ALIBABA CLOUD

Alibaba Cloud

ApsaraDB for Redis

Data Migration And Synchronization

Document Version: 20220711

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.Migrate data

1.1. Overview

ApsaraDB for Redis provides multiple data migration solutions based on Data Transmission Service (DTS) and redis-shake. This allows you to migrate data in different scenarios based on your business requirements.

Data migration tools

- DTS (recommended)
  DTS is a real-time data streaming service that is provided by Alibaba Cloud. The service allows you to migrate, subscribe to, and synchronize data by using stable and secure transmission channels. DTS supports full data migration and incremental data migration with minimal downtime. This ensures service availability.
- Redis-shake
  redis-shake is an open source Linux-based tool that is developed by Alibaba Cloud. You can use this tool to parse (decode mode), restore (restore mode), back up (dump mode), and synchronize (sync or rump mode) data in ApsaraDB for Redis instances. For scenarios in which DTS is not suitable, you can use redis-shake to migrate data.

The following table describes the differences between DTS and redis-shake for Redis data migration.

> Note
>
> - You can migrate data between different database types and architectures described in the following table. For example, you can use DTS to migrate data from a self-managed Redis database in the master-replica architecture to an ApsaraDB for Redis Community Edition instance in the cluster architecture.
> - The data synchronization feature of DTS is suitable for more data synchronization scenarios. We recommend that you use the data synchronization feature of DTS to migrate data. For more information, see Overview.

Differences

Tool	Supported source database	Supported destination database	Supported architecture	Supported migration type

Tool	Supported source database	Supported destination database	Supported architecture	Supported migration type
DTS (recommended)	- Self-managed Redis database Redis 2.8, 3.0, 3.2, 4.0, and 5.0 - ApsaraDB for Redis Community Edition Redis 4.0 and 5.0 - ApsaraDB for Redis Enhanced Edition (Tair)	- ApsaraDB for Redis Community Edition Redis 4.0 and 5.0 - ApsaraDB for Redis Enhanced Edition (Tair)	- Source database - Standard master-replica instances - Cluster master-replica instances - Read/write splitting instances - Destination database - Standard master-replica instances - Cluster master-replica instances - Read/write splitting instances	- Full data migration - Incremental data migration - Cross-account data migration
Redis-shake	- Self-managed Redis database Redis 2.8, 3.0, 3.2, 4.0, and 5.0	- ApsaraDB for Redis Community Edition Redis 4.0 and 5.0 - ApsaraDB for Redis Enhanced Edition (Tair) Redis 5.0	The source and destination databases support the following architectures: - Standard master-replica instances - Cluster master-replica instances - Read/write splitting instances	- Full data migration - Incremental data migration

Required permissions

The permissions that are required by DTS and redis-shake vary based on different migration scenarios. Before you configure data migration, you must obtain the required permissions on the source and destination databases based on the migration scenarios. The following table describes the required permissions.

> ? **Note** For more information about how to create an account for an ApsaraDB for Redis instance and grant the required permissions to the account, see Create and manage database accounts.

Tool	Scenario	Permission on the source database	Permission on the destination database
DTS (recommended)	• Migrate data from a self-managed Redis database to an ApsaraDB for Redis instance • Migrate data between self-managed databases	Permissions to run the **SYNC** or **PSYNC** command	Read and write permissions
Redis-shake	• Migrate data from a self-managed Redis database to an ApsaraDB for Redis instance • Migrate data between self-managed databases	Permissions to run the **SYNC** or **PSYNC** command	Read and write permissions
	• Migrate data between ApsaraDB for Redis instances that belong to the same Alibaba Cloud account • Migrate data between ApsaraDB for Redis instances that belong to different Alibaba Cloud accounts	Replication permissions **Note** You cannot create an account that has **replication** permissions for the source instance in the cluster or read/write splitting architecture. If the destination instance is an ApsaraDB for Redis instance, you can submit a ticket to eliminate the limit.	Read and write permissions

Data migration solutions

> **Notice** If your migration scenario is not included in the following table, we recommend that you use the data synchronization feature of DTS to migrate data because this feature is more frequently used to migrate Redis data. For more information, see Overview.

Migration solutions

Scenario	Tool	Solution
		Migrate data from a self-managed Redis database to an ApsaraDB for Redis instance

Scenario	Tool	Solution
Migrate data from a self-managed Redis database to an ApsaraDB for Redis instance	DTS	Synchronize data from a Codis cluster hosted on ECS to an ApsaraDB for Redis instance **Note** This solution uses the data synchronization feature of DTS to migrate data.
		Synchronize data from a Twemproxy Redis cluster hosted on ECS to an ApsaraDB for Redis instance **Note** This solution uses the data synchronization feature of DTS to migrate data.
	Redis-shake	Use redis-shake to migrate data from a self-managed Redis database to Alibaba Cloud
		Use redis-shake to migrate data from on-premises Codis or Redis to ApsaraDB for Redis
		Use redis-shake to migrate the data of a self-managed Redis database from a backup file to an ApsaraDB for Redis instance
		Use AOF files to migrate data
Migrate data between ApsaraDB for Redis instances	DTS	Configure one-way data synchronization between ApsaraDB for Redis instances **Note** This solution uses the data synchronization feature of DTS to migrate data.
	Redis-shake	Migrate data between ApsaraDB for Redis instances by using redis-shake
Migrate data from a third-party database to an ApsaraDB for Redis instance	Redis-shake	Migrate data from Amazon ElastiCache for Redis to ApsaraDB for Redis
		Migrate data from SSDB to ApsaraDB for Redis
		Migrate data from Google Cloud Memorystore for Redis to ApsaraDB for Redis

Related information

- FAQ about data migration and data synchronization
- Overview
- Verify data after migration

1.2. Migrate data from on-premises Redis to ApsaraDB for Redis

1.2.1. Migrate data from a self-managed Redis database to an ApsaraDB for Redis instance

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

Prerequisites

- The version of the self-managed Redis database is 2.8, 3.0, 3.2, 4.0, 5.0, or 6.0.
- The self-managed Redis database is deployed in the standalone architecture rather than the cluster architecture.

  > **Note** If the self-managed Redis database is deployed in the cluster architecture, you can migrate data by using the data synchronization feature. For more information, see Synchronize data from a self-managed Redis cluster to an ApsaraDB for Redis cluster instance.

- The `PSYNC` or `SYNC` command can be run on the self-managed Redis database.
- The available storage of the destination ApsaraDB for Redis instance is larger than the total size of the data that is stored in the self-managed Redis database.

Precautions

- DTS consumes resources of the source and destination databases during full data migration. This may increase the loads on database servers. If you migrate a large volume of data or the server specifications cannot meet your requirements, database services may become unavailable. 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.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction` , data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. For this reason, the number of keys in the destination database may be less than that in the source database. You can run the INFO command to view the number of keys in the destination database.

  > **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination database. During incremental data migration, the destination database does not explicitly return the execution results of Lua scripts.
- When you run the `PSYNC` or `SYNC` command to transfer data of the LIST type, DTS does not

perform the `flush` operation on the existing data in the destination database. For this reason, the destination database may contain duplicate data records.

- During data migration, if the number of shards in the self-managed Redis database increases or decreases, or if the specifications of the database are changed (for example, the memory capacity is scaled up), you must reconfigure the task. We recommend that you clear the data that has been migrated to the destination ApsaraDB for Redis instance before you reconfigure the task to ensure data consistency.
- If the endpoint of the self-managed Redis database is changed during data migration, you must submit a ticket to update the change. Otherwise, the append-only files (AOF) of the self-managed Redis database may be reset. In this case, you must reconfigure the task.
- 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. If you do not stop or release the data migration task, the data in the source database overwrites 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

- Full data migration
  DTS migrates historical data of the required objects from the self-managed Redis database to the destination ApsaraDB for Redis instance.

> **Note** If you select only full data migration, do not write new data to the source Redis database during data migration to ensure data consistency.

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

Commands that can be used for synchronization during incremental data migration

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- FLUSHALL and FLUSHDB
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX

- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, PSETEX, and PUBLISH
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE

Preparations before incremental data migration

To ensure that the incremental data migration task runs as expected, we recommend that you remove the limit on the replication output buffer. This topic uses a Linux server as an example.

> **Note** If you perform only full data migration, skip the following steps.

1. Use redis-cli to connect to the self-managed Redis database.

   > **Note** You can use redis-cli after you install the Redis client. For more information, visit Redis community official website.

   ```
   redis-cli -h <host> -p <port> -a <password>
   ```

   > **Note**
   >
   > - <host>: the endpoint that is used to connect to the self-managed Redis database. You can use 127.0.0.1 in this example.
   > - <port>: the service port number of the self-managed Redis database. The default port number is 6379.
   > - <password>: the password of the self-managed Redis database.

   Example:

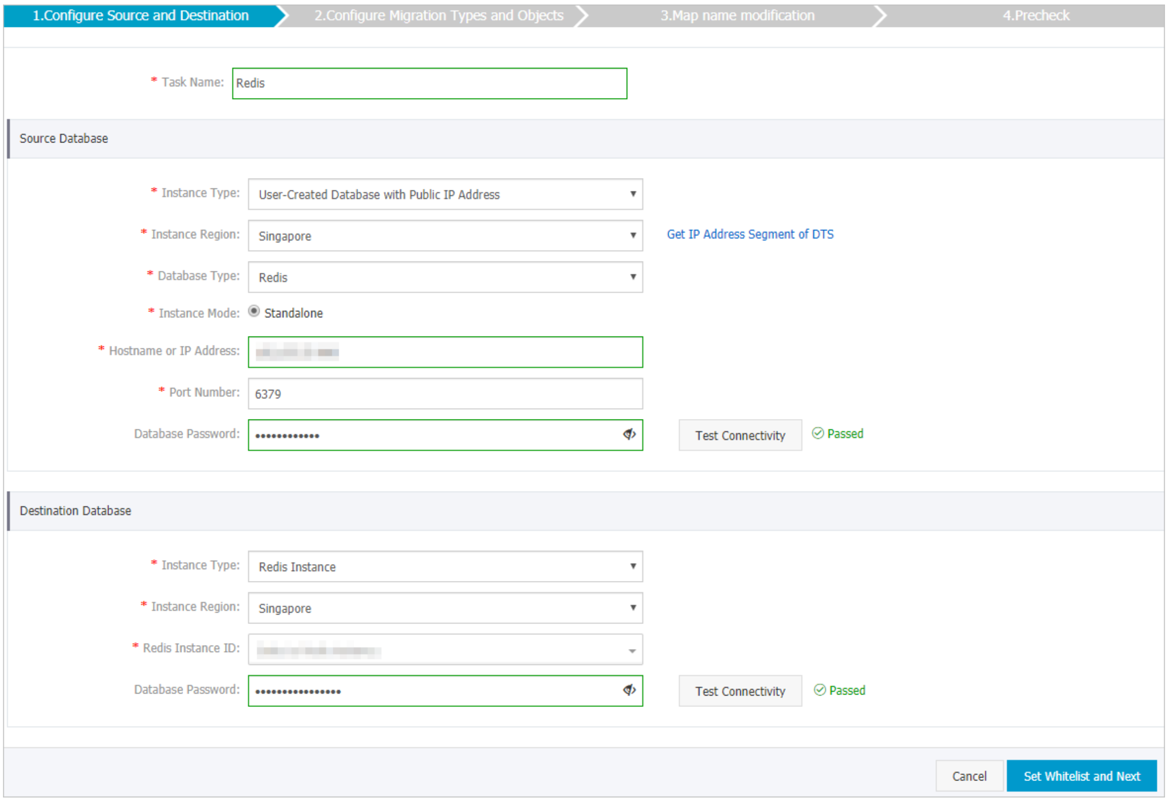
   ```
   redis-cli -h 127.0.0.1 -p 6379 -a Test123456
   ```

2. Run the following command to remove the limit on the replication output buffer:

   ```
   config set client-output-buffer-limit 'slave 0 0 0'
   ```

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 for the data migration task.



1.Configure Source and Destination
2.Configure Migration Types and Objects
3.Map name modification
4.Precheck

* Task Name: Redis

Source Database

* Instance Type: User-Created Database with Public IP Address
* Instance Region: Singapore — Get IP Address Segment of DTS
* Database Type: Redis
* Instance Mode: ● Standalone
* Hostname or IP Address: [redacted]
* Port Number: 6379

Database Password: •••••••••••• — Test Connectivity — ✓ Passed

Destination Database

* Instance Type: Redis Instance
* Instance Region: Singapore
* Redis Instance ID: [redacted]

Database Password: •••••••••••••••• — Test Connectivity — ✓ Passed

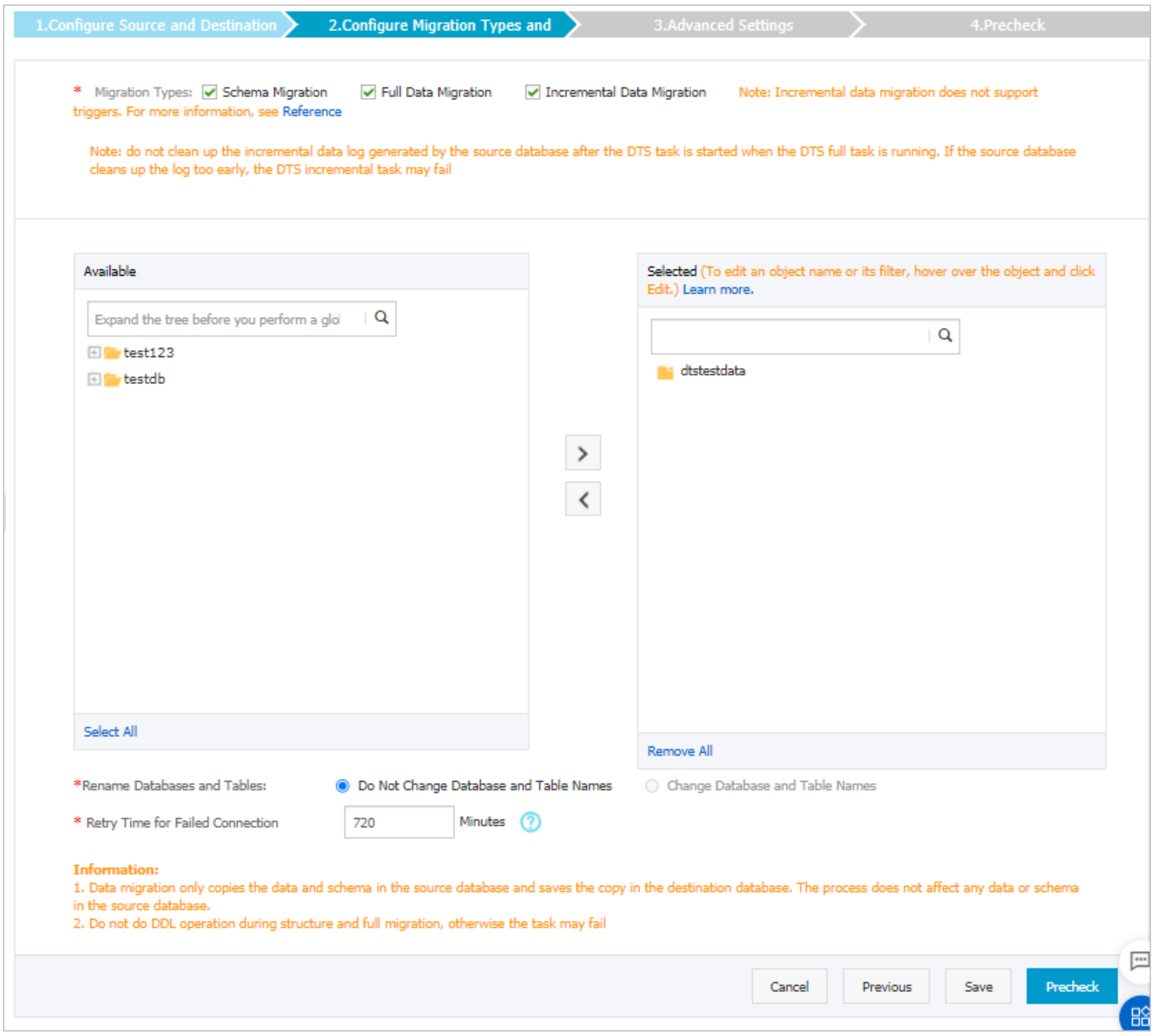
Section	Parameter	Description
N/A	Task Name	DTS automatically generates a task name. We recommend that you specify a descriptive name that helps identify the migration task. You do not need to use a unique task name.
Source Database	Instance Type	Select an instance type based on the deployment of the source database. In this example, select **User-Created Database with Public IP Address**. **Note** If you select other instance types, you must prepare the environment that is required by the source database. For more information, see Preparation overview.
	Instance Region	If the instance type is set to **Self-managed Database with Public IP Address**, you do not need to specify the **instance region**. **Note** If a whitelist is configured for the self-managed Redis 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 **Redis**.
	Instance Mode	The value of this parameter is set to **Standalone** and cannot be changed to Cluster.

Section	Parameter	Description
	Hostname or IP Address	Enter the endpoint that is used to connect to the self-managed Redis database. In this example, enter the public IP address.
	Port Number	Enter the service port number of the self-managed Redis database. The default port number is **6379**. **Note** The service port of the self-managed Redis database must be accessible over the Internet.
	Database Password	Enter the password of the self-managed Redis database. **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 **Redis Instance**.
	Instance Region	Select the region where the destination instance is deployed.
	Redis Instance ID	Select the ID of the destination instance.
	Database Password	Enter the database password of the destination ApsaraDB for Redis instance. **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**.

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

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



1.Configure Source and Destination 2.Configure Migration Types and 3.Advanced Settings 4.Precheck

* Migration Types: ☑ Schema Migration ☑ Full Data Migration ☑ Incremental Data Migration Note: Incremental data migration does not support triggers. For more information, see Reference

Note: do not clean up the incremental data log generated by the source database after the DTS task is started when the DTS full task is running. If the source database cleans up the log too early, the DTS incremental task may fail

Available

Expand the tree before you perform a glo

test123

testdb

Select All

Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.

dtstestdata

Remove All

*Rename Databases and Tables: ● Do Not Change Database and Table Names ○ Change Database and Table Names

* Retry Time for Failed Connection 720 Minutes

Information:
1. Data migration only copies the data and schema in the source database and saves the copy in the destination database. The process does not affect any data or schema in the source database.
2. Do not do DDL operation during structure and full migration, otherwise the task may fail

Cancel Previous Save Precheck

Setting	Description
Select the migration types	◦ To perform only full data migration, select **Full Data Migration**. ◦ To ensure service continuity during data migration, select both **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 self-managed Redis database during data migration. This ensures data consistency between the source and destination databases.
Select the objects to be migrated	Select objects from the **Available** section and click the > icon to move the objects to the **Selected** section. Note You can select only databases as the objects to be migrated.

Setting	Description
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 period of 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**.

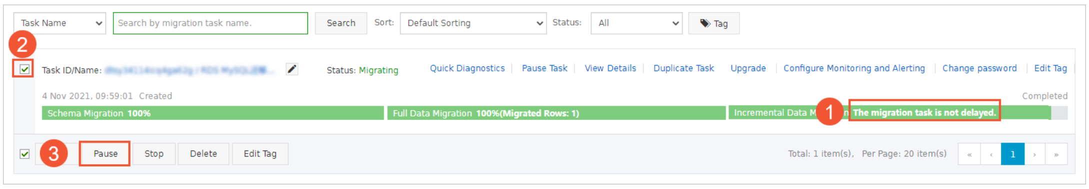
   > **Note**
   >
   > - A precheck is performed before the migration task starts. The migration task only starts after the precheck succeeds.
   > - If the precheck fails, click the
   >
   >   ⓘ
   >
   >   icon next to each failed check item to view the related details. Fix the issues as instructed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



Task ID/Name: [redacted]

Status: Migrating

4 Nov 2021, 09:59:01 Created

Schema Migration 100%

Full Data Migration 100%(Migrated Rows: 1)

Incremental Data M... The migration task is not delayed.

Completed

Total: 1 item(s), Per Page: 20 item(s)

12. Switch your workloads to the destination ApsaraDB for Redis instance.

What to do next

The database accounts that are used for data migration have the read and write permissions. After the data migration task is completed, you must change the passwords of the self-managed Redis database and the ApsaraDB for Redis instance to ensure database security.

1.2.2. Use redis-shake to migrate data from a self-managed Redis database to Alibaba Cloud

You can set redis-shake to sync mode and then use it to migrate data from a self-managed Redis database to an ApsaraDB for Redis instance in real time. The migration does not cause service interruption.

Prerequisites

- The version of the self-managed Redis database is Redis 2.8, 3.0, 3.2, 4.0, or 5.0.

> **Note** If you want to migrate data between different versions of Redis databases, for example, from Redis 2.8 to Redis 4.0, we recommend that you create a pay-as-you-go ApsaraDB for Redis instance to verify the compatibility. If no compatibility issue is found, you can change the billing method of the instance to subscription. For more information about how to change the billing method, see Change the billing method to subscription.

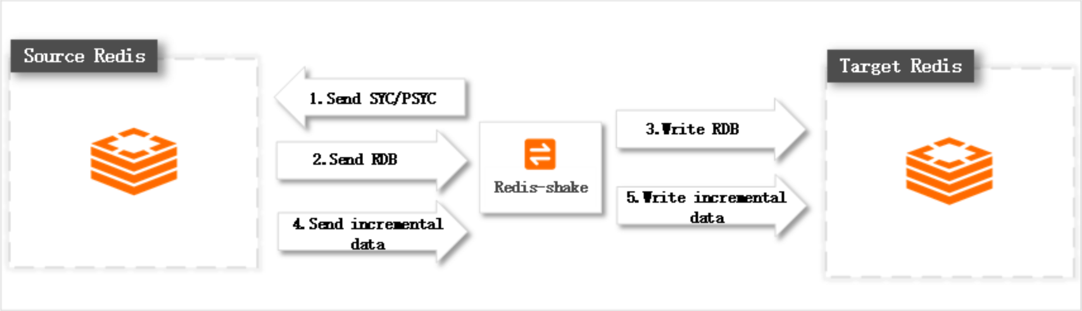
- The destination ApsaraDB for Redis instance must meet the following requirements:
    - The destination ApsaraDB for Redis instance is of Community Edition (Redis 4.0 or 5.0) or Enhanced Edition (Redis 5.0).
    - The destination ApsaraDB for Redis instance use a standard or cluster architecture.

> **Note** If you have not created an instance, see Step 1: Create an ApsaraDB for Redis instance.

Redis-shake introduction

Redis-shake is an open source tool developed by Alibaba Cloud for Redis data transmission. You can use this flexible and efficient tool to parse (decode mode), restore (restore mode), back up (dump mode), and synchronize (sync or rump mode) data on ApsaraDB for Redis instances. When redis-shake works in sync mode, it supports both full data migration and incremental data migration. The detailed migration process is shown in the following figure.

Redis-shake data migration diagram



Note You can also use the data synchronization feature of Data Transmission Service (DTS) to migrate data. DTS is applicable to more scenarios and supports more features. For more information, see Overview.

Precautions

- If the data eviction policy (maxmemory-policy) of the destination database is not set to *noeviction*, data may become inconsistent between the source and destination databases. For more information about the data eviction policy, see How does ApsaraDB for Redis evict data by default?
- After you run the info command to query the keys in the destination database, you may find that the destination database contains fewer keys than the source database. This is caused by the key expiration mechanism of Redis. This situation occurs if the source database contains keys that are not deleted upon expiration.

  Note The numbers of keys that do not have a validity period in the source and destination databases are the same.

Procedure

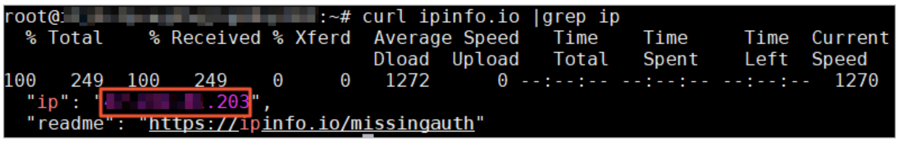
1. Perform the following operations based on the location where redis-shake is installed:

   Note We recommend that you install redis-shake on the host where the self-managed database is deployed. This reduces network latency and prevents connection failures caused by the firewall settings of the self-managed database.

Location where redis-shake is installed	Operation

<table>
<tr><th>Location where redis-shake is installed</th><th>Operation</th></tr>
<tr><td>ECS instances</td><td>
i. Make sure that the ECS instance and the ApsaraDB for Redis instance are deployed in the same VPC. In this case, the same VPC ID is displayed in the Basic Information section of the instances.

Note
■ If the instances are deployed in different VPCs, you can change the VPC to which the ECS instance belongs. For more information, see Change the VPC of an ECS instance.
■ The network types of the ECS instance and the ApsaraDB for Redis instance may be different. For example, the ECS instance belongs to the classic network and the ApsaraDB for Redis instance belongs to a VPC. For information about how to connect to an ApsaraDB for Redis instance from an ECS instance when the instances are deployed in different types of networks, see Connect an ECS instance to an ApsaraDB for Redis instance in different types of networks.

ii. Obtain the internal IP address of the ECS instance. For more information, see Network FAQ.
iii. Add the internal IP address of the ECS instance to a whitelist of the ApsaraDB for Redis instance. For more information, see Configure whitelists.
</td></tr>
<tr><td>On-premises machine</td><td>
i. By default, only internal endpoints are available for ApsaraDB for Redis instances. If you want to connect to an ApsaraDB for Redis instance over the Internet, you must apply for a public endpoint. For more information, see Apply for a public endpoint for an ApsaraDB for Redis instance.
ii. Run the <b>curl ipinfo.io |grep ip</b> command on the on-premises device to obtain its public IP address. The following figure shows a sample result.

<pre>
root@[redacted]:~# curl ipinfo.io |grep ip
  % Total   % Received % Xferd Average Speed   Time   Time   Time Current
                                 Dload Upload   Total   Spent   Left Speed
100   249 100   249   0   0   1272   0 --:--:-- --:--:-- --:--:-- 1270
  "ip": "[redacted].203",
  "readme": "https://ipinfo.io/missingauth"
</pre>
iii. Add the public IP address of the on-premises device to a whitelist of the ApsaraDB for Redis instance. For more information, see Configure whitelists.
</td></tr>
</table>

2. Install redis-shake.

   i. Log on to the host where you want to install redis-shake. The host may be an ECS instance or an on-premises machine.

ii. Run the following command to download the redis-shake file.

```
wget 'http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/attach/120287/cn_zh/
1608173646665/redis-shake-v2.0.3.tar.gz'
```

> Note This example shows how to install redis-shake 2.0.3. You can also install redis-shake of other versions. For more information, see RedisShake.

iii. Run the following command to decompress the redis-shake file:

```
tar xzf redis-shake-v2.0.3.tar.gz
```

3. Migrate data from the host where redis-shake is installed.

i. Run the following command to access the directory of the decompressed redis-shake file and modify the configuration file:

```
cd redis-shake-v2.0.3/ && vim redis-shake.conf
```

> Note After the command is run, the system will open an editing interface. Enter *a* to enter the editing mode.

Parameters

Parameter	Required	Description	Example
source.type	Yes	Select a value based on the architecture of the source database (self-managed database). Valid values: ■ *standalone*: master-replica architecture. ■ *cluster*: cluster architecture.	standalone
source.address	Yes	The connection address and port number of the source database. Separate the connection address and port number with a colon (:). Note ■ If the source database and redis-shake are deployed on the same host, enter `127.0.0.1:6379`. ■ If the source database is a cluster instance, you must add the `master@` prefix to the connection address, for example, `master@127.0.0.1:6379`.	127.0.0.1:6379

Parameter	Required	Description	Example
source.password_raw	Yes	The password of the source database. If no password is set for the source database, you do not need to specify a password.	Rp829dlwa
target.type	Yes	Select a value based on the architecture of the destination ApsaraDB for Redis instance. Valid values: - *standalone*: standard (master-replica) architecture. - *cluster*: cluster architecture.	standalone
target.address	Yes	The connection address and port number of the destination ApsaraDB for Redis instance. Separate the connection address and port number with a colon (:). For more information about how to obtain the connection address and port number of the ApsaraDB for Redis instance, see . - The ECS instance is connected through a virtual private cloud (VPC): You must obtain the internal endpoint of the ApsaraDB for Redis instance. - If you connect an on-premises machine to the ApsaraDB for Redis instance over the Internet, you must obtain the public IP address of the ApsaraDB for Redis instance. **Note** If the destination instance uses the cluster architecture, you must connect to the instance deployed in a VPC by using a private endpoint. You must add the prefix `master@` to the connection address, for example, `master@r-bp1mfnrflszg75w****.redis.rds.aliyuncs.com:6379`. For more information about how to request a private endpoint, see Enable the direct connection mode.	r-bp1wcw2rlw76acc5k****.redis.rds.aliyuncs.com:6379
target.password_raw	Yes	The account with **read and write** permissions and the password of the destination ApsaraDB for Redis instance. Separate the account and password with a colon (:). For more information about how to create an account, see Create and manage database accounts.	testaccount:Rp829dlwa

Parameter	Required	Description	Example
target.db	No	Migrates data from all databases in the source database to the specified databases of the destination database. Valid values: *0* to *15*. Note The default value *-1* indicates that the feature is disabled.	-1
key_exists	No	The data writing policy that is applied when the keys in the source database are the same as those in the destination database. Valid values: ■ *rewrite*: overwrites the existing keys in the destination database that are the same as those in the source database. ■ *none*: This is the default value. Redis-shake stops running and a message that indicates conflicting keys appears. ■ *ignore*: skips the current key, retains the data in the destination database, and continues to migrate data.	rewrite
filter.db.whitelist	No	The names of the databases to be migrated. Separate multiple database names with semicolons (;). By default, this parameter is left empty. This indicates that all databases are migrated.	0;1
filter.db.blacklist	No	The names of databases that you do not want to migrate. This parameter is equivalent to a blacklist. Separate multiple database names with semicolons (;). By default, this parameter is left empty. This indicates that no database is added to the blacklist.	1;2
parallel	No	The number of concurrent threads for redis-shake to perform the migration tasks. You can increase this value to improve synchronization performance. Note The default value is 32. The minimum value is 1. The maximum value depends on the performance of the server where redis-shake is deployed.	32

Note You do not need to configure other parameters unless otherwise required. For more information, see the comments on each parameter in the *redis-shake.conf* file.

ii. Press the Esc key to exit the editing mode and enter *:wq* to save and close the file.

iii. Run the following command to start redis-shake and migrate data:

```
./redis-shake.linux -type=sync -conf=redis-shake.conf
```

Redis-shake prints the operational log on the screen.

> ? **Note** For more information about the causes of related errors and how to fix these errors, see FAQ.

4. (Optional)Stop the data migration task as required.

> ? **Note** If you need to run redis-shake for a long time to migrate data in real time, skip this step.

i. Check the log data and wait until the migration task enters the incremental data migration state.

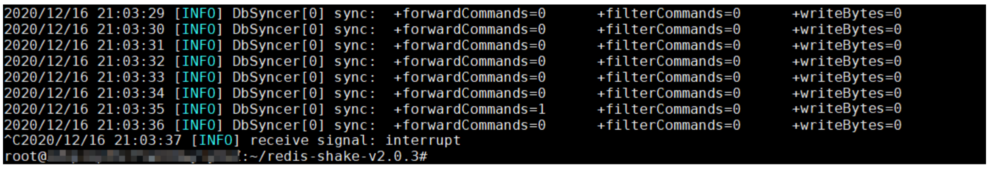
Task stage	Logs

Full data migration	`2020/12/16 21:02:36 [INFO] DbSyncer[0] total = 4.00MB - 2.18MB [ 54%]  entry=52199`
	`2020/12/16 21:02:36 [INFO] DbSyncer[0] total = 4.00MB - 4.00MB [100%]  entry=97531`
	`2020/12/16 21:02:36 [INFO] DbSyncer[0] sync rdb done`
	? **Note** When `sync rdb done` appears in the log data, full synchronization is completed and incremental synchronization starts.

<table>
<tr><th>Task stage</th><th>Logs</th></tr>
<tr><td>Incremental data migration</td><td>
<pre>
2020/12/16 21:03:07 [INFO] DbSyncer[0] sync: +forwardCommands=5
+filterCommands=0   +writeBytes=5095
2020/12/16 21:03:08 [INFO] DbSyncer[0] sync: +forwardCommands=7
+filterCommands=0   +writeBytes=7133
2020/12/16 21:03:09 [INFO] DbSyncer[0] sync: +forwardCommands=0
+filterCommands=0   +writeBytes=0
2020/12/16 21:03:10 [INFO] DbSyncer[0] sync: +forwardCommands=645
+filterCommands=0   +writeBytes=657255
2020/12/16 21:03:11 [INFO] DbSyncer[0] sync: +forwardCommands=28
+filterCommands=0   +writeBytes=28532
2020/12/16 21:03:12 [INFO] DbSyncer[0] sync: +forwardCommands=0
+filterCommands=0   +writeBytes=0
</pre>
The log contains the following parameters:
<ul>
<li><code>forwardCommands</code>: the number of commands that are sent from the source database.</li>
<li><code>filterCommands</code>: the number of commands that are filtered out. For example, you can set the configuration file of redis-shake to filter out specific databases.</li>
<li><code>writeBytes</code>: the number of bytes that are sent from the source database.</li>
</ul>
</td></tr>
</table>

ii. Stop writing data to the source database and wait until the value of the `writeBytes` parameter in the returned log remains 0 for multiple times in succession, and then press Ctrl+C to stop the running of redis-shake.

Stop running example



```
2020/12/16 21:03:29 [INFO] DbSyncer[0] sync:  +forwardCommands=0       +filterCommands=0       +writeBytes=0
2020/12/16 21:03:30 [INFO] DbSyncer[0] sync:  +forwardCommands=0       +filterCommands=0       +writeBytes=0
2020/12/16 21:03:31 [INFO] DbSyncer[0] sync:  +forwardCommands=0       +filterCommands=0       +writeBytes=0
2020/12/16 21:03:32 [INFO] DbSyncer[0] sync:  +forwardCommands=0       +filterCommands=0       +writeBytes=0
2020/12/16 21:03:33 [INFO] DbSyncer[0] sync:  +forwardCommands=0       +filterCommands=0       +writeBytes=0
2020/12/16 21:03:34 [INFO] DbSyncer[0] sync:  +forwardCommands=0       +filterCommands=0       +writeBytes=0
2020/12/16 21:03:35 [INFO] DbSyncer[0] sync:  +forwardCommands=1       +filterCommands=0       +writeBytes=0
2020/12/16 21:03:36 [INFO] DbSyncer[0] sync:  +forwardCommands=0       +filterCommands=0       +writeBytes=0
^C2020/12/16 21:03:37 [INFO] receive signal: interrupt
root@[redacted]:~/redis-shake-v2.0.3#
```

> **Note** The preceding figure indicates that the data in the source databases and destination databases is consistent. You can switch workloads from the self-managed Redis database to the ApsaraDB for Redis instance.

1.2.3. Use redis-shake to migrate the data of a self-managed Redis database from a backup file to an ApsaraDB for Redis instance

This topic describes how to migrate the data of a self-managed Redis database from a backup file in the RDB format to an ApsaraDB for Redis instance by using redis-shake.

Prerequisites

- The self-managed Redis database runs Redis 2.8, 3.0, 3.2, 4.0, or 5.0.

> **Note** If the self-managed Redis database and the ApsaraDB for Redis instance run different Redis versions, we recommend that you create a pay-as-you-go ApsaraDB for Redis instance to test the compatibility between the two versions. If no compatibility issues are found, you can change the billing method of the ApsaraDB for Redis instance to subscription and migrate the data of the self-managed Redis database to the ApsaraDB for Redis instance. For more information about how to change the billing method of an ApsaraDB for Redis instance, see Change the billing method to subscription.

- The ApsaraDB for Redis instance runs the ApsaraDB for Redis Community Edition in Redis 4.0 or 5.0 or the ApsaraDB for Redis Enhanced Edition (Tair) in Redis 5.0. For more information about how to create an ApsaraDB for Redis instance, see Step 1: Create an ApsaraDB for Redis instance.

Introduction to redis-shake

Redis-shake is an open source tool that is developed by Alibaba Cloud. Redis-shake is easy to deploy and can be used to efficiently parse (decode mode), restore (restore mode), back up (dump mode), and synchronize (sync or rump mode) the data of Redis databases. In this topic, the restore mode of redis-shake is used to restore the data of the self-managed Redis database from an RDB file to the ApsaraDB for Redis instance.

> **Note**
>
> - For more information about how to migrate incremental data by using redis-shake, see Use redis-shake to migrate data from a self-managed Redis database to Alibaba Cloud.
> - You can also use the data synchronization feature of Data Transmission Service (DTS) to migrate data. DTS provides more features and can be used for more scenarios compared with redis-shake. For more information, see Overview.

Precautions

- If the data eviction policy (maxmemory-policy) of the destination database is not set to *noeviction*, data may become inconsistent between the source and destination databases. For more information about the data eviction policy, see How does ApsaraDB for Redis evict data by default?
- After you run the info command to query the keys in the destination database, you may find that the destination database contains fewer keys than the source database. This is caused by the key expiration mechanism of Redis. This situation occurs if the source database contains keys that are not deleted upon expiration.

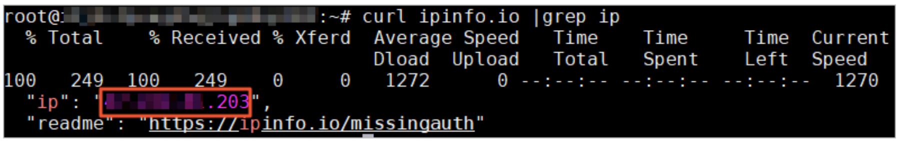
> **Note** The numbers of keys that do not have a validity period in the source and destination databases are the same.

Procedure

1. Perform the following operations based on the installation location of redis-shake:

> **Note** We recommend that you install redis-shake on an Elastic Compute Service (ECS) instance. You can connect the ECS instance and the ApsaraDB for Redis instance over a virtual private cloud (VPC) to increase security and reduce network latencies.

Installation location of redis-shake	Operation
(Recommended) ECS instance	i. Make sure that the ECS instance and the ApsaraDB for Redis instance are deployed in the same VPC. In this case, the same VPC ID is displayed in the Basic Information section of the instances. Note ■ If the instances are deployed in different VPCs, you can change the VPC to which the ECS instance belongs. For more information, see Change the VPC of an ECS instance. ■ The network types of the ECS instance and the ApsaraDB for Redis instance may be different. For example, the ECS instance belongs to the classic network and the ApsaraDB for Redis instance belongs to a VPC. For information about how to connect to an ApsaraDB for Redis instance from an ECS instance when the instances are deployed in different types of networks, see Connect an ECS instance to an ApsaraDB for Redis instance in different types of networks. ii. Obtain the internal IP address of the ECS instance. For more information, see Network FAQ. iii. Add the internal IP address of the ECS instance to a whitelist of the ApsaraDB for Redis instance. For more information, see Configure whitelists.

On-premises machine	i. By default, only internal endpoints are available for ApsaraDB for Redis instances. If you want to connect to an ApsaraDB for Redis instance over the Internet, you must apply for a public endpoint. For more information, see Apply for a public endpoint for an ApsaraDB for Redis instance.
	ii. Run the **curl ipinfo.io |grep ip** command on the on-premises device to obtain its public IP address. The following figure shows a sample result.
	
	`root@[redacted]:~# curl ipinfo.io |grep ip`
	`  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current`
	`                                 Dload  Upload   Total   Spent    Left  Speed`
	`100   249  100   249    0     0   1272      0 --:--:-- --:--:-- --:--:--  1270`
	`  "ip": "[redacted].203",`
	`  "readme": "https://ipinfo.io/missingauth"`
	iii. Add the public IP address of the on-premises device to a whitelist of the ApsaraDB for Redis instance. For more information, see Configure whitelists.

2. Install redis-shake.

   i. Log on to the host where you want to install redis-shake. The host may be an ECS instance or an on-premises machine.

ii. Run the following command to download the redis-shake file.

```
wget 'http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/attach/120287/cn_zh/
1608173646665/redis-shake-v2.0.3.tar.gz'
```

> ⓘ **Note** This example shows how to install redis-shake 2.0.3. You can also install redis-shake of other versions. For more information, see RedisShake.

iii. Run the following command to decompress the redis-shake file:

```
tar xzf redis-shake-v2.0.3.tar.gz
```

3. Start the migration on the ECS instance or the on-premises machine.

i. Copy the RDB file that contains the backup data of the self-managed Redis database to the ECS instance or the on-premises machine. In this example, the RDB file is stored in the following path: */root/redisdump*.

ii. Run the following command to open and edit the redis-shake.conf file in the directory that is generated after you decompress the redis-shake software package:

```
cd redis-shake-v2.0.3/ && vim redis-shake.conf
```

> ⓘ **Note** After you run the command, the system opens an editor. You can type *a* to enter the edit mode.

Parameter description

Parameter	Required	Description	Example
source.rdb.input	Yes	The save path of the RDB file. You can specify a relative path or an absolute path. ⓘ **Note** If you need to specify more than one RDB file, separate the save paths of the RDB files with semicolons (;).	/root/redisdump/dump.rdb

Parameter	Required	Description	Example
target.address	Yes	The endpoint and port number of the ApsaraDB for Redis instance. Separate the endpoint and port number with a colon (:). For more information about how to obtain the endpoint and port number of an ApsaraDB for Redis instance, see View endpoints. ■ If you install redis-shake on an ECS instance and connect the ECS instance and the ApsaraDB for Redis instance over a VPC, obtain the internal endpoint and port number of the ApsaraDB for Redis instance. ■ If you install redis-shake on an on-premises machine and connect the on-premises machine and the ApsaraDB for Redis instance over the Internet, obtain the public endpoint and port number of the ApsaraDB for Redis instance.	r-bp1wcw2rlw76acc5k****.redis.rds.aliyuncs.com:6379
target.password_raw	Yes	The username and password of the account that has the **Read/Write** permissions on the ApsaraDB for Redis instance. Separate the username and password with a colon (:). For more information about how to create an account, see Create and manage database accounts.	testaccount:Rp829dlwa
key_exists	No	The data write policy that is used when the keys in the self-managed Redis database are the same as the keys in the ApsaraDB for Redis instance. Valid values: ■ *rewrite*: The keys in the self-managed Redis database overwrite the keys with identical names in the ApsaraDB for Redis instance. ■ *none*: Redis-shake stops running, and a message that indicates conflicting keys appears. This value is the default value. ■ *ignore*: Redis-shake skips the keys in the self-managed Redis database, retains the keys with identical names in the ApsaraDB for Redis instance, and continues to migrate the other data.	rewrite

Parameter	Required	Description	Example
parallel	No	The number of concurrent threads that are invoked by redis-shake to perform the migration. You can increase the value of this parameter to accelerate the migration. **Note** The default value is 32. The minimum value is 1. The maximum value varies based on the performance of the server on which redis-shake is installed.	32

> **Note** You do not need to configure other parameters unless otherwise specified. For more information, see the comments on each parameter in the *redis-shake.conf* file.

iii. Press Esc to exit the edit mode. Then, type *:wq* and press Enter to save the redis-shake.conf file and exit the editor.

iv. Run the following command to start redis-shake and migrate data:

```
./redis-shake.linux -type=restore -conf=redis-shake.conf
```

Redis-shake displays the operational log on the screen.

> **Note** For more information about the causes of and solutions to possible errors, see FAQ.

4. Wait until the migration is complete. Then, end the migration task.

i. Check the log data and wait until the migration task completes incremental data synchronization.

```
2021/06/18 10:44:25 [INFO] restore from '[/root/redisdump/dump.rdb]' to '[r-bp1bjmq
wlelmkhy****.redis.rds.aliyuncs.com:6379]'
2021/06/18 10:44:25 [INFO] routine[0] starts restoring data from /root/redisdump/du
mp.rdb to [r-bp1bjmqwlelmkhy5ry933.redis.rds.aliyuncs.com:6379]
2021/06/18 10:44:26 [INFO] routine[0] total = 11.15MB -       6.35MB [ 56%]  entry=
5692
2021/06/18 10:44:27 [INFO] routine[0] total = 11.15MB -      11.15MB [100%]  entry=
10001
2021/06/18 10:44:27 [INFO] routine[0] restore: rdb done
2021/06/18 10:44:27 [INFO] restore from '[/root/redisdump/dump.rdb]' to '[r-bp1bjmq
wlelmkhy****.redis.rds.aliyuncs.com:6379]' done
2021/06/18 10:44:27 [INFO] Enabled http stats, set status (incr), and wait forever.
```

> **Note** When `restore: rdb done` appears in the operational log, the migration is complete.

ii. Press Ctrl+C to stop redis-shake.

5. (Optional)Verify the data in the ApsaraDB for Redis instance. For more information, see Verify data after migration.

1.2.4. Use redis-shake to migrate data from on-premises Codis or Redis to ApsaraDB for Redis

You can use the sync mode of the redis-shake tool to migrate the data of an on-premises Codis or Redis cluster to an ApsaraDB for Redis instance.

> ⓘ **Note** For more information about how to migrate the data of a standalone Redis instance, see Use redis-shake to migrate data from a self-managed Redis database to Alibaba Cloud.

Prerequisites

- An ApsaraDB for Redis instance is created as the destination for data migration. For more information about how to create an ApsaraDB for Redis instance, see Create an ApsaraDB for Redis instance.
- An Elastic Compute Service (ECS) instance is created to run the redis-shake tool. The 64-bit Linux operating system is running on the ECS instance. For more information about how to create an ECS instance, see Create an ECS instance.
- The ECS instance is allowed to access the ApsaraDB for Redis instance.

  > ⓘ **Note**
  >
  > - If the ECS instance and the ApsaraDB for Redis instance are deployed in the same virtual private cloud (VPC), you must add the internal IP address of the ECS instance to the whitelist of the ApsaraDB for Redis instance. For more information, see Configure whitelists.
  > - If the ECS instance and the ApsaraDB for Redis instance are not deployed in the same VPC, the ECS instance can access the ApsaraDB for Redis instance by using the public endpoint. For more information, see Use a public endpoint to connect to an ApsaraDB for Redis instance.

Limits

- The sync mode applies to scenarios such as data migration from on-premises Redis to the cloud. It also applies to data synchronization between on-premises Redis and ApsaraDB for Redis, and data synchronization between on-premises Redis databases. ApsaraDB for Redis cluster instances cannot be used as the source for data migration in sync mode.
- In sync mode, the source Redis must support the **SYNC** and **PSYNC** commands.

Introduction to redis-shake

The redis-shake tool is an open source tool developed by Alibaba Cloud. You can use it to parse (decode mode), recover (restore mode), back up (dump mode), and synchronize (sync or rump mode) Redis data. In sync mode, redis-shake runs the **SYNC** or **PSYNC** command to synchronize full or incremental data from the source Redis to ApsaraDB for Redis. Incremental synchronization automatically starts after full synchronization is completed. This topic describes how to use the sync mode of the redis-shake tool to migrate the data of an on-premises Codis or Redis cluster to an ApsaraDB for Redis instance.

Note

- The sync mode supports data synchronization between different Redis versions. For example, when you synchronize data between Redis 2.8 and Redis 4.0.
- For more information about the redis-shake tool, see redis-shake on GitHub or FAQ.

Procedure

1. Log on to the ECS instance that can access the ApsaraDB for Redis instance.
2. Download redis-shake in the ECS instance.

   Note We recommend that you download the latest version of redis-shake.

3. Run the following command to decompress the downloaded *redis-shake.tar.gz* package:

```
tar -xvf redis-shake.tar.gz
```

   Note The decompressed *redis-shake* file is a binary file that runs on a 64-bit Linux operating system. The *redis-shake.conf* file is the configuration file of the redis-shake tool. You must modify this configuration file in the next step.

4. Modify the *redis-shake.conf* file. The following table describes the parameters for the sync mode of the redis-shake tool.

Parameter	Description	Example
source.type	The type of the source Codis or Redis cluster.	`cluster`
source.address	The endpoint and port of the source Codis or Redis cluster. Note Data migration will consume resources. To minimize adverse impacts on your services, we recommend that you use the endpoints and ports of replica nodes.	`10.xx.xx.1:7000;10.xx.xx.1:7002;10.xx.xx.1:7003;10.xx.xx.1:7004`
source.password_raw	The password of the source Codis or Redis cluster.	`SourcePass233`
target.type	The type of the ApsaraDB for Redis instance.	`proxy`
target.address	The endpoint and port of the ApsaraDB for Redis instance. For more information, see View endpoints.	`r-bpxxxxxxxxxxxxxx.redis.rds.aliyuncs.com:6379`
target.password_raw	The password of the ApsaraDB for Redis instance.	`TargetPass233`

Parameter	Description	Example
rewrite	Specifies whether to overwrite the existing keys in ApsaraDB for Redis that are same as those in the RDB file. Valid values: ○ true: overwrites the existing keys. ○ false: does not overwrite the existing keys. **Note** Default value: true. We recommend that you back up the valid data in ApsaraDB for Redis before data migration. If you set this parameter to false and a key exists in both the source and destination databases, an error is returned.	`true`
target.db	The database to which the data is migrated in the ApsaraDB for Redis instance. For example, to migrate data from the on-premises Redis to DB10 of the ApsaraDB for Redis instance, set this parameter to 10. If you set this parameter to -1, data in the source Redis cluster is migrated to the same database in the ApsaraDB for Redis instance. For example, the data of DB0 in the source Redis cluster is migrated to DB0 in the ApsaraDB for Redis instance, the data of DB1 in the source Redis cluster is migrated to DB1 in the ApsaraDB for Redis instance.	`0`

Parameter	Description	Example
parallel	The number of concurrent threads used to synchronize the RDB file. More concurrent threads improve synchronization performance. Note - Minimum value: 1. - Maximum value: depends on the server performance. - We recommend that you set this parameter to 64.	`64`

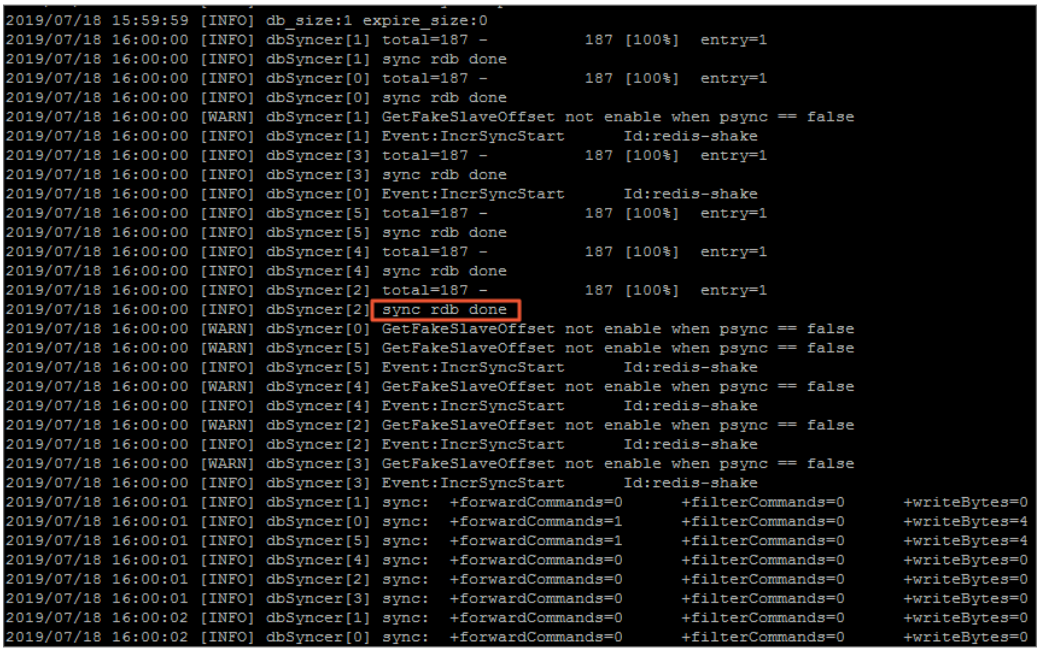
5. Run the following command to migrate data:

```
./redis-shake -type=sync -conf=redis-shake.conf
```

> Note You must run this command in the same directory as the *redis-shake* and *redis-shake.conf* files. Otherwise, you must specify the correct file path in the command.

6. Check the synchronization status in synchronization logs. When `sync rdb done` appears in the log, full synchronization is completed and incremental synchronization starts.

   Synchronization logs



```
2019/07/18 15:59:59 [INFO] db_size:1 expire_size:0
2019/07/18 16:00:00 [INFO] dbSyncer[1] total=187 -          187 [100%]  entry=1
2019/07/18 16:00:00 [INFO] dbSyncer[1] sync rdb done
2019/07/18 16:00:00 [INFO] dbSyncer[0] total=187 -          187 [100%]  entry=1
2019/07/18 16:00:00 [INFO] dbSyncer[0] sync rdb done
2019/07/18 16:00:00 [WARN] dbSyncer[1] GetFakeSlaveOffset not enable when psync == false
2019/07/18 16:00:00 [INFO] dbSyncer[1] Event:IncrSyncStart      Id:redis-shake
2019/07/18 16:00:00 [INFO] dbSyncer[3] total=187 -          187 [100%]  entry=1
2019/07/18 16:00:00 [INFO] dbSyncer[3] sync rdb done
2019/07/18 16:00:00 [INFO] dbSyncer[0] Event:IncrSyncStart      Id:redis-shake
2019/07/18 16:00:00 [INFO] dbSyncer[5] total=187 -          187 [100%]  entry=1
2019/07/18 16:00:00 [INFO] dbSyncer[5] sync rdb done
2019/07/18 16:00:00 [INFO] dbSyncer[4] total=187 -          187 [100%]  entry=1
2019/07/18 16:00:00 [INFO] dbSyncer[4] sync rdb done
2019/07/18 16:00:00 [INFO] dbSyncer[2] total=187 -          187 [100%]  entry=1
2019/07/18 16:00:00 [INFO] dbSyncer[2] sync rdb done
2019/07/18 16:00:00 [WARN] dbSyncer[0] GetFakeSlaveOffset not enable when psync == false
2019/07/18 16:00:00 [WARN] dbSyncer[5] GetFakeSlaveOffset not enable when psync == false
2019/07/18 16:00:00 [INFO] dbSyncer[5] Event:IncrSyncStart      Id:redis-shake
2019/07/18 16:00:00 [WARN] dbSyncer[4] GetFakeSlaveOffset not enable when psync == false
2019/07/18 16:00:00 [INFO] dbSyncer[4] Event:IncrSyncStart      Id:redis-shake
2019/07/18 16:00:00 [WARN] dbSyncer[2] GetFakeSlaveOffset not enable when psync == false
2019/07/18 16:00:00 [INFO] dbSyncer[2] Event:IncrSyncStart      Id:redis-shake
2019/07/18 16:00:00 [WARN] dbSyncer[3] GetFakeSlaveOffset not enable when psync == false
2019/07/18 16:00:00 [INFO] dbSyncer[3] Event:IncrSyncStart      Id:redis-shake
2019/07/18 16:00:01 [INFO] dbSyncer[1] sync:  +forwardCommands=0      +filterCommands=0      +writeBytes=0
2019/07/18 16:00:01 [INFO] dbSyncer[0] sync:  +forwardCommands=1      +filterCommands=0      +writeBytes=4
2019/07/18 16:00:01 [INFO] dbSyncer[5] sync:  +forwardCommands=1      +filterCommands=0      +writeBytes=4
2019/07/18 16:00:01 [INFO] dbSyncer[4] sync:  +forwardCommands=0      +filterCommands=0      +writeBytes=0
2019/07/18 16:00:01 [INFO] dbSyncer[2] sync:  +forwardCommands=0      +filterCommands=0      +writeBytes=0
2019/07/18 16:00:01 [INFO] dbSyncer[3] sync:  +forwardCommands=0      +filterCommands=0      +writeBytes=0
2019/07/18 16:00:02 [INFO] dbSyncer[1] sync:  +forwardCommands=0      +filterCommands=0      +writeBytes=0
2019/07/18 16:00:02 [INFO] dbSyncer[0] sync:  +forwardCommands=0      +filterCommands=0      +writeBytes=0
```

> Note Whenever the data of a node in Codis or Redis is synchronized, `sync rdb done` appears.

7. After the data of all nodes is synchronized, `+forwardCommands=0` indicates that no data is written to the source database and no incremental data is being synchronized. In this case, switch your services to the new database.

1.2.5. Synchronize data from a Codis cluster hosted on ECS to an ApsaraDB for Redis instance

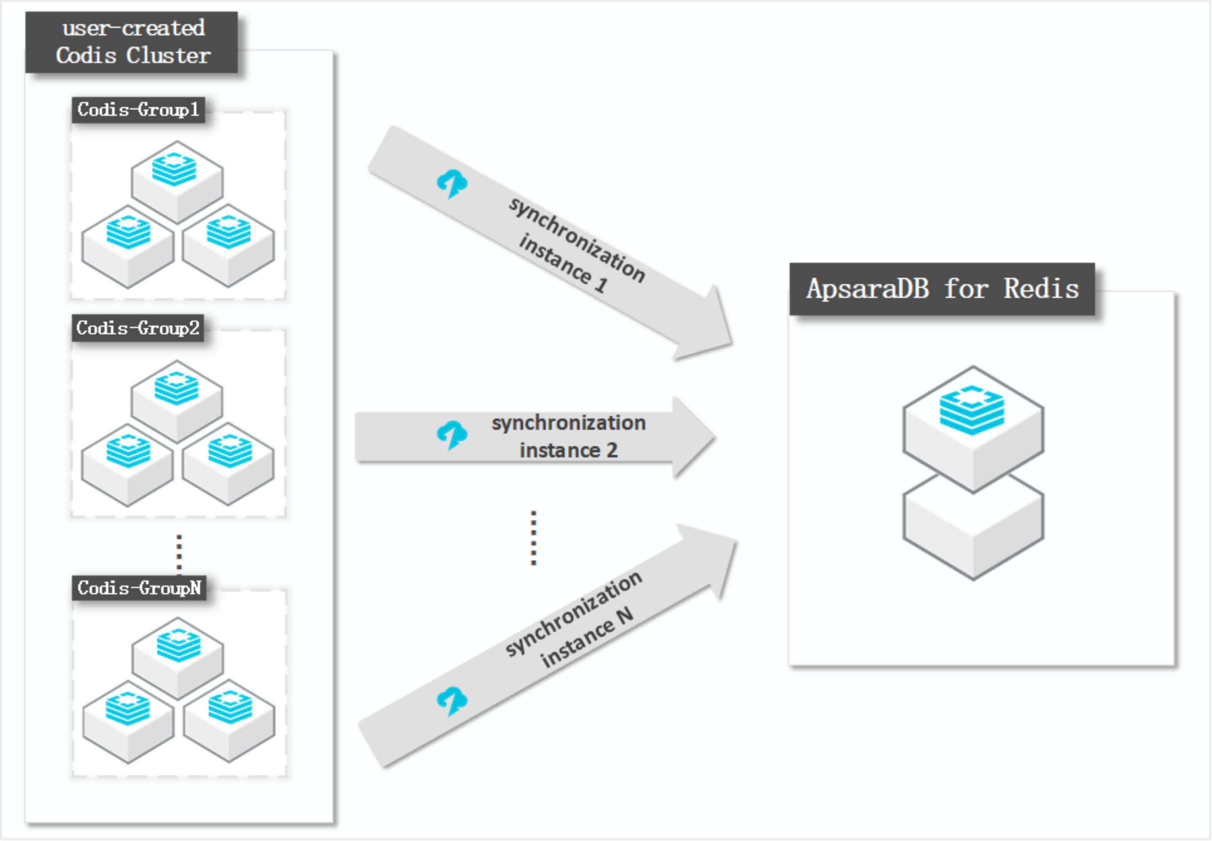
ApsaraDB for Redis is a database service compatible with the open source Redis protocol and provides hybrid storage of memory and disks. Based on reliable hot standby architectures and scalable cluster architectures, ApsaraDB for Redis is suitable for scenarios that require flexible configuration changes, high throughput, and low latency. This topic describes how to synchronize data from a Codis cluster to an ApsaraDB for Redis instance by using Data Transmission Service (DTS).

Prerequisites

- An ApsaraDB for Redis instance is created as the destination instance, and the database version of the instance is 2.8, 4.0, or 5.0. For more information, see Step 1: Create an ApsaraDB for Redis instance.
- The available storage capacity of the destination ApsaraDB for Redis instance is larger than the total size of data stored in the source Codis cluster.
- All master nodes in the source Codis cluster support the `PSYNC` command.

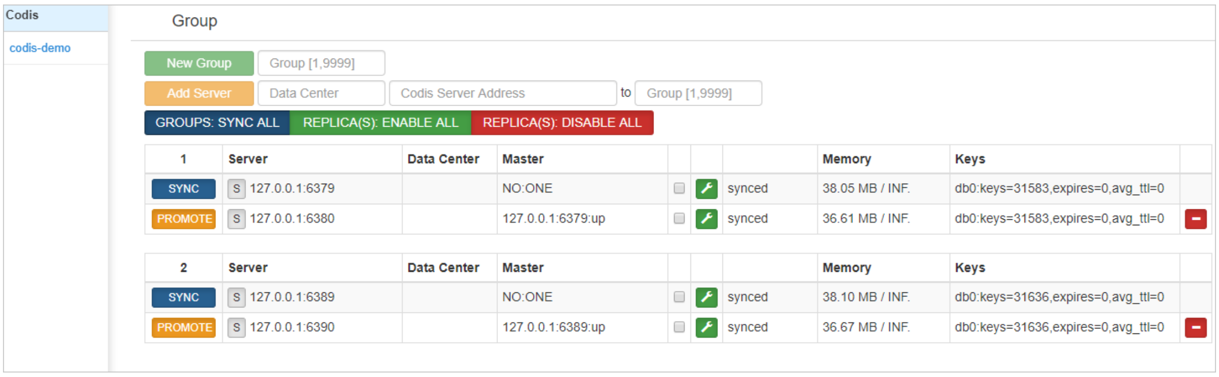
How DTS synchronizes data from a Codis cluster

A Codis cluster consists of multiple codis-groups. You must create a data synchronization task for each codis-group. DTS synchronizes each codis-group in a data synchronization task until the whole cluster is synchronized.



Architecture of the Codis cluster

In this topic, a Codis cluster that consists of two codis-groups is used. Each codis-group runs in a master-replica architecture. The following figure shows the architecture of the cluster.



1	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6379		NO:ONE	□		synced	38.05 MB / INF.	db0:keys=31583,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6380		127.0.0.1:6379:up	□		synced	36.61 MB / INF.	db0:keys=31583,expires=0,avg_ttl=0	−

2	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6389		NO:ONE	□		synced	38.10 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6390		127.0.0.1:6389:up	□		synced	36.67 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0	−

Precautions

- DTS uses the resources of the source and destination instances during initial full data synchronization. This may increase the loads of the database servers. If you migrate a large amount of data or if the server specifications cannot meet your requirements, database services may become unavailable. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.
- If the `bind` parameter is configured in the *redis.conf* file of the source database, you must set the value of this parameter to the internal IP address of the ECS instance. The setting ensures that DTS can connect to the source database.
- To ensure the stability of data synchronization, we recommend that you increase the value of the *re*

pl-backlog-size parameter in the `redis.conf` file of the source Codis cluster.

- To ensure the synchronization quality, DTS adds the following key to the source Codis cluster: `DTS_REDIS_TIMESTAMP_HEARTBEAT`. This key is used to record the time when data is synchronized to the destination instance.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command in the source Codis cluster. Otherwise, data may become inconsistent between the source and destination databases.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction`, data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

  > **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- The database version of the destination ApsaraDB for Redis instance must be 2.8, 4.0, or 5.0. The version of the destination database must be the same as or later than the version of the source database. If you synchronize data between different versions of Redis databases, make sure that the versions of the source and destination databases are compatible with each other. For example, you can create a destination pay-as-you-go ApsaraDB for Redis instance to verify the compatibility between the source and destination databases. Then, you can release the instance or change the billing method of the instance to subscription.

Supported synchronization topologies

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

For more information, see Synchronization topologies.

Commands that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, PSETEX, and PUBLISH
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE

- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE

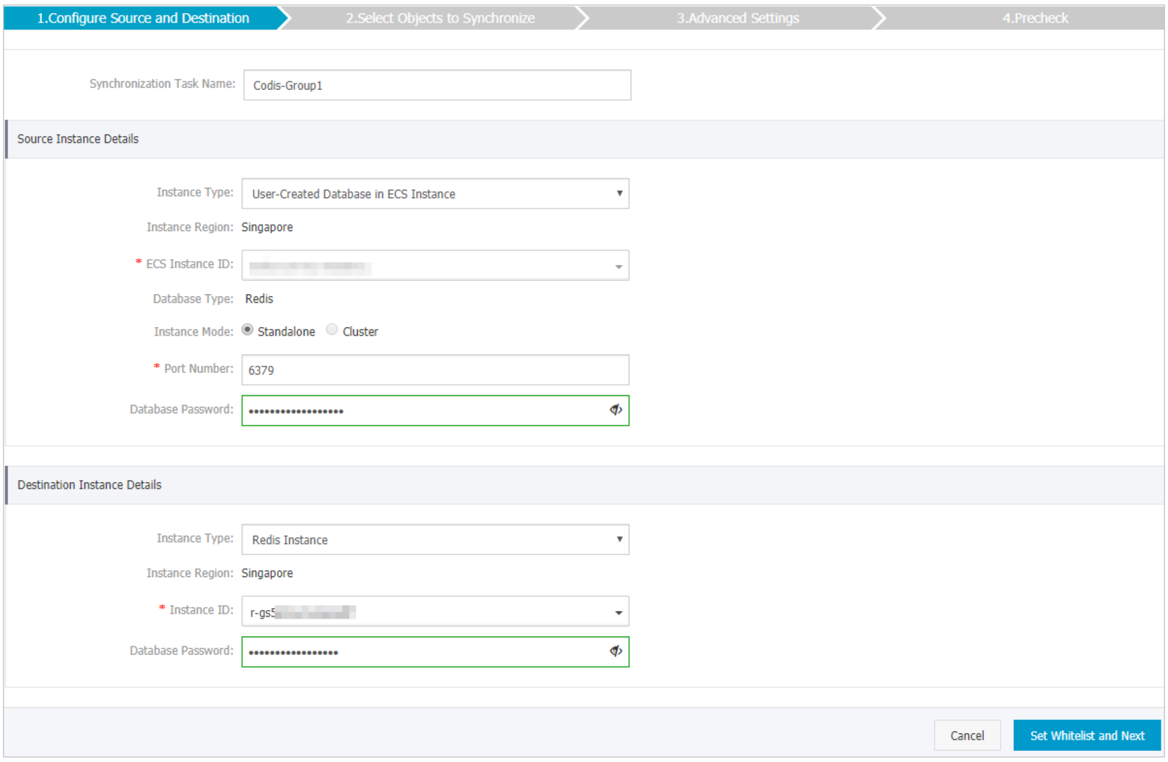
> ? Note
>
> - PUBLISH operations cannot be synchronized.
> - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
> - When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Procedure

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

   > ? **Note** On the buy page, set Source Instance to **Redis**, set Destination Instance to **Redis**, 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.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Task Name: Codis-Group1

Source Instance Details

Instance Type: User-Created Database in ECS Instance

Instance Region: Singapore

* ECS Instance ID: [redacted]

Database Type: Redis

Instance Mode: ● Standalone ○ Cluster

* Port Number: 6379

Database Password: •••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs5[redacted]

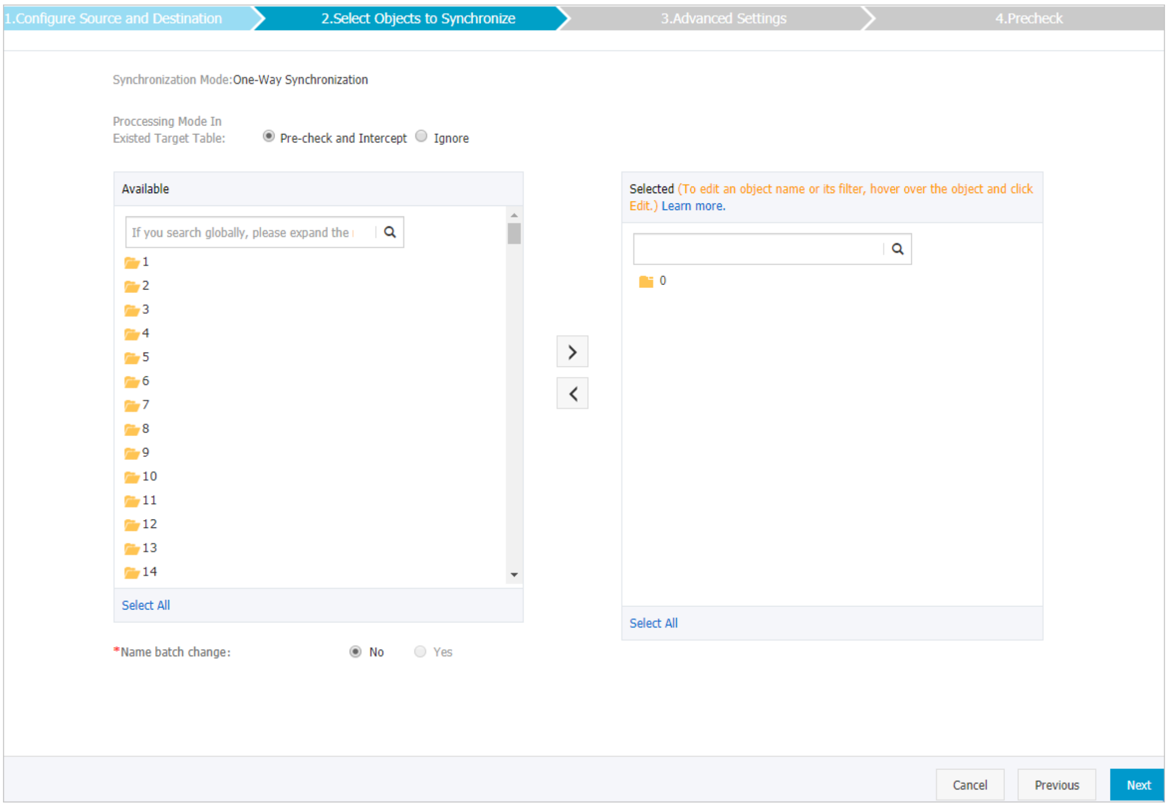
Database Password: ••••••••••••••••

Section	Parameter	Description
None	Synchronization Task Name	DTS generates a task name. We recommend that you specify an informative name to identify the task. 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. The value of this parameter cannot be changed.
	ECS Instance ID	Select the ID of the Elastic Compute Service (ECS) instance that hosts the master node of the codis-group. **Note** DTS synchronizes each codis-group of the Codis cluster by using a data synchronization task until the entire cluster is synchronized. In this step, enter the ECS instance ID of the master node for codis-group 1. When you configure the data synchronization task for codis-group 2, enter the ECS instance ID of the master node for codis-group 2. You can configure data synchronization tasks for all codis-groups by following the procedure described in this topic.
	Database Type	The value of this parameter is set to **Redis**.

Section	Parameter	Description
	Instance Mode	Select **Standalone**. **Note** You must select **Standalone** for this parameter because data from a Codis cluster cannot be synchronized at a time. DTS synchronizes each codis-group of the cluster in a data synchronization task until all codis-groups are synchronized.
	Port Number	Enter the service port number of the master node in the codis-group.
	Database Password	The database password of the master node. **Note** This parameter is optional and can be left blank if no database password is set.
Destination Instance Details	Instance Type	Select **Redis Instance**.
	Instance Region	The region of the destination cluster. The region is the same as the destination region that you selected on the buy page. The value of this parameter cannot be changed.
	Instance ID	Select the ID of the destination ApsaraDB for Redis instance.
	Database Password	Enter the database password of the ApsaraDB for Redis instance. **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

7.

8. Select the processing mode of conflicting tables and the objects to be synchronized.



1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Mode:One-Way Synchronization

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

Available

If you search globally, please expand the

1
2
3
4
5
6
7
8
9
10
11
12
13
14

Select All

Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.

0

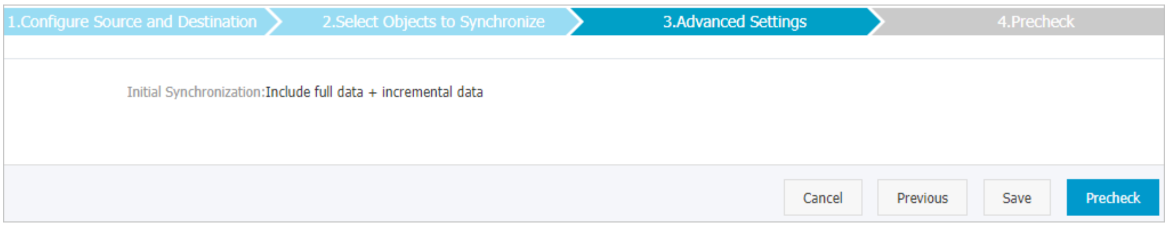
Select All

*Name batch change: ● No ○ Yes

Parameter	Description
Select the processing mode of conflicting tables	DTS synchronizes each codis-group of the Codis cluster in a data synchronization task until the whole cluster is synchronized. When you configure data synchronization for codis-group 1, select **Pre-check and Intercept** if the ApsaraDB for Redis instance has no data. When you configure data synchronization for codis-groups 2 to N, select **Ignore**. Otherwise, errors may occur during data synchronization. Note ◦ **Pre-check and Intercept**: checks whether the destination database is empty. If the destination database is empty, the precheck is passed. If the database is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ◦ **Ignore**: skips the precheck for empty destination databases and continues with data synchronization. If the keys in the destination database are the same as those in the source database during data synchronization, the keys in the source database overwrite those in the destination database.

Parameter	Description
Select the objects to be synchronized	◦ Select one or more databases from the **Available** section and click > to move the databases to the **Selected** section. ◦ You can select only databases as the objects to be synchronized. You cannot select keys as the objects to be synchronized.
Rename Databases and Tables	In this scenario, you cannot rename objects.
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 initial synchronization.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Previous
Save
Precheck

Initial Synchronization:Include full data + incremental data

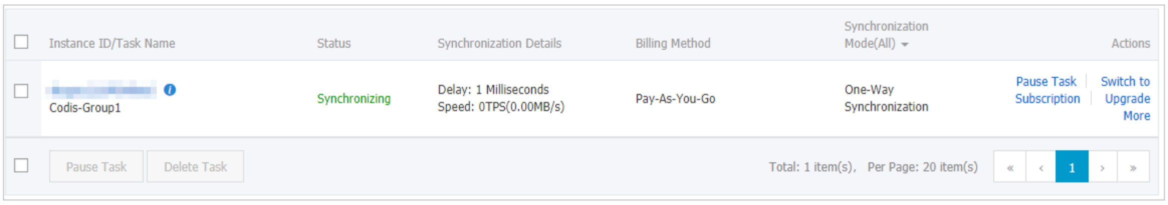
> ? **Note** The value is set to **Include full data + incremental data**. DTS synchronizes historical data from the source Codis cluster to the destination Redis databases and then synchronizes incremental data.

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: **The precheck is passed.** Then, the data synchronization task starts.

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



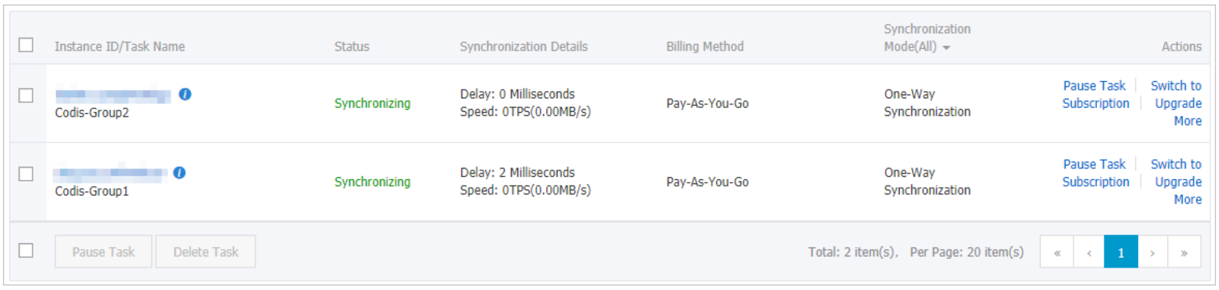
□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All) ▾	Actions
□	[redacted] ⓘ Codis-Group1	Synchronizing	Delay: 1 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

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

14. Create and configure a data synchronization task for the other codis-group by repeating Steps 1 to 13.

Result

In this topic, the Codis cluster consists of two codis-groups. You must create two data synchronization tasks. The following figure shows that the initial synchronization is complete for both tasks and both tasks are in the **Synchronizing** state.

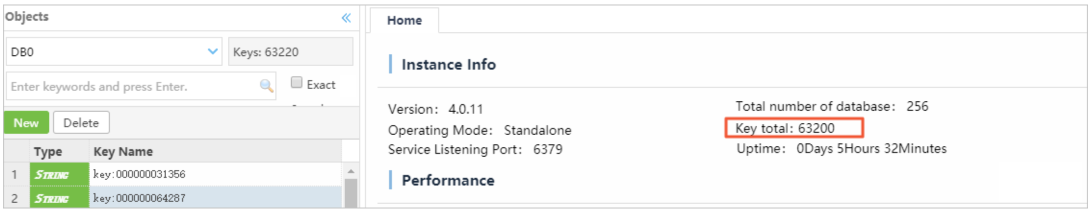


□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
□	[redacted] Codis-Group2	Synchronizing	Delay: 0 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More
□	[redacted] Codis-Group1	Synchronizing	Delay: 2 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

□ Pause Task Delete Task Total: 2 item(s), Per Page: 20 item(s) « ‹ 1 › »

In this topic, databases DB0 and DB1 are synchronized. You can use Data Management (DMS) to log on to the ApsaraDB for Redis instance and check the total number of keys in the ApsaraDB for Redis instance. The total number of keys is the same as that in the source Codis cluster. For more information about how to use DMS to log on to an ApsaraDB for Redis instance, see Use DMS to log on to an ApsaraDB for Redis instance.

ApsaraDB for Redis instance



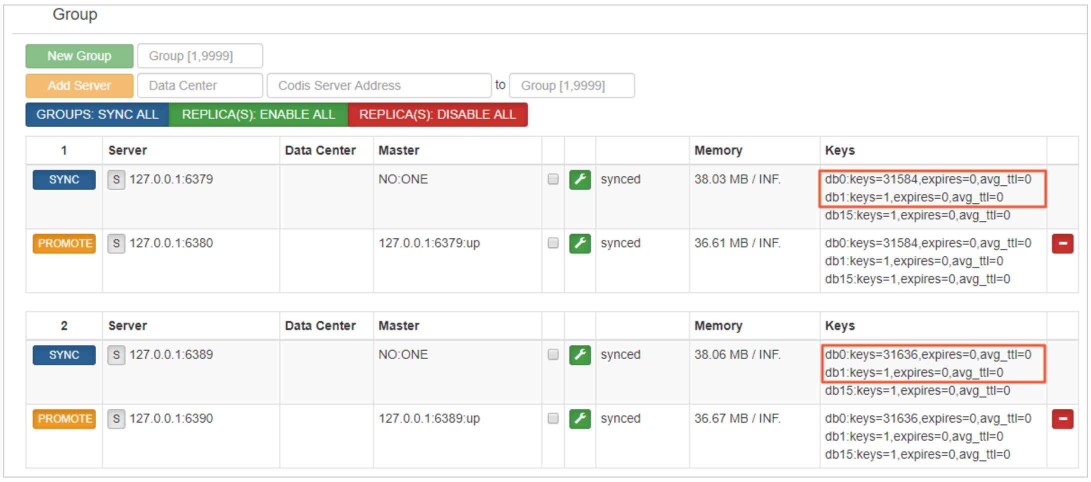
	Type	Key Name
1	STRING	key:000000031356
2	STRING	key:000000064287

Instance Info

Version: 4.0.11
Operating Mode: Standalone
Service Listening Port: 6379
Total number of database: 256
Key total: 63200
Uptime: 0Days 5Hours 32Minutes

Performance

Source Codis cluster



1	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6379		NO:ONE	□		synced	38.03 MB / INF.	db0:keys=31584,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6380		127.0.0.1:6379:up	□		synced	36.61 MB / INF.	db0:keys=31584,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	−

2	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6389		NO:ONE	□		synced	38.06 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6390		127.0.0.1:6389:up	□		synced	36.67 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	−

1.2.6. Synchronize data from a Twemproxy Redis cluster hosted on ECS to an ApsaraDB for Redis instance

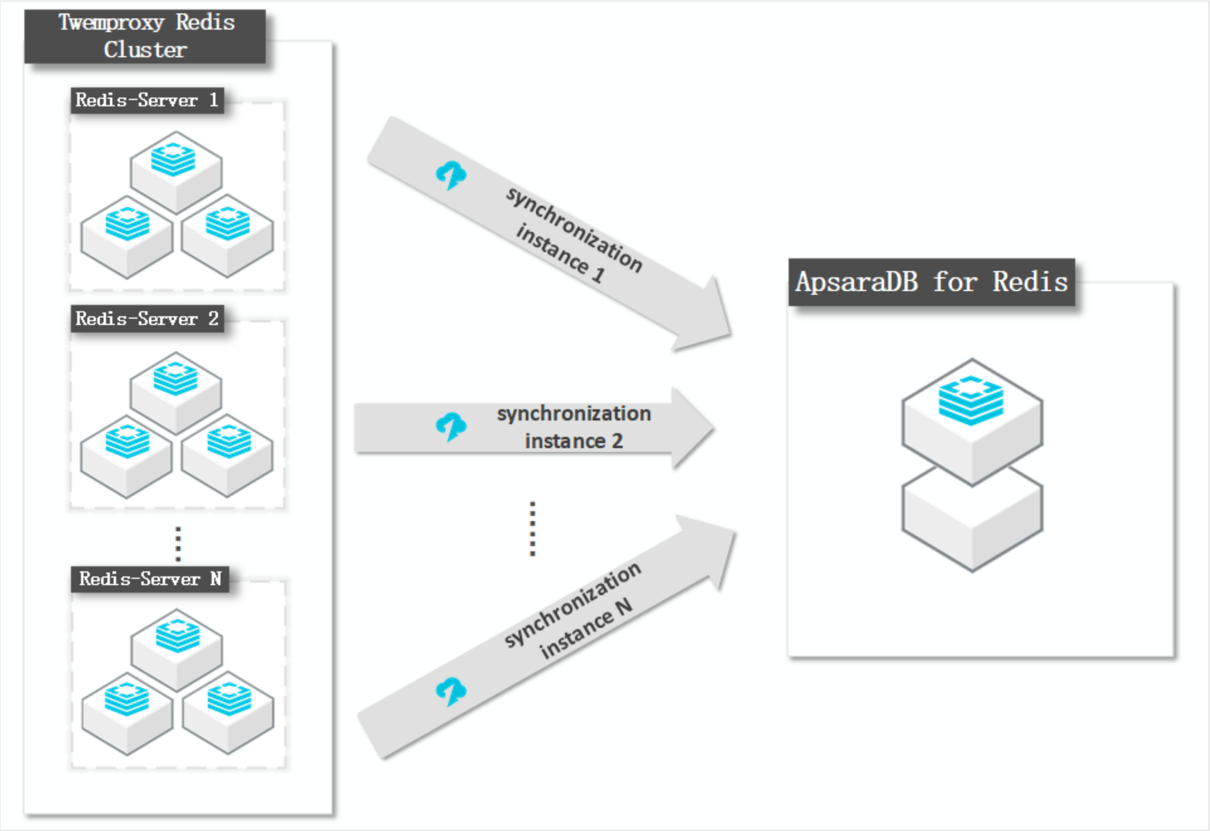
ApsaraDB for Redis is a database service compatible with the open source Redis protocol and provides hybrid storage of memory and hard disks. Based on reliable hot standby architecture and scalable cluster architecture, ApsaraDB for Redis is suitable for scenarios that require flexible configuration changes, high throughput, and low latency. This topic describes how to synchronize data from a Twemproxy Redis cluster to an ApsaraDB for Redis instance by using Data Transmission Service (DTS).

Prerequisites

- An ApsaraDB for Redis instance is created. For more information, see Create an ApsaraDB for Redis instance.
- The available storage space of the ApsaraDB for Redis instance is larger than the total size of data stored in the Twemproxy Redis cluster.
- All master nodes in the Twemproxy Redis cluster support the `PSYNC` command.

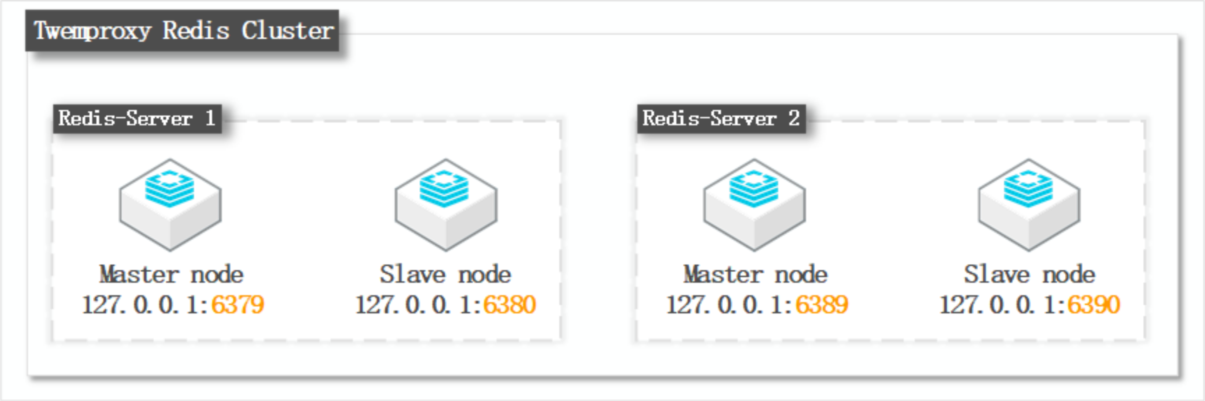
How DTS synchronizes data from a Twemproxy Redis cluster

A Twemproxy Redis cluster consists of multiple Redis-Servers. DTS synchronizes data of each Redis-Server in a data synchronization task until the whole cluster is synchronized.



Architecture of the Twemproxy Redis cluster

In this topic, the Twemproxy Redis cluster consists of two Redis-Servers. Each Redis-Server runs in a master-replica architecture. The following figure shows the architecture of the cluster.



Precautions

- DTS uses the resources of the source and destination instances during initial full data synchronization. This may increase the loads of the database servers. If you migrate a large amount of data or if the server specifications cannot meet your requirements, database services may become unavailable. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.
- If the `bind` parameter is configured in the *redis.conf* file of the source database, you must set the value of this parameter to the internal IP address of the ECS instance. The setting ensures that DTS can connect to the source database.
- To ensure the stability of data synchronization, we recommend that you increase the value of the *repl-backlog-size* parameter in the `redis.conf` file of the source Codis cluster.
- To ensure the synchronization quality, DTS adds the following key to the source Codis cluster: `DTS_REDIS_TIMESTAMP_HEARTBEAT` . This key is used to record the time when data is synchronized to the destination instance.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command in the source Codis cluster. Otherwise, data may become inconsistent between the source and destination databases.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction` , data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

  > **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- The database version of the destination ApsaraDB for Redis instance must be 2.8, 4.0, or 5.0. The version of the destination database must be the same as or later than the version of the source database. If you synchronize data between different versions of Redis databases, make sure that the versions of the source and destination databases are compatible with each other. For example, you can create a destination pay-as-you-go ApsaraDB for Redis instance to verify the compatibility between the source and destination databases. Then, you can release the instance or change the billing method of the instance to subscription.

Supported synchronization topologies

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

For more information, see Synchronization topologies.

Commands that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT

- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, PSETEX, and PUBLISH
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE

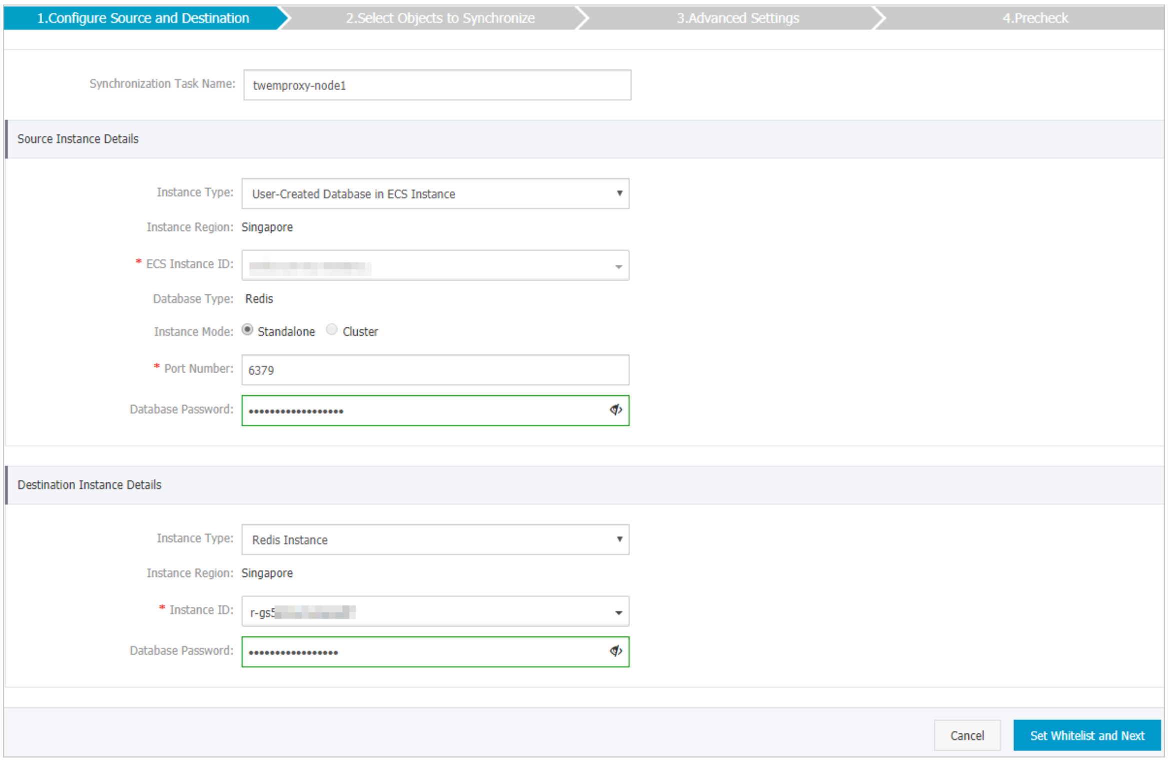
> Note
>
> - PUBLISH operations cannot be synchronized.
> - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
> - When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Procedure

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

   > Note On the buy page, set Source Instance to **Redis**, set Target Instance to **Redis**, 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.



1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Task Name: twemproxy-node1

Source Instance Details

Instance Type: User-Created Database in ECS Instance

Instance Region: Singapore

* ECS Instance ID: [redacted]

Database Type: Redis

Instance Mode: ● Standalone ○ Cluster

* Port Number: 6379

Database Password: ••••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs5[redacted]

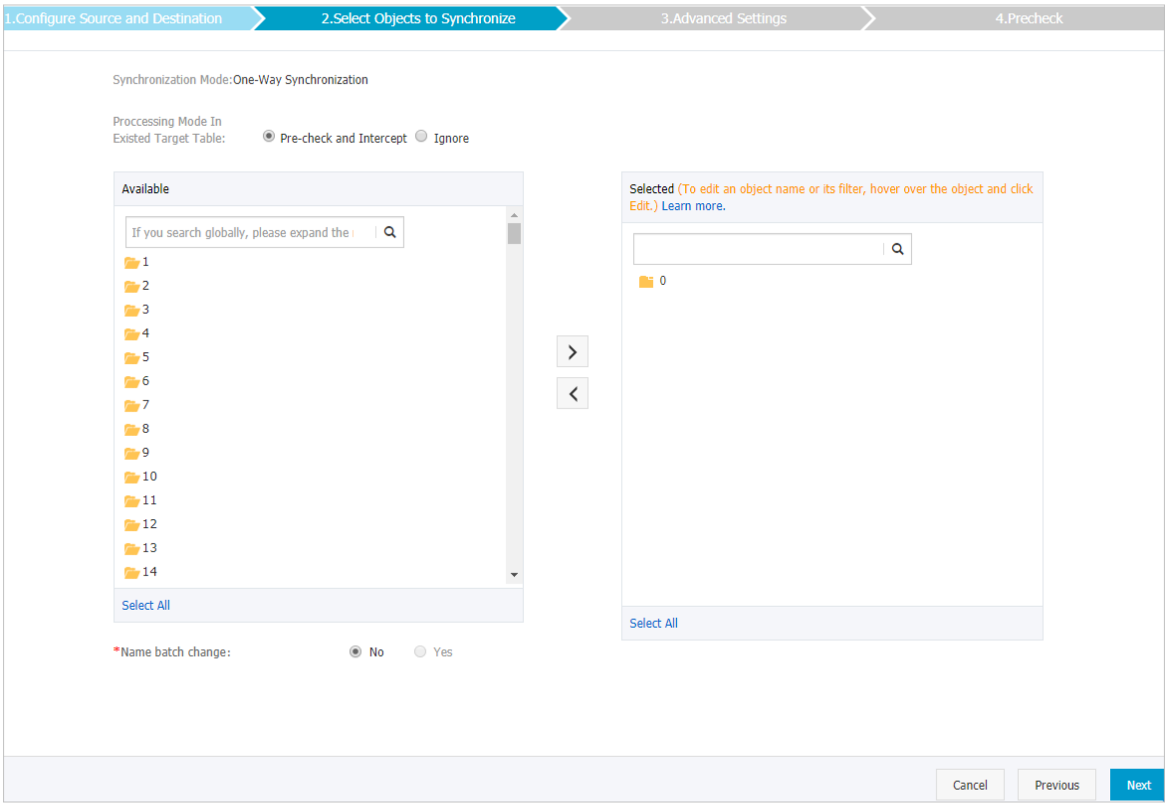
Database Password: •••••••••••••••••

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 Elastic Compute Service (ECS) instance that hosts the master node of the Redis-Server. **Note** DTS synchronizes each Redis-Server of the Twemproxy Redis cluster by using a data synchronization task until the whole cluster is synchronized. In this step, enter the ECS instance ID for the master node of Redis-Server 1. When you configure the data synchronization task for Redis-Server 2, enter the ECS instance ID for the master node of Redis-Server 2. You can configure data synchronization tasks for all Redis-Servers by following the procedure described in this topic.
	Database Type	The value of this parameter is set to **Redis**.

Section	Parameter	Description
	Instance Mode	Select **Standalone**. ⓘ **Note** You must select **Standalone** for this parameter because data from a Twemproxy Redis cluster cannot be synchronized at a time. DTS synchronizes data of each Redis-Server of the cluster in a data synchronization task until all Redis-Servers are synchronized.
	Port Number	Enter the service port number of the master node in the Redis-Server.
	Database Password	The database password of the master node. ⓘ **Note** This parameter is optional and can be left blank if no database password is set.
Destination Instance Details	Instance Type	Select **Redis 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 ApsaraDB for Redis instance.
	Database Password	Enter the database password of the ApsaraDB for Redis instance. ⓘ **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

7.

8. Select the processing mode of conflicting tables, and the objects to be synchronized.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Mode:One-Way Synchronization

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

Available

If you search globally, please expand the

1
2
3
4
5
6
7
8
9
10
11
12
13
14

Select All

Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.

0

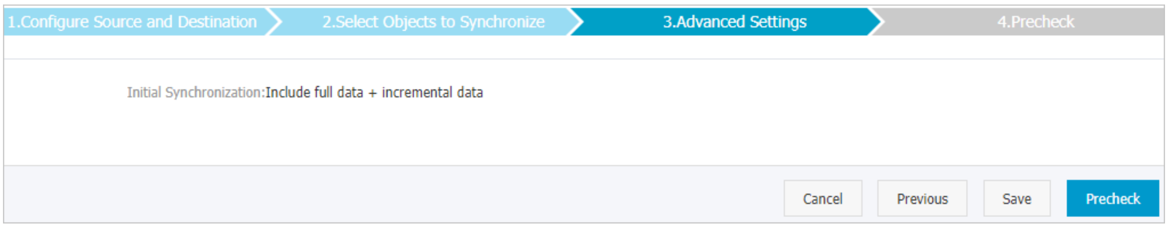
Select All

*Name batch change: ● No ○ Yes

Setting	Description
Select the processing mode of conflicting tables	DTS synchronizes each Redis-Server of the Twemproxy Redis cluster in a data synchronization task until the whole cluster is synchronized. When you configure data synchronization for Redis-Server 1, if the ApsaraDB for Redis instance has no data, select **Pre-check and Intercept**. When you configure data synchronization for Redis-Server 2 to N, select **Ignore**. Otherwise, errors may occur during data synchronization. Note ◦ **Pre-check and Intercept**: checks whether the destination database is empty. If the destination database is empty, the precheck is passed. If the database is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ◦ **Ignore**: skips the precheck for empty destination databases and continues with data synchronization. If the keys in the destination database are the same as those in the source database during data synchronization, the keys in the source database overwrite those in the destination database.
Select the objects to be synchronized	◦ Select one or more databases from the **Available** section and click > to move the databases to the **Selected** section. ◦ You can select only databases as the objects to be synchronized. You cannot select keys as the objects 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. Select the initial synchronization types.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Previous
Save
Precheck

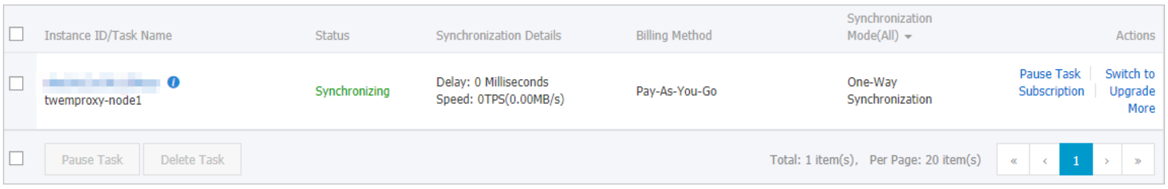
Initial Synchronization:Include full data + incremental data

Note The value is set to **Include full data + incremental data**. DTS synchronizes historical data from the source Twemproxy Redis cluster to the destination Redis database and then synchronizes incremental data.

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: **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.



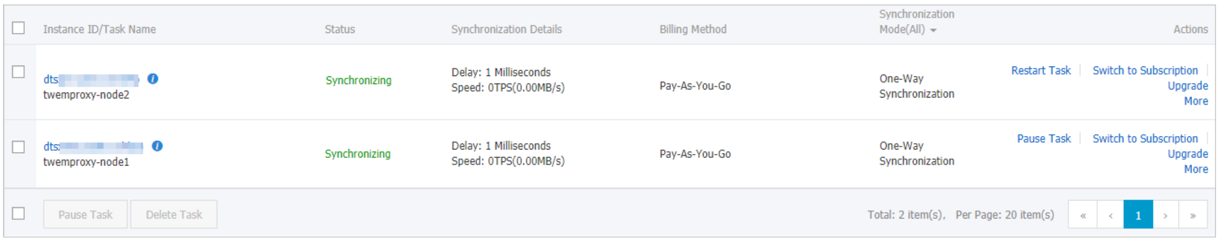
□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
□	[redacted] ⓘ twemproxy-node1	Synchronizing	Delay: 0 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

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

14. Create and configure a data synchronization task for every other Redis-Server by repeating Steps 1 to 13.

Result

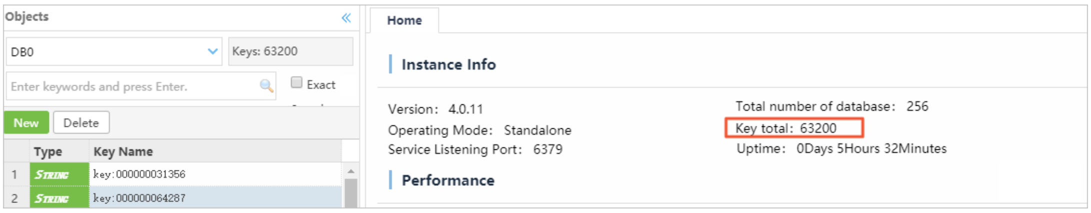
In this topic, the Twemproxy Redis cluster consists of two Redis-Servers. You must create two data synchronization tasks. The following figure shows that the initial synchronization is complete for both tasks and both tasks are in the **Synchronizing** state.



□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
□	dts[redacted] ⓘ twemproxy-node2	Synchronizing	Delay: 1 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Restart Task | Switch to Subscription | Upgrade More
□	dts[redacted] ⓘ twemproxy-node1	Synchronizing	Delay: 1 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

In this topic, the database DB0 is synchronized. You can use Data Management (DMS) to log on to the destination ApsaraDB for Redis instance and check the total number of keys in the ApsaraDB for Redis instance. The total number of keys is the same as that in the Twemproxy Redis cluster.

ApsaraDB for Redis instance



DB0 Keys: 63200

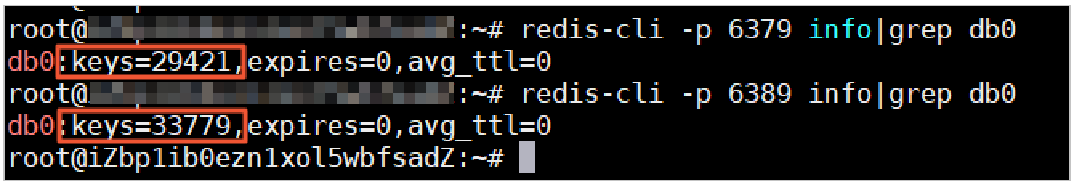
Enter keywords and press Enter. □ Exact

	Type	Key Name
1	STRING	key:000000031356
2	STRING	key:000000064287

Version: 4.0.11
Operating Mode: Standalone
Service Listening Port: 6379

Total number of database: 256
Key total: 63200
Uptime: 0Days 5Hours 32Minutes

Source Twemproxy Redis cluster



```
root@[redacted]:~# redis-cli -p 6379 info|grep db0
db0:keys=29421,expires=0,avg_ttl=0
root@[redacted]:~# redis-cli -p 6389 info|grep db0
db0:keys=33779,expires=0,avg_ttl=0
root@iZbp1ib0ezn1xol5wbfsadZ:~# 
```

1.2.7. Use AOF files to migrate data

The redis-cli tool allows you to use append-only files (AOFs) to migrate data from self-managed Redis databases to ApsaraDB for Redis instances.

The redis-cli tool is a command-line tool of Redis. ApsaraDB for Redis allows you to use the redis-cli tool to seamlessly migrate existing data in Redis databases to ApsaraDB for Redis instances. You can also use Data Transmission Service (DTS) to migrate data. For more information, see Migrate data from a self-managed Redis database to an ApsaraDB for Redis instance.

Precautions

- ApsaraDB for Redis can be accessed only from the Alibaba Cloud network. Therefore, you can use this migration solution only with Alibaba Cloud Elastic Compute Service (ECS) instances. If your self-managed Redis databases are not deployed on an Alibaba Cloud ECS instance, you must copy an existing AOF file to an ECS instance before you migrate data.
- The redis-cli tool is a command-line tool of Redis. If you cannot use the redis-cli tool on an ECS instance, you must download and install Redis.

Procedure

For a self-managed Redis instance that is hosted on an ECS instance, perform the following operations:

1. Enable the AOF feature for the existing self-managed Redis instance. You can skip this step if the AOF feature is enabled. Sample command:

   ```
   redis-cli -h old_instance_ip -p old_instance_port config set appendonly yes
   ```

2. Use an AOF file to import data from the existing Redis instance to a new ApsaraDB for Redis instance. Sample command:

   ```
   redis-cli -h aliyun_redis_instance_ip -p 6379 -a password --pipe < appendonly.aof
   ```

   > **Notice** If your self-managed Redis instance does not need the AOF feature enabled all the time, you can disable the feature after data is imported. Sample command:
   >
   > ```
   > redis-cli -h old_instance_ip -p old_instance_port config set appendonly no
   > ```

FAQ

- Q: What do I do if the `ERR Protocol error: too big inline request` error message is returned when I import data?

A: Run the `config get aof-use-rdb-preamble` command to check whether the `aof-use-rdb-preamble` instance parameter is set to `yes`. If the parameter is set to `yes`, run the `config set aof-use-rdb-preamble no` command to set the parameter to `no` and then create another AOF file to re-import data.

> **Note** If the `aof-use-rdb-preamble` parameter is set to `yes`, some data of an AOF file is compressed and cannot be parsed and imported by using redis-cli commands. For more information, see Redis persistence.

1.3. Migrate data between ApsaraDB for Redis instances

1.3.1. Configure unidirectional data migration between ApsaraDB for Redis instances

This topic describes how to use Data Transmission Service (DTS) to configure unidirectional data migration between ApsaraDB for Redis instances. You can also follow the procedure to configure data migration tasks for self-managed Redis databases. DTS supports full data migration and incremental data migration. When you configure a data migration task, you can select both of the supported migration types to ensure service continuity.

Prerequisites

- The source and destination ApsaraDB for Redis instances are created. For more information, see Step 1: Create an ApsaraDB for Redis instance.
- For self-managed Redis databases, the database version is 2.8, 3.0, 3.2, 4.0, 5.0, or 6.0.
- The engine version is 4.0 for Community Edition instances or 5.0 for Community Edition or Enhanced Edition (Tair) instances.
- The amount of available storage space of the destination ApsaraDB for Redis instance is larger than the total size of the data stored in the source ApsaraDB for Redis instance.

Supported source and destination databases

You can use DTS to migrate data between the following types of databases.

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

Precautions

Category	Description
Limits on the source database	• Bandwidth requirements: The server to which the source database belongs must have sufficient egress bandwidth. Otherwise, the data migration speed is affected. • If you perform only incremental data migration, you must enable the data logging feature. In addition, the append-only file (AOF) logs of the source database must be stored for more than 24 hours. • If you perform both full data migration and incremental data migration, you must enable the data logging feature. In addition, the AOF logs of the source database must be stored for at least seven days. After full data migration is completed, you can set the retention period to more than 24 hours. Otherwise, DTS may fail to obtain the AOF logs and the task may fail. In extreme cases, data may be inconsistent or lost. Make sure that you set the retention period of AOF logs based on the preceding requirements. Otherwise, the Service Level Agreement (SLA) of DTS does not guarantee service reliability and performance. • Limits on operations: If you perform only full data migration, do not write data to the source database during data migration. Otherwise, data will be inconsistent between the source and destination databases. To ensure data consistency, we recommend that you select full data migration and incremental data migration as the migration types. • Limits on migrating a standalone ApsaraDB for Redis instance to an ApsaraDB for Redis cluster instance: Each command can be run only on a single slot in an ApsaraDB for Redis cluster instance. If you perform operations on multiple keys in the source database and the keys belong to different slots, an error occurs. `CROSSSLOT Keys in request don't hash to the same slot` We recommend that you perform operations on only one key during data migration. Otherwise, the migration task will be interrupted.

Category	Description
Other limits	- ApsaraDB for Redis Enhanced Edition instances (storage-optimized instances) cannot be used as the source and destination databases. - If the data eviction policy (`maxmemory-policy`) of the destination database is not set to `noeviction` , data may become inconsistent between the source and destination databases. - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination database. During incremental data migration, the destination database does not explicitly return the execution results of Lua scripts. - When you run the `PSYNC` or `SYNC` command to transfer data of the LIST type, DTS does not perform the `flush` operation on the existing data. Therefore, the destination database may contain duplicate data records. - If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database. **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases. - To ensure compatibility, the version of the destination Redis database must be the same as or later than the version of the source Redis database. **Note** If the version of the destination database is earlier than the version of the source database, database compatibility issues may occur. - 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. During full data migration, DTS uses read and write resources of the source and destination databases. This may increase the loads of the database servers. - 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. - DTS attempts to resume data migration tasks that failed within the last seven days. Before you switch workloads to the destination instance, stop or release the data migration task. You can also run the `REVOKE` command to revoke the write permissions from the accounts that are used by DTS to access the destination instance. Otherwise, the data in the source database will overwrite the data in the destination instance after the task is resumed. - If both the source and destination databases are ApsaraDB for Redis Community Edition or Enhanced Edition, take note of the following limits: If the database specifications are changed (for example, the specifications are upgraded or the port number is changed), DTS cannot obtain continuous log data and correct connection information. As a result, the data migration task is interrupted. To ensure data consistency, we recommend that you delete the data in the destination database and reconfigure the data migration task after the database specifications are changed.

Category	Description
Special cases	If the source database is a self-managed Redis database, take note of the following limits: • If you perform a primary/secondary switchover on the source database when the data migration task is running, the task fails. • DTS calculates migration latency based on the timestamp of the latest migrated data in the destination database and the current timestamp in the source database. If no data manipulation language (DML) operation is performed on the source database for a long time, the migration latency may be inaccurate. If the latency of the migration task is too high, you can perform a DML operation on the source database to update the latency. **Note** If you select an entire database as the object to be migrated, you can create a heartbeat table. The heartbeat table is updated or receives data every second.

Migration types

Migration type	Description
Full data migration	
Incremental data migration	

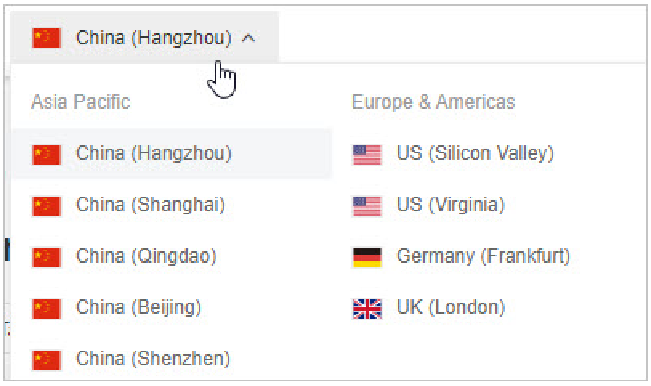
Commands that can be incrementally migrated

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- FLUSHALL and FLUSHDB
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, PSETEX, and PUBLISH
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE

Procedure

1.
2. In the upper-left corner of the page, select the region where the instance to be migrated is

located.

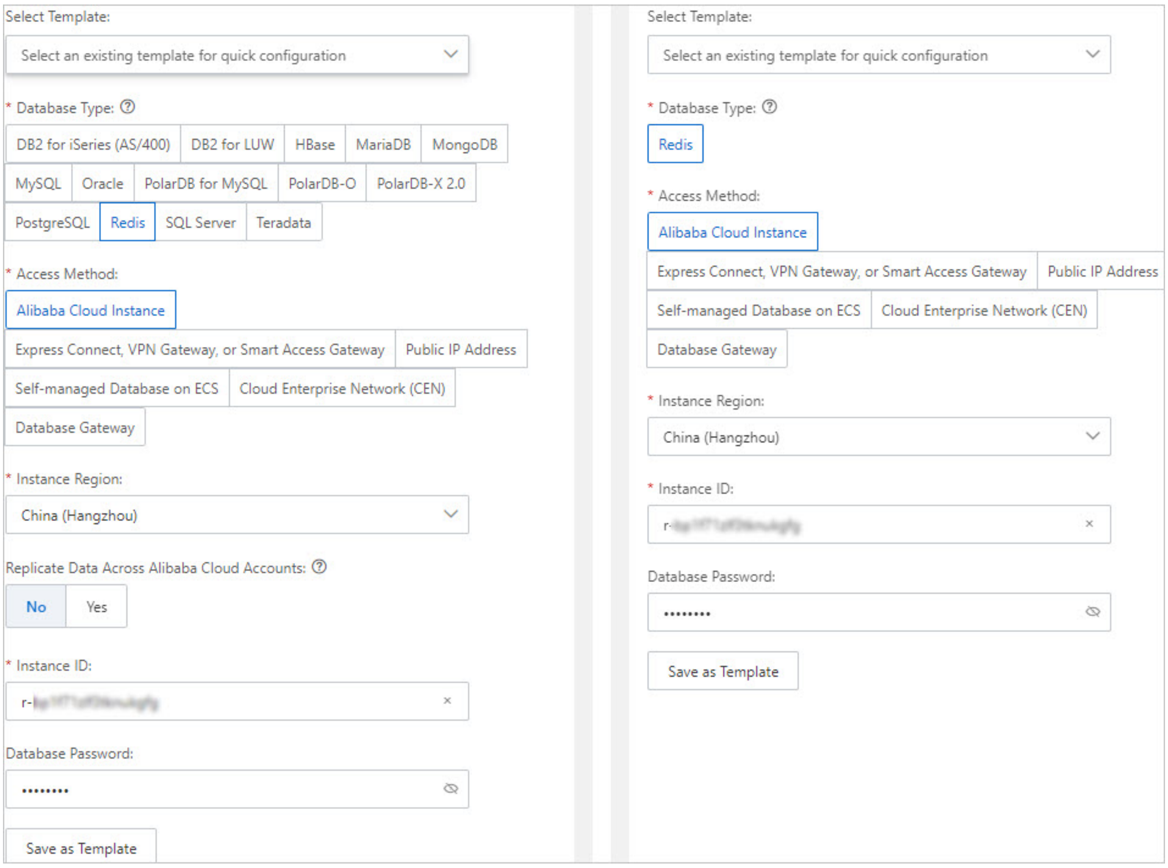


China (Hangzhou)

Asia Pacific
- China (Hangzhou)
- China (Shanghai)
- China (Qingdao)
- China (Beijing)
- China (Shenzhen)

Europe & Americas
- US (Silicon Valley)
- US (Virginia)
- Germany (Frankfurt)
- UK (London)

3. Warning



Select Template: Select an existing template for quick configuration

* Database Type: DB2 for iSeries (AS/400) | DB2 for LUW | HBase | MariaDB | MongoDB | MySQL | Oracle | PolarDB for MySQL | PolarDB-O | PolarDB-X 2.0 | PostgreSQL | Redis | SQL Server | Teradata

* Access Method: Alibaba Cloud Instance | Express Connect, VPN Gateway, or Smart Access Gateway | Public IP Address | Self-managed Database on ECS | Cloud Enterprise Network (CEN) | Database Gateway

* Instance Region: China (Hangzhou)

Replicate Data Across Alibaba Cloud Accounts: No | Yes

* Instance ID: r-[redacted]

Database Password: ••••••••

Save as Template

Select Template: Select an existing template for quick configuration

* Database Type: Redis

* Access Method: Alibaba Cloud Instance | Express Connect, VPN Gateway, or Smart Access Gateway | Public IP Address | Self-managed Database on ECS | Cloud Enterprise Network (CEN) | Database Gateway

* Instance Region: China (Hangzhou)

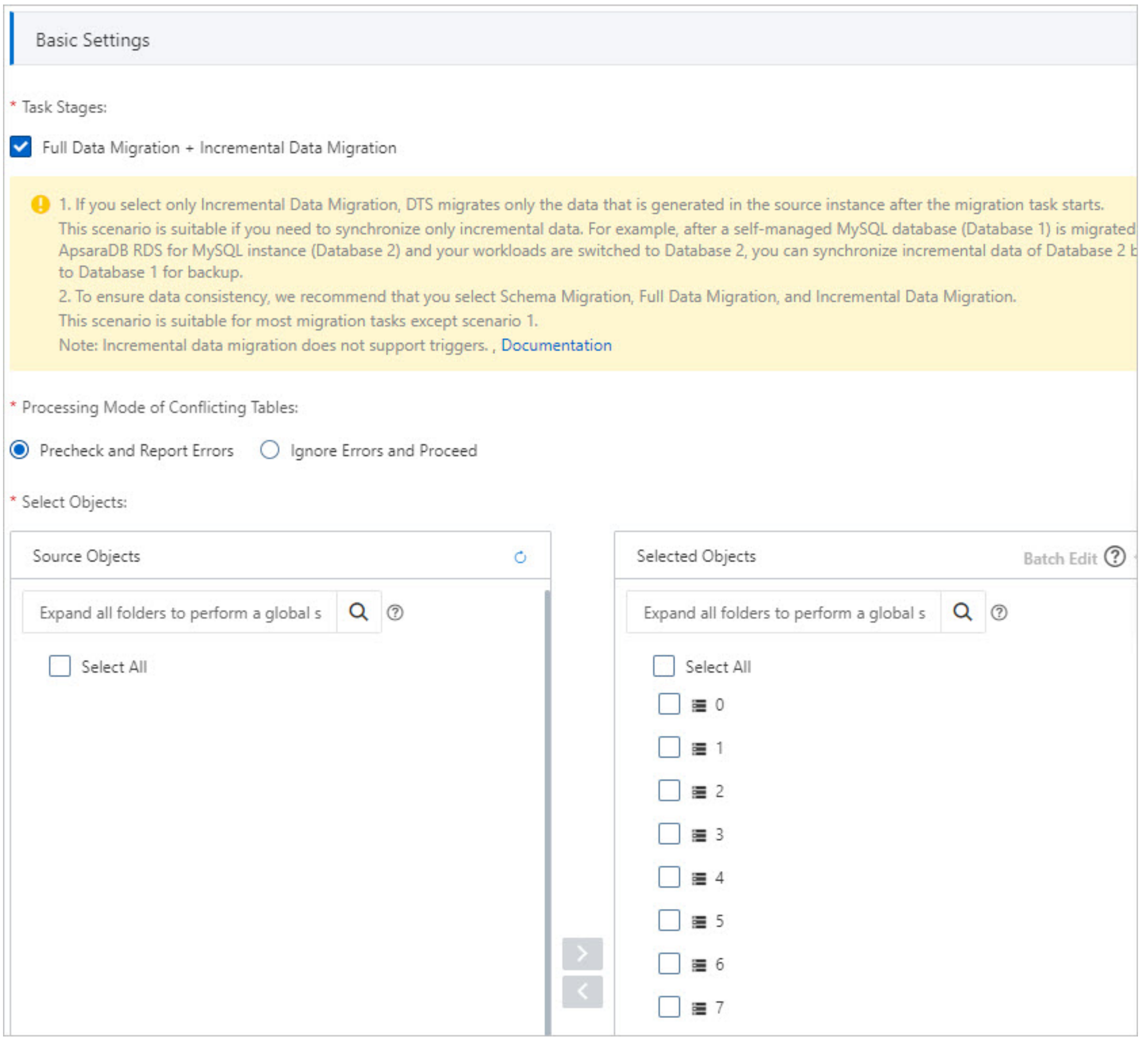
* Instance ID: r-[redacted]

Database Password: ••••••••

Save as Template

4.

5. ○ Basic Settings



Basic Settings

* Task Stages:

☑ Full Data Migration + Incremental Data Migration

1. If you select only Incremental Data Migration, DTS migrates only the data that is generated in the source instance after the migration task starts. This scenario is suitable if you need to synchronize only incremental data. For example, after a self-managed MySQL database (Database 1) is migrated ApsaraDB RDS for MySQL instance (Database 2) and your workloads are switched to Database 2, you can synchronize incremental data of Database 2 b to Database 1 for backup.
2. To ensure data consistency, we recommend that you select Schema Migration, Full Data Migration, and Incremental Data Migration. This scenario is suitable for most migration tasks except scenario 1.

Note: Incremental data migration does not support triggers. , Documentation

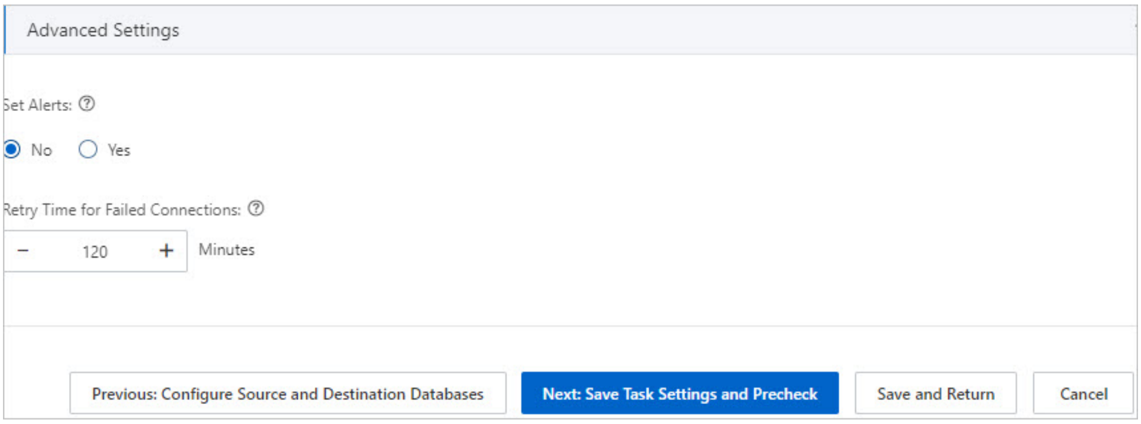
* Processing Mode of Conflicting Tables:

● Precheck and Report Errors ○ Ignore Errors and Proceed

* Select Objects:

Parameter	Description
	For more information, see Migration types.
	Note You can select only databases as the objects to be migrated. You cannot select keys as the objects to be migrated.
	In this scenario, you cannot rename objects.
Select the commands to be migrated	In the **Selected Objects** section, right-click an object. In the dialog box that appears, select the commands that you want to migrate. For more information, see Commands that can be incrementally migrated.

- **Advanced Settings**



Advanced Settings

Set Alerts: ● No ○ Yes

Retry Time for Failed Connections: 120 Minutes

Previous: Configure Source and Destination Databases
Next: Save Task Settings and Precheck
Save and Return
Cancel

Parameter	Description

6.
7.
8.
9.
10.

1.4. Migrate data from a third-party database to ApsaraDB for Redis

1.4.1. Migrate data from an Amazon ElastiCache for Redis cluster to an ApsaraDB for Redis instance

This topic describes how to migrate data from an AWS ElastiCache for Redis instance to an Alibaba Cloud ApsaraDB for Redis instance.

Prerequisites

The ApsaraDB for Redis instance runs the ApsaraDB for Redis Community Edition in Redis 4.0 or 5.0 or the ApsaraDB for Redis Enhanced Edition (Tair) in Redis 5.0. If no ApsaraDB for Redis instances are created, you must create an ApsaraDB for Redis instance. For more information, see Step 1: Create an ApsaraDB for Redis instance.

Introduction to redis-shake

Redis-shake is an open source tool that is developed by Alibaba Cloud. Redis-shake is easy to deploy and can be used to efficiently parse (decode mode), restore (restore mode), back up (dump mode), and synchronize (sync or rump mode) the data of Redis databases. In this topic, the restore mode of redis-shake is used to restore the data of the self-managed Redis database from an RDB file to the ApsaraDB for Redis instance.

Note

- For more information about how to migrate incremental data by using redis-shake, see Use redis-shake to migrate data from a self-managed Redis database to Alibaba Cloud.
- You can also use the data synchronization feature of Data Transmission Service (DTS) to migrate data. DTS provides more features and can be used for more scenarios compared with redis-shake. For more information, see Overview.

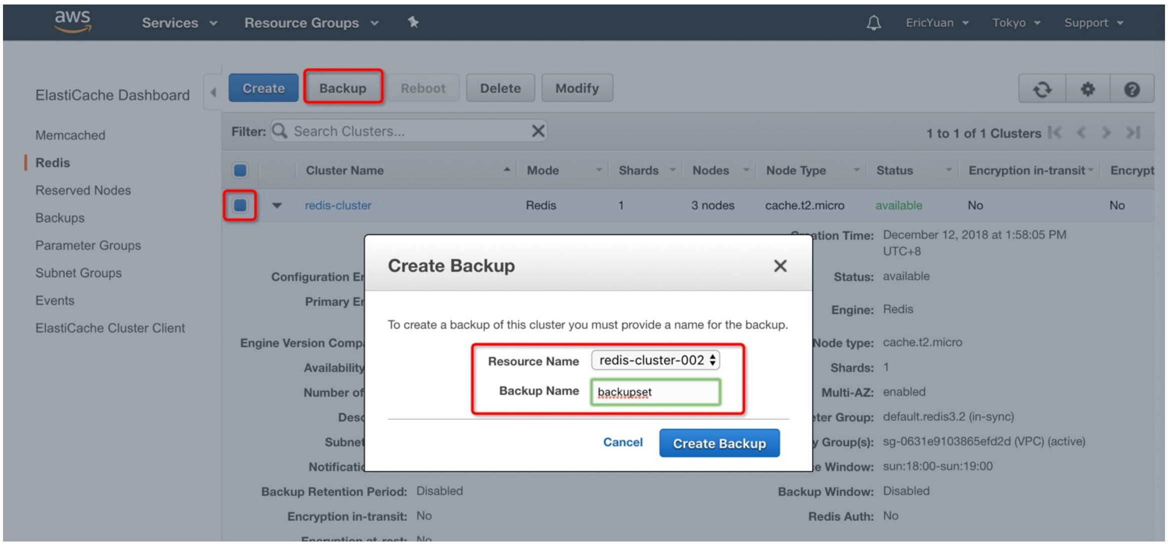
Precautions

- Before you start the migration, we recommend that you stop the write operations on the AWS ElastiCache for Redis cluster.
- Before you start the migration, we recommend that you back up the AWS ElastiCache for Redis cluster and prepare a migration schedule that minimizes downtime.
- If the data eviction policy (maxmemory-policy) of the destination database is not set to *noeviction*, data may become inconsistent between the source and destination databases. For more information about the data eviction policy, see How does ApsaraDB for Redis evict data by default?
- After you run the info command to query the keys in the destination database, you may find that the destination database contains fewer keys than the source database. This is caused by the key expiration mechanism of Redis. This situation occurs if the source database contains keys that are not deleted upon expiration.

Note The numbers of keys that do not have a validity period in the source and destination databases are the same.

Step 1: Export the data of the AWS ElastiCache for Redis cluster as an RDB file

1. Create a backup and store the generated backup file to a different AWS ElastiCache for Redis cluster. Log on to the AWS ElastiCache console. In the left-side navigation pane, click **Redis**. On the page that appears, select the **AWS ElastiCache for Redis cluster** and click **Backup**. In the dialog box that appears, set the **Resource Name** parameter to the name of the AWS ElastiCache for Redis cluster, enter the name of the backup in the **Backup Name** field, and then click **Create Backup**.



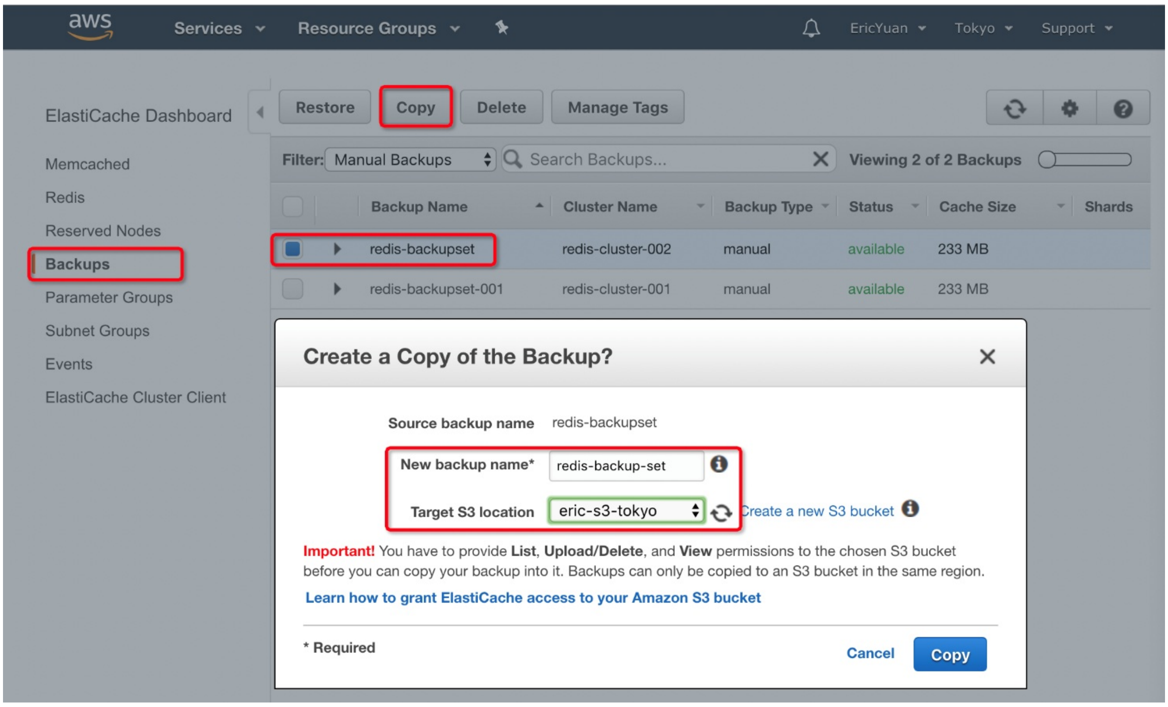
Create Backup

To create a backup of this cluster you must provide a name for the backup.

Resource Name: redis-cluster-002

Backup Name: backupset

2. Export the backup file to an AWS S3 bucket as an RDB object. In the left-side navigation pane, click **Backups**. On the page that appears, select the backup file and click **Copy** above the file list. In the dialog box that appears, configure the **New backup name** and **Target S3 location** parameters and click **Copy**.



Backup Name	Cluster Name	Backup Type	Status	Cache Size	Shards
redis-backupset	redis-cluster-002	manual	available	233 MB	
redis-backupset-001	redis-cluster-001	manual	available	233 MB	

Create a Copy of the Backup?

Source backup name redis-backupset

New backup name* redis-backup-set

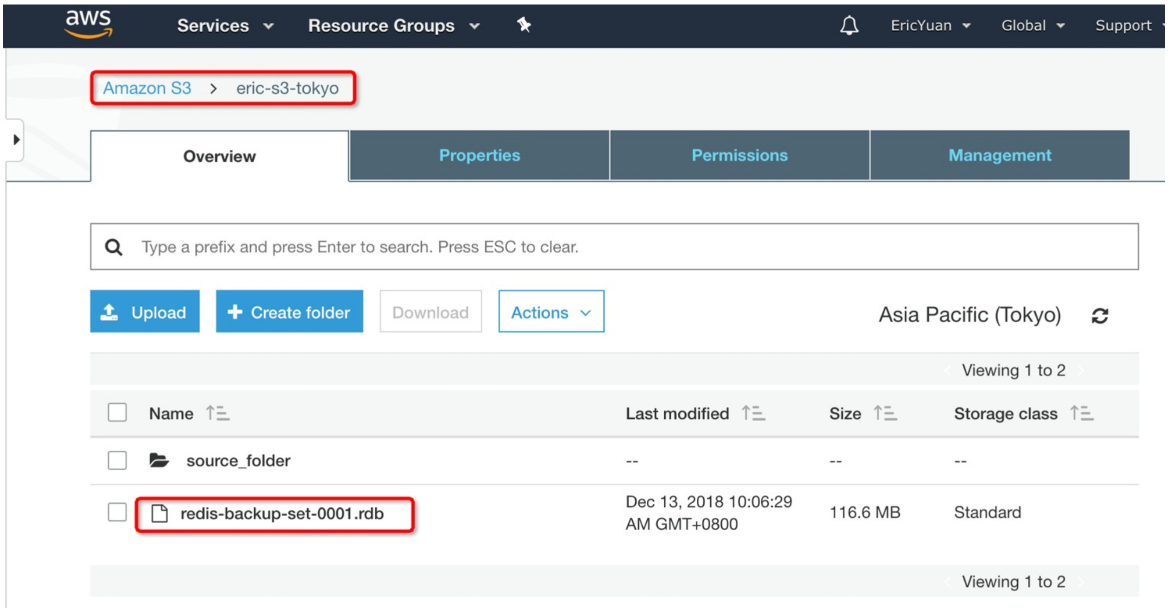
Target S3 location eric-s3-tokyo

Important! You have to provide List, Upload/Delete, and View permissions to the chosen S3 bucket before you can copy your backup into it. Backups can only be copied to an S3 bucket in the same region.

Learn how to grant ElastiCache access to your Amazon S3 bucket

* Required

3. View the RDB object in the AWS S3 bucket that you specify.



Name	Last modified	Size	Storage class
source_folder	--	--	--
redis-backup-set-0001.rdb	Dec 13, 2018 10:06:29 AM GMT+0800	116.6 MB	Standard

4. Download the RDB object from the AWS S3 bucket to your computer as an RDB file.

Step 2: Migrate the RDB file to the ApsaraDB for Redis instance by using redis-shake

1. Perform the following operations based on the installation location of redis-shake:

> Note We recommend that you install redis-shake on an Elastic Compute Service (ECS) instance. You can connect the ECS instance and the ApsaraDB for Redis instance over a virtual private cloud (VPC) to increase security and reduce network latencies.

<table>
<tr><th>Installation location of redis-shake</th><th>Operation</th></tr>
<tr><td>(Recommended) ECS instance</td><td>

i. Make sure that the ECS instance and the ApsaraDB for Redis instance are deployed in the same VPC. In this case, the same VPC ID is displayed in the Basic Information section of the instances.

> Note
> - If the instances are deployed in different VPCs, you can change the VPC to which the ECS instance belongs. For more information, see Change the VPC of an ECS instance.
> - The network types of the ECS instance and the ApsaraDB for Redis instance may be different. For example, the ECS instance belongs to the classic network and the ApsaraDB for Redis instance belongs to a VPC. For information about how to connect to an ApsaraDB for Redis instance from an ECS instance when the instances are deployed in different types of networks, see Connect an ECS instance to an ApsaraDB for Redis instance in different types of networks.

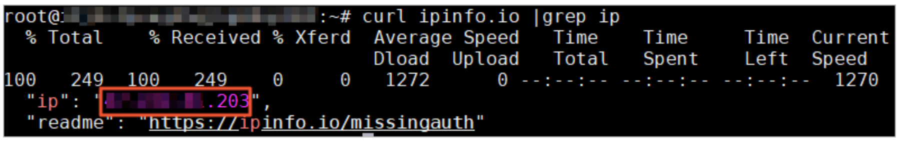
ii. Obtain the internal IP address of the ECS instance. For more information, see Network FAQ.

iii. Add the internal IP address of the ECS instance to a whitelist of the ApsaraDB for Redis instance. For more information, see Configure whitelists.

</td></tr>
<tr><td>On-premises machine</td><td>

i. By default, only internal endpoints are available for ApsaraDB for Redis instances. If you want to connect to an ApsaraDB for Redis instance over the Internet, you must apply for a public endpoint. For more information, see Apply for a public endpoint for an ApsaraDB for Redis instance.

ii. Run the **curl ipinfo.io |grep ip** command on the on-premises device to obtain its public IP address. The following figure shows a sample result.



```
root@[redacted]:~# curl ipinfo.io |grep ip
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   249  100   249    0     0   1272      0 --:--:-- --:--:-- --:--:--  1270
  "ip": "[redacted].203",
  "readme": "https://ipinfo.io/missingauth"
```

iii. Add the public IP address of the on-premises device to a whitelist of the ApsaraDB for Redis instance. For more information, see Configure whitelists.

</td></tr>
</table>

2. Install redis-shake.

    i. Log on to the host where you want to install redis-shake. The host may be an ECS instance or an on-premises machine.

ii. Run the following command to download the redis-shake file.

```
wget 'http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/attach/120287/cn_zh/
1608173646665/redis-shake-v2.0.3.tar.gz'
```

> ⓘ **Note** This example shows how to install redis-shake 2.0.3. You can also install redis-shake of other versions. For more information, see RedisShake.

iii. Run the following command to decompress the redis-shake file:

```
tar xzf redis-shake-v2.0.3.tar.gz
```

3. Start the migration on the ECS instance or the on-premises machine.

i. Download the RDB file to the ECS instance or the on-premises machine.

> ⓘ **Note** You can run the `wget <The URL from which you can download the RDB file>` command to download the RDB file to the ECS instance. You can also download the RDB file to the on-premises machine and then use MobaXterm Personal Edition to upload the RDB file from the on-premises machine to the ECS instance over SFTP.

ii. Run the following command to open and edit the redis-shake.conf file in the directory that is generated after you decompress the redis-shake software package:

```
cd redis-shake-v2.0.3/ && vim redis-shake.conf
```

> ⓘ **Note** After you run the command, the system opens an editor. You can type *a* to enter the edit mode.

Parameter description

Parameter	Required	Description	Example
source.rdb.input	Yes	The save path of the RDB file. You can specify a relative path or an absolute path. ⓘ **Note** If you need to specify more than one RDB file, separate the save paths of the RDB files with semicolons (;).	/root/tools/RedisShake/demo.rdb

Parameter	Required	Description	Example
target.address	Yes	The endpoint and port number of the ApsaraDB for Redis instance. Separate the endpoint and port number with a colon (:). For more information about how to obtain the endpoint and port number of an ApsaraDB for Redis instance, see View endpoints. ■ If you install redis-shake on an ECS instance and connect the ECS instance and the ApsaraDB for Redis instance over a VPC, obtain the internal endpoint and port number of the ApsaraDB for Redis instance. ■ If you install redis-shake on an on-premises machine and connect the on-premises machine and the ApsaraDB for Redis instance over the Internet, obtain the public endpoint and port number of the ApsaraDB for Redis instance.	r-bp1wcw2rlw76acc5k****.redis.rds.aliyuncs.com:6379
target.password_raw	Yes	The username and password of the account that has the **Read/Write** permissions on the ApsaraDB for Redis instance. Separate the username and password with a colon (:). For more information about how to create an account, see Create and manage database accounts.	testaccount:Rp829dlwa
key_exists	No	The data write policy that is applied if the keys in AWS ElastiCache for Redis cluster are the same as the keys in the ApsaraDB for Redis instance. Valid values: ■ *rewrite*: The keys in the AWS ElastiCache for Redis cluster overwrite the keys with identical names in the ApsaraDB for Redis instance. ■ *none*: Redis-shake stops running, and a message that indicates conflicting keys appears. This is the default value. ■ *ignore*: Redis-shake skips the conflicting keys, retains the data of the ApsaraDB for Redis instance, and continues to migrate the other data.	rewrite

Parameter	Required	Description	Example
parallel	No	The number of concurrent threads that are invoked by redis-shake to perform the migration. You can increase the value of this parameter to accelerate the migration. **Note** The default value is 32. The minimum value is 1. The maximum value varies based on the performance of the server on which redis-shake is installed.	32

> **Note** You do not need to configure other parameters unless otherwise specified. For more information, see the comments on each parameter in the *redis-shake.conf* file.

iii. Press Esc to exit the edit mode. Then, type *:wq* and press Enter to save the .redis-shake.conf file and exit the editor.

iv. Run the following command to start redis-shake and migrate data:

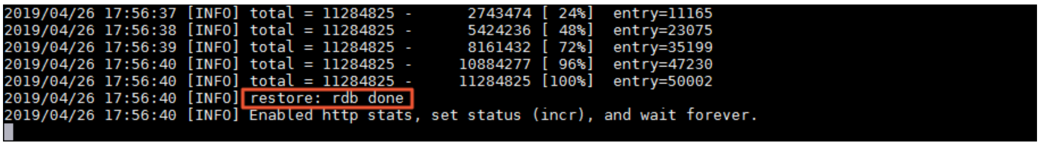
```
./redis-shake.linux -type=restore -conf=redis-shake.conf
```

Redis-shake displays the operational log on the screen.

> **Note** For more information about the causes of and solutions to possible errors, see FAQ.

4. When `restore: rdb done` appears, press Ctrl+C to stop redis-shake.

Example



```
2019/04/26 17:56:37 [INFO] total = 11284825 -      2743474 [ 24%]  entry=11165
2019/04/26 17:56:38 [INFO] total = 11284825 -      5424236 [ 48%]  entry=23075
2019/04/26 17:56:39 [INFO] total = 11284825 -      8161432 [ 72%]  entry=35199
2019/04/26 17:56:40 [INFO] total = 11284825 -     10884277 [ 96%]  entry=47230
2019/04/26 17:56:40 [INFO] total = 11284825 -     11284825 [100%]  entry=50002
2019/04/26 17:56:40 [INFO] restore: rdb done
2019/04/26 17:56:40 [INFO] Enabled http stats, set status (incr), and wait forever.
```

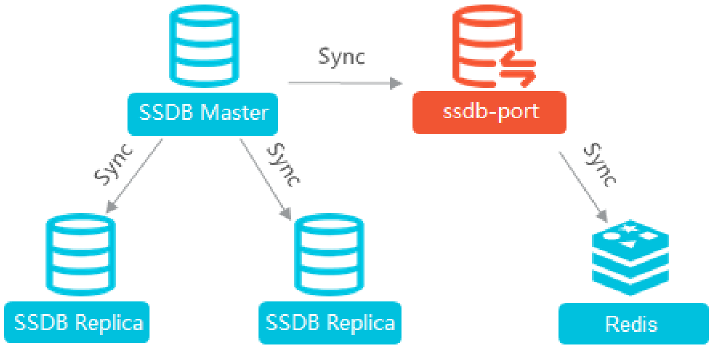
1.4.2. Migrate data from SSDB to ApsaraDB for Redis

You can use the ssdb-port tool to migrate data from SSDB to ApsaraDB for Redis.

Background

How it works

As an SSDB replica node, the ssdb-port tool synchronizes data from the SSDB master node that serves as the source database. Then, the ssdb-port tool parses and converts the data to the format supported by ApsaraDB for Redis, and sends the data to the ApsaraDB for Redis instance that is specified in the configuration file. The following figure shows the migration process.



After full synchronization is completed, incremental data in the SSDB database is synchronized to the ApsaraDB for Redis instance before ssdb-port is disconnected from the instance.

Note

- If you run a command that is not supported by the ssdb-port tool to modify data in the source SSDB database, the modified data cannot be synchronized to the ApsaraDB for Redis instance. For more information about the SSDB commands that are supported by the ssdb-port tool, see the following list.
- If you need to synchronize more commands, log on to the Connect Platform to propose a suggestion.

Commands supported by the ssdb-port tool

Notice If you run the hset or hget command to modify data, the modified data cannot be synchronized if the target keys are in Chinese. This restriction does not apply to other supported commands.

- set
- setx
- setnx
- expire
- del
- get
- incr
- qpop_front
- qpush_front
- qclear
- qtrim_front
- qtrim_back
- zset
- zdel
- zincr
- multi_zdel
- multi_zset
- hset

- hdel
- hclear
- multi_hset
- multi_hdel
- hincr

Prerequisites

- A Linux-based Elastic Compute Service (ECS) instance is created in the same Virtual Private Cloud (VPC) as the destination ApsaraDB for Redis instance, and can connect to the destination ApsaraDB for Redis instance.
- The version of the source SSDB database is 1.9.2 or later.

Procedure

1. Run the following commands in the ECS instance to download and decompress the ssdb-port.tar.gz package:

   ```
   # wget http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/attach/94155/cn_zh/1547627852086/ssdb-port.tar.gz
   # tar -xvf ssdb-port.tar.gz
   # cd ssdb-port
   ```

2. Run the following command to modify the configuration file of the ssdb-port tool based on the example:

   ```
   vi ssdb_port.conf
   ```

   The following code shows an example of the *ssdb_port.conf* file. Modify the connection information of the source SSDB database and the destination ApsaraDB for Redis instance based on the comments.

```
# SSDB server configuration for replication
# You MUST indent the code with the Tab key.
# The relative path of this file. The directory must exist.
work_dir = ./var_ssdb_port
pidfile = ./var_ssdb_port/ssdb.pid
# The connection information of the ssdb-port tool, which does not need to be modified.
server:
    ip: 127.0.0.1
    port: 8890
    #readonly: yes
replication:
    binlog: yes
        capacity: 100000000
    # The maximum synchronization speed. Unit: MB/s. A value of -1 indicates that the speed is not limited.
    sync_speed: -1
    slaveof:
        # The ID of the master node. This ID takes effect even when the IP address or port number of the master node is changed.
        # If this parameter is left empty or not specified, the IP address and port number (in the format of IP address:port) will be used.
        id: svc_1
        # The replication type. Valid values: sync and mirror. Default value: sync.
        type: sync
        host: localhost # The connection address of the SSDB master node (the source SSDB database).
        port: 8888 # The port of the SSDB master node (the source SSDB database).
        #auth: password
        redis_host: localhost # The connection address of the destination ApsaraDB for Redis instance.
        redis_port: 6379 # The port number of the destination ApsaraDB for Redis instance.
        redis_auth: password # The password of the destination ApsaraDB for Redis instance.
logger:
    level: debug
    output: log_ssdb_port.txt
    rotate:
        size: 1000000000
leveldb:
    # Unit: MB.
    cache_size: 500
    # Unit: MB.
    write_buffer_size: 64
    # Unit: MB/s.
    compaction_speed: 1000
    # Compression flag. Value values: yes and no.
    compression: yes
```

3. Run the `./ssdb-port-2.17 ssdb_port.conf` command to start synchronization.
4. Connect to the ApsaraDB for Redis instance to check whether data synchronization is completed.

Note You can use the redis-cli tool or Data Management Service (DMS) to connect to the ApsaraDB for Redis instance. For more information about how to connect to the instance, see Quick Start.

1.4.3. Migrate data from Google Cloud Memorystore for Redis to ApsaraDB for Redis

You can use the rump tool to migrate data from Google Cloud Memorystore for Redis to ApsaraDB for Redis.

Prerequisites

- An Elastic Compute Service (ECS) instance and an ApsaraDB for Redis instance are created in Alibaba Cloud, and they can communicate with each other through the internal network.
- A public endpoint is configured so that you can access the ApsaraDB for Redis instance from the ECS instance over the Internet.

  Notice The public connection address is used only for data migration through the Internet. After data is migrated, clear the related public connection address and only access the ApsaraDB for Redis instance through the internal network to ensure data security.

- The rump tool is downloaded in Google Compute Engine, and the owner of this file is granted the execute permission.

How it works

The rump tool uses the **SCAN** command to obtain the key list from the source Cloud Memorystore for Redis instance, and uses the **DUMP** command to retrieve the key content. Then, it uses the **PTTL** command to obtain the expiration time, and uses the **RESTORE** command to synchronize the keys to the destination instance through pipelines.

Note The rump tool does not support incremental migration.

Procedure

1. Run the following command in Google Compute Engine to migrate data:

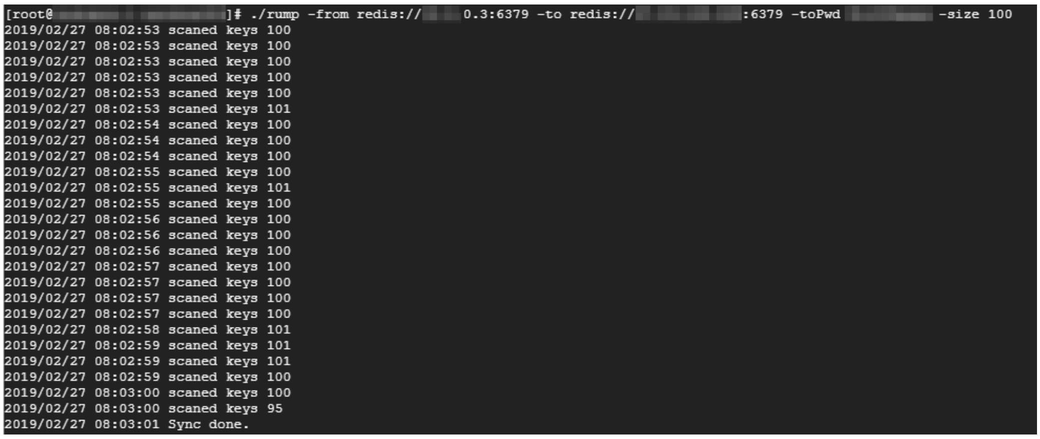
```
./rump -from source_addr -fromPwd source_pwd -to dest_addr -toPwd dest_pwd [-size size]
[-replace]
```

Parameters for the rump tool

Parameter	Description
source_addr	The address of the source Cloud Memorystore for Redis instance, in the format of `redis://host:port/db` . Set this parameter to the private IP address of the Cloud Memorystore for Redis instance. The host and port fields are required. If the db field is not set, the default value 0 is used.

Parameter	Description
source_pwd	The password of the source Cloud Memorystore for Redis instance. You do not need to set this parameter if the Cloud Memorystore for Redis instance does not have a password.
dest_addr	The address of the destination ApsaraDB for Redis instance, in the format of `redis://host:port/db` . Set this parameter to the public connection address of the ApsaraDB for Redis instance. The host and port fields are required. If the db field is not set, the default value 0 is used.
dest_pwd	The password of the destination ApsaraDB for Redis instance.
size	The number of keys scanned and synchronized at a time. Default value: 10.
replace	Specifies whether to overwrite the existing keys in the destination ApsaraDB for Redis instance that are identical to those synchronized from the source Cloud Memorystore for Redis instance. If you do not set this parameter and any keys are duplicate in the source and destination databases, an error message is returned. **Note** If you set this parameter, make sure that you have backed up important data in the destination ApsaraDB for Redis instance to avoid data loss after overwriting.

Migration example



```
[root@[redacted]]# ./rump -from redis://[redacted]0.3:6379 -to redis://[redacted]:6379 -toPwd [redacted] -size 100
2019/02/27 08:02:53 scaned keys 100
2019/02/27 08:02:53 scaned keys 100
2019/02/27 08:02:53 scaned keys 100
2019/02/27 08:02:53 scaned keys 100
2019/02/27 08:02:53 scaned keys 100
2019/02/27 08:02:53 scaned keys 101
2019/02/27 08:02:54 scaned keys 100
2019/02/27 08:02:54 scaned keys 100
2019/02/27 08:02:54 scaned keys 100
2019/02/27 08:02:55 scaned keys 100
2019/02/27 08:02:55 scaned keys 101
2019/02/27 08:02:55 scaned keys 100
2019/02/27 08:02:56 scaned keys 100
2019/02/27 08:02:56 scaned keys 100
2019/02/27 08:02:56 scaned keys 100
2019/02/27 08:02:57 scaned keys 100
2019/02/27 08:02:57 scaned keys 100
2019/02/27 08:02:57 scaned keys 100
2019/02/27 08:02:57 scaned keys 100
2019/02/27 08:02:58 scaned keys 101
2019/02/27 08:02:59 scaned keys 101
2019/02/27 08:02:59 scaned keys 101
2019/02/27 08:02:59 scaned keys 100
2019/02/27 08:03:00 scaned keys 100
2019/02/27 08:03:00 scaned keys 95
2019/02/27 08:03:01 Sync done.
```

2. Check whether all data is migrated to the ApsaraDB for Redis instance.

1.5. Verify data after migration

After Redis data is migrated, you must check whether the data is consistent between the source and destination instances.

Prerequisites

- Redis data is migrated.

> **Note** For more information about how to migrate data in ApsaraDB for Redis, see Overview.

- The source and destination Redis instances are in the following architectures: master-replica, standalone, open-source cluster, ApsaraDB for Redis cluster with proxy nodes, and TencentDB for Redis cluster with proxy nodes.
- A Linux-based Elastic Compute Service (ECS) instance is created for running the redis-full-check tool. For more information, see Create an ECS instance.
- The ECS instance can access the source and destination Redis instances.

> **Note**
>
> - If the ECS instance and the source or destination Redis instance are in the same Virtual Private Cloud (VPC), you must add the internal IP address of the ECS instance to the whitelist of the Redis instance. For more information, see Configure whitelists.
> - If the ECS instance and the source or destination Redis instance are not in the same VPC, the ECS instance can access the Redis instance through the public endpoint. For more information, see Use a public endpoint to connect to an ApsaraDB for Redis instance.

Introduction to the redis-full-check tool

If an exception occurs during Redis data migration, the data is inconsistent between the source and destination Redis instances. You can use the redis-full-check tool to find the inconsistent data.

The redis-full-check tool is a Redis data verification tool developed by Alibaba Cloud. It can extract data from the source and destination instances, compare them for multiple times, and then record the comparison results in an SQLite3 database. This tool can be used to verify full data.

> **Note** For more information about the redis-full-check tool, see redis-full-check on GitHub.

Procedure

1. Log on to the ECS instance. For more information, see Connect to an instance.
2. Download the redis-full-check tool on the ECS instance.

   > **Note** We recommend that you download the latest version.

3. Run the following command to decompress the *redis-full-check.tar.gz* package:

   ```
   tar -xvf redis-full-check.tar.gz
   ```

4. Run the following command to verify data:

   ```
   ./redis-full-check -s "<Endpoint 1 of the source Redis cluster:Port of endpoint 1;Endpoint 2 of the source Redis cluster:Port of endpoint 2;Endpoint 3 of the source Redis cluster:Port of endpoint 3;>" -p <Password of the source Redis cluster> -t <Endpoint of the destination Redis instance:Port> -a <Password of the destination Redis instance> --comparemode=1 --comparetimes=1 --qps=10 --batchcount=100 --sourcedbtype=1 --targetdbfilterlist=0
   ```

The following table describes common parameters of the redis-full-check tool. For more information, see Configure redis-full-check.

Common parameters of redis-full-check

Parameter	Description	Example

-s	The endpoint and port of the source Redis instance.
	Note
	- If the source Redis instance is a cluster, separate cluster endpoints with semicolons (;).
	- Enclose the cluster endpoints in a pair of double quotation marks (").
	- This parameter is required.
		`r-bp1xxxxxxxxxxxxx.redis.rds.aliyuncs.com:6379`
		`"10.xx.xx.1:7000;10.xx.xx.1:7001;10.xx.xx.2:7002;10.xx.xx.2:7003"`

-p	The password of the source Redis instance.	`SourcePwd233`

-t	The endpoint and port of the destination Redis instance.
	Note
	- If the destination Redis instance is a cluster, separate cluster endpoints with semicolons (;).
	- Enclose the cluster endpoints in a pair of double quotation marks (").
	- This parameter is required.
		`r-bp1xxxxxxxxxxxxx.redis.rds.aliyuncs.com:6379`
		`"10.xx.xx.1:7000;10.xx.xx.1:7001;10.xx.xx.2:7002;10.xx.xx.2:7003"`

-a	The password of the destination Redis instance.	`TargetPwd233`

Parameter	Description	Example
--sourcedbtype	The type of the source Redis instance. Valid values: - 0: standalone or master-replica. - 1: cluster. - 2: ApsaraDB for Redis or TencentDB for Redis.	`--sourcedbtype=1`
--sourcedbfilterlist	The databases for which you want to verify data in the source Redis instance. Note - This parameter is not required for open-source Redis clusters. - If you do not set this parameter, the data in all databases is verified. - Separate multiple databases with semicolons (;).	`--sourcedbfilterlist=0;1;2`
--targetdbtype	The type of the destination Redis instance. Valid values: - 0: standalone or master-replica. - 1: cluster. - 2: ApsaraDB for Redis or TencentDB for Redis.	`--targetdbtype=0`

Parameter	Description	Example
--targetdbfilterlist	The databases for which you want to verify data in the destination Redis instance. Note - This parameter is not required for open-source Redis clusters. - If you do not set this parameter, the data in all databases is verified. - Separate multiple databases with semicolons (;).	`--targetdbfilterlist=0;1;2`
-d	The name of the file for storing inconsistent keys. Default value: result.db.	`xxx.db`
--comparetimes	The number of times that the data is verified. - Default value: 3. - Minimum value: 1. - We recommend that you set this parameter to a value that does not exceed 5.	--comparetimes=1
-m	The verification mode. Valid values: - 1: verifies full data. - 2: only verifies the length of the value. - 3: only checks whether the keys exist. - 4: verifies full data except for big keys.	1

Parameter	Description	Example
--qps	The queries per second (QPS). Note - Minimum value: 1. - Maximum value: depends on the server performance.	--qps=10
--filterlist	The keys to verify. Separate multiple keys with vertical bars (|). Note - *abc**: matches all keys that start with abc. - *abc*: matches the abc key.	--filterlist=abc*|efg|m*

> Note After the verification is completed, the comparison result appears on the CLI. For example, the following result indicates that two keys are inconsistent between the source and destination Redis instances. If the number of inconsistent keys is 0, the data is consistent between the source and destination Redis instances.
>
> ```
> all finish successfully, totally 2 keys or fields conflict
> ```

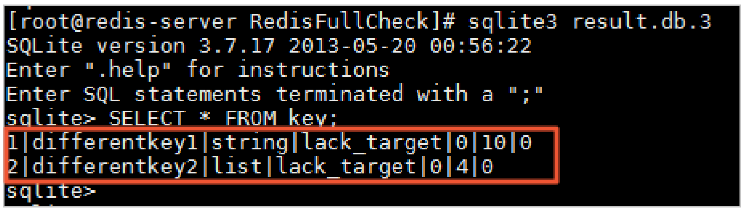
5. Query the inconsistent keys in the SQLite3 database.

   i. Run the `sqlite3 result.db.3` command.

   > Note Inconsistent keys are stored in result.db.3 by default.

ii. Run the `SELECT * FROM key;` statement.

Query inconsistent keys



```
[root@redis-server RedisFullCheck]# sqlite3 result.db.3
SQLite version 3.7.17 2013-05-20 00:56:22
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> SELECT * FROM key;
1|differentkey1|string|lack_target|0|10|0
2|differentkey2|list|lack_target|0|4|0
sqlite>
```

Note The SQLite3 database provides the key and field tables.

- The key table stores the inconsistent keys.
- The field table stores details about inconsistent data of the hash, set, zset, and list types.

2.Data Synchronization

2.1. Overview

ApsaraDB for Redis provides multiple data synchronization solutions based on Data Transmission Service (DTS) and redis-shake. These solutions apply to the following scenarios: data synchronization between on-premises Redis databases and Alibaba Cloud services, active geo-redundancy, disaster recovery, and data analysis.

Data synchronization tools

- DTS (recommended)
  DTS is a real-time data streaming service that is provided by Alibaba Cloud. The service allows you to migrate, subscribe to, and synchronize data by using stable and secure transmission channels. DTS supports one-way and two-way data synchronization. These synchronization solutions apply to scenarios such as active geo-redundancy and geo-disaster recovery.
- redis-shake
  redis-shake is an open source Linux-based tool that is developed by Alibaba Cloud. You can use this tool to parse (decode mode), restore (restore mode), back up (dump mode), and synchronize (sync or rump mode) data in ApsaraDB for Redis instances. For scenarios in which DTS is not suitable, you can use redis-shake to synchronize data.

The following table describes the differences between DTS and redis-shake for Redis data synchronization.

> **Note** You can synchronize data between different database types and architectures described in the following table. For example, you can synchronize data from a self-managed Redis database in the cluster architecture to an ApsaraDB for Redis Community Edition instance in the standard architecture.

Differences

Tool	Supported source database	Supported destination database	Supported architecture	Supported synchronization topology

Tool	Supported source database	Supported destination database	Supported architecture	Supported synchronization topology
DTS (recommended)	- Self-managed Redis database Redis 2.8, 3.0, 3.2, 4.0, and 5.0 - ApsaraDB for Redis Community Edition Redis 4.0 and 5.0 - ApsaraDB for Redis Enhanced Edition (Tair) Redis 5.0	- ApsaraDB for Redis Community Edition Redis 4.0 and 5.0 - ApsaraDB for Redis Enhanced Edition (Tair) Redis 5.0	- Standard master-replica instances - Cluster master-replica instances - Read/write splitting instances Note Read/write splitting instances do not support data synchronization between ApsaraDB for Redis instances that belong to different Alibaba Cloud accounts.	- One-way synchronization - Cross-account synchronization - Two-way synchronization Note Only ApsaraDB for Redis Enhanced Edition (Tair) instances whose database engine version is Redis 5.0 support two-way synchronization.

Tool	Supported source database	Supported destination database	Supported architecture	Supported synchronization topology
redis-shake	- Self-managed Redis database Redis 2.8, 3.0, 3.2, 4.0, and 5.0	- Self-managed Redis database Redis 2.8, 3.0, 3.2, 4.0, and 5.0 - ApsaraDB for Redis Community Edition Redis 4.0 and 5.0 - ApsaraDB for Redis Enhanced Edition (Tair) Redis 5.0	- Standard master-replica instances - Cluster master-replica instances - Read/write splitting instances	One-way synchronization

Required permissions

The permissions that are required by DTS and redis-shake vary based on different synchronization scenarios. Before you configure data synchronization, you must obtain the required permissions on the source and destination databases based on the synchronization scenarios. The following table describes the required permissions.

> **Note** For more information about how to create an account for an ApsaraDB for Redis instance and grant the required permissions to the account, see Create and manage database accounts.

Required permissions

Tool	Synchronization scenario	Permission on the source database	Permission on the destination database
	- Synchronize data between ApsaraDB for Redis instances that belong to the same Alibaba Cloud account - Synchronize data from an ApsaraDB for Redis instance to a self-managed Redis database	Read permissions	Read and write permissions

Tool	Synchronization scenario	Permission on the source database	Permission on the destination database
DTS (recommended)	• Synchronize data between ApsaraDB for Redis instances that belong to different Alibaba Cloud accounts	Replication permissions **Note** You cannot create an account that has **replication** permissions for the source instance in the cluster or read/write splitting architecture. If the destination instance is an ApsaraDB for Redis instance, you can submit a ticket to eliminate the limit.	Read and write permissions
	• Synchronize data from a self-managed Redis database to an ApsaraDB for Redis instance • Synchronize data between self-managed Redis databases	Permissions to run the **SYNC** or **PSYNC** command	Read and write permissions
redis-shake	• Synchronize data from a self-managed Redis database to an ApsaraDB for Redis instance • Synchronize data between self-managed Redis databases	Permissions to run the **SYNC** or **PSYNC** command	Read and write permissions
	• Synchronize data between ApsaraDB for Redis instances that belong to the same Alibaba Cloud account • Synchronize data between ApsaraDB for Redis instances that belong to different Alibaba Cloud accounts • Synchronize data from an ApsaraDB for Redis instance to a self-managed Redis database	Replication permissions	Read and write permissions

Data synchronization solutions

Notice The configuration methods for different scenarios are similar. The following table describes the solutions for general data synchronization scenarios. If your synchronization scenario is not included in the following table, you can view the topics about similar scenarios and configure the Required permissions.

Scenario	Tool	Solution
Synchronize data from a self-managed database to an ApsaraDB for Redis instance	DTS	Synchronize data from a self-managed Redis cluster to an ApsaraDB for Redis cluster instance
		Synchronize data from a self-managed Redis database hosted on ECS to an ApsaraDB for Redis instance
		Synchronize data from a Codis cluster hosted on ECS to an ApsaraDB for Redis instance
		Synchronize data from a Twemproxy Redis cluster hosted on ECS to an ApsaraDB for Redis instance
	ssdb-port	Use ssdb-port to synchronize data from SSDB to ApsaraDB for Redis **Note** You can use ssdb-port to synchronize data from SSDB to ApsaraDB for Redis in real time.
Synchronize data between ApsaraDB for Redis instances	DTS	Configure one-way data synchronization between ApsaraDB for Redis instances
		Configure two-way data synchronization between ApsaraDB for Redis Enhanced Edition (Tair) instances
		Synchronize data between ApsaraDB for Redis cluster instances across Alibaba Cloud accounts
Synchronize data between self-managed Redis databases	DTS	Synchronize data from a self-managed Redis database connected over Express Connect networks, VPN gateways, or Smart Access Gateway (SAG) devices to a self-managed Redis database hosted on an ECS instance

Related information

- FAQ about data migration and data synchronization
- Overview

2.2. Synchronize data from on-premises Redis to ApsaraDB for Redis

2.2.1. Synchronize data from a self-managed Redis cluster to an ApsaraDB for Redis cluster instance

Data Transmission Service (DTS) supports one-way data synchronization between Redis clusters. This feature is applicable to scenarios such as data migration, active geo-redundancy, and geo-disaster recovery. This topic describes how to configure one-way data synchronization from a self-managed Redis cluster to an ApsaraDB for Redis cluster instance.

You can also follow the procedure to configure data synchronization from an ApsaraDB for Redis cluster instance to a self-managed Redis cluster. However, you must configure parameters for the source and destination instances based on the actual scenarios.

> **Warning** After you configure a data synchronization task, do not change the architecture type of the source or destination database. For example, if you change the master-replica architecture to the cluster architecture, data synchronization fails. For more information about the architecture types, see .

Prerequisites

- The database version of the self-managed Redis cluster is 2.8, 3.0, 3.2, 4.0, or 5.0.

  > **Note** The database version of the destination ApsaraDB for Redis cluster instance can be 2.8, 4.0, or 5.0. The version of the destination database must be the same as or later than the version of the source database. If you synchronize data between different versions of Redis databases, make sure that the versions of the source and destination databases are compatible. You can create a pay-as-you-go ApsaraDB for Redis cluster instance to verify database compatibility. After verification, you can release the instance or change the billing method to subscription.

- The available storage space of the destination ApsaraDB for Redis cluster instance is larger than the total size of the data in the source Redis database.
- All nodes of the source Redis cluster support the `PSYNC` command and share the same password.

Precautions

- DTS uses the resources of the source and destination instances during initial full data synchronization. This may increase the loads of the database servers. If you migrate a large amount of data or if the server specifications cannot meet your requirements, database services may become unavailable. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.
- To ensure the stability of data synchronization, we recommend that you increase the value of the *repl-backlog-size* parameter in the `redis.conf` file.
- To ensure the synchronization quality, DTS adds the following key to the source Redis database: DTS_REDIS_TIMESTAMP_HEARTBEAT. This key is used to record the time when data is synchronized to ApsaraDB for Redis.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command in the source Redis

cluster. Otherwise, data may become inconsistent between the source and destination databases.

- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction` , data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

  > **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- During data synchronization, if the number of shards in the self-managed Redis database is increased or decreased, or if the specifications of the database are changed (for example, the memory capacity is scaled up), you must reconfigure the task. To ensure data consistency, we recommend that you clear the data that has been synchronized to the destination Redis database before you reconfigure the task.
- During data synchronization, if the endpoint of the self-managed Redis database is changed, you must submit a ticket to update the change. Otherwise, the append-only files (AOF) of the self-managed Redis database may be reset. In this case, you must reconfigure the task.
- Limits on synchronizing a standalone ApsaraDB for Redis instance to an ApsaraDB for Redis cluster instance: Each command can be run only on a single slot in an ApsaraDB for Redis cluster instance. If you perform operations on multiple keys in the source database and the keys belong to different slots, an error occurs.

  ```
  CROSSSLOT Keys in request don't hash to the same slot
  ```

  We recommend that you perform operations on only one key during data synchronization. Otherwise, the synchronization task will be interrupted.

Supported synchronization topologies

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

For more information, see Synchronization topologies.

SQL operations that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI

- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, and PSETEX
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE
- SWAPDB and UNLINK (supported only if the engine version of the source instance is 4.0)

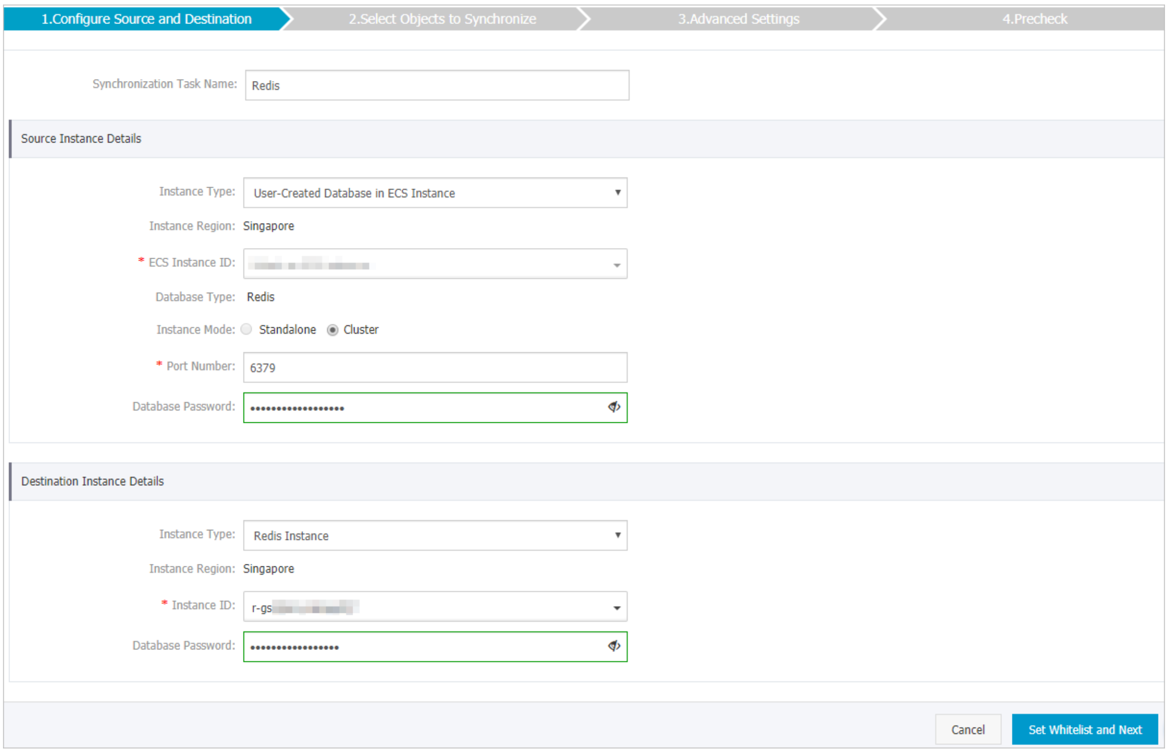
> **Note**
>
> - PUBLISH operations cannot be synchronized.
> - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
> - When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Procedure

1. Purchase a data synchronization instance.

   > **Note** On the buy page, set both Source Instance and Destination Instance to **Redis**.

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.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Task Name: Redis

Source Instance Details

Instance Type: User-Created Database in ECS Instance

Instance Region: Singapore

* ECS Instance ID: [redacted]

Database Type: Redis

Instance Mode: ○ Standalone ● Cluster

* Port Number: 6379

Database Password: ••••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs[redacted]

Database Password: •••••••••••••••••

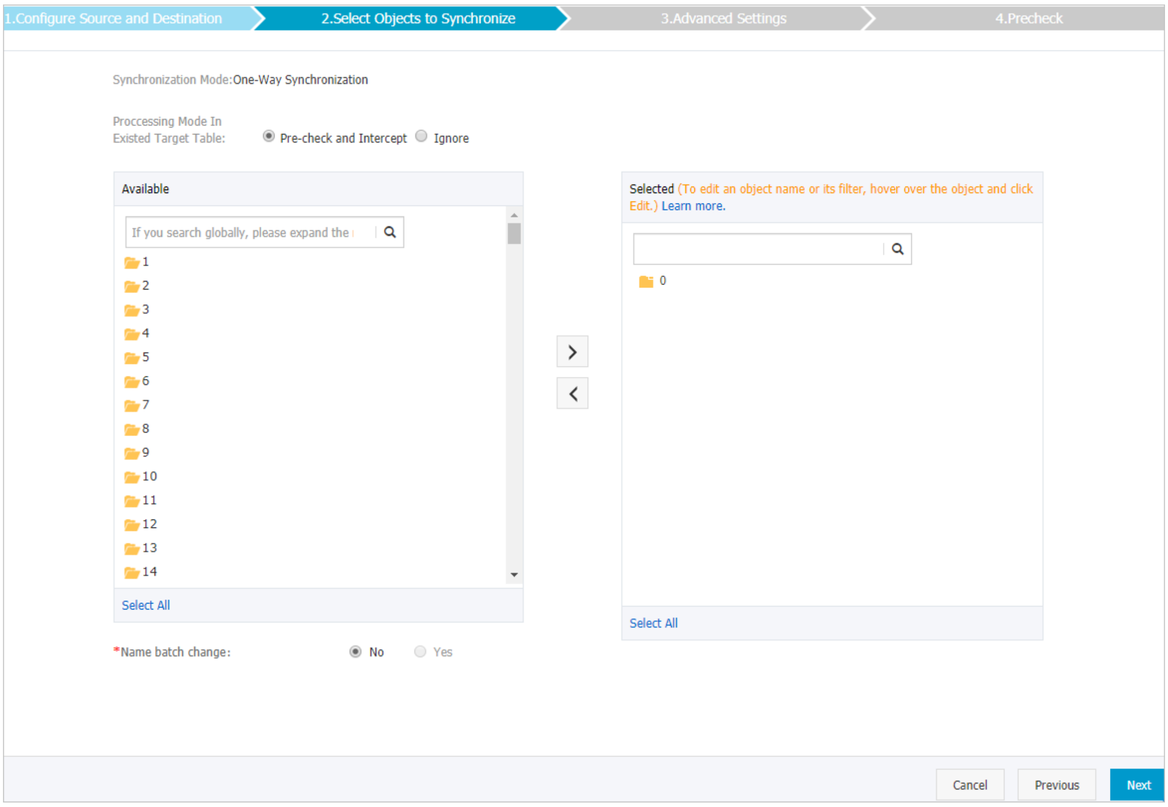
Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **User-Created Database in ECS Instance**. You can select **User-Created Database in ECS Instance** or **User-Created Database Connected Over Express Connect, VPN Gateway, or Smart Access Gateway** based on the type of the source database. The procedure in this topic uses **User-Created Database in ECS Instance** as an example. You can also follow the procedure to configure data synchronization tasks for other types of self-managed Redis databases.
	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 Elastic Compute Service (ECS) instance where a master node in the self-managed Redis cluster resides.
	Database Type	The value of this parameter is set to **Redis**.
	Instance Mode	Select **Cluster**.
	Port Number	Enter the service port number of a master node in the self-managed Redis cluster. In this example, enter **7000**.

Section	Parameter	Description
	Database Password	Enter the password that is used to log on to the self-managed Redis database. Note This parameter is optional and can be left blank if no database password is set.
Destination Instance Details	Instance Type	Select **Redis 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 ApsaraDB for Redis cluster instance.
	Database Password	Enter the database password of the destination ApsaraDB for Redis cluster instance. Note The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

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 inbound rule of the source ECS instance and the whitelist of the destination ApsaraDB for Redis cluster instance. This ensures that DTS servers can connect to the source and destination instances.

8. Select the processing mode of conflicting tables and the objects to synchronize.



1.Configure Source and Destination 2.Select Objects to Synchronize 3.Advanced Settings 4.Precheck

Synchronization Mode:One-Way Synchronization

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

Available

If you search globally, please expand the

1
2
3
4
5
6
7
8
9
10
11
12
13
14

Select All

Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.

0

Select All

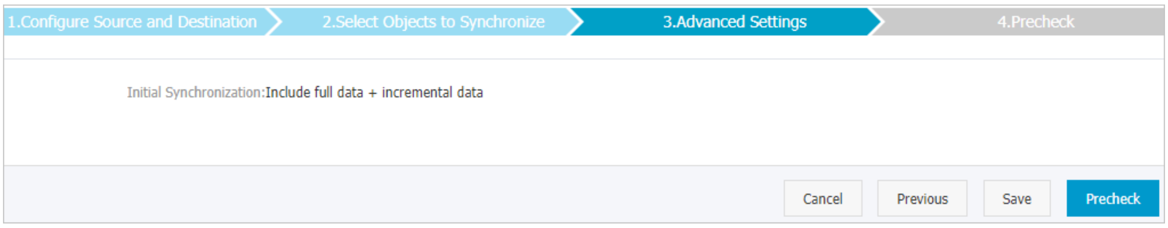
*Name batch change: ● No ○ Yes

Setting	Description
Select the processing mode of conflicting tables	◦ **Pre-check and Intercept**: checks whether the destination instance is empty. If the destination instance is empty, the precheck is passed. If the instance is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ◦ **Ignore**: skips the check for empty destination instances. **Warning** If you select **Ignore**, data records in the source instance overwrite the data records that have the same keys in the destination instance. Proceed with caution.
Select the objects to synchronize	◦ Select one or more databases from the **Available** section and click the > icon to add the databases to the **Selected** section. ◦ You can select only databases as objects to synchronize. Keys cannot be selected as objects to synchronize.
Rename Databases and Tables	In this scenario, you cannot rename objects.

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 initial synchronization types. The value is set to **Include full data + incremental data** and cannot be changed.



1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Previous
Save
Precheck

Initial Synchronization:Include full data + incremental data

Note

- DTS synchronizes historical data from the source instance to the destination instance. Then, DTS synchronizes incremental data.
- If a version-related error message is displayed, you must upgrade the source instance to a specified version. For more information about how to upgrade the version, see Upgrade the major version and Update the minor version.

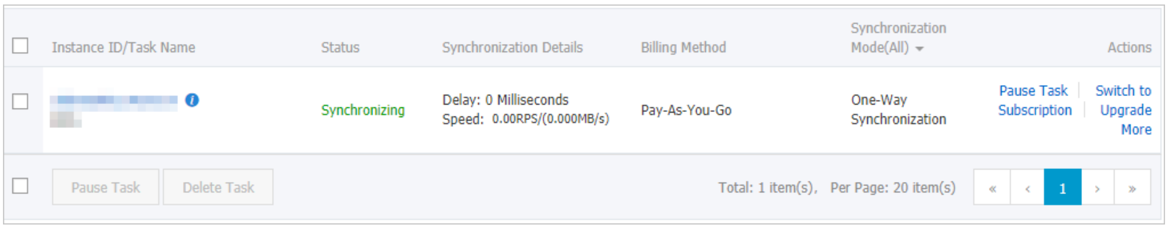
11. In the lower-right corner of the page, click **Precheck**.

    ⓘ 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.
        - 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 **The precheck is passed.** message is displayed in the **Precheck** dialog box. Then, the data synchronization task starts.

13. Wait until initial synchronization is complete and the data synchronization task enters the **Synchronizing** state.



□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
□	[redacted] ⓘ	Synchronizing	Delay: 0 Milliseconds Speed: 0.00RPS/(0.000MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

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

2.2.2. Synchronize data from a self-managed Redis database hosted on ECS to an ApsaraDB for Redis instance

Data Transmission Service (DTS) supports one-way data synchronization between Redis databases. This feature is applicable to scenarios such as active geo-redundancy and geo-disaster recovery. This topic describes how to configure one-way data synchronization from a self-managed Redis database hosted on Elastic Compute Service (ECS) to an ApsaraDB for Redis instance.

Warning After you configure a data synchronization task, do not change the architecture type of the source or destination database. For example, if you change the master-replica architecture to the cluster architecture, data synchronization fails. For more information about the architecture types, see .

Prerequisites

- The version of the source Redis database is 2.8, 3.0, 3.2, 4.0, or 5.0.

> Note The version of the destination instance must be the same as or later than that of the source instance. If you want to synchronize data between different versions of instances, make sure that the versions of the source and destination instances are compatible. For example, you can create a destination pay-as-you-go instance to verify the compatibility between the source and destination instances. Then, you can release this instance or change the billing method of the instance to subscription.

- The available storage space of the destination ApsaraDB for Redis instance is larger than the total size of the data in the source Redis database.
- If the source Redis database is deployed in a cluster architecture, all nodes of the Redis cluster must support the `PSYNC` command and share the same password.

Precautions

- DTS uses the resources of the source and destination instances during initial full data synchronization. This may increase the loads of the database servers. If you migrate a large amount of data or if the server specifications cannot meet your requirements, database services may become unavailable. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

> Note The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- If the `bind` parameter is configured in the *redis.conf* file of the source database, you must set the value of this parameter to the internal IP address of the ECS instance. The setting ensures that DTS can connect to the source database.
- To ensure the stability of data synchronization, we recommend that you increase the value of the *repl-backlog-size* parameter in the `redis.conf` file of the source Redis database.
- To ensure the synchronization quality, DTS adds the following key to the source Redis database: `DTS_REDIS_TIMESTAMP_HEARTBEAT`. This key is used to record the time when data is synchronized to ApsaraDB for Redis.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command in the source database during data synchronization. Otherwise, data may become inconsistent between the source and destination databases.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction`, data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- During data synchronization, if the number of shards in the self-managed Redis database is increased or decreased, or if the specifications of the database are changed (for example, the memory capacity is scaled up), you must reconfigure the task. To ensure data consistency, we recommend that you clear the data that has been synchronized to the destination Redis database before you reconfigure the task.
- During data synchronization, if the endpoint of the self-managed Redis database is changed, you

must submit a ticket to update the change. Otherwise, the append-only files (AOF) of the self-managed Redis database may be reset. In this case, you must reconfigure the task.

- Limits on synchronizing a standalone ApsaraDB for Redis instance to an ApsaraDB for Redis cluster instance: Each command can be run only on a single slot in an ApsaraDB for Redis cluster instance. If you perform operations on multiple keys in the source database and the keys belong to different slots, an error occurs.

  ```
  CROSSSLOT Keys in request don't hash to the same slot
  ```

  We recommend that you perform operations on only one key during data synchronization. Otherwise, the synchronization task will be interrupted.

Supported synchronization topologies

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

For more information, see Synchronization topologies.

SQL operations that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, and PSETEX
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE
- SWAPDB and UNLINK (supported only if the engine version of the source instance is 4.0)

Note

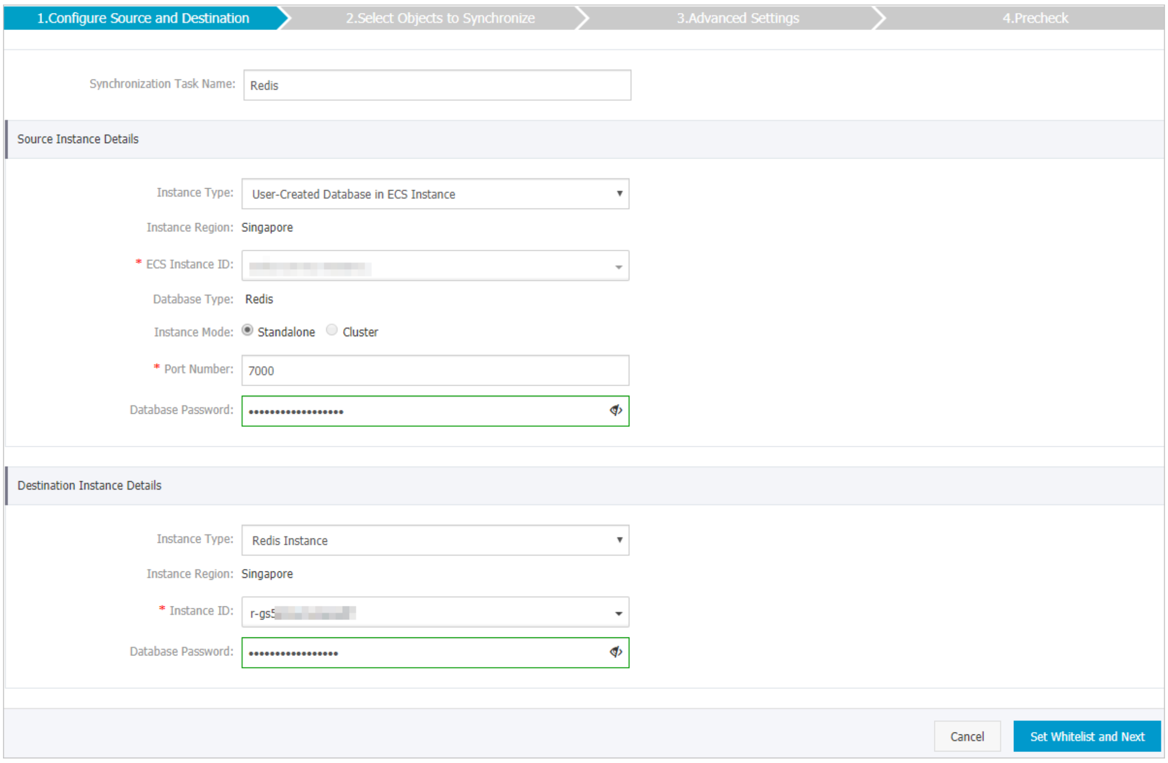
- PUBLISH operations cannot be synchronized.
- If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
- When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Procedure

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

   Note On the buy page, set both Source Instance and Destination Instance to **Redis**.

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.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Task Name: Redis

Source Instance Details

Instance Type: User-Created Database in ECS Instance

Instance Region: Singapore

* ECS Instance ID: [redacted]

Database Type: Redis

Instance Mode: ● Standalone ○ Cluster

* Port Number: 7000

Database Password: •••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs5[redacted]

Database Password: ••••••••••••••••

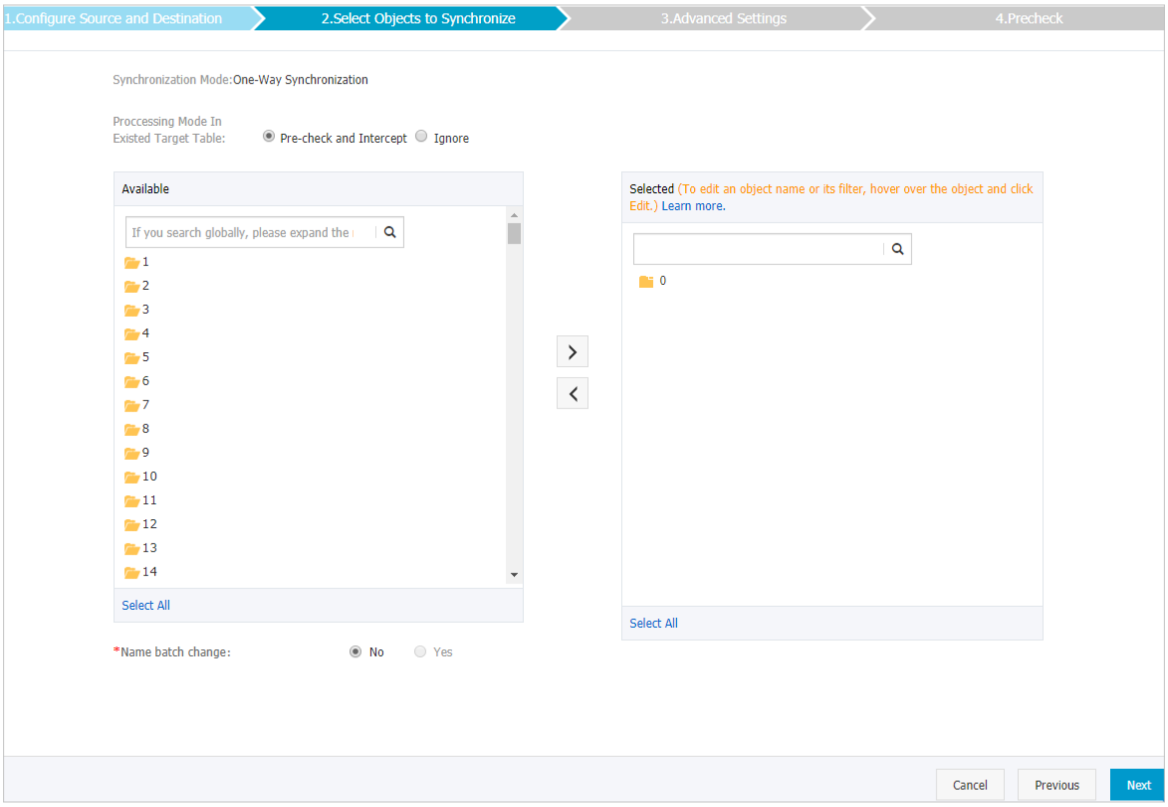
Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name for easy identification. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **User-Created Database in ECS Instance**.
	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 source Redis database. **Note** If the source Redis database is deployed in a cluster architecture, select the ID of the ECS instance where a master node resides.
	Database Type	The value of this parameter is set to **Redis**.
	Instance Mode	Select **Standalone** or **Cluster** based on the architecture of the source Redis database.
	Port Number	Enter the service port number of the source Redis database. The default port number is **6379**. In this example, enter **7000**. **Note** If the source Redis database is deployed in a cluster architecture, enter the service port number of a master node.
	Database Password	Enter the password of the source Redis database. **Note** This parameter is optional and can be left blank if no database password is set.
Destination Instance Details	Instance Type	Select **Redis 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 Redis instance.

Section	Parameter	Description
	Database Password	Enter the database password of the destination ApsaraDB for Redis instance. **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

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 inbound rule of the source ECS instance and the whitelist of the destination ApsaraDB for Redis instance. This ensures that DTS servers can connect to the source and destination instances.

8. Select the processing mode of conflicting tables and the objects to synchronize.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Available
If you search globally, please expand the
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Select All
0
Select All
Cancel
Previous
Next

Synchronization Mode:One-Way Synchronization

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.

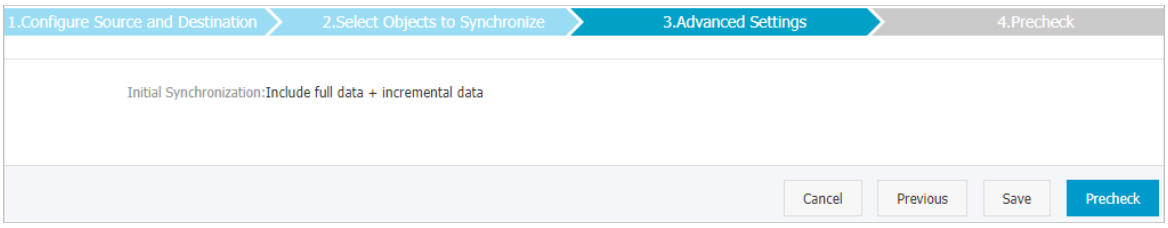
*Name batch change: ● No ○ Yes

Setting	Description

Setting	Description
Select the processing mode of conflicting tables	- **Pre-check and Intercept**: checks whether the destination instance is empty. If the destination instance is empty, the precheck is passed. If the instance is not empty, an error is returned during the precheck and the data synchronization task cannot be started. - **Ignore**: skips the check for empty destination instances. **Warning** If you select **Ignore**, data records in the source instance overwrite the data records that have the same keys in the destination instance. Proceed with caution.
Select the objects to synchronize	- Select one or more databases from the **Available** section and click the > icon to add the databases to the **Selected** section. - You can select only databases as objects to synchronize. Keys cannot be selected as objects to synchronize.
Rename Databases and Tables	In this scenario, you cannot rename objects.
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. Select initial synchronization types. The value is set to **Include full data + incremental data** and cannot be changed.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Initial Synchronization:Include full data + incremental data

Cancel
Previous
Save
Precheck

Note

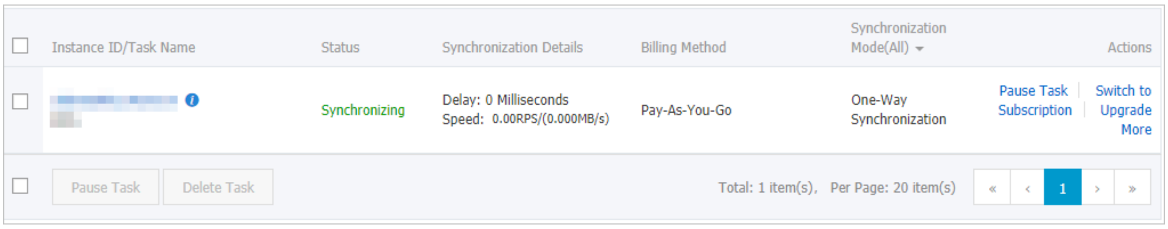
- DTS synchronizes historical data from the source instance to the destination instance. Then, DTS synchronizes incremental data.
- If a version-related error message is displayed, you must upgrade the source instance to a specified version. For more information about how to upgrade the version, see Upgrade the major version and Update the minor version.

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.
>   - 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 **The precheck is passed.** message is displayed in the **Precheck** dialog box. Then, the data synchronization task starts.

13. Wait until initial synchronization is complete and the data synchronization task enters the **Synchronizing** state.



□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All) ▾	Actions
□	[redacted] ⓘ	Synchronizing	Delay: 0 Milliseconds Speed: 0.00RPS/(0.000MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

□ Pause Task Delete Task Total: 1 item(s), Per Page: 20 item(s) « ‹ 1 › »

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

2.2.3. Synchronize data from a Codis cluster hosted on ECS to an ApsaraDB for Redis instance

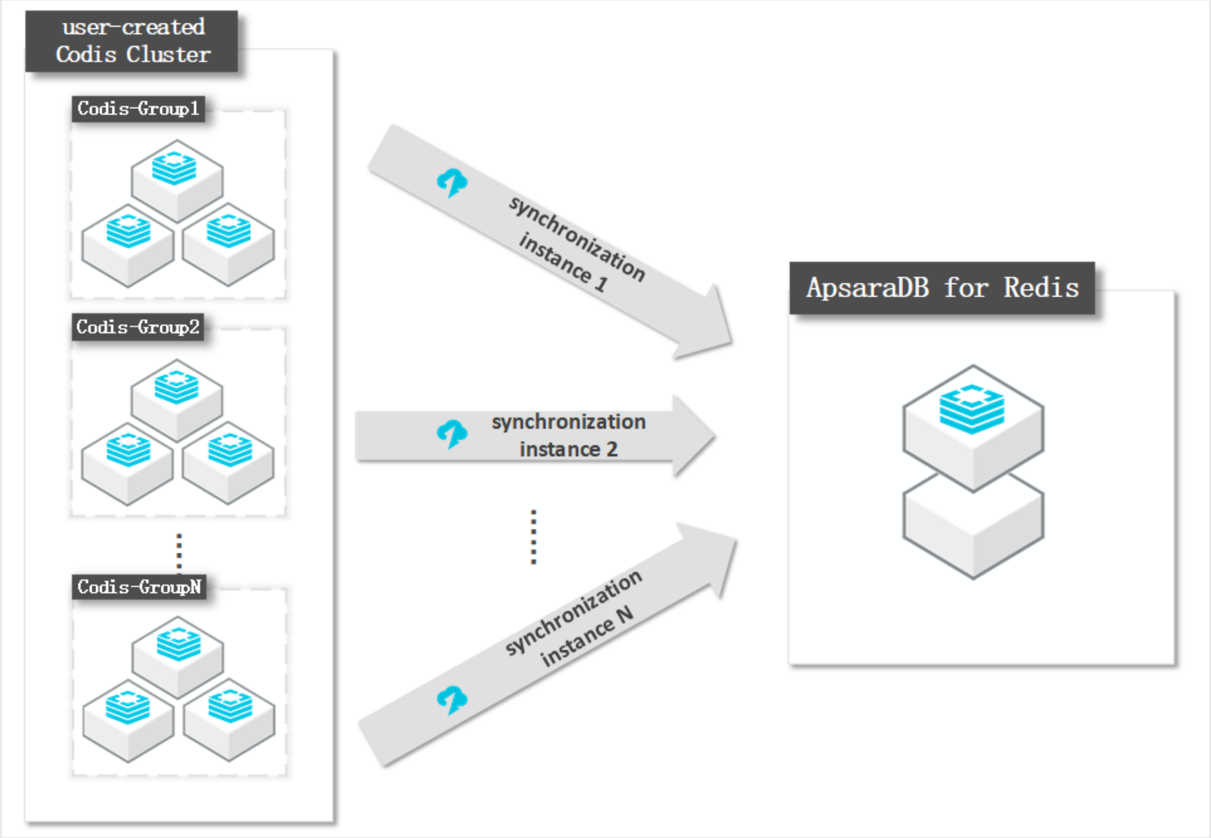
ApsaraDB for Redis is a database service compatible with the open source Redis protocol and provides hybrid storage of memory and disks. Based on reliable hot standby architectures and scalable cluster architectures, ApsaraDB for Redis is suitable for scenarios that require flexible configuration changes, high throughput, and low latency. This topic describes how to synchronize data from a Codis cluster to an ApsaraDB for Redis instance by using Data Transmission Service (DTS).

Prerequisites

- An ApsaraDB for Redis instance is created as the destination instance, and the database version of the instance is 2.8, 4.0, or 5.0. For more information, see Step 1: Create an ApsaraDB for Redis instance.
- The available storage capacity of the destination ApsaraDB for Redis instance is larger than the total size of data stored in the source Codis cluster.
- All master nodes in the source Codis cluster support the `PSYNC` command.

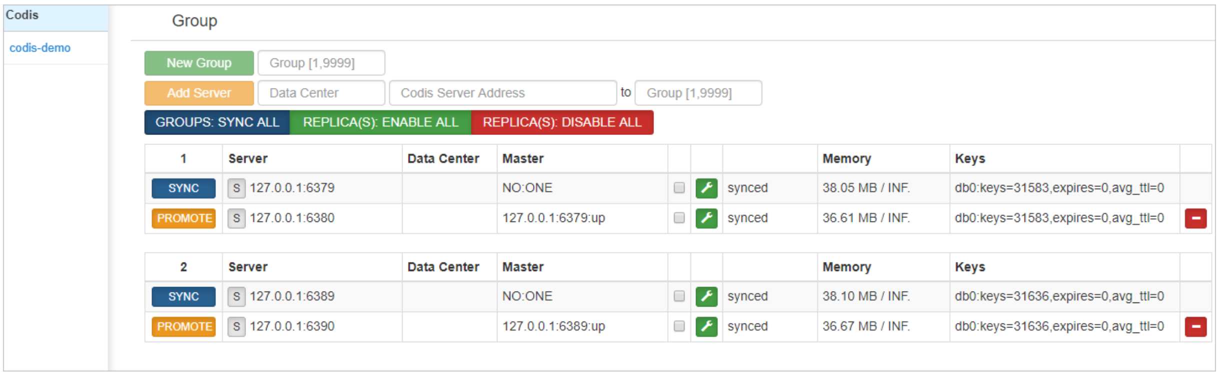
How DTS synchronizes data from a Codis cluster

A Codis cluster consists of multiple codis-groups. You must create a data synchronization task for each codis-group. DTS synchronizes each codis-group in a data synchronization task until the whole cluster is synchronized.



Architecture of the Codis cluster

In this topic, a Codis cluster that consists of two codis-groups is used. Each codis-group runs in a master-replica architecture. The following figure shows the architecture of the cluster.



1	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6379		NO:ONE	□		synced	38.05 MB / INF.	db0:keys=31583,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6380		127.0.0.1:6379:up	□		synced	36.61 MB / INF.	db0:keys=31583,expires=0,avg_ttl=0	−

2	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6389		NO:ONE	□		synced	38.10 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6390		127.0.0.1:6389:up	□		synced	36.67 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0	−

Precautions

- DTS uses the resources of the source and destination instances during initial full data synchronization. This may increase the loads of the database servers. If you migrate a large amount of data or if the server specifications cannot meet your requirements, database services may become unavailable. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.

- If the `bind` parameter is configured in the *redis.conf* file of the source database, you must set the value of this parameter to the internal IP address of the ECS instance. The setting ensures that DTS can connect to the source database.
- To ensure the stability of data synchronization, we recommend that you increase the value of the *repl-backlog-size* parameter in the `redis.conf` file of the source Codis cluster.
- To ensure the synchronization quality, DTS adds the following key to the source Codis cluster: `DTS_REDIS_TIMESTAMP_HEARTBEAT`. This key is used to record the time when data is synchronized to the destination instance.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command in the source Codis cluster. Otherwise, data may become inconsistent between the source and destination databases.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction`, data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

  > **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- The database version of the destination ApsaraDB for Redis instance must be 2.8, 4.0, or 5.0. The version of the destination database must be the same as or later than the version of the source database. If you synchronize data between different versions of Redis databases, make sure that the versions of the source and destination databases are compatible with each other. For example, you can create a destination pay-as-you-go ApsaraDB for Redis instance to verify the compatibility between the source and destination databases. Then, you can release the instance or change the billing method of the instance to subscription.

Supported synchronization topologies

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

For more information, see Synchronization topologies.

Commands that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI

- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, PSETEX, and PUBLISH
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE

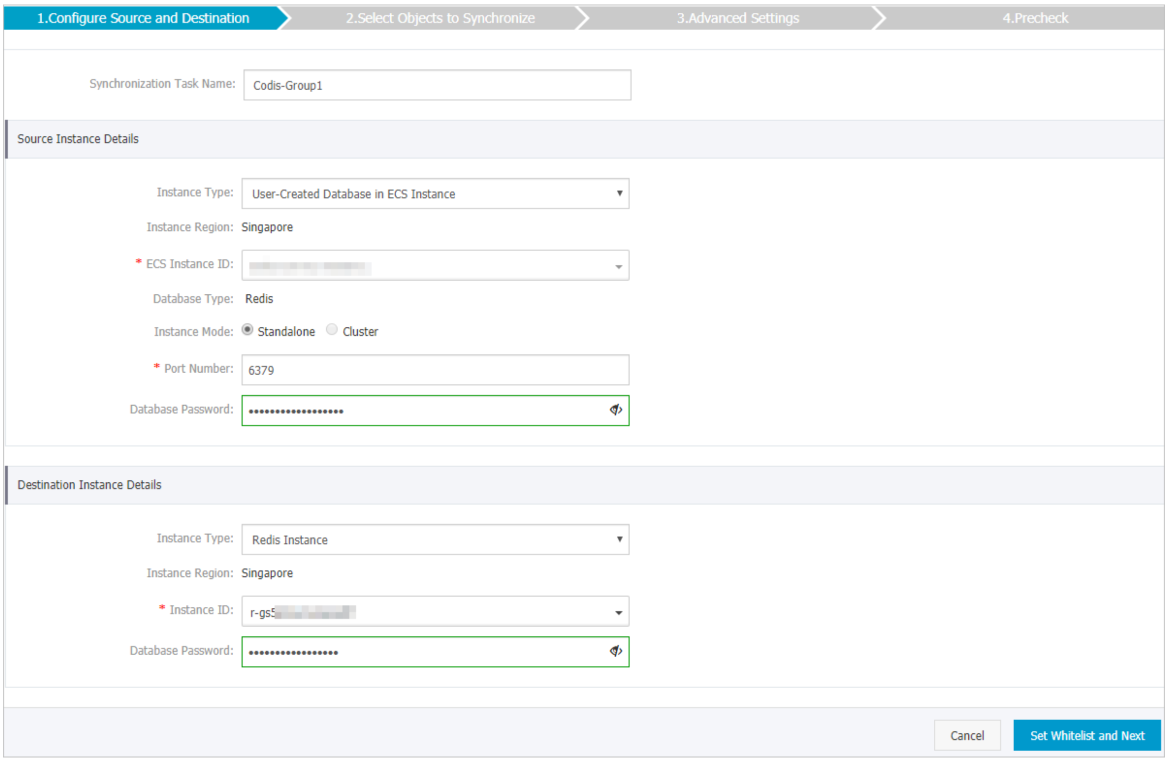
> Note
>
> - PUBLISH operations cannot be synchronized.
> - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
> - When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Procedure

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

   > Note On the buy page, set Source Instance to **Redis**, set Destination Instance to **Redis**, 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.



1.Configure Source and Destination > 2.Select Objects to Synchronize > 3.Advanced Settings > 4.Precheck

Synchronization Task Name: Codis-Group1

Source Instance Details

Instance Type: User-Created Database in ECS Instance

Instance Region: Singapore

* ECS Instance ID: [redacted]

Database Type: Redis

Instance Mode: ● Standalone ○ Cluster

* Port Number: 6379

Database Password: ••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs5[redacted]

Database Password: ••••••••••••••••

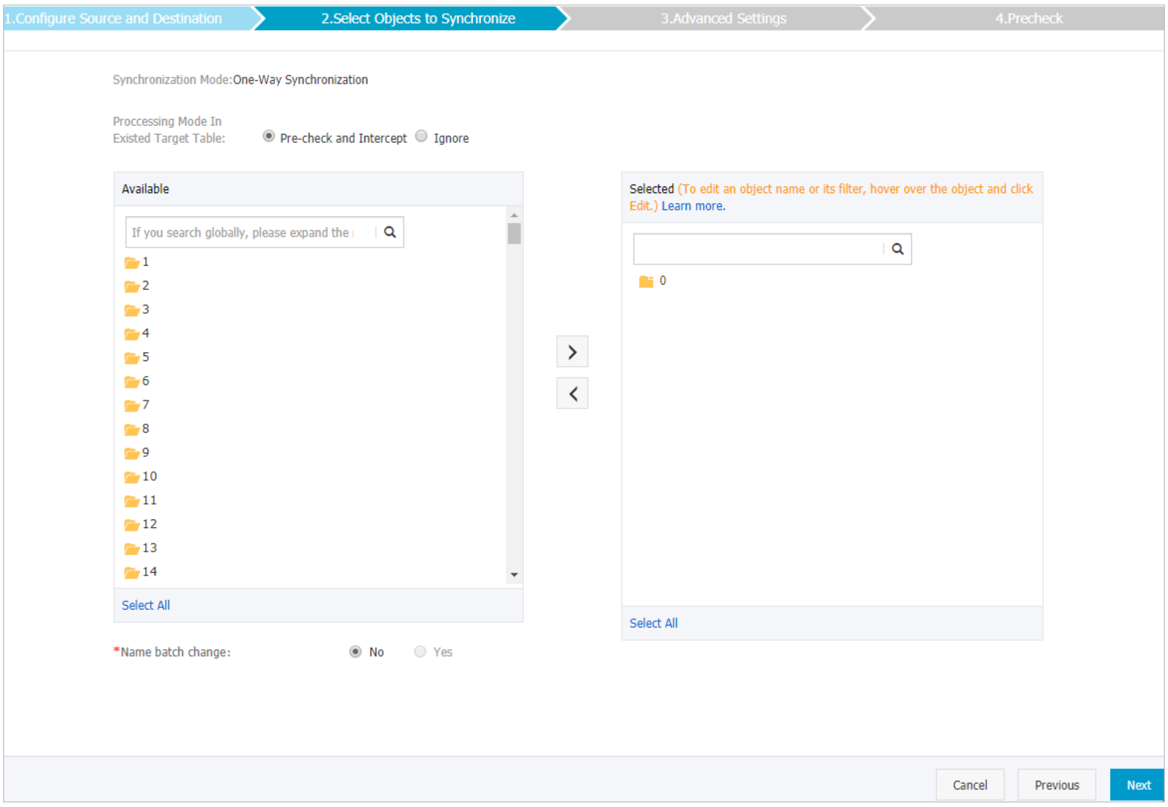
Cancel | Set Whitelist and Next

Section	Parameter	Description
None	Synchronization Task Name	DTS generates a task name. We recommend that you specify an informative name to identify the task. 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. The value of this parameter cannot be changed.
	ECS Instance ID	Select the ID of the Elastic Compute Service (ECS) instance that hosts the master node of the codis-group. **Note** DTS synchronizes each codis-group of the Codis cluster by using a data synchronization task until the entire cluster is synchronized. In this step, enter the ECS instance ID of the master node for codis-group 1. When you configure the data synchronization task for codis-group 2, enter the ECS instance ID of the master node for codis-group 2. You can configure data synchronization tasks for all codis-groups by following the procedure described in this topic.
	Database Type	The value of this parameter is set to **Redis**.

Section	Parameter	Description
	Instance Mode	Select **Standalone**. **Note** You must select **Standalone** for this parameter because data from a Codis cluster cannot be synchronized at a time. DTS synchronizes each codis-group of the cluster in a data synchronization task until all codis-groups are synchronized.
	Port Number	Enter the service port number of the master node in the codis-group.
	Database Password	The database password of the master node. **Note** This parameter is optional and can be left blank if no database password is set.
Destination Instance Details	Instance Type	Select **Redis Instance**.
	Instance Region	The region of the destination cluster. The region is the same as the destination region that you selected on the buy page. The value of this parameter cannot be changed.
	Instance ID	Select the ID of the destination ApsaraDB for Redis instance.
	Database Password	Enter the database password of the ApsaraDB for Redis instance. **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

7.

8. Select the processing mode of conflicting tables and the objects to be synchronized.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Available
If you search globally, please expand the
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Select All
Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.
0
Select All
Cancel
Previous
Next

Synchronization Mode:One-Way Synchronization

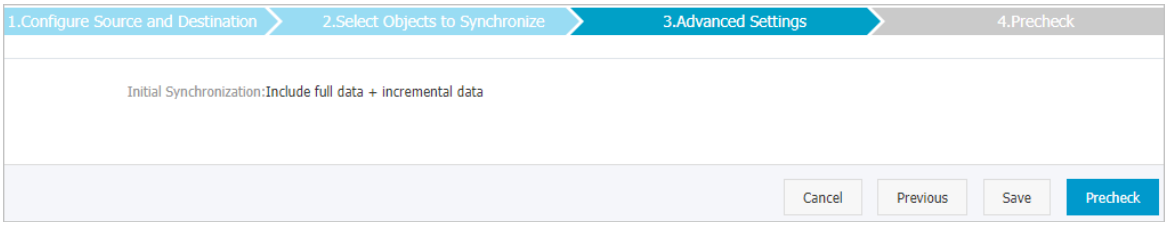
Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

*Name batch change: ● No ○ Yes

Parameter	Description
Select the processing mode of conflicting tables	DTS synchronizes each codis-group of the Codis cluster in a data synchronization task until the whole cluster is synchronized. When you configure data synchronization for codis-group 1, select **Pre-check and Intercept** if the ApsaraDB for Redis instance has no data. When you configure data synchronization for codis-groups 2 to N, select **Ignore**. Otherwise, errors may occur during data synchronization. **Note** ◦ **Pre-check and Intercept**: checks whether the destination database is empty. If the destination database is empty, the precheck is passed. If the database is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ◦ **Ignore**: skips the precheck for empty destination databases and continues with data synchronization. If the keys in the destination database are the same as those in the source database during data synchronization, the keys in the source database overwrite those in the destination database.

Parameter	Description
Select the objects to be synchronized	○ Select one or more databases from the **Available** section and click > to move the databases to the **Selected** section. ○ You can select only databases as the objects to be synchronized. You cannot select keys as the objects to be synchronized.
Rename Databases and Tables	In this scenario, you cannot rename objects.
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 initial synchronization.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Previous
Save
Precheck

Initial Synchronization:Include full data + incremental data

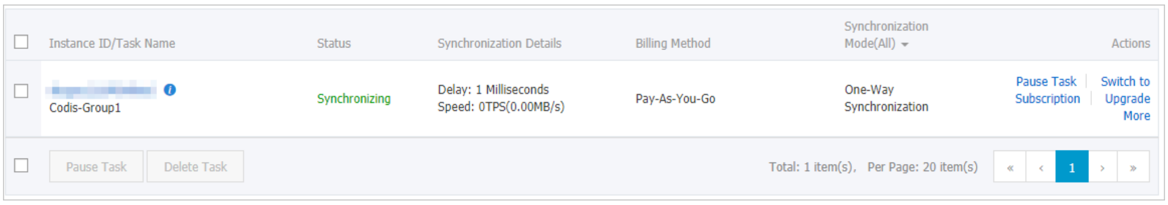
Note The value is set to **Include full data + incremental data**. DTS synchronizes historical data from the source Codis cluster to the destination Redis databases and then synchronizes incremental data.

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: **The precheck is passed.** Then, the data synchronization task starts.
13. Wait until initial synchronization is completed and the data synchronization task enters the **Synchronizing** state.



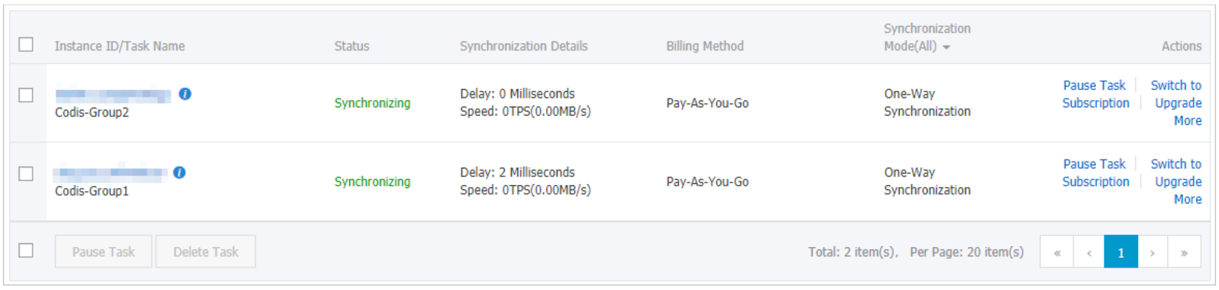
□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
□	[redacted] Codis-Group1	Synchronizing	Delay: 1 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

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

14. Create and configure a data synchronization task for the other codis-group by repeating Steps 1 to 13.

Result

In this topic, the Codis cluster consists of two codis-groups. You must create two data synchronization tasks. The following figure shows that the initial synchronization is complete for both tasks and both tasks are in the **Synchronizing** state.

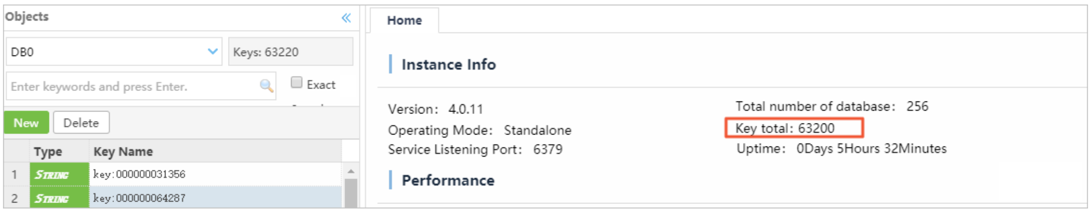


☐	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
☐	[redacted] Codis-Group2	Synchronizing	Delay: 0 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More
☐	[redacted] Codis-Group1	Synchronizing	Delay: 2 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

Total: 2 item(s), Per Page: 20 item(s)

In this topic, databases DB0 and DB1 are synchronized. You can use Data Management (DMS) to log on to the ApsaraDB for Redis instance and check the total number of keys in the ApsaraDB for Redis instance. The total number of keys is the same as that in the source Codis cluster. For more information about how to use DMS to log on to an ApsaraDB for Redis instance, see Use DMS to log on to an ApsaraDB for Redis instance.

ApsaraDB for Redis instance



Keys: 63220

	Type	Key Name
1	STRING	key:000000031356
2	STRING	key:000000064287

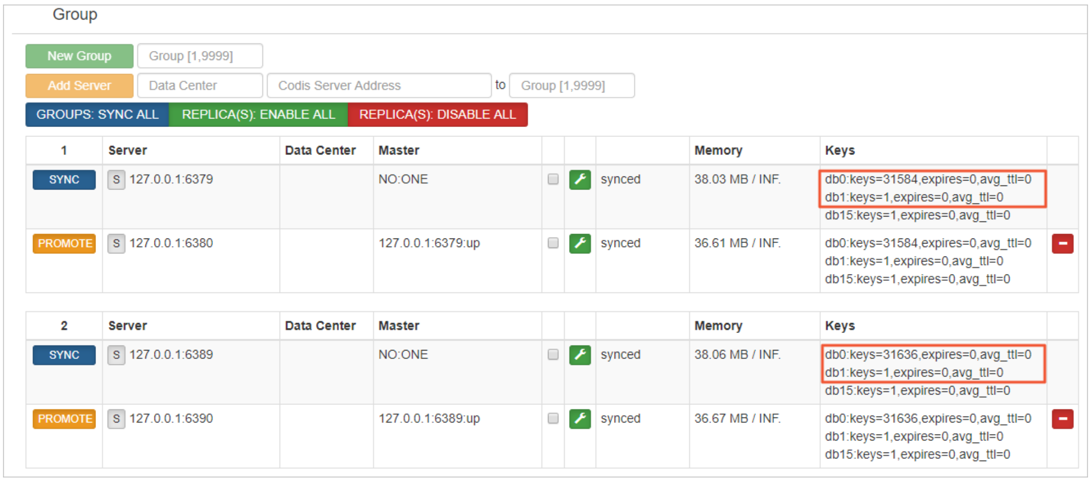
Instance Info

Version: 4.0.11
Operating Mode: Standalone
Service Listening Port: 6379

Total number of database: 256
Key total: 63200
Uptime: 0Days 5Hours 32Minutes

Performance

Source Codis cluster



1	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6379		NO:ONE	☐	🔧	synced	38.03 MB / INF.	db0:keys=31584,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6380		127.0.0.1:6379:up	☐	🔧	synced	36.61 MB / INF.	db0:keys=31584,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	−

2	Server	Data Center	Master				Memory	Keys	
SYNC	S 127.0.0.1:6389		NO:ONE	☐	🔧	synced	38.06 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	
PROMOTE	S 127.0.0.1:6390		127.0.0.1:6389:up	☐	🔧	synced	36.67 MB / INF.	db0:keys=31636,expires=0,avg_ttl=0 db1:keys=1,expires=0,avg_ttl=0 db15:keys=1,expires=0,avg_ttl=0	−

2.2.4. Synchronize data from a Twemproxy Redis cluster hosted on ECS to an ApsaraDB for Redis instance

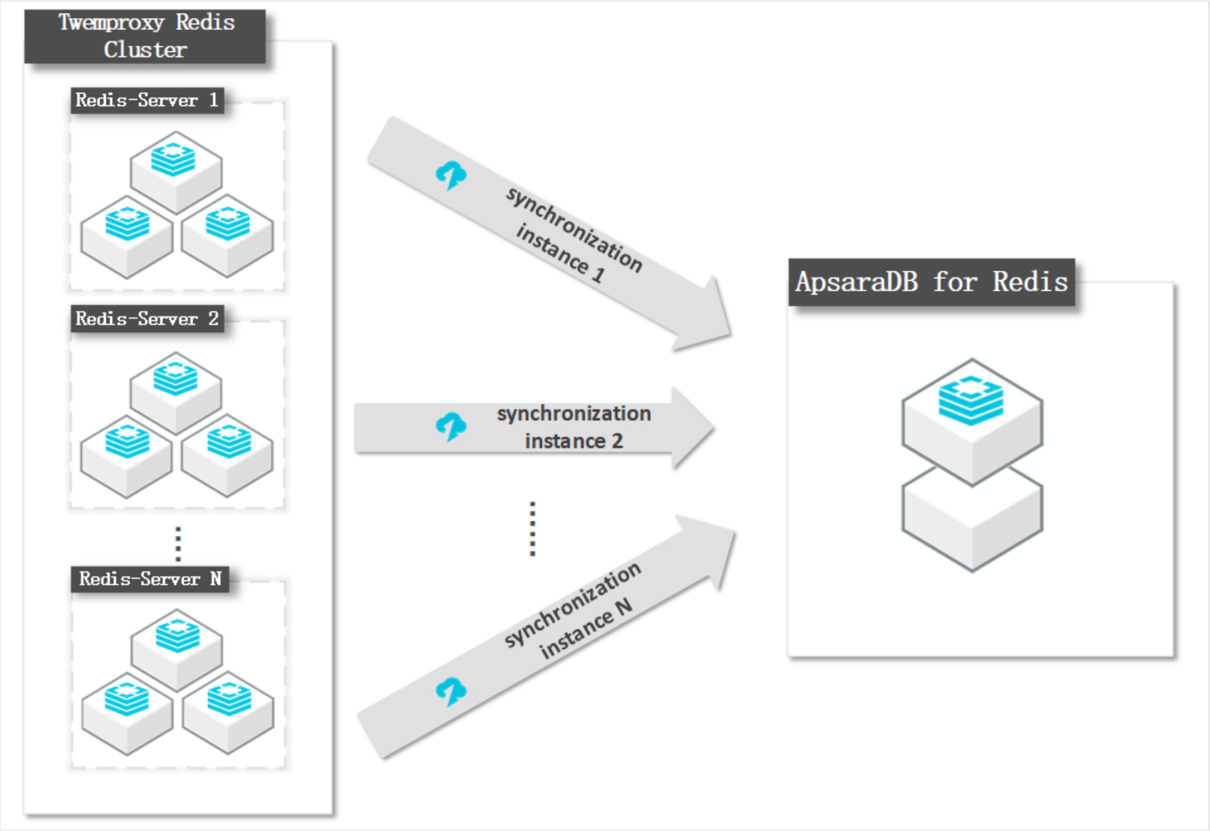
ApsaraDB for Redis is a database service compatible with the open source Redis protocol and provides hybrid storage of memory and hard disks. Based on reliable hot standby architecture and scalable cluster architecture, ApsaraDB for Redis is suitable for scenarios that require flexible configuration changes, high throughput, and low latency. This topic describes how to synchronize data from a Twemproxy Redis cluster to an ApsaraDB for Redis instance by using Data Transmission Service (DTS).

Prerequisites

- An ApsaraDB for Redis instance is created. For more information, see Create an ApsaraDB for Redis instance.
- The available storage space of the ApsaraDB for Redis instance is larger than the total size of data stored in the Twemproxy Redis cluster.
- All master nodes in the Twemproxy Redis cluster support the `PSYNC` command.

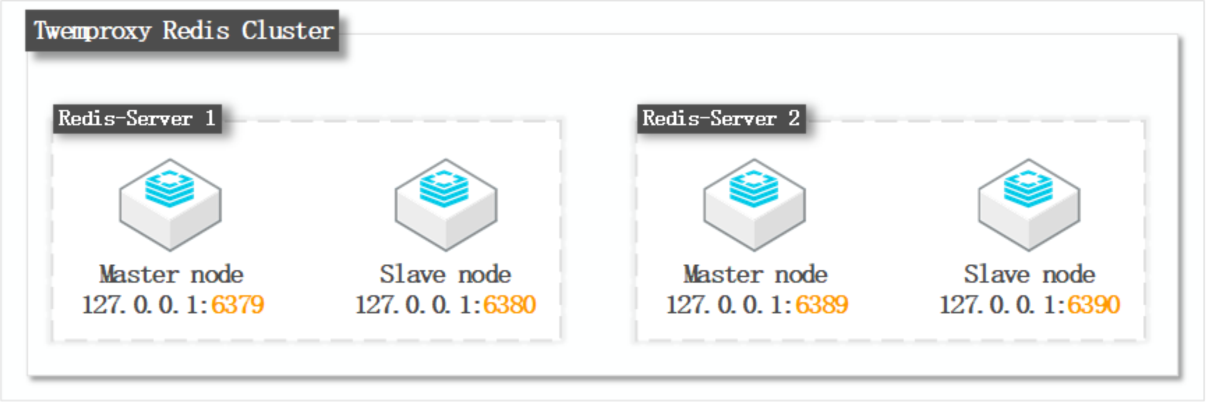
How DTS synchronizes data from a Twemproxy Redis cluster

A Twemproxy Redis cluster consists of multiple Redis-Servers. DTS synchronizes data of each Redis-Server in a data synchronization task until the whole cluster is synchronized.



Architecture of the Twemproxy Redis cluster

In this topic, the Twemproxy Redis cluster consists of two Redis-Servers. Each Redis-Server runs in a master-replica architecture. The following figure shows the architecture of the cluster.



Precautions

- DTS uses the resources of the source and destination instances during initial full data synchronization. This may increase the loads of the database servers. If you migrate a large amount of data or if the server specifications cannot meet your requirements, database services may become unavailable. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.
- If the `bind` parameter is configured in the *redis.conf* file of the source database, you must set the value of this parameter to the internal IP address of the ECS instance. The setting ensures that DTS can connect to the source database.
- To ensure the stability of data synchronization, we recommend that you increase the value of the *repl-backlog-size* parameter in the `redis.conf` file of the source Codis cluster.
- To ensure the synchronization quality, DTS adds the following key to the source Codis cluster: `DTS_REDIS_TIMESTAMP_HEARTBEAT`. This key is used to record the time when data is synchronized to the destination instance.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command in the source Codis cluster. Otherwise, data may become inconsistent between the source and destination databases.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction`, data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

  > **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- The database version of the destination ApsaraDB for Redis instance must be 2.8, 4.0, or 5.0. The version of the destination database must be the same as or later than the version of the source database. If you synchronize data between different versions of Redis databases, make sure that the versions of the source and destination databases are compatible with each other. For example, you can create a destination pay-as-you-go ApsaraDB for Redis instance to verify the compatibility between the source and destination databases. Then, you can release the instance or change the billing method of the instance to subscription.

Supported synchronization topologies

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

For more information, see Synchronization topologies.

Commands that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT

- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, PSETEX, and PUBLISH
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE

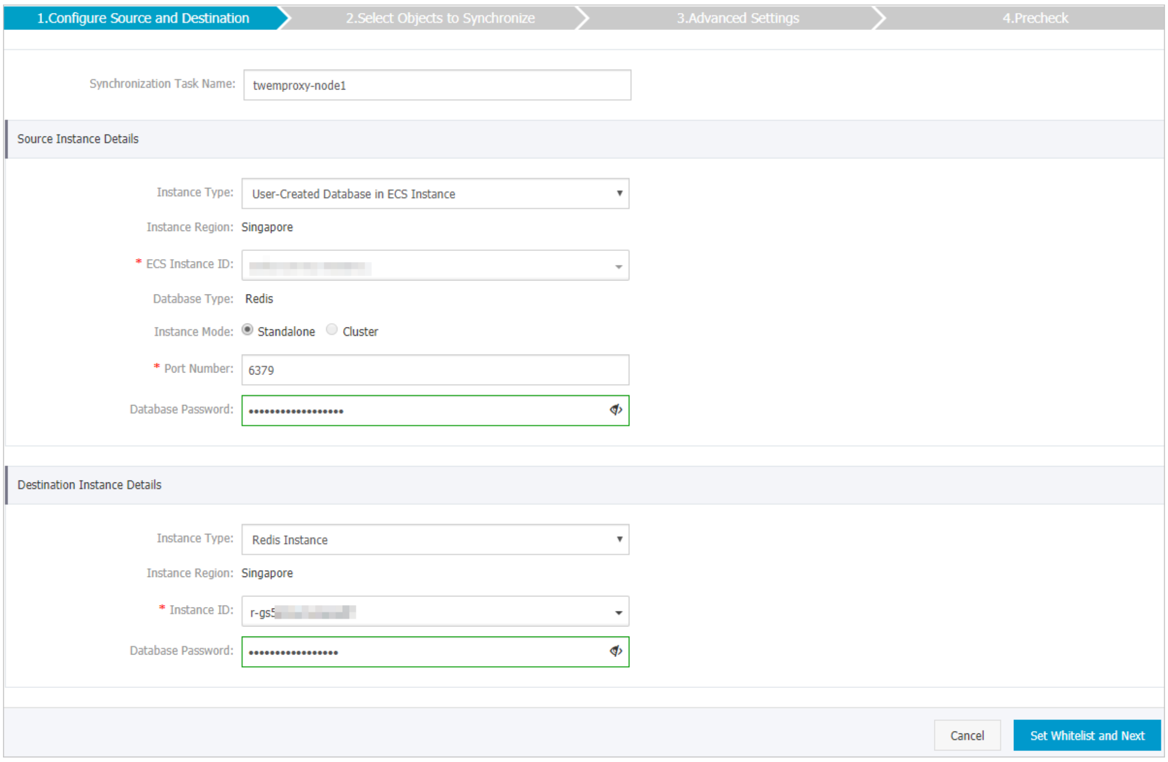
> Note
>
> - PUBLISH operations cannot be synchronized.
> - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
> - When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Procedure

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

   > Note On the buy page, set Source Instance to **Redis**, set Target Instance to **Redis**, 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.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Set Whitelist and Next

Synchronization Task Name: twemproxy-node1

Source Instance Details

Instance Type: User-Created Database in ECS Instance

Instance Region: Singapore

* ECS Instance ID: [redacted]

Database Type: Redis

Instance Mode: ● Standalone ○ Cluster

* Port Number: 6379

Database Password: ••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs5[redacted]

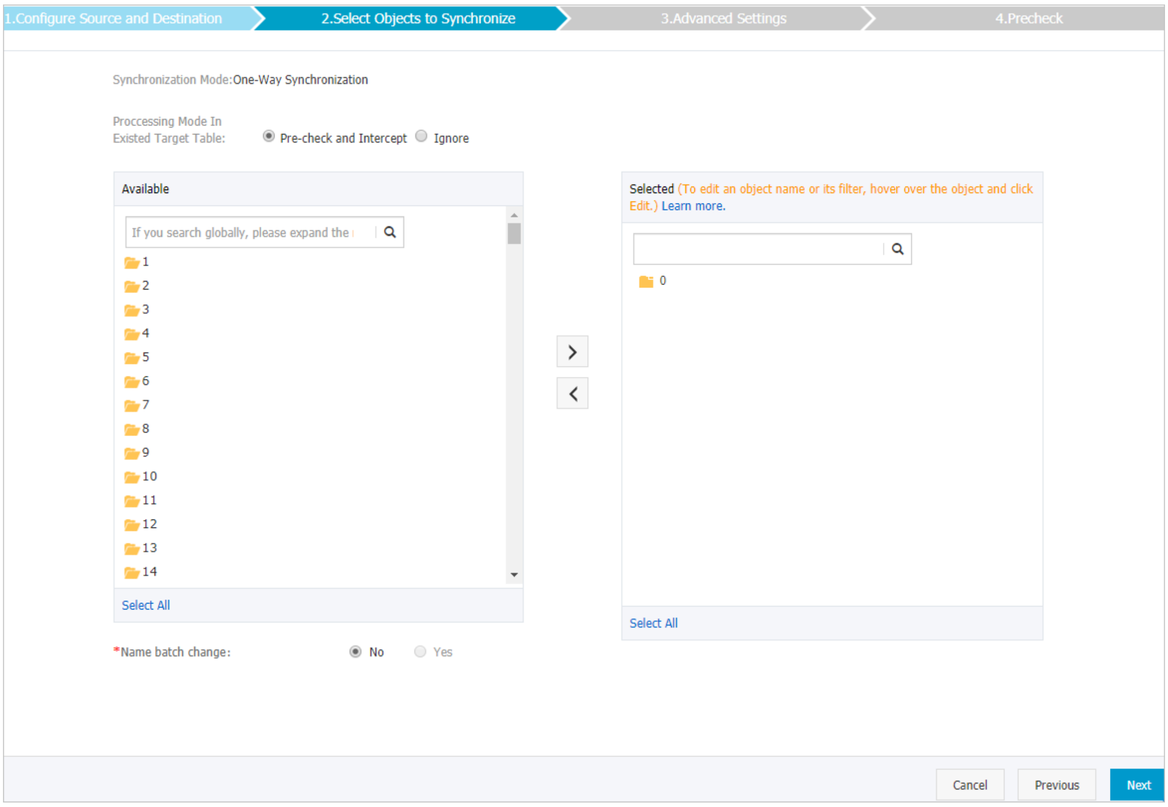
Database Password: ••••••••••••••••

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 Elastic Compute Service (ECS) instance that hosts the master node of the Redis-Server. **Note** DTS synchronizes each Redis-Server of the Twemproxy Redis cluster by using a data synchronization task until the whole cluster is synchronized. In this step, enter the ECS instance ID for the master node of Redis-Server 1. When you configure the data synchronization task for Redis-Server 2, enter the ECS instance ID for the master node of Redis-Server 2. You can configure data synchronization tasks for all Redis-Servers by following the procedure described in this topic.
	Database Type	The value of this parameter is set to **Redis**.

Section	Parameter	Description
	Instance Mode	Select **Standalone**. Note You must select **Standalone** for this parameter because data from a Twemproxy Redis cluster cannot be synchronized at a time. DTS synchronizes data of each Redis-Server of the cluster in a data synchronization task until all Redis-Servers are synchronized.
	Port Number	Enter the service port number of the master node in the Redis-Server.
	Database Password	The database password of the master node. Note This parameter is optional and can be left blank if no database password is set.
Destination Instance Details	Instance Type	Select **Redis 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 ApsaraDB for Redis instance.
	Database Password	Enter the database password of the ApsaraDB for Redis instance. Note The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

7.

8. Select the processing mode of conflicting tables, and the objects to be synchronized.



1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Mode:One-Way Synchronization

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

Available

If you search globally, please expand the

1
2
3
4
5
6
7
8
9
10
11
12
13
14

Select All

Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.

0

Select All

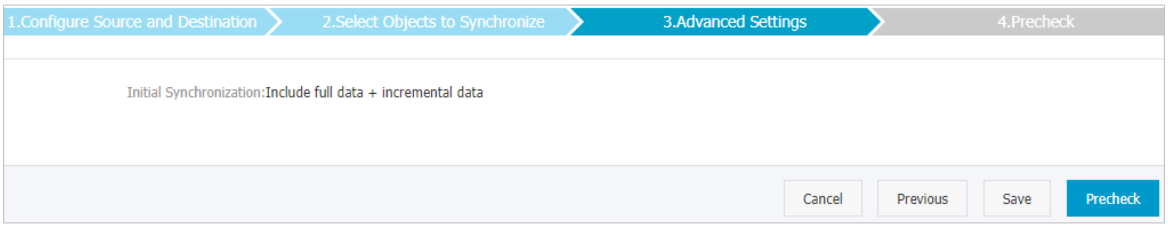
*Name batch change: ● No ○ Yes

Setting	Description
Select the processing mode of conflicting tables	DTS synchronizes each Redis-Server of the Twemproxy Redis cluster in a data synchronization task until the whole cluster is synchronized. When you configure data synchronization for Redis-Server 1, if the ApsaraDB for Redis instance has no data, select **Pre-check and Intercept**. When you configure data synchronization for Redis-Server 2 to N, select **Ignore**. Otherwise, errors may occur during data synchronization. Note ◦ **Pre-check and Intercept**: checks whether the destination database is empty. If the destination database is empty, the precheck is passed. If the database is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ◦ **Ignore**: skips the precheck for empty destination databases and continues with data synchronization. If the keys in the destination database are the same as those in the source database during data synchronization, the keys in the source database overwrite those in the destination database.
Select the objects to be synchronized	◦ Select one or more databases from the **Available** section and click > to move the databases to the **Selected** section. ◦ You can select only databases as the objects to be synchronized. You cannot select keys as the objects 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. Select the initial synchronization types.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Previous
Save
Precheck

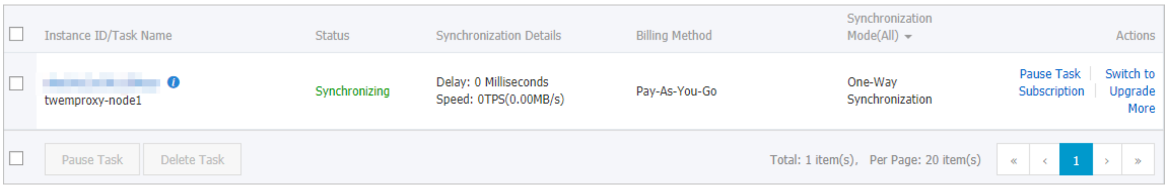
Initial Synchronization:Include full data + incremental data

> **Note** The value is set to **Include full data + incremental data**. DTS synchronizes historical data from the source Twemproxy Redis cluster to the destination Redis database and then synchronizes incremental data.

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: **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.



□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All) ▾	Actions
□	[redacted] ⓘ twemproxy-node1	Synchronizing	Delay: 0 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

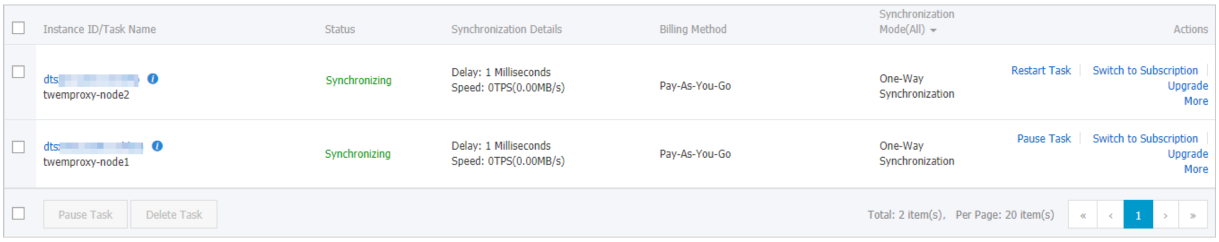
□ Pause Task Delete Task Total: 1 item(s), Per Page: 20 item(s) « ‹ 1 › »

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

14. Create and configure a data synchronization task for every other Redis-Server by repeating Steps 1 to 13.

Result

In this topic, the Twemproxy Redis cluster consists of two Redis-Servers. You must create two data synchronization tasks. The following figure shows that the initial synchronization is complete for both tasks and both tasks are in the **Synchronizing** state.

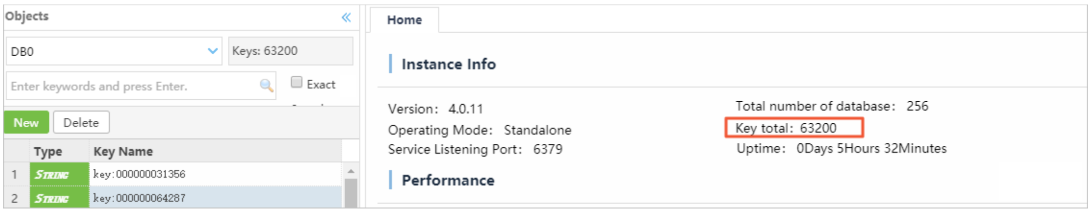


□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All) ▾	Actions
□	dts[redacted] ⓘ twemproxy-node2	Synchronizing	Delay: 1 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Restart Task | Switch to Subscription | Upgrade More
□	dts[redacted] ⓘ twemproxy-node1	Synchronizing	Delay: 1 Milliseconds Speed: 0TPS(0.00MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task | Switch to Subscription | Upgrade More

□ Pause Task Delete Task Total: 2 item(s), Per Page: 20 item(s) « ‹ 1 › »

In this topic, the database DB0 is synchronized. You can use Data Management (DMS) to log on to the destination ApsaraDB for Redis instance and check the total number of keys in the ApsaraDB for Redis instance. The total number of keys is the same as that in the Twemproxy Redis cluster.

ApsaraDB for Redis instance



Objects

DB0 | Keys: 63200

Enter keywords and press Enter. □ Exact

New | Delete

	Type	Key Name
1	STRING	key:000000031356
2	STRING	key:000000064287

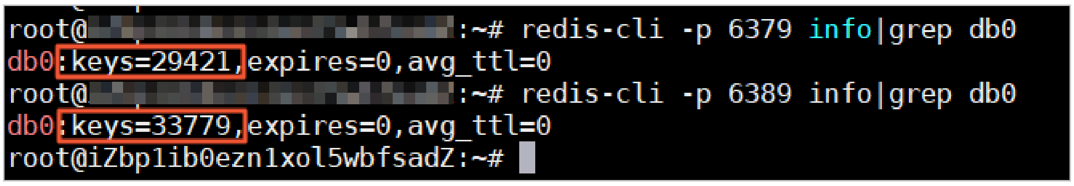
Home

Instance Info

Version: 4.0.11
Operating Mode: Standalone
Service Listening Port: 6379
Total number of database: 256
Key total: 63200
Uptime: 0Days 5Hours 32Minutes

Performance

Source Twemproxy Redis cluster



```
root@[redacted]:~# redis-cli -p 6379 info|grep db0
db0:keys=29421,expires=0,avg_ttl=0
root@[redacted]:~# redis-cli -p 6389 info|grep db0
db0:keys=33779,expires=0,avg_ttl=0
root@iZbp1ib0ezn1xol5wbfsadZ:~#
```

2.2.5. Use ssdb-port to synchronize data from SSDB to ApsaraDB for Redis

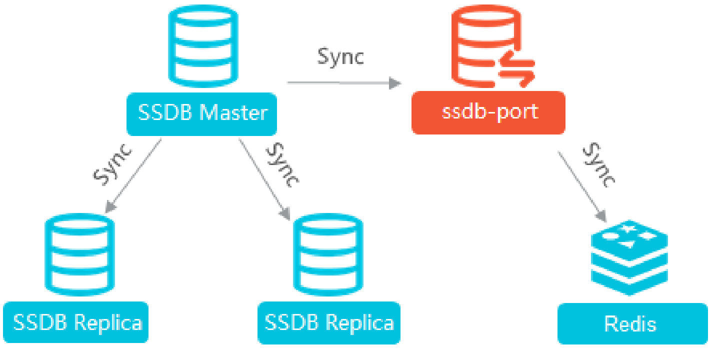
ssdb-port is a tool that can be used to synchronize data from Sequence Similarity DataBase (SSDB) to ApsaraDB for Redis in real time.

Prerequisites

- A Linux-based Elastic Compute Service (ECS) instance is created in the Virtual Private Cloud (VPC) network that is connected to the destination ApsaraDB for Redis instance. The ECS instance can connect to the destination ApsaraDB for Redis instance.
- The version of the source SSDB database is 1.9.2 and later.

Context

- Synchronization process
  The source SSDB database acts as the master node in the SSDB cluster, and the ssdb-port tool acts as an SSDB replica node. The source SSDB database synchronizes data to the ssdb-port tool. Then, the ssdb-port tool parses the data, converts the data to the format supported by ApsaraDB for Redis, and sends the data to the ApsaraDB for Redis instance that is specified in the configuration file. The following figure shows the synchronization process.

  After full synchronization is complete, incremental data in the SSDB database is synchronized to the ApsaraDB for Redis instance before ssdb-port is disconnected from the instance.

Note

- If you run a command that is not supported by the ssdb-port tool to update data in the source SSDