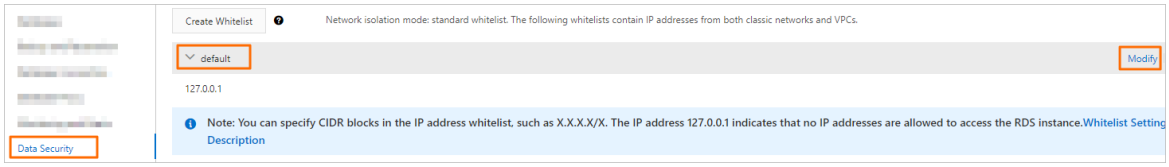


Enhanced whitelist mode



4. Click **Modify** to the right of the IP address whitelist labeled **default**.

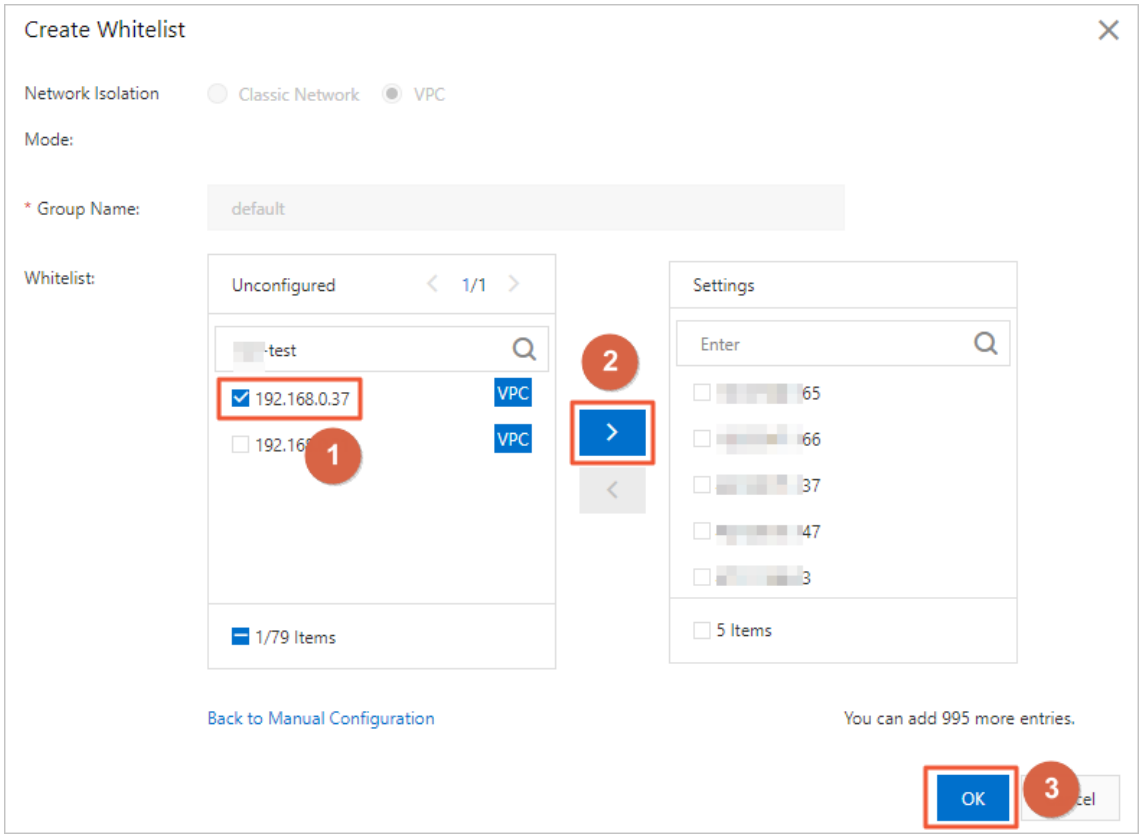
> **Note** You can also click **Create Whitelist** to create an IP address whitelist.



5. Use one of the following methods to configure an IP address whitelist for the RDS instance:
   - Method 1: Add the IP address of the server on which your application is deployed to the **IP Addresses** box. For more information about how to obtain the IP address of a server, see the "How to obtain IP addresses" section of this topic.

> **Note**
>
> - If you add multiple IP addresses and CIDR blocks to an IP address whitelist, you must separate the IP addresses and CIDR blocks with commas (,) and leave no spaces before and after each comma.
> - You can add a maximum of 1,000 IP addresses and CIDR blocks in total for each RDS instance. If you want to add a large number of IP addresses, we recommend that you merge the IP addresses into CIDR blocks, such as 10.10.10.0/24.
> - If an RDS instance runs in standard whitelist mode, you do not need to take note of special considerations when you configure IP address whitelists for the RDS instance. **If an RDS instance runs in enhanced whitelist mode, you must take note of the following considerations when you configure IP address whitelists for the RDS instance:**
>   - Add the **public IP addresses** or private IP addresses of -hosted Elastic Compute Service (ECS) instances to IP address whitelists of the **classic network type**. classic network
>   - Add the private IP addresses of VPC-hosted ECS instances to IP address whitelists of the **VPC network type**.

- Method 2: Click **Loading ECS Inner IP** to load the IP addresses of all ECS instances that are created within your Alibaba Cloud account. Then, select IP addresses and add them to an IP address whitelist.



The server on which your application is deployed can access the RDS instance only after you add the IP address of the server to an IP address whitelist of the RDS instance.

6. Click **OK**.

What to do next

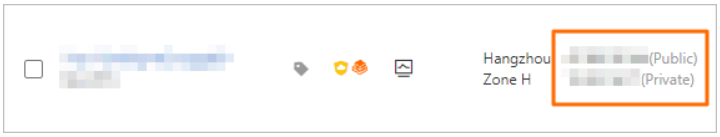
Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance

References

- For more information about how to modify an IP address whitelist for an RDS instance by calling an API operation, see Modify IP address whitelists.
- For more information about how to query the IP address whitelists of an RDS instance by calling an API operation, see Query IP address whitelists.
- For more information about how to configure an IP address whitelist for an RDS instance that runs a different database engine, see the following topics:
  - Configure an IP address whitelist for an ApsaraDB RDS for SQL Server instance
  - Configure an IP address whitelist for an ApsaraDB RDS for PostgreSQL instance
  - Configure an IP address whitelist for an ApsaraDB RDS for MariaDB TX instance

How to obtain IP addresses

Obtain IP addresses

<table>
<tr><th>Connection scenario</th><th>IP address to be obtained</th><th>How to obtain the IP address</th></tr>
<tr><td>the conditions for communication over an internal network</td><td>The private IP address of the ECS instance</td><td rowspan="2">1. Log on to the ECS console and go to the Instances page.
2. In the top navigation bar, select the region where the ECS instance resides.
3. View the public IP address and private IP address of the ECS instance.
  </td></tr>
<tr><td>You want to connect to the RDS instance from an ECS instance. The ECS instance and the RDS instance do not meet the conditions for communication over an internal network.</td><td>The public IP address of the ECS instance</td></tr>
</table>

Connection scenario	IP address to be obtained	How to obtain the IP address
You want to connect to the RDS instance from an on-premises device.	The public IP address of the on-premises device	On the on-premises device, use a search engine such as Google to search for **IP**. **Note** The IP address that you obtain by using this method may be inaccurate. For more information about how to obtain the accurate IP address of an on-premises device, see Why am I unable to connect to my ApsaraDB RDS for MySQL or ApsaraDB RDS for MariaDB instance from a local server over the Internet?

22.2.2. Configure a security group for an ApsaraDB RDS for MySQL instance

This topic describes how to configure a security group for an ApsaraDB RDS for MySQL instance. A security group is a virtual firewall that is used to control the inbound and outbound traffic of the Elastic Compute Service (ECS) instances in that security group. After you add a security group to your RDS instance, all the ECS instances in that security group can access the instance.

Prerequisites

Your RDS instance runs one of the following MySQL versions:

- MySQL 8.0
- MySQL 5.7
- MySQL 5.6

Scenario

After your RDS instance is created, you must configure IP address whitelists or security groups for the instance. This allows the specified devices to access the instance . For more information about how to configure an IP address whitelist, see Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance.

For more information about security groups, see Create a security group.

Note You can configure both IP address whitelists and security groups for your RDS instance. For more information, see Configure an IP address whitelist for an ApsaraDB RDS for MySQL instance. All the IP addresses in the configured IP address whitelists and all the ECS instances in the configured security groups are granted access to your RDS instance.

Precautions

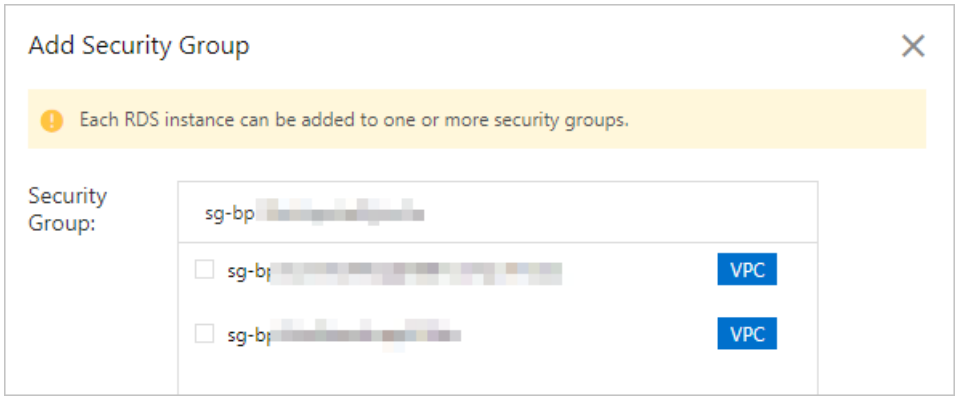
- The security groups that you can add to your RDS instance must have the same network type as the instance. For example, if your RDS instance uses the VPC network type, you can add only the security groups of the VPC network type.
- After you change the network type of your RDS instance, the configured security groups become invalid. In this case, you must reconfigure the security groups with the new network type.

- A maximum of 10 security groups are allowed per RDS instance.

Procedure

1.
2. In the left-side navigation pane, click **Data Security**. On the page that appears, click the **Security Group** tab.
3. Click **Add Security Group**.

   > **Note** Security groups that are followed by a **VPC** tag contain ECS instances that reside in virtual private clouds (VPCs).

4. Select the security group that you want to add, and then click **OK**.

What to do next

Create databases and accounts for an ApsaraDB RDS for MySQL instance

Related operations

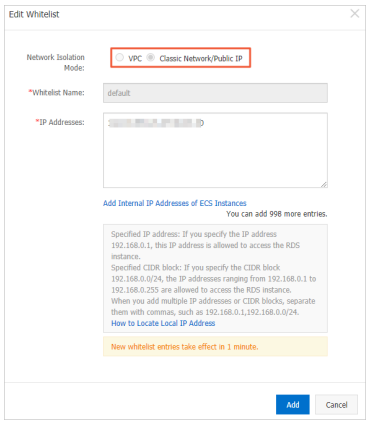
API	Description
DescribeSecurityGroupConfiguration	Queries details about the ECS security groups that are associated with an ApsaraDB RDS instance.
ModifySecurityGroupConfiguration	Modifies details about the ECS security groups that are associated with an ApsaraDB RDS instance.

22.2.3. Errors and FAQ about IP address whitelist settings in ApsaraDB RDS for MySQL

This topic introduces the common errors and provides answers to some commonly asked questions about the IP address whitelist settings of an ApsaraDB RDS for MySQL instance.

Common errors

Error	Description	Solution
No IP address whitelists are configured. This means that your RDS instance has only one default IP address whitelist, and this whitelist contains only the default IP address 127.0.0.1.	The 127.0.0.1 IP address indicates that no devices can access your RDS instance.	Add the IP addresses of the devices that require access to your RDS to an IP address whitelist.
The 0.0.0.0 IP address is added to an IP address whitelist during a connectivity test.	The IP address format is invalid.	Add the 0.0.0.0/0 Classless Inter-Domain Routing (CIDR) block to the IP address whitelist. **Notice** The 0.0.0.0/0 CIDR block indicates that all IP addresses are granted access to your RDS instance. We recommend that you add this CIDR block only for a connectivity test. When you run online workloads, do not add this CIDR block to an IP address whitelist.
The public IP addresses in the configured IP address whitelist are inaccessible.	• The public IP addresses dynamically change. • The tool or website that you use to query public IP addresses returns inaccurate results.	For more information, see Why am I unable to connect to my ApsaraDB RDS for MySQL or ApsaraDB RDS for MariaDB instance from a local server over the Internet?

Error	Description	Solution
The IP addresses of the devices that require access to your RDS instance are added to an enhanced IP address whitelist, and the network type of this whitelist differs from the network types of the devices.	In enhanced whitelist mode, ApsaraDB RDS distinguishes between the classic network and virtual private networks (VPCs).	Add the IP addresses to an IP address whitelist whose network type is the same as the network types of the devices. For example, if an IP address is added to an IP address whitelist of the VPC network type, you can connect to your RDS instance from the IP address only over a VPC. 

FAQ

- Can I configure both IP address whitelists and security groups for my RDS instance?

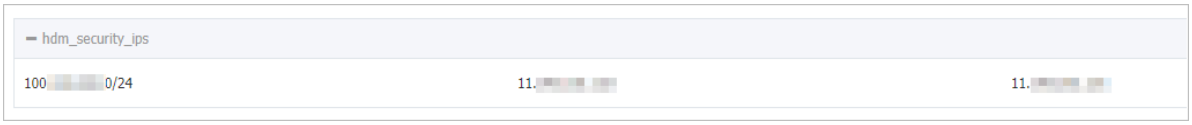
  Yes, you can configure both IP address whitelists and security groups for your RDS instance. All the IP addresses in the configured IP address whitelists and all the Elastic Compute Service (ECS) instances in the configured security groups are granted access to your RDS instance.

- After I configure an IP address whitelist for my RDS instance, does the IP address whitelist immediately take effect?

  After you configure an IP address whitelist for your RDS instance, the IP address whitelist requires about 1 minute to take effect.

- What are the IP address whitelists labeled **ali_dms_group** and **hdm_security_ips**?

  When you connect to your RDS instance from other Alibaba Cloud services, these services generate IP address whitelists upon your authorization. The generated IP address whitelists contain the IP addresses of the servers on which these services run. The IP address whitelist labeled ali_dms_group is generated by Data Management (DMS). The IP address whitelist labeled hdm_security_ips is generated by Database Autonomy Service (DAS). Do not modify or delete the IP address whitelists. If you modify or delete the IP address whitelists, these services cannot access your RDS instance. These services do not perform operations on your business data.

22.3. Configure SSL encryption for an ApsaraDB RDS for MySQL instance

This topic describes how to make data transmission for an ApsaraDB RDS for MySQL instance more secure by configuring SSL encryption. You must enable SSL encryption and install SSL certificates that are issued by certificate authorities (CAs) in the required applications. SSL encrypts the network connections at the transport layer between your RDS instance and your application. This enhances the security and integrity of data in transit but increases the response time.

Prerequisites

Your RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS Enterprise Edition
- MySQL 8.0 on RDS High-availability Edition
- MySQL 5.7 on RDS Enterprise Edition
- MySQL 5.7 on RDS High-availability Edition
- MySQL 5.6

> **Note** SSL encryption cannot be enabled for some read-only RDS instances that are created before September 2021. If you want to enable SSL encryption for these instances, you must submit a ticket.

Context

SSL is developed by Netscape to allow encrypted communication between a web server and a browser. SSL supports various encryption algorithms, such as RC4, MD5, and RSA. The Internet Engineering Task Force (IETF) upgraded SSL 3.0 to Transport Layer Security (TLS). However, the term "SSL encryption" is still used in the industry. In this topic, SSL encryption refers to TLS encryption.

> **Note** ApsaraDB RDS supports TLS 1.0, TLS 1.1, and TLS 1.2.

For more information about how to configure SSL encryption for an RDS instance that runs a different database engine, see the following topics:

- Configure SSL encryption for an ApsaraDB RDS for SQL Server instance
- Configure SSL encryption for an ApsaraDB RDS for PostgreSQL instance

Usage notes

- An SSL certificate remains valid for one year. If an SSL certificate is about to expire, Alibaba Cloud notifies you by mail and internal message and updates the SSL certificate within a specific time range. You can view internal messages on the Event Center page. The update of an SSL certificate causes transient connections on your RDS instance. You can customize the time to update the SSL certificate in the **Schedule Event** dialog box. For more information, see Manage scheduled events.

> **Note** SSL certificates are signed by using the private key of a root certificate. The automatic update of an SSL certificate is to sign the SSL certificate by using the private key of the root certificate. After the SSL certificate is automatically updated, the client can establish encrypted connections to the database without the need to download the SSL certificate file or configure the SSL certificate again. When the root certificate expires, you must manually update the validity period of the root certificate, download the SSL certificate file, and then configure the SSL certificate again. Otherwise, the client cannot establish encrypted connections to the database. For more information, see Update the validity period of an SSL certificate.

- SSL encryption may cause a significant increase in CPU utilization. We recommend that you enable SSL encryption only when you need to encrypt the connections to the public endpoint of your RDS instance. In most cases, connections that are established to the internal endpoint of your RDS instance are secure and do not require SSL encryption.
- SSL encryption is not supported for the connections to the read/write splitting endpoint of your RDS instance.
- If you disable SSL encryption, your application can connect to your RDS instance only over a non-SSL connection.
- If you disable SSL encryption, your RDS instance restarts. Proceed with caution.

Enable SSL encryption

1.
2. In the left-side navigation pane, click **Data Security**.
3. Click the **SSL Encryption** tab.

   > **Note** If the SSL Encryption tab cannot be found, you must check whether the RDS instance meets all requirements that are described in the "**Prerequisites**" section of this topic.

4. In the SSL Settings section, turn on **SSL Encryption**.
5. In the dialog box that appears, select the endpoint that you want to protect and click **OK**.

   > **Note** You can encrypt the link to the internal or public endpoint based on your business requirements. You can encrypt only one link.

6. Click **Download CA Certificate** to download the SSL certificate files as a compressed package.

   The downloaded package contains the following files:

   - P7B file: the SSL certificate file that is used for a Windows operating system
   - PEM file: the SSL certificate file that is used for an operating system other than Windows or an application that is not run on Windows
   - JKS file: the SSL certificate file that is stored in the Java-supported truststore. You can use this file to import the SSL certificate files from an SSL certificate chain into Java-based applications. The default password is apsaradb.

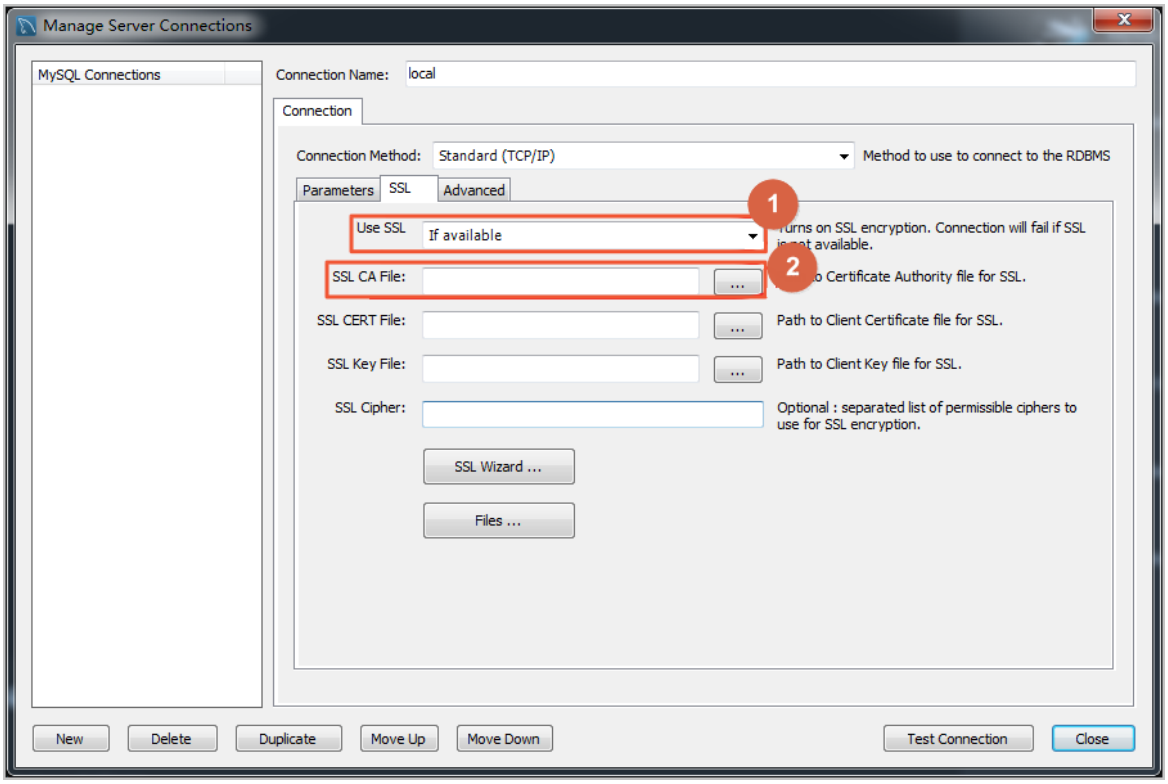
> **Note** When you use the JKS file in JDK 7 or JDK 8, you must modify the following default JDK security configuration items in the `jre/lib/security/Java.security` file on the host on which your application resides:
>
> ```
> jdk.tls.disabledAlgorithms=SSLv3, RC4, DH keySize < 224
> jdk.certpath.disabledAlgorithms=MD2, RSA keySize < 1024
> ```
>
> If you do not modify these configurations, the following error is returned. In most cases, similar errors are caused by invalid Java security configurations.
>
> ```
> Javax.net.ssl.SSLHandshakeException: DHPublicKey does not comply to algorithm con
> straints
> ```

Configure an SSL certificate

After SSL encryption is enabled, you must configure an SSL certificate on your application or client. If you do not configure an SSL certificate, your application or client cannot connect to your RDS instance. In this topic, MySQL Workbench and Navicat are used as examples. If you want to use other applications or clients, see the related instructions.

Perform the following steps to configure an SSL certificate on MySQL Workbench:

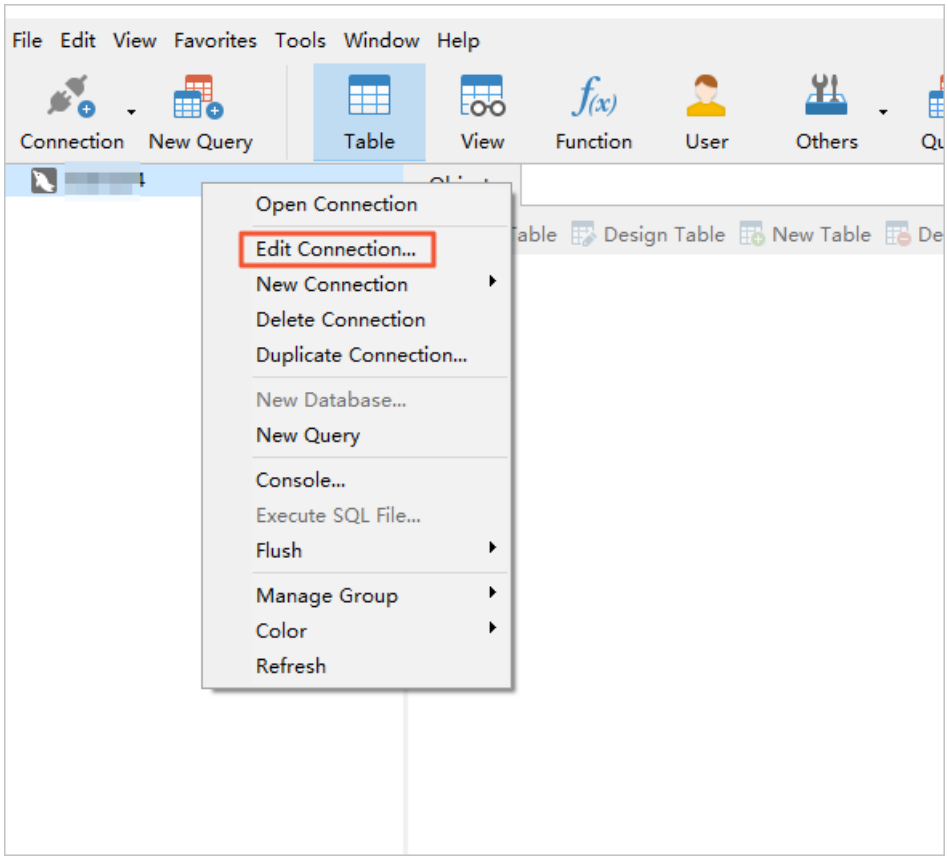
1. Start MySQL Workbench.
2. Choose **Database > Manage Connections**.
3. Configure the **Use SSL** parameter and import the SSL certificate file.



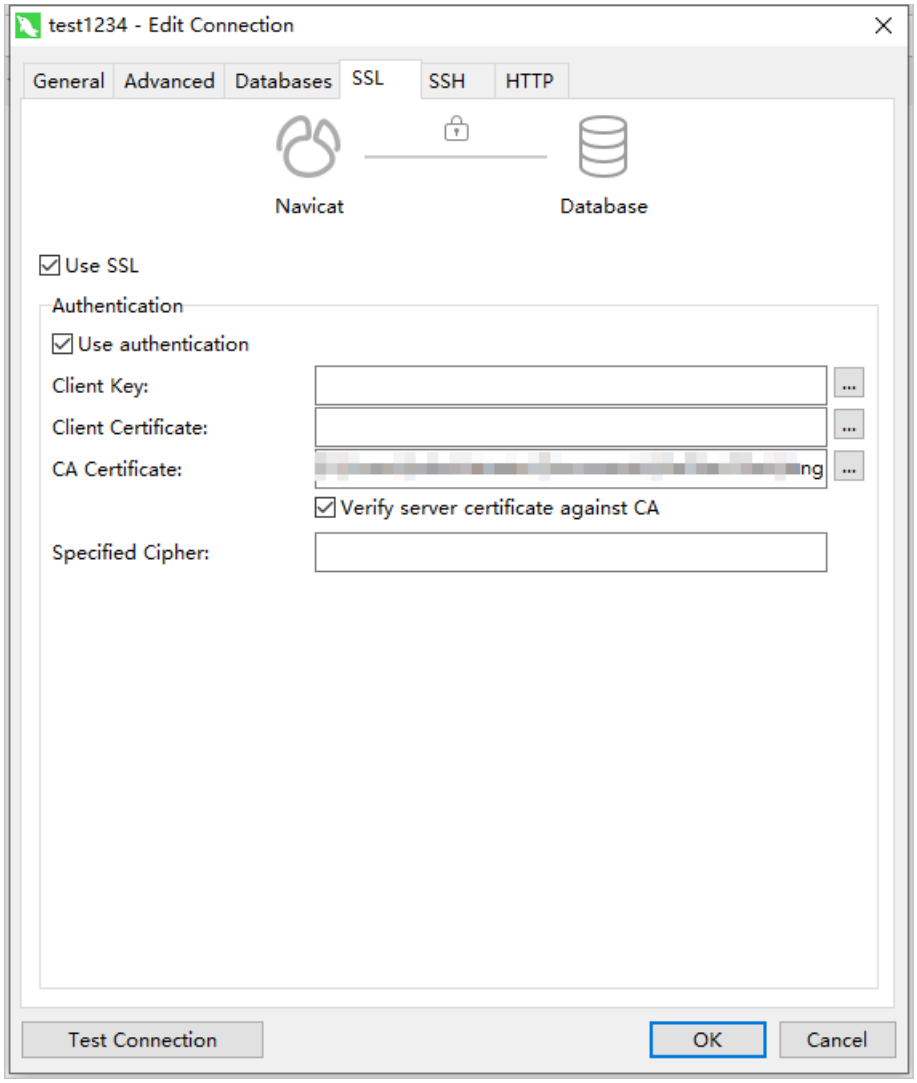
Perform the following steps to configure an SSL certificate on Navicat:

1. Start Navicat.

2. Right-click the database that you want to connect. Then, select **Edit Connection**.



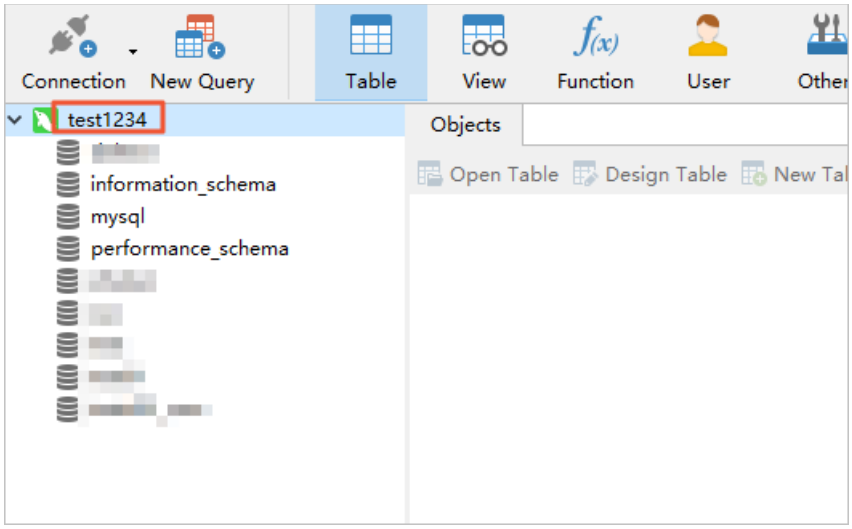
3. Click the **SSL** tab and select the path of the PEM certificate file, as shown in the following figure.



4. Click **OK**.

> **Note** If the "`connection is being used`" error is reported, the previous session remains connected. In this case, you must restart Navicat.

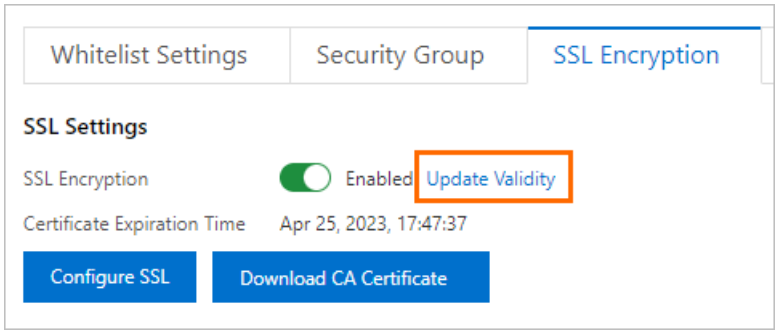
5. Double-click your database to check whether Navicat can connect to the database.



Update the validity period of an SSL certificate

> **Note**
>
> - The **Update Validity** operation causes your RDS instance to restart. Proceed with caution.
> - After you perform the **Update Validity** operation, you must download the SSL certificate file and configure the SSL certificate again.

1.
2. In the left-side navigation pane, click **Data Security**.
3. On the page that appears, click the **SSL Encryption** tab. Then, click **Update Validity**.



> **Notice** The Update Validity operation causes your RDS instance to restart. We recommend that you update the validity period during off-peak hours.

Disable SSL encryption

> **Note**
>
> - When you disable SSL encryption, your RDS instance restarts. In this case, ApsaraDB RDS triggers a primary/secondary switchover to reduce the impacts on your workloads. We still recommend that you disable SSL encryption during off-peak hours.
> - After you disable SSL encryption, access performance increases, but security decreases. We recommend that you disable SSL encryption only in secure environments.
> - If you disable SSL encryption, your application can connect to your RDS instance only over a non-SSL connection.

1.
2. In the left-side navigation pane, click **Data Security**.
3. Click the **SSL Encryption** tab.
4. Turn off **SSL Encryption**. In the message that appears, click **OK**.

Appendix: Sample code for connections over SSL

- Sample code in Java:

```
<dependency>
       <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
       <version>8.0.11</version>
</dependency>
-----------------------------------------------------------demo-----------------------------
---------------------------------
package com.aliyun.sample;
import com.mysql.cj.jdbc.MysqlDataSource;
import java.sql.Connection;
import java.sql.SQLException;
public class Sample {
    public static void main(String[] args) {
        Connection conn = null;
        MysqlDataSource mysqlDS=null;
        try{
            mysqlDS = new MysqlDataSource();
            // set useSSL=true and provide truststore for server certificate verification
.
            mysqlDS.setUseSSL(true);
            mysqlDS.setClientCertificateKeyStoreType("JKS");
            // Path of the ApsaraDB-CA-Chain.jks file:/D:\ApsaraDB-CA-Chain\ApsaraDB-CA-C
hain.jks.
            mysqlDS.setClientCertificateKeyStoreUrl("file:/D:\\xxxx\\ApsaraDB-CA-Chain.jk
s");
            // Specify the password that is used to establish a connection. In Java, the
default password is apsaradb.
            mysqlDS.setClientCertificateKeyStorePassword("apsaradb");
            mysqlDS.setTrustCertificateKeyStoreType("JKS");
            // Path of the ApsaraDB-CA-Chain.jks file:/D:\ApsaraDB-CA-Chain\ApsaraDB-CA-C
hain.jks.
            mysqlDS.setTrustCertificateKeyStoreUrl("file:/D:\\ApsaraDB-CA-Chain\\ApsaraDB
-CA-Chain.jks");
```

```
            // Specify the password that is used to establish a connection. In Java, the
default password is apsaradb.
            mysqlDS.setTrustCertificateKeyStorePassword("apsaradb");
            // Specify the endpoint that is used to connect to the specified database in
your RDS instance.
            mysqlDS.setServerName("rm-xxxxxx.mysql.rds.aliyuncs.com");
            // Specify the port number that is used to connect to the specified database
in your RDS instance.
            mysqlDS.setPort(3306);
            // Specify the username of the account that is used to connect to the specifi
ed database in your RDS instance.
            mysqlDS.setUser("xxxxxx");
            // Specify the password of the account that is used to connect to the specifi
ed database in your RDS instance.
            mysqlDS.setPassword("xxxxxx");
            // Specify the name of the database that you want to connect on your RDS inst
ance.
            mysqlDS.setDatabaseName("xxxxxx");
            conn = mysqlDS.getConnection();
        }catch(Exception e){
            e.printStackTrace();
        } finally {
            try {
                if (conn != null)
                    conn.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

- Sample code in Python:

```
# Run the pip install pymysql command to install pymysql.
import pymysql
try:
    conn = pymysql.connect(host='******.mysql.rds.aliyuncs.com',user='*****',passwd='****
**',db='*****',ssl=True,
            ssl_ca='/path/to/path/ApsaraDB-CA-Chain.pem')
    cursor = conn.cursor()
    cursor.execute('select version()')
    data = cursor.fetchone()
    print('Database version:', data[0])
    cursor.close()
except pymysql.Error as e:
    print(e)
```

FAQ

If I do not update the validity period of an expired SSL certificate, does my RDS instance malfunction or data security deteriorate?

If you do not update an expired SSL certificate, your RDS instance still runs as expected and no security risks occur. However, your application cannot establish encrypted connections to your RDS instance.

22.4. Configure TDE for an ApsaraDB RDS for MySQL instance

This topic describes how to configure Transparent Data Encryption (TDE) for an ApsaraDB RDS for MySQL instance. You can use TDE to perform real-time I/O encryption and decryption on data files. Data is encrypted before it is written to a disk and is decrypted when it is read from a disk to the memory. TDE does not increase the size of data files. Developers can use TDE without the need to modify the configurations of their applications.

Prerequisites

- Your RDS instance runs one of the following MySQL versions and RDS editions:
  - MySQL 8.0 (with a minor engine version of 20191015 or later) on RDS High-availability Edition with local SSDs
  - MySQL 5.7 (with a minor engine version of 20191015 or later) on RDS High-availability Edition with local SSDs
  - MySQL 5.6

  > **Note** For more information about how to update the minor engine version of an ApsaraDB RDS for MySQL instance, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.

- Key Management Service (KMS) is activated. If KMS is not activated, you can activate KMS when you enable TDE.

Context

The key that is used for TDE is created and managed by KMS. ApsaraDB RDS does not provide the key or certificates that are required for TDE. In some zones, you can use an automatically generated key or a custom key that is generated from your own key material. After you select a key, you must authorize your RDS instance to use that key.

> **Note** After you enable TDE, the AES_128_ECB algorithm is used for TDE.

Precautions

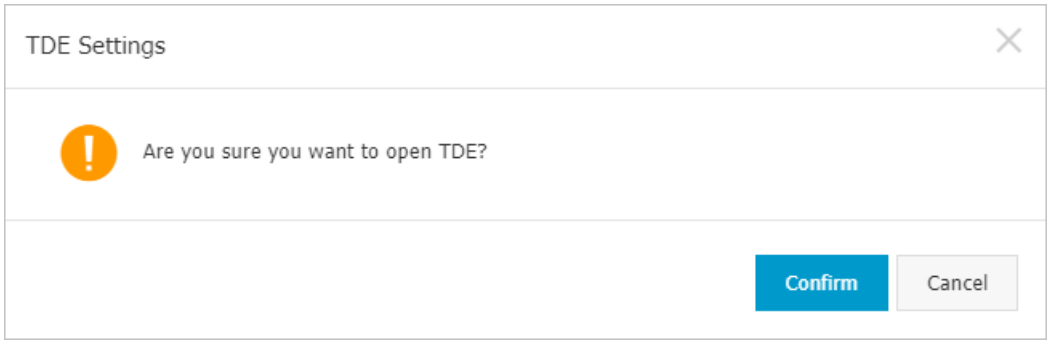
- We recommend that you update the minor engine version of your RDS instance to the latest version to ensure the stability of the instance. If read-only RDS instances are attached to your RDS instance, we recommend that you update the minor engine versions of your RDS instance and all the read-only RDS instances to the latest version. For more information, see Update the minor engine version of an ApsaraDB RDS for MySQL instance.
- When you enable TDE, your RDS instance restarts. As a result, a transient connection occurs. Proceed with caution. Before you enable TDE, we recommend that you make appropriate arrangements for your workloads based on your business requirements.
- After you enable TDE, you cannot disable TDE.

- After you enable TDE, you cannot change the key that is used for TDE.
- After you enable TDE, you must decrypt the data on your RDS instance if you want to restore the data to an on-premises database. For more information, see the "Decrypt a table" section of this topic.
- After you enable TDE, the CPU utilization of your RDS instance significantly increases.
- If you use an existing custom key for TDE, take note of the following information:
  - If you disable the key, configure a plan to delete the key, or delete the key material, the key becomes unavailable.
  - If your RDS instance restarts after you revoke the key, your RDS instance becomes unavailable.
  - You must use an Alibaba Cloud account or an account to which the AliyunSTSAssumeRoleAccess policy is attached.

> **Note** For more information about keys, see What is Key Management Service?

Use an automatically generated key

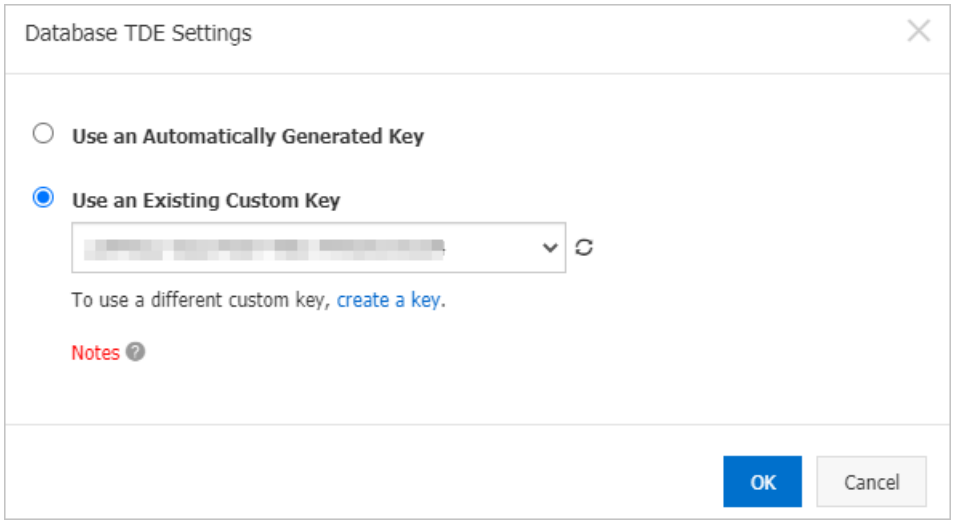
1.
2. In the left-side navigation pane, click **Data Security**.
3. On the **TDE** tab, turn on **TDE Status**.
4. In the dialog box that appears, select **Use an Automatically Generated Key** and click **OK**.



Use an existing custom key

1.
2. In the left-side navigation pane, click **Data Security**.
3. On the **TDE** tab, turn on **TDE Status**.
4. In the dialog box that appears, select **Use an Existing Custom Key** and click **OK**.

> **Note** If you do not have a custom key, you can click **go to the KMS console** to go to the KMS console. In the KMS console, you can create a key and import your own key material. For more information, see Create a CMK.



Encrypt a table

If you want to encrypt a table on an RDS instance, you must log on to the RDS instance and execute the following statements:

- If the RDS instance runs MySQL 5.6, execute the following statement:

```
alter table <tablename> engine=innodb,block_format=encrypted;
```

- If the RDS instance runs MySQL 5.7 or MySQL 8.0, execute the following statement:

```
alter table <tablename> encryption='Y';
```

Decrypt a table

If you want to decrypt a table on an RDS instance, you must log on to the RDS instance and execute the following statements:

- If the RDS instance runs MySQL 5.6, execute the following statement:

```
alter table <tablename> engine=innodb,block_format=default;
```

- If the RDS instance runs MySQL 5.7 or MySQL 8.0, execute the following statement:

```
alter table <tablename> encryption='N';
```

FAQ

- After I enable TDE, can I use common database tools such as Navicat?

  Yes, after you enable TDE, you can use common database tools such as Navicat.

- After I enable TDE, can I migrate data from my RDS instance to a different RDS instance?

  Yes, after you enable TDE, you can migrate data from your RDS instance to a different RDS instance.

- After I enable TDE, why is my data still in plaintext?

After you enable TDE, your data is stored in ciphertext. However, when the data is queried, it is decrypted and then loaded in plaintext to the memory. TDE encrypts backup files to prevent data leaks. Before you restore the data of your RDS instance from an encrypted backup file to your computer, you must decrypt the file. For more information, see the "Decrypt a table" section of this topic.

References

Configure TDE for an ApsaraDB RDS for SQL Server instance

Related operations

Operation	Description
Enable TDE	Enables TDE for an ApsaraDB RDS instance.

22.5. Configure the disk encryption feature for an ApsaraDB RDS for MySQL instance

This topic describes how to configure the disk encryption feature for an ApsaraDB RDS for MySQL instance that is equipped with standard SSDs or enhanced SSDs (ESSDs). The disk encryption feature encrypts the data on each disk of your RDS instance by using block storage. This way, your data cannot be cracked even if it is leaked.

For more information about the disk encryption feature in other database engines, see the following topics:

- Configure disk encryption for an ApsaraDB RDS for SQL Server instance
- Configure disk encryption for an ApsaraDB RDS for PostgreSQL instance

Prerequisites

- Your RDS instance is being created. The disk encryption feature cannot be enabled after your RDS instance is created. For more information, see Create an ApsaraDB RDS for MySQL instance.
- The ESSD storage type is selected for your RDS instance. For more information, see Storage types.
- High-availability Edition is selected for your RDS instance. For more information, see Overview of ApsaraDB RDS editions.

Billing

The disk encryption feature is free of charge. You are not charged for the read and write operations that you perform on the encrypted disks.

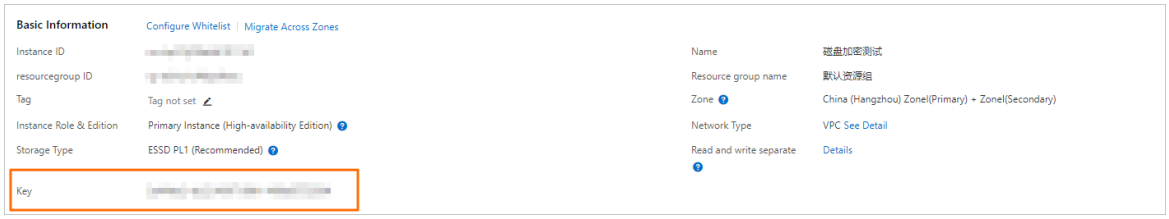
Precautions

- The disk encryption feature cannot be disabled after you enable it.
- If you enable the disk encryption feature for your RDS instance, your RDS instance does not support cross-region backups. For more information, see Enable cross-region backups for an ApsaraDB RDS for MySQL instance.

- The disk encryption feature does not interrupt your business, and you do not need to modify your application.
- After you enable the disk encryption feature for your RDS instance, the snapshots that are created for your RDS instance are automatically encrypted. If you use the encrypted snapshots to create an RDS instance that uses standard SSDs or ESSDs, the disk encryption feature is automatically enabled for the new RDS instance.
- If your Alibaba Cloud Key Management Service (KMS) is overdue, the standard SSDs or ESSDs of your RDS instance become unavailable. Make sure that your KMS is normal. For more information, see What is KMS?
- If you disable or delete the customer master key (CMK) that is used for disk encryption, your RDS instance cannot run as normal. For example, you cannot create snapshots, restore data from snapshots, or rebuild the secondary RDS instance of your RDS instance.

Check whether the disk encryption feature is enabled for an RDS instance

1.
2. In the **Basic Information** section, check whether the **Key** parameter can be found. If you can find the parameter, the disk encryption feature is enabled for the RDS instance.



Enable the disk encryption feature for an RDS instance

When you create an RDS instance, set Edition to **High-availability**, select the **ESSD** storage type, select **Disk Encryption**, and then configure the Key parameter. For more information, see Create an ApsaraDB RDS for MySQL instance.

> **Note** For information about how to create a key, see Create a CMK.

Related operations

Operation	Description
Create an instance	Creates an ApsaraDB RDS instance.

22.6. Enable or disable the release protection feature for an ApsaraDB RDS for MySQL instance

This topic describes how to enable or disable the release protection feature for an ApsaraDB RDS for MySQL instance. If your RDS instance uses the pay-as-you-go billing method and runs critical workloads, you can enable the release protection feature for pay-as-you-go RDS instances. This feature prevents your RDS instance from being manually released due to unintended operations or lack of communication among team members.

Prerequisites

Your RDS instance is a pay-as-you-go RDS instance.

Precautions

The release protection feature cannot prevent the automatic release of RDS instances in normal scenarios such as the following scenarios:

- A payment in your account is overdue for more than 15 days.
- The RDS instance does not comply with the applicable security compliance policies.

Benefits of release protection

If you release an RDS instance for which the release protection feature is enabled, the following result is returned:

- If you release the RDS instance in the ApsaraDB RDS console, the "`The instance cannot be released because release protection has been enabled. Disable release protection first`" message is displayed.
- If you call the DeleteDBInstnace operation to release the RDS instance, the error code `OperationDenied.DeletionProtection` is returned.

Enable the release protection feature when you create an RDS instance

This section describes how to configure release protection settings when you create an RDS instance. For more information about how to create an instance, see Create an ApsaraDB RDS for MySQL instance.

1.
2. On the **Instances** page, click **Create Instance**.
3. In the **Basic Configurations** step, set **Billing Method** to **Pay-As-You-Go** and complete the remaining configurations. Click **Next: Instance Configuration**.
4. In the **Instance Configurations** step, select **Prevent release through the console or API by mistake** and complete the remaining configurations. Click **Next: Confirm Order**.
5. Complete the remaining configurations until the RDS instance is created.

> **Note** When you can call the CreateDBInsance or CloneDBInstance operation to create an RDS instance, you can enable or disable the release protection feature for the RDS instance by setting the DeletionProtection parameter.

Modify release protection settings

You can also enable or disable the release protection feature for an RDS instance by modifying the settings of the RDS instance.

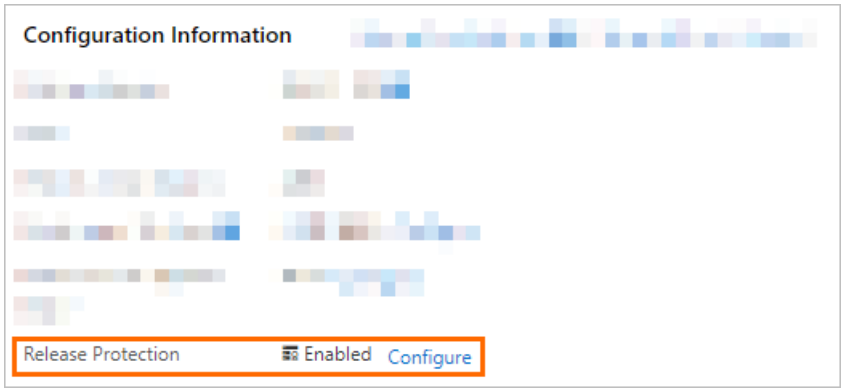
1.

2. On the **Instances** page, find the RDS instance whose release protection settings you want to modify. In the **Actions** column, click **More** and select **Change Instance Release Protection Settings**.
3. In the Change Release Protection Setting dialog box, turn on or turn off **Release Protection**.
4. Click **OK**.

> **Note** You can also call the ModifyDBInstanceDeletionProtection operation to enable or disable the release protection feature for an RDS instance.

Check whether the release protection feature is enabled

1.
2. On the **Basic Information** page, view the **Release Protection** section of the **Configuration Information** section.



Related operations

Operation	Description
Create an instance	Creates an ApsaraDB RDS instance.
Restore data to a new ApsaraDB RDS instance	Restores the data of an ApsaraDB RDS instance to a new instance. The new instance is also called a cloned instance.
Enable or disable the release protection feature	Enables or disables the release protection feature for an ApsaraDB RDS instance.

22.7. Best practices for data security

ApsaraDB RDS provides basic protection for critical data. This topic describes how to create and configure ApsaraDB RDS instances to further improve the security level of data.

Instance disaster recovery

- Enterprise Edition (formerly known as Finance Edition)

To further meet the high reliability and data security requirements in various business scenarios, ApsaraDB RDS provides Enterprise Edition. In Enterprise Edition, your database system consists of a primary RDS instance and two secondary RDS instances. Data is replicated between these instances to ensure strong data consistency and financial-grade reliability.

You can select Enterprise Edition when you create an RDS instance. For more information, see Create an ApsaraDB RDS for MySQL instance.

- Multi-zone deployment

  Each region consists of multiple zones. The network latency between the zones in the same region is less than 3 ms. A fault in one zone does not affect the services in the other zones. If you select the multi-zone deployment method, the physical hosts on which your RDS instance resides can reside in different zones. This way, if one zone fails, your workloads can be switched over to another zone within a short period of time. The switchover process is invisible to users and does not require changes to the code of your application.

  You can select the multi-zone deployment method when you create an RDS instance. For more information, see Create an ApsaraDB RDS for MySQL instance.

  If you select the single-zone deployment method, you can migrate your RDS instance to multiple zones. You can do this only when the region where your RDS instance resides can provide multiple available zones. For more information, see Migrate an ApsaraDB RDS for MySQL instance across zones in the same region.

- Cross-region disaster recovery RDS instances

  ApsaraDB RDS uses Data Transmission Service (DTS) to synchronize data in real time between a primary RDS instance and its disaster recovery RDS instance that resides in a different region than the region of the primary RDS instance. Both the primary RDS instance and the disaster recovery RDS instance are deployed based on a primary/secondary high-availability architecture. If the primary RDS instance and the secondary RDS instance are inaccessible due to unexpected natural disasters, you can switch your workloads over to the disaster recovery RDS instance and then update the endpoint information on your application to minimize downtime.

  For more information, see Create a disaster recovery ApsaraDB RDS for MySQL instance.

- Cross-region backups

  ApsaraDB RDS supports cross-region backups. After you enable cross-region backups, the backup files of your RDS instance are automatically replicated to an Object Storage Service (OSS) bucket in a different region. The cross-region data backup files can be used for monitoring and disaster recovery. Cross-region backup files are independent of RDS instances. After your RDS instance is released, its cross-region backup files are still retained based on the cross-region backup retention period that you specify.

  For more information, see Enable cross-region backups for an ApsaraDB RDS for MySQL instance.

Access control

- RAM user authorization

  Resource Access Management (RAM) allows you to create and manage RAM users and control the permissions of RAM users on the resources within your Alibaba Cloud account. If multiple users in your enterprise need to simultaneously use the same resources, we recommend that you follow the principle of least privilege (PoLP) when you assign permissions to the users. This prevents the users from sharing the same key and reduces information security risks for your enterprise.

  For more information, see Use RAM for resource authorization.

- Prohibition to create RDS instances with unencrypted disks

  You can configure a RAM policy for a RAM user to prevent the RAM user from creating RDS instances whose disks are not encrypted.

  For more information, see Use RAM policies to manage the permissions of RAM users on ApsaraDB RDS instances.

- Database account authorization

  ApsaraDB RDS allows you to grant permissions to database accounts based on your business requirements in the production environment.

  You can create an account and grant the permissions on specific databases to the account in the ApsaraDB RDS console. For more information, see Create databases and accounts for an ApsaraDB RDS for MySQL instance.

  If you want to use an account to manage a specific table in a database, you can execute an SQL statement to grant the permissions on the table to the account. For more information, see Authorize accounts to manage tables, views, and fields.

Network isolation

- Virtual Private Cloud (VPC)

  ApsaraDB RDS supports multiple network types. We recommend that you use the **VPC** network type.

  A VPC is an isolated network that provides higher security and higher performance than the classic network. Before you can create an RDS instance in a VPC, you must create a VPC. For more information, see Default VPC and default vSwitch.

  If your RDS instance resides in the classic network, you can migrate your RDS instance to a VPC. For more information, see Change the network type of an ApsaraDB RDS for MySQL instance. If your RDS instance is deployed in a VPC, no additional configuration is required.

- IP address whitelists

  After you create an RDS instance, you must configure IP address whitelists for the RDS instance to allow access from external devices.

  For more information, see Use a database client or the CLI to connect to an ApsaraDB RDS for MySQL instance.

Log audit

- SQL Explorer

  ApsaraDB RDS provides the SQL Explorer feature. You can use this feature to perform security audits and performance diagnostics on your RDS instance.

  For more information, see Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance.

- Log management

  ApsaraDB RDS provides the log management feature. You can use this feature to view the error logs, slow query log details, slow query log summary, and primary/secondary switchover logs of your RDS instance. These logs help you locate faults.

  For more information, see View the logs of an ApsaraDB RDS for MySQL instance.

- Event history

ApsaraDB RDS provides the event history feature. You can use this feature to view the O&M events that are performed by users and Alibaba Cloud on your RDS instance. These events include instance creation and parameter reconfiguration.

For more information, see View the event history of an ApsaraDB RDS instance.

Data encryption

- SSL encryption

  When you connect to your RDS instance over the Internet, you can enable SSL encryption and install SSL CA certificates on your application. SSL encrypts the network connections at the transport layer between your RDS instance and your application. This enhances the security and integrity of data in transit but increases the response time.

  For more information, see Configure SSL encryption for an ApsaraDB RDS for MySQL instance.

- Transparent Data Encryption (TDE)

  You can use TDE to perform real-time I/O encryption and decryption on data files. Data is encrypted before it is written to the disk and is decrypted when it is read from the disk to the memory. After you enable TDE for your RDS instance, the size of data files in your RDS instance does not increase. You can use TDE without the need to modify the configurations of your application.

  For more information, see Configure TDE for an ApsaraDB RDS for MySQL instance.

23.Events Management

23.1. View the event history of an ApsaraDB RDS instance

This topic describes how to view the operation and maintenance (O&M) events that are performed by users and Alibaba Cloud on an ApsaraDB RDS for SQL Server instance. These events include instance creation and parameter reconfiguration.

Billing

The event history feature is free of charge in the public preview phase, but starts to be charged after the public preview phase ends.

Scenarios

- Track instance management operations.
- Audit the security of instance management operations.
- Audit the compliance of the instance management operations that are performed by Alibaba Cloud. This applies to security-demanding sectors, such as finance and government affairs.

View the event history feature

1. Log on to the RDS management console , in the let-side navigation pane, click **Event Center**, and then select a region above.
2. Click the **Historical Events** tab.

Introduction to the Historical Events page

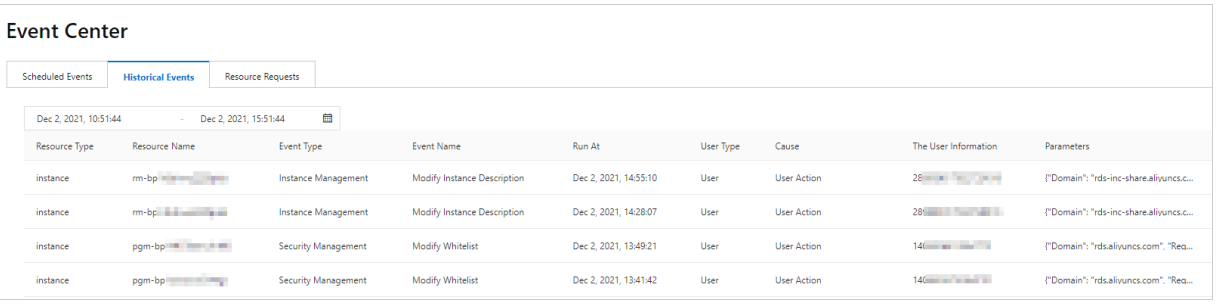
The Historical Events page shows details about historical events in the selected region. These details include the resource types, resource names, and event types. The following table describes the parameters of a historical event.

Parameter	Description
Resource Type	The type of the RDS resource managed in the event. Only the **Instance** resource type is supported.
Resource Name	The name of the RDS resource managed in the event. If the value of the **Resource Type** parameter is **Instance**, the **Resource Name** column displays the ID of the involved RDS instance.
Event Type	The type of the event, for example, **Instance Management**, **Database Management**, **Read-write Splitting**, and **Network Management**. For more information, see Events.
Event Name	The operation executed in the event. For example, if the event type is **Instance Management**, supported operations include **Create Instance**, **Delete Instance**, **Change Specifications**, and **Restart Instance**. For more information, see Events.

Parameter	Description
Run At	The time when the event was executed.
User Type	The initiator of the event. Valid values: • User: initiates operations by using the ApsaraDB RDS console or the API. • System: initiates automatic O&M operations or periodic tasks. • O&M Administrator: initiates manual O&M operations.
Cause	The cause of the event. Valid values: • User Action: The event was initiated from a user by using the ApsaraDB RDS console or the API. • System Action or O&M Action: The event was initiated from the system or an O&M administrator.
The user information	The ID of the account that is used by a user to perform the event.
Parameters	The request parameters used by a user to initiate the event in the ApsaraDB RDS console.

Note

- The Historical Events page shows the historical events that were generated about 5 minutes earlier.
- Historical Events are presented specific to regions. You can select a region in the top navigation bar and then view the historical events in the selected region.



Events

Event type	Operation
	Restart Instance (RestartDBInstance)
	Renew (RenewInstance)
	Change Specifications (ModifyDBInstanceSpec)
	Migrate Across Zones (MigrateToOtherZone)

Event type	Operation
Instance Management	Shrink Log (PurgeDBInstanceLog)
	Upgrade Kernel Version (UpgradeDBInstanceEngineVersion)
	Modify Instance Description (ModifyDBInstanceDescription)
	Modify Maintenance Window (ModifyDBInstanceMaintainTime)
	Create Read-only Instance (CreateReadOnlyDBInstance)
	Destroy Instance (DestroyDBInstance)
	Modify Upgrade Mode of Kernel Version (ModifyDBInstanceAutoUpgradeMinorVersion)
	Edit Parameters (ModifyParameter)
CloudDBA	Create Diagnostics Report (CreateDiagnosticReport)
Database Management	Create Database (CreateDatabase)
	Delete Database (DeleteDatabase)
	Modify Database Description (ModifyDBDescription)
	Replicate Database Between Instances (CopyDatabaseBetweenInstances)
	Modify System Collation and Time Zone (ModifyCollationTimeZone)
Read-write Splitting	Create Read-write Splitting Endpoint (AllocateReadWriteSplittingConnection)
	Query System-assigned Weight (CalculateDBInstanceWeight)
	Modify Read-write Splitting Policy (ModifyReadWriteSplittingConnection)
	Release Read-write Splitting Endpoint (ReleaseReadWriteSplittingConnection)
Security Management	Enable Enhanced Whitelist (MigrateSecurityIPMode)
	Enable SSL (ModifyDBInstanceSSL)
	Enable TDE (ModifyDBInstanceTDE)
	Modify Whitelist (ModifySecurityIps)
	Create Account (CreateAccount)
	Delete Account (DeleteAccount)

Event type	Operation
Account Management	Authorize Account to Access Database (GrantAccountPrivilege)
	Revoke Database Permissions from Account (RevokeAccountPrivilege)
	Modify Description of Database Account (ModifyAccountDescription)
	Reset Account Password (ResetAccountPassword)
	Reset Permissions of Superuser Account (ResetAccount)
High Availability (HA)	Trigger Switchover Between Primary and Secondary Instances (SwitchDBInstanceHA)
	Modify HA Mode (ModifyDBInstanceHAConfig)
Network Management	Apply for Public Endpoint (AllocateInstancePublicConnection)
	Modify Expiry Time of Endpoint (ModifyDBInstanceNetworkExpireTime)
	Modify Endpoint and Port (ModifyDBInstanceConnectionString)
	Switch Network Type (ModifyDBInstanceNetworkType)
	Release Public Endpoint (ReleaseInstancePublicConnection)
	Switch Between Internal and Public Endpoints (SwitchDBInstanceNetType)
Log Management	Enable/disable Log Audit (ModifySQLCollectorPolicy)
Backup Restoration	Create Data Backup (CreateBackup)
	Clone Instance (CloneDBInstance)
	Create Temporary Instance (CreateTempDBInstance)
	Modify Backup Policy (ModifyBackupPolicy)
	Restore Backup Set to Original Instance (RestoreDBInstance)
	Delete Data Backup (DeleteBackup)
	Restore Database (RecoveryDBInstance)
Cross-region Backup Restoration	Restore Data to New Instance Across Regions (CreateDdrInstance)
	Modify Cross-region Backup Settings (ModifyInstanceCrossBackupPolicy)
SQL Server Backup Migration to	Restore Backup File in OSS to RDS Instance (CreateMigrateTask)

Event type	Operation
Cloud	Make Database Available While Migrating Backup Data to Cloud (CreateOnlineDatabaseTask)
Monitoring	Set Monitoring Frequency (ModifyDBInstanceMonitor)
Data Migration	Create Upload Path for SQL Server (CreateUploadPathForSQLServer)
	Import Data from Other RDS (ImportDatabaseBetweenInstances)
	Cancel Migration Task (CancelImport)
Tag Management	Bind Tags to Instance (AddTagsToResource)
	Remove Tag (RemoveTagsFromResource)

Related operations

Operation	Description
Query historical events	Queries the events of an ApsaraDB RDS instance.
Query status of the event history feature	Queries the status of the historical events feature of an ApsaraDB RDS instance.
Enable or disable the event history feature	Enables or disables the historical events feature of an ApsaraDB RDS instance.

23.2. Manage scheduled events

If an O&M event, such as instance migration or version upgrade, needs to be managed, you will receive phone calls, emails, or internal messages. In addition, you are prompted to manage the event after you log on to the ApsaraDB RDS console. You can view the types, regions, and IDs of the RDS instances that are affected by the O&M event, and the cause and impacts of the event. You can also change the scheduled time of switchovers that are triggered by the O&M event.

Prerequisites

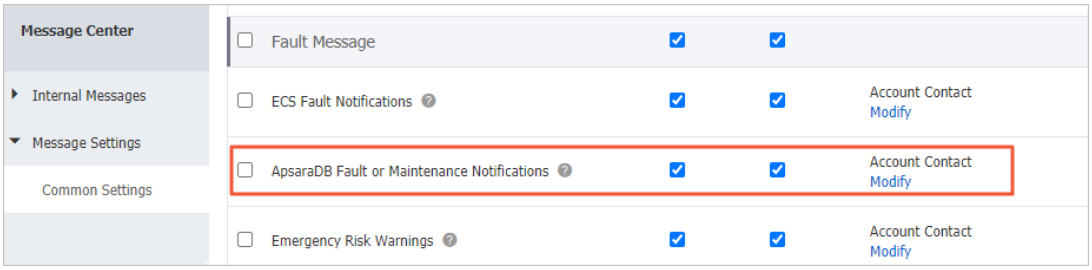
A pending O&M event exists.

> **Note** To check whether pending O&M events exist, you can move the pointer over the icon in the upper-right corner of the ApsaraDB RDS console.

Precautions

You are notified of ApsaraDB for Redis pending events such as instance migrations or version upgrades at least three days before the events occur. Notifications for high-risk vulnerability fixes are sent three or fewer days before execution due to the urgency of these events. Event notifications are sent by usingphone calls, emails, internal messages, or the ApsaraDB for Redis console. To use this feature, log on to the Message Center console, enable **ApsaraDB Fault or Maintenance Notifications**, and then specify a contact. We recommend that you specify an O&M engineer as the contact.

Message Center settings



Procedure

1. Log on to the ApsaraDB RDS console.
2. In the left-side navigation pane, click **Event Center**. In the top navigation bar, select a region.

   > **Note** If a pending O&M event requires you to schedule a time for the O&M operation, a message appears, which prompts you to schedule the time at your earliest opportunity.

3. (Optional)On the **Scheduled Events** tab, configure the periodic switching time.

   > **Note** The **Set Global Schedule** panel provides the global configuration items of proactive O&M events except events to fix high-risk vulnerabilities. After you configure the periodic switching time, the scheduled time of switchovers for newly generated proactive O&M events is automatically changed based on the configured time interval. If you do not configure the periodic switching time, the scheduled time of switchovers for newly generated proactive O&M events is automatically changed based on the maintenance window of your RDS instance. For more information, see Set the maintenance window of an ApsaraDB RDS for MySQL instance.

   i. Click **Set Global Schedule**.
   ii. In the panel that appears, configure the required parameters and click **Save**.
4. On the **Scheduled Events** tab, view the details of the event. To change the switching time of the event, select the RDS instance that you want to manage and click **Schedule Event**.

   > **Note** The displayed information varies based on the event type.

5. In the **Schedule Event** dialog box, configure the scheduled switching time and click **OK**.

> **Note**
>
> - If you select **Earliest Execution Time**, the system automatically provides the earliest switching date and time for the next O&M operation. After you save the settings, the instance prepares to switch over and enters the Pending state. If you do not select this option, you can customize the date and time.
> - The time that is specified by the **Scheduled Disconnection Time** parameter cannot be later than the time that is specified by the **Set Before** parameter.

Causes and impacts of events

Cause	Impact	Description
Instance migration Switchover between primary and secondary instances SSL certificate update Backup mode change	Transient connections	After a switchover is performed at the the following impacts occur: • Your RDS instance or the data shard in your RDS instance experiences transient connections and stays in the read-only state for up to 30 seconds before all data is synchronized. We recommend that you perform the switchover during off-peak hours and make sure that your application is configured to automatically reconnect to your RDS instance. • You cannot manage your RDS instance by using Data Management (DMS) or Data Transmission Service (DTS). This impact is temporary. scheduled switching time
Minor engine version update	Transient connections	After a switchover is performed at the the following impacts occur: • Your RDS instance or the data shard in your RDS instance experiences transient connections and stays in the read-only state for up to 30 seconds before all data is synchronized. We recommend that you perform the switchover during off-peak hours and make sure that your application is configured to automatically reconnect to your RDS instance. • You cannot manage your RDS instance by using Data Management (DMS) or Data Transmission Service (DTS). This impact is temporary.
	Differences between minor engine versions	Different minor engine versions have different updates. You must take note of the differences between the current minor engine version and the version to which you want to update. For more information, see the relevant release notes. Release notes are available only for the following services: • ApsaraDB RDS: Release notes of minor AliSQL versions, Release notes for AliPG, and Release notes. • PolarDB: Release notes of the PolarDB kernel, Release notes of minor PolarDB for Oracle versions, and Release notes of minor PolarDB for PostgreSQL versions.

Cause	Impact	Description
Minor version update for proxies	Transient connections	After a switchover is performed at the the following impacts occur: • Your RDS instance or the data shard in your RDS instance experiences transient connections and stays in the read-only state for up to 30 seconds before all data is synchronized. We recommend that you perform the switchover during off-peak hours and make sure that your application is configured to automatically reconnect to your RDS instance. • You cannot manage your RDS instance by using Data Management (DMS) or Data Transmission Service (DTS). This impact is temporary.
	Differences between minor engine versions	Different minor versions have different updates. You must take note of the differences between the current minor version and the minor version to which you want to update
Network upgrade	Transient connections	After a switchover is performed at the the following impacts occur: • Your RDS instance or the data shard in your RDS instance experiences transient connections and stays in the read-only state for up to 30 seconds before all data is synchronized. We recommend that you perform the switchover during off-peak hours and make sure that your application is configured to automatically reconnect to your RDS instance. • You cannot manage your RDS instance by using Data Management (DMS) or Data Transmission Service (DTS). This impact is temporary.
	Change of virtual IP addresses (VIPs)	Some network upgrades may involve cross-zone migrations that change the VIP of your RDS instance. If a client uses a VIP to connect to a cloud database, the connection is interrupted. **Note** To prevent transient connections, you must use the endpoint in the form of a domain name that is provided by your RDS instance and disable the DNS cache feature of the application and the server.

23.3. Subscribe to event notifications

You can configure alert rules for system events of ApsaraDB RDS in the CloudMonitor console. After you configure alert rules, CloudMonitor generates alerts when the specified thresholds are reached or events are detected. This helps you make business decisions.

Context

CloudMonitor is a service that monitors Internet applications and Alibaba Cloud resources. You can configure CloudMonitor to notify you of system exceptions. Then, you can automate the event handling process based on alert notifications. CloudMonitor supports the following notification methods:

- Send alert notifications by using emails or DingTalk chatbots.
- Push events to Message Service (MNS), Function Compute, Log Service, or a specified callback URL. This allows you to automate the event handling process.

Step 1: Create an alert rule

1. Create an alert contact or alert contact group.

   CloudMonitor sends notifications to alert contacts in contact groups. You must create an alert contact and a contact group and add the alert contact to the contact group.
2. Log on to the CloudMonitor console.
3. In the left-side navigation pane, choose **Alerts > Alert Rules**.
4. On the page that appears, click the **Event Alert** tab. On the Event Alert tab, click **Create Event Alert**.
5. In the panel that appears, configure the parameters to create an alert rule.

Parameter	Description
Alert Rule Name	Enter the name of the alert rule. The name can be up to 30 characters in length and can contain letters, digits, and underscores (_).
Event Type	Select **System Event**.
Product Type	Select **ApsaraDB for RDS**. You can follow a similar procedure to create alert rules for other cloud services.
Event Type	Retain the default value **All types**.
Event Level	Select one or more event levels. Valid values: **CRITICAL**, **WARN**, and **INFO**.
Event Name	Select the name of the event. **Note** ◦ The valid values of this parameter vary based on the value of the Event Level parameter. For more information about the relationship between event types and event levels, see System events for ApsaraDB RDS. For more information about other cloud services, see System events overview. ◦ If you want to test the event notification feature in Step 2: Test the alert rule, do not select **All Events** for this parameter.
Resource Range	Select **All Resources** or **Application Groups**. If you select **Application Groups**, you must specify the groups. For more information, see Create an application group.

Parameter	Description
Alert Type	Select the following notification methods based on your business requirements: ○ **Alert Notification**: sends alert notifications to a specific contact group by using a specific notification method. This is the default value. You must specify a contact group and a notification method. ○ **MNS queue**: pushes the event alert to a specific queue in MNS. For more information, see What is MNS?. ○ **Function service**: pushes the event alert to a specific function in Function Compute. For more information, see Overview. ○ **URL callback**: pushes the event alert to a specific callback URL. CloudMonitor delivers event alerts to the specific callback URL by using the POST or GET method. For more information, see EventBridge. ○ **Log Service**: pushes the event alert to a specific Logstore in Log Service. For more information, see What is Log Service?.

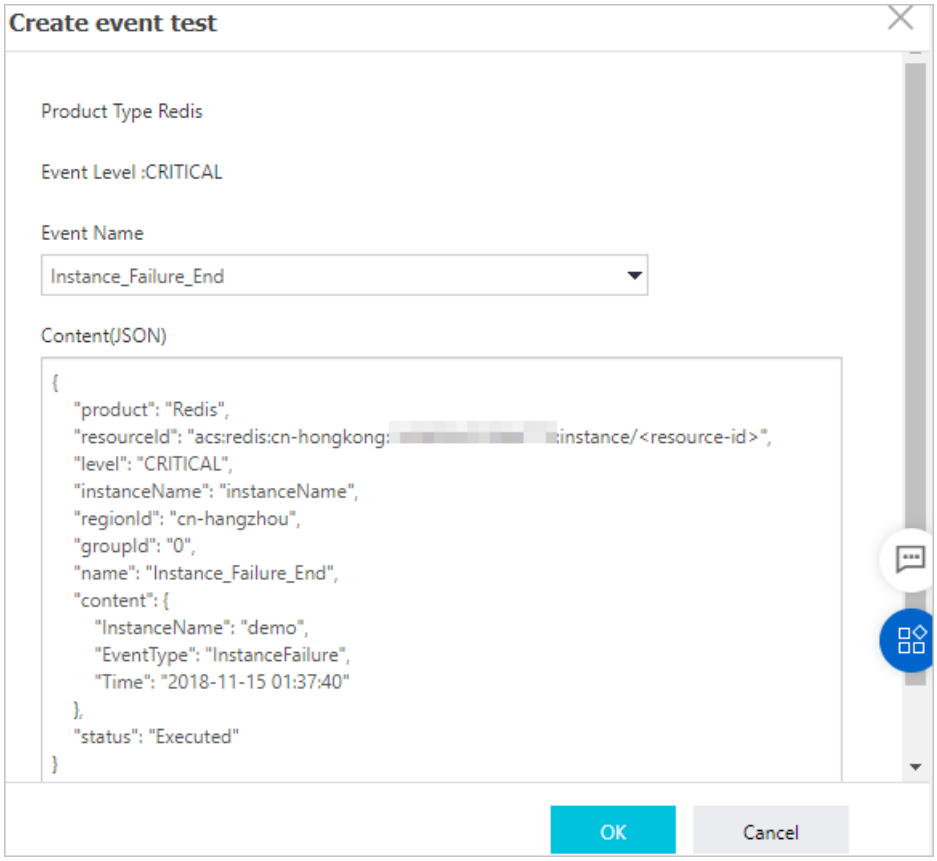
6. Click **OK**.

Step 2: Test the alert rule

After an alert rule is created, you can test the alert rule. You can check whether alert notifications can be received or whether event alerts can be pushed to MNS, Function Compute, Log Service, or the specified callback URL.

1. Log on to the CloudMonitor console.
2. In the left-side navigation pane, choose **Alerts > Alert Rules**.
3. Click the **Event Alert** tab.
4. Find the alert rule that you want to test and click **Test** in the **Actions** column.
5. In the **Create event test** panel, select the event that you want to test and modify the content.

   Create an event test



6. Click **OK**.
   CloudMonitor sends an event that contains specific content and an alert is sent by using the specified notification methods. For example, the alert may be sent through a notification and by using MNS.

Operations

CloudMonitor API operation	Description
PutEventRule	Creates or modifies an alert rule.

24.Audit

24.1. Use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance

This topic describes how to use the SQL Explorer feature on an ApsaraDB RDS for MySQL instance. The SQL Audit feature is upgraded to the SQL Explorer feature to provide more value-added capabilities such as security audit and performance diagnosis at lower costs. The upgrade does not interrupt the workloads on your RDS instance.

Prerequisites

- Your RDS instance does not run RDS Basic Edition.
- The RAM user whose credentials you use is granted the read and write permissions on ApsaraDB RDS. For example, the AliyunRDSFullAccess policy is attached to the RAM user. This prerequisite must be met if you log on to your RDS instance by using the credentials of a RAM user. For more information about how to grant permissions to a RAM user, see Use RAM to manage ApsaraDB RDS permissions.

Context

After you enable the SQL Explorer feature for your RDS instance, the SQL Explorer feature records the information about all data query language (DQL), DML, and DDL operations that are performed on your RDS instance. ApsaraDB RDS obtains the information by using network protocol analysis techniques. This way, only a small number of vCPU resources are consumed. SQL Explorer Trial Edition allows you to store SQL audit log files for up to one day free of charge. If you want to store SQL audit log files for more than one day, you are charged additional fees.

Notification

ApsaraDB RDS for MySQL supports the SQL Explorer and Audit feature. This feature provides capabilities such as **Source Analysis**, **SQL Review**, and **Related SQL Identification**. You can open the Upgrade from SQL Explorer to SQL Explorer and Audit dialog box to view the differences in functionality and billing between the **SQL Explorer** feature and the SQL Explorer and Audit feature. For more information about the SQL Explorer and Audit feature, see Use the SQL Explorer and Audit feature on an ApsaraDB RDS for MySQL instance.

Billing

- SQL Explorer Trial Edition: Since August 20, 2020, new billing rules are applied to SQL Explorer Trial Edition in all Alibaba Cloud regions.

  SQL Explorer Trial Edition provides a 15-day free trial of SQL Explorer. After the 15-day free trial expires, the SQL Explorer feature becomes unavailable. If you want to continue using the SQL Explorer feature, we recommend that you purchase SQL Explorer Paid Edition before the free trial expires.

  > **Note** You can enable SQL Explorer Trial Edition only once for each RDS instance.

- SQL Explorer Paid Edition: You are charged on an hourly basis. The pricing varies in Alibaba Cloud regions.
  - USD 0.0015 per GB per hour: China (Hong Kong), US (Silicon Valley), and US (Virginia).

  - USD 0.0018 per GB per hour: Singapore (Singapore), Japan (Tokyo), Germany (Frankfurt), UAE (Dubai), Australia (Sydney), Malaysia (Kuala Lumpur), India (Mumbai), Indonesia (Jakarta), and UK (London).
  - USD 0.0012 per GB per hour: all regions except the preceding regions.

- SQL Explorer and Audit: After you enable the SQL Explorer and Audit feature, the fees that you must pay for the original SQL Explorer feature are billed to DAS Professional Edition. The fees are no longer billed to your RDS instance. For more information, see Use the SQL Explorer and Audit feature on an ApsaraDB RDS for MySQL instance and Pricing of DAS Professional Edition.

Scenarios

- Your RDS instance is used for sectors, such as finance, security, stocks, public service, and insurance sectors, that require high data security.
- You want to analyze the status of your RDS instance to troubleshoot issues and check the performance of SQL statements in extreme circumstances.
- You want to restore the data of your RDS instance by using the logged information of executed SQL statements in extreme circumstances.

Differences between SQL audit logs and binary logs

Both SQL audit logs and binary logs contain the incremental data of your RDS instance. The two types of logs differ in the following aspects:

- SQL audit logs are similar to audit logs in MySQL and include information about all executed DQL, DML, and DDL operations. ApsaraDB RDS obtains the information by using network protocol analysis techniques. The SQL Explorer feature does not parse actual parameter values. If a large number of SQL statements are executed to query data, a small number of records may be lost. As a result, the incremental data that is obtained from SQL audit logs may be inaccurate.
- Binary logs record all add, delete, and modify operations that are performed and the incremental data that can be used to restore data. After a binary log file is generated, it is temporarily stored on your RDS instance. ApsaraDB RDS periodically transfers the binary log files whose sizes reach the specified threshold to an Object Storage Service (OSS) bucket. Binary log files can be stored for seven days in the OSS bucket. A binary log file to which data is being written cannot be transferred to an OSS bucket. After a periodic transfer is complete, you may find binary log files that cannot be transferred to the OSS bucket. Binary logs are not generated in real time. However, you can still use binary log files to obtain accurate incremental data.

Precautions

- The time range for an online query extends up to 24 hours. This is because the SQL Explorer feature logs a large number of SQL statements. You can use the logs to trace all operations that are performed on your RDS instance. If the time range for an online query exceeds 24 hours, the query requires a long period of time and may time out.

  > **Note** If you want to query SQL audit logs over a time range that exceeds 24 hours, we recommend that you export SQL audit logs as a file in asynchronous mode and download the file to your computer.

- You can specify a combination of conditions for an online query. For example, you can enter *test1 test2* in the Keywords field to query the SQL audit logs that contain the keyword test1 or test2.
- Fuzzy match is not supported for online queries.

- Each keyword for an online query must contain at least four characters.
- The size per SQL statement is limited to 2,000 bytes. If the size of an SQL statement exceeds 2,000 bytes, the excessive bytes cannot be logged.
- If you enable the SQL Explorer Trial Edition for your RDS instance, you cannot call the DescribeSQLLogRecords operation to query the logs that are generated by the SQL Explorer feature for your RDS instance.

Functionality

- SQL audit logging

  The SQL Explorer feature logs all operations that are performed on your RDS instance. You can use SQL audit logs to analyze faults and behavior and audit security.

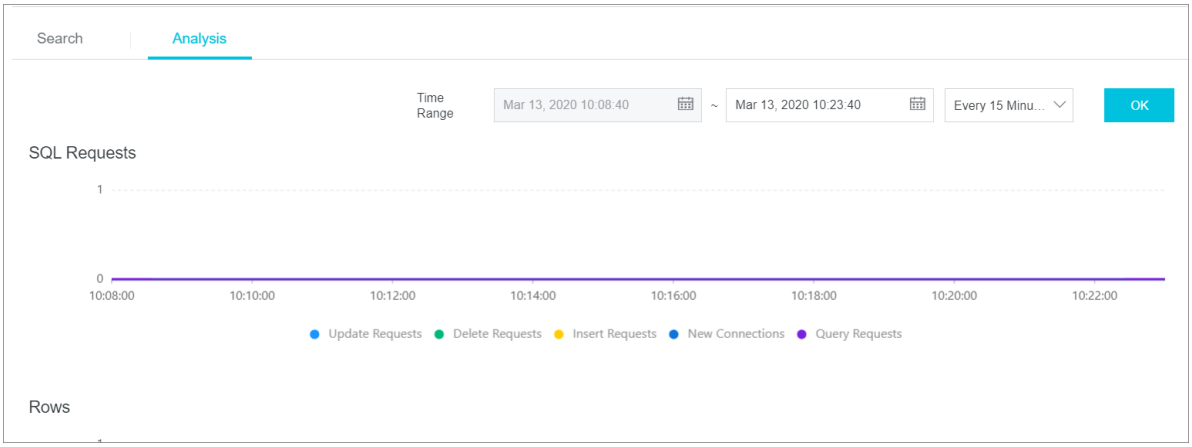
- Advanced search

  The SQL Explorer feature allows you to query data in various dimensions, such as database, user, client IP address, thread ID, execution duration, and number of scanned rows. You can export and download query results.

  > **Note**
  > - If you query data in a single dimension, you can specify more than one search condition. ApsaraDB RDS applies the OR operator to the specified search conditions. For example, if you specify two search conditions, user1 and user2, in the **Users** field, ApsaraDB RDS returns all SQL statements that are executed by user1 and those that are executed by user2.
  > - If you query data in more than one dimension, ApsaraDB RDS applies the AND operator to the specified dimensions. For example, if you enter user1 in the **Users** field and select the SELECT statement for the **Operation Type** parameter, ApsaraDB RDS returns all SELECT statements that are executed by user1.
  > - Fuzzy match is not supported.

- SQL analysis

  The SQL Explorer feature allows you to view and analyze the SQL audit logs that are generated over a specified time range. You can identify abnormal SQL statements and troubleshoot performance issues by using the analysis results.



- Cost reduction

The SQL Explorer feature uses columnar storage and compression technologies to reduce the storage usage for SQL audit logs. This reduces the overall storage costs by approximately 60%.

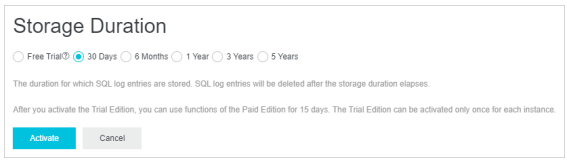
Enable the SQL Explorer feature

1.
2. In the left-side navigation pane, click **SQL Explorer**.
3. On the **SQL Explorer Upgraded to SQL Explorer and Audit** page, click **Close**.

> **Note**
>
> - On this page, you can select **Do not show again**. This way, this page is not displayed in the future.
> - To use the SQL Explorer and Audit feature, click **Upgrade** to go to the page on which you can use the SQL Explorer and Audit feature. For more information, see Use the SQL Explorer and Audit feature on an ApsaraDB RDS for MySQL instance.

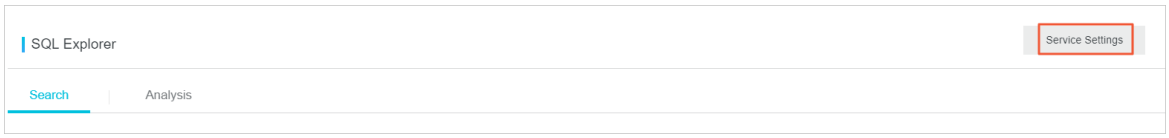
4. Click **Activate Now**.
5. Select a retention period and click **Activate**.

> **Note** ApsaraDB RDS deletes all SQL audit log files that are stored for longer than the specified retention period.

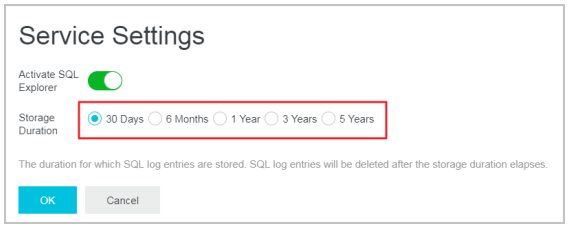


Modify the retention period of SQL audit logs

1.
2. In the left-side navigation pane, click **SQL Explorer**.
3. Click **Service Settings**.



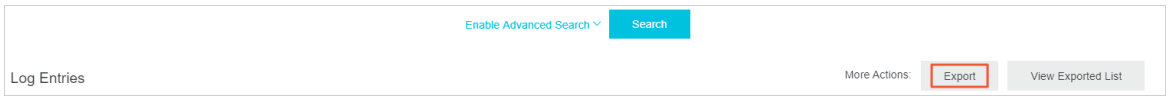
4. Select a retention period and click **OK**.



Disable the SQL Explorer feature

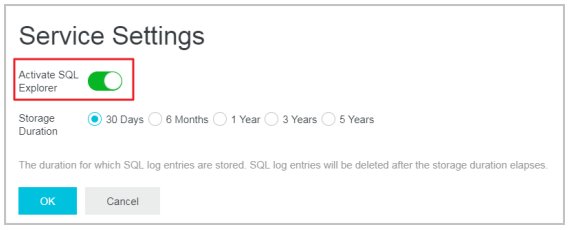
> **Note** After you disable the SQL Explorer feature, all SQL audit logs are deleted. Before you disable the SQL Explorer feature, we recommend that you export the SQL audit logs as a file and download the file to your computer.

1.
2. In the left-side navigation pane, click **SQL Explorer**.
3. Click **Export**.



4. In the message that appears, click **OK**.
5. After the export is complete, click **View Exported List** and download the SQL audit log file that you exported to your computer.
6. Click **Service Settings**.

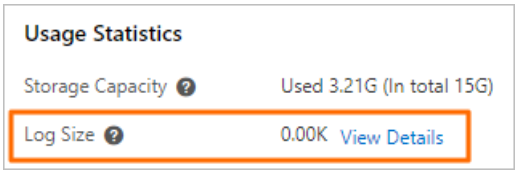
7. Turn off the switch next to Activate SQL Explorer and click **OK**.



FAQ

- How do I obtain the size of the logs that are generated by the SQL Explorer feature?

  Log on to the ApsaraDB RDS console, find your RDS instance, and then go to the **Basic Information** page. In the **Usage Statistics** section of the page, you can view the size of the logs that are generated by the SQL Explorer feature.



- Can I delete a specified part of the generated SQL audit logs?

  No, you cannot delete a specified part of the generated SQL audit logs. To reduce costs, you can disable the SQL Explorer feature.

24.2. View the logs of an ApsaraDB RDS for MySQL instance

This topic describes how to view the logs of an ApsaraDB RDS for MySQL instance in the ApsaraDB RDS console. The logs include error logs, slow query logs, and primary/secondary switchover logs. You can use the logs to troubleshoot issues on the RDS instance.

> **Note** For more information about binary logs, see Enable the automatic backup feature for an ApsaraDB RDS for MySQL instance and Download the backup files of an ApsaraDB RDS for MySQL instance.

For more information about how to view the logs of an RDS instance that runs a different database engine, see the following topics:

- View the logs of an ApsaraDB RDS for SQL Server instance
- View the logs of an ApsaraDB RDS for PostgreSQL instance
- View the logs of an ApsaraDB RDS for MariaDB TX instance

Procedure

1.
2. In the left-side navigation pane, click **Logs**.
3. On the **Logs** page, click the Error Logs, Slow Log Details, Slow Log Summary, or Primary/Secondary Switching Logs tab, select a time range, and then click **OK**. You can also subscribe to binary logs on the Binlog Subscription tab of the Logs page.

Tab	Description
Binlog Subscription	Allows you to subscribe to binary logs by using Data Transmission Service (DTS). For more information, see Track data changes from an ApsaraDB RDS for MySQL instance.
Error Logs	Provides statistics about the database running errors that occurred over the last 30 days.
Slow Log Details	Provides details about the SQL statements that each took more than 1 second to run over the last 7 days. Duplicate SQL statements are removed. You can change the 1-second threshold by reconfiguring the long_query_time parameter. For more information, see Reconfigure the parameters of an ApsaraDB RDS for MySQL instance. **Note** This tab is refreshed once every minute.
Slow Log Summary	Provides a summary of the SQL statements that each took more than 1 second to run over the last 7 days and allows you to export the summary as a report file. You can change the 1-second threshold by reconfiguring the long_query_time parameter. For more information, see Reconfigure the parameters of an ApsaraDB RDS for MySQL instance. **Note** Slow log statistics are not collected in real time and may have a delay of 6 hours to 8 hours.

Tab	Description
Primary/Secondary Switching Logs	Provides statistics about the primary/secondary switchovers that occurred over the last 30 days. Primary/secondary switchover logs are supported only for RDS instances that do not run the RDS Basic Edition.

Note If an RDS instance resides in the China (Zhangjiakou) region, ApsaraDB RDS retains only the error logs that are generated over the last 9 days, slow query log details, and slow query log summary that are generated over the last 7 days of the RDS instance.

24.3. View the slow log details of an ApsaraDB RDS for MySQL instance

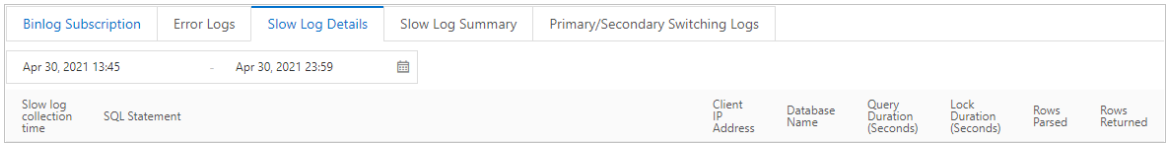
This topic describes how to view the slow log details of an ApsaraDB RDS for MySQL instance.

Precautions

If more than 600 slow query log entries are generated within 1 minute on your RDS instance, some entries may be lost. To mitigate the loss of slow query log entries, you can Update the minor engine version of an ApsaraDB RDS for MySQL instance to 20191212.

View slow log details by using the ApsaraDB RDS console

1.
2. In the left-side navigation pane, click **Logs**.
3. Click the **Slow Log Details** tab and on the tab view the slow log details of your RDS instance.



Note The Slow Log Details tab provides information such as **Slow Log Collection Time**, **SQL Statement**, **Client IP Address**, **Database Name**, and **Query Duration (Seconds)**. The time in the **Slow Log Collection Time** column is the time when ApsaraDB RDS detects the slow SQL statement. It is not the time when the slow SQL statement is executed.

View slow log details by using commands

Note This operation is not supported if your RDS instance runs MySQL 5.5.

1. Log on to your RDS instance by using Alibaba Cloud Data Management (DMS). For more information, see Use DMS to log on to an ApsaraDB RDS for MySQL instance.
2. In the top navigation bar, choose **SQL Operations** > **SQL Window**.
3. Run the following command to obtain the slow log details of your RDS instance:

```
select * from mysql.slow_log
```

24.4. Delete the binary log files of an ApsaraDB RDS for MySQL instance

This topic describes how to delete the binary log files of an ApsaraDB RDS for MySQL instance. You can configure rules based on which ApsaraDB RDS automatically deletes binary log files. You can also manually delete binary log files.

After binary log files are deleted, the storage that is occupied by the binary log files is released.

> **Note** After binary log files are deleted from an RDS instance, the size of the log backup files of the RDS instance does not decrease and you can still restore the data of the RDS instance to a specific point in time. Take note that you can restore the data of an RDS instance to a specific point in time only after you enable the log backup feature for the RDS instance.

Procedure

To configure rules based on which ApsaraDB RDS automatically deletes the binary log files of an RDS instance, perform the following steps:

1.
2. In the left-side navigation pane, click **Backup and Restoration**.
3. On the **Backup Settings** tab, click Edit in the **Local Log Backup Settings** section to configure the rules.
   - Rule 1: ApsaraDB RDS deletes the binary log files whose retention period exceeds the maximum retention period that is specified by the **Retention Period** parameter. By default, the maximum retention period is 18 hours.
   - Rule 2: If one of the following conditions is met, ApsaraDB RDS deletes earlier binary log files until the condition is not met.
     - The storage usage that is calculated by using the following formula exceeds the threshold that is specified by the **Max Storage Usage** parameter: **Storage usage = (Size of binary log files/Size of storage capacity) × 100%**. The default threshold is 30%.
     - The number of binary log files exceeds the maximum number that is specified by the **Retained Files** parameter. By default, the maximum number is 60.
     - The storage usage exceeds 80% or the amount of available storage is less than 5 GB. Make sure that you set the **Protect Available Storage** parameter to Enable.

To manually delete the binary log files of an RDS instance, perform the following steps:

1.
2. In the left-side navigation pane, click **Backup and Restoration**.
3. Click **Upload Binlogs**. Then, click **OK**.

   If you perform this operation, all binary log files except the most recent two binary log files are deleted. Examples:

   - If the mysql-bin.000192, mysql-bin.000193, and mysql-bin.000194 binary log files are stored in the RDS instance, only the mysql-bin.000192 binary log file is deleted upon this operation.
   - If only two binary log files are stored in the RDS instance, no binary log files are deleted upon this

operation.

> **Note**
>
> - ApsaraDB RDS deletes binary log files at a latency.
> - This operation is not supported for RDS instances that run RDS Basic Edition.
> - If binary log files cannot be deleted and the following warning is displayed on the **Error Logs** tab of the **Logs** page, the database client may encounter errors or a change tracking task is in progress. Before you can delete the binary log files, you must stop the task.
>
>   ```
>   [Warning] file /home/mysql/data3001/mysql/mysql-bin.069435 was not purged bec
>   ause it was being readby thread number 17126285
>   ```
>
> - If you use the **Upload Binlogs** feature, the RDS instance uploads binary log files to the specified Object Storage Service (OSS) bucket. The retention period of binary log files that are stored as objects in the OSS bucket is the same as the log backup retention period that you specify on the RDS instance.

4. Log on to the RDS instance and run the `show binary logs;` command to view the binary log files of the RDS instance. For more information, see Use DMS to log on to an ApsaraDB RDS for MySQL instance.
5. View the size of binary log files on the **Monitoring and Alerts** page.

Related operations

Operation	Description
PurgeDBInstanceLog	Deletes the binary log files of an ApsaraDB RDS instance.

FAQ

- Why is the size of binary log files on the primary RDS instance different from the size of binary log files on the secondary RDS instance in my database system?

  By default, the size of binary log files on the primary RDS instance is the same as the size of binary log files on the secondary RDS instance. The two sizes may be different due to the following reasons:

  - The primary RDS instance replicates data to the secondary RDS instance at a latency, the Binlog dump thread dumps the binary log files at a latency, or Data Transmission Service (DTS) is migrating data from or to the primary RDS instance.
  - The secondary RDS instance is reading and applying log records.

- Why does ApsaraDB RDS delete a binary log file of my RDS instance immediately after the binary log file is generated?

The storage usage of the RDS instance may have reached 80%, or the amount of available storage on the RDS instance is less than 5 GB. At this time, if you set the **Protect Available Storage** parameter in the **Local Log File Settings** dialog box to Enable, all binary log files are deleted immediately after they are generated. We recommend that you manually increase the storage capacity of the RDS instance or enable the automatic storage expansion feature for the RDS instance. This way, the RDS instance is not locked even if the storage usage reaches 80% or the amount of available storage is less than 5 GB. For more information, see Change the specifications of an ApsaraDB RDS for MySQL instance or Configure automatic storage expansion for an ApsaraDB RDS for MySQL instance.

25.Tag

25.1. Add tags to ApsaraDB RDS instances

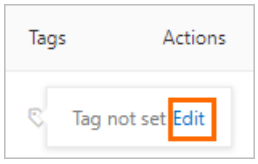
This topic describes how to add tags to one or more ApsaraDB RDS instances. You can use tags to classify a large number of RDS instances. Each tag consists of a key and a value. You can use tag keys and values to further classify RDS instances.

Limits

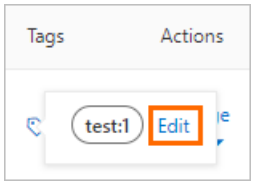
- You can add up to 20 tags to each RDS instance. Each tag must have a unique key. If two tags have the same key, the tag that is created later overwrites the earlier tag.
- You can add tags to up 50 RDS instances at a time.
- RDS instances in different regions do not share the same tag namespace.
- After you remove a tag from an RDS instance, ApsaraDB RDS checks whether the tag is added to other RDS instances. If the tag is not added to other RDS instances, ApsaraDB RDS deletes the tag.

Add tags to an RDS instance

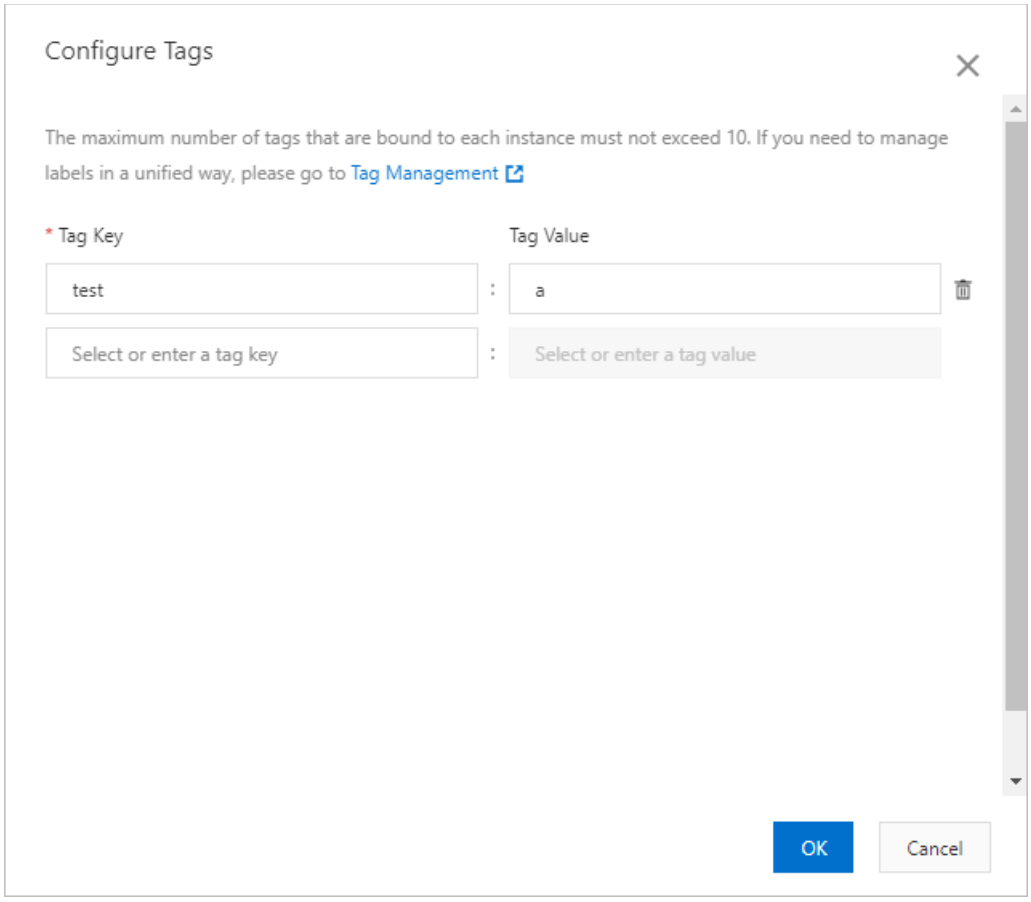
1.
2. Click the icon in the **Tags** column of the required RDS instance and then click **Edit**.



If you have added a tag to the RDS instance, you can click **Edit** to edit the tag.



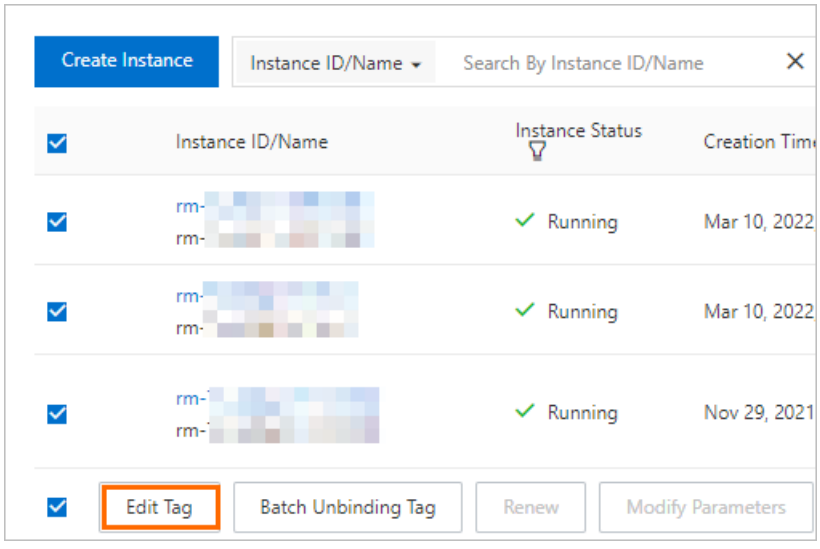
3. In the **Configure Tags** dialog box, configure the **Tag Key** and **Tag Value** parameters and click **OK**.



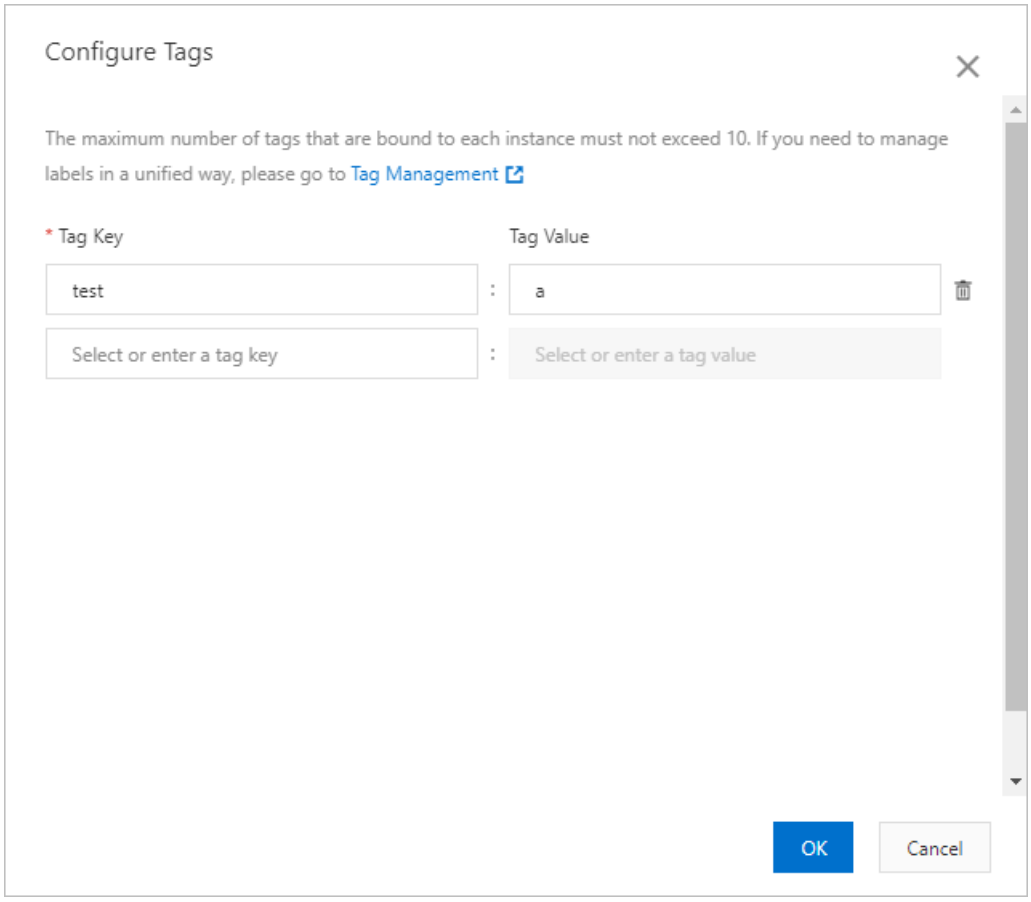
Add tags to multiple RDS instances at a time

1.
2. Select the RDS instances to which you want to add tags and click **Edit Tag** below the instance list.

> **Note** The Edit Tag button is displayed in the lower part of the page.



3. In the **Configure Tags** dialog box, configure the **Tag Key** and **Tag Value** parameters and click **OK**.



Related operations

Operation	Description
Create and bind tags	Adds tags to one or more ApsaraDB RDS instances.

25.2. Remove tags from an ApsaraDB RDS for MySQL instance

This topic describes how to remove tags from an ApsaraDB RDS for MySQL instance. If you change the configuration of your RDS instance or you no longer require specific tags, you can remove these tags from your RDS instance.

Limits

- You can remove a maximum of 20 tags at a time.
- After you remove a tag from your RDS instance, ApsaraDB RDS checks whether the tag is added to other RDS instances. If the tag is not added to other RDS instances, ApsaraDB RDS deletes the tag.

Procedure

1.
2. Use one of the following methods to remove tags:

- Remove a tag from an RDS instance
  a. Move the pointer over the icon on the right of the instance. In the dialog box that appears, click **Edit**.
  b. Click the icon on the right of the tag that you want to remove.
  c. Click **OK**.
- Remove tags from multiple RDS instances at a time
  a. Select the RDS instances from which you want to remove tags.
  b. Click **Batch Unbinding Tag** below the instance list.
  c. In the dialog box that appears, select the tags that you want to remove.
  d. Click **Unbind X tags**. You can query operation details in the **Configure Tags successfully** message.

Related operations

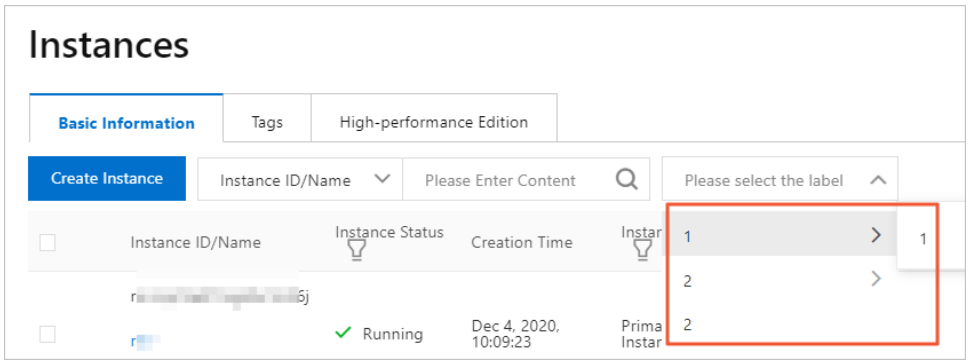
Operation	Description
Unbind tags	Removes tags from ApsaraDB RDS instances.

25.3. Use tags to filter ApsaraDB RDS for MySQL instances

This topic describes how to filter ApsaraDB RDS for MySQL instances based on the tags that are added to these instances.

1.
2. Select a **key** and a **value**. Then, ApsaraDB RDS filters your RDS instances based on the specified tag.

> **Note** To cancel the filter condition that is specified by the tag, you can click the X icon to the right of the **tag**.



Related operations

Operation	Description
Query the tags of ApsaraDB RDS instances	Queries the tags that are added to one or more RDS instances.

26.Appendixes

26.1. Reserved keywords of an ApsaraDB RDS for MySQL instance

This topic describes the keywords that are reserved in an ApsaraDB RDS for MySQL instance. Do not use these reserved keywords when you create user-defined functions (UDFs).

ApsaraDB RDS provides a few more reserved keywords in addition to the reserved keywords that are provided by open source MySQL. The following table lists the new reserved keywords that are provided by ApsaraDB RDS. For more information about the reserved keywords that are provided by open source MySQL, see the following topics:

- MySQL 8.0 Reference Manual
- MySQL 5.7 Reference Manual
- MySQL 5.6 Reference Manual

MySQL version	Reserved keyword	Description
8.0	NEXTVAL	An operator that is used to access the value of a sequence in SQL statements. If a UDF has the same name as the NEXTVAL operator, the priorities of the UDF and the NEXTVAL operator vary based on the minor engine version of your RDS instance. • If the minor engine version is 20201031 or earlier, the priority of the NEXTVAL operator is higher than the priority of the UDF. • If the minor engine version is later than 20201031, the priority of the UDF is higher than the priority of NEXTVAL operator.
	CURRVAL	An operator that is used to access the value of a sequence in SQL statements. If a UDF has the same name as the CURRVAL operator, the priorities of the UDF and the CURRVAL operator vary based on the minor engine version of your RDS instance. • If the minor engine version is 20201031 or earlier, the priority of the CURRVAL operator is higher than the priority of the UDF. • If the minor engine version is later than 20201031, the priority of the UDF is higher than the priority of CURRVAL operator.
	NEXTVAL	An operator that is used to access the value of a sequence in SQL statements. If a UDF has the same name as the NEXTVAL operator, the priorities of the UDF and the NEXTVAL operator vary based on the minor engine version of your RDS instance. • If the minor engine version is 20201231 or earlier, the priority of the NEXTVAL operator is higher than the priority of the UDF. • If the minor engine version is later than 20201231, the priority of the UDF is higher than the priority of NEXTVAL operator.

5.7 MySQL version	Reserved keyword	Description
	CURRVAL	An operator that is used to access the value of a sequence in SQL statements. If a UDF has the same name as the CURRVAL operator, the priorities of the UDF and the CURRVAL operator vary based on the minor engine version of your RDS instance. • If the minor engine version is 20201231 or earlier, the priority of the CURRVAL operator is higher than the priority of the UDF. • If the minor engine version is later than 20201231, the priority of the UDF is higher than the priority of CURRVAL operator.
	RDS_AUDIT	None.
5.6	ASYNC_COMMIT	None.
	CACHED	None.
	CLUSTERING	None.
	FORCE_UPDATE_PLAN_CACHE	None.
	NO_PLAN_CACHE	None.

26.2. Commonly used SQL statements for MySQL

This topic lists some of the commonly used SQL statements.

For more information about the SQL statements including parameters and restrictions, see MySQL 5.7 Reference Manual.

Database-related SQL statements

Operation	Example
Create a database and designate a character set.	`create database db01 DEFAULT CHARACTER SET gbk COLLATE gbk_chinese_ci;`
Delete a database.	`drop database db01;`

Account-related SQL statementss

Note If an RDS instance has a premier account, the passwords of the other accounts under this instance cannot be changed by using the premier account. To change the password of another account, you must delete this account and create a new one.

Operation	Example
Create an account.	`CREATE USER 'username'@'host' IDENTIFIED BY 'password';`
Delete an account.	`DROP USER 'username'@'host';`
Authorize the account.	`GRANT SELECT ON db01. * TO 'username'@'host';`

Query the created accounts in the database.	`SELECT user,host,password FROM mysql.user_view;`
	or
	`show grants for xxx`

Reclaim permissions.	• Reclaim all permissions:
	`REVOKE ALL PRIVILEGES,GRANT OPTION FROM 'username'@'host';`
	• Reclaim specific permissions:
	`REVOKE UPDATE ON *. * FROM 'username'@'host';`

26.3. Grant backup file download permissions to a RAM user with read-only permissions

This topic describes how to grant backup file download permissions to a RAM user who only has read permissions. For security purposes, a RAM user with read-only permissions cannot download backup files.

Procedure

1. Log on to the RAM console.
2. In the left-side navigation pane, choose **Permissions > Policies**.

3. Click **Create Policy** and specify the parameters:

   The policy contains the following content:

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "rds:Describe*",
                "rds:ModifyBackupPolicy",
                "rds:CheckRegionSupportBackupEncryption"
            ],
            "Resource": "*"
        }
    ],
    "Version": "1"
}
```

4. Click **OK**.
5. In the left-side navigation pane, choose **Permissions > Grants**.
6. Click **Add Authorization** to attach the new permission policy to the RAM user.
7. Click **OK**.

26.4. Authorize an ApsaraDB RDS for MySQL instance to access KMS

To use the disk encryption feature for an ApsaraDB RDS for MySQL instance, you must authorize the instance to access Key Management Service (KMS). This topic describes how to authorize your RDS instance to access KMS by using the RAM console.

Prerequisites

You are logged on to the RAM console by using your Alibaba Cloud account.

Create a permission policy named AliyunRDSInstanceEncryptionRolePolicy

1. Go to the Policies page.
2. Click **Create Policy**.

   > **Note** A permission policy is a set of permissions that are described by using a specific syntax. You can use permission policies to describe the authorized resource sets, authorized operation sets, and authorization conditions. For more information, see Terms.

3. Configure the following parameters.

Parameter	Description

Parameter	Description
Policy Name	The name of the permission policy. Enter **AliyunRDSInstanceEncryptionRolePolicy**.
Note	The information that is used to identify the permission policy. Example: Allows ApsaraDB RDS to access KMS.
Configuration Mode	The configuration mode of the permission policy. Select the **Script** configuration mode. Then, copy the following script and paste it to the edit box below Policy Document.

Copy and paste the following script:

```
{
    "Version": "1",
    "Statement": [
        {
            "Action": [
                "kms:List*",
                "kms:DescribeKey",
                "kms:TagResource",
                "kms:UntagResource"
            ],
            "Resource": [
                "acs:kms:*:*:*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": [
                "acs:kms:*:*:*"
            ],
            "Effect": "Allow",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "kms:tag/acs:rds:instance-encryption": "true"
                }
            }
        }
    ]
}
```

4. Click **OK**.

Create and authorize a RAM role named AliyunRDSInstanceEncryptionDefaultRole

After you create the AliyunRDSInstanceEncryptionRolePolicy permission policy, you must create a RAM role and attach the permission policy to the RAM role. Then, ApsaraDB RDS can access KMS.

1. Go to the RAM Roles page.
2. Click **Create RAM Role**.
3. In the Create RAM Role panel, select **Alibaba Cloud Service** and click **Next**.
4. Configure the following parameters and click **OK**.

Parameter	Description
Role Type	The type of the RAM role. Select **Normal Service Role**.
RAM Role Name	The name of the RAM role. Enter **AliyunRDSInstanceEncryptionDefaultRole**.
Note	The information that is used to identify the RAM role.
Select Trusted Service	The trusted service of the RAM role. Select **RDS**.

5. After the message "**The Role has been created**" appears, click **Add Permissions to RAM Role**.

> **Note** If you have closed the panel in which the message "**The Role has been created**" appears, you can open the RAM Roles page, find the **AliyunRDSInstanceEncryptionDefaultRole** role, and then click **Add Permissions** in the Actions column.

6. In the **Add Permissions** panel, click the **AliyunRDSInstanceEncryptionRolePolicy** permission policy to add the permission policy to the **Selected** list.
7. Click **OK**.

(Optional) View the ARN of a RAM user

Alibaba Cloud Resource Name (ARN) is the global resource descriptor of a RAM role. The ARN of a RAM role describes the resources that the RAM role can access. When you call an API operation to enable the disk encryption feature, you must specify the ARN of a RAM role that has the permissions to access KMS. For more information, see CreateDBInstance.

1. Go to the RAM Roles page.
2. Find the RAM role that you want to use. Then, click the name of the RAM role.
3. In the Basic Information section of the page that appears, view the ARN of the RAM role.

26.5. Cached data persistence

ApsaraDB RDS can be used together with ApsaraDB Memcache and Redis to form storage solutions with high throughput and low delay. This document describes the cached data persistence solution based on the combined use of RDS and Memcache.

Background information

Compared with RDS, Memcache and Redis have the following features:

- Quick response: The request delay of ApsaraDB Memcache and Redis is usually within several

milliseconds.

- The cache area supports a higher Queries Per Second (QPS) than RDS.

System requirements

- bmemcached (with support for SASL extension) has been installed in the local environment or ECS.

  bmemcached download address: Click Here to download.

  The bmemcached installation command is as follows:

  ```
  pip install python-binary-memcached
  ```

- Python is used as an example. Python and pip must be installed in the local environment or ECS.

Sample code

The following sample code realizes the combined use of ApsaraDB RDS and Memcache:

```
/usr/bin/env python
import bmemcached
Memcache_client = bmemcached.Client((‘ip:port’), ‘user’, ‘passwd’)
#Search for a value in ApsaraDB Memcache
res = os.client.get(‘test’)
if res is not None:
    return res #Return the value found
else:
    #Query RDS if the value is not found
    res = mysql_client.fetchone(sql)
     Memcache_client.put(‘test’, res) #Write cached data to ApsaraDB for Memcache
    return res
```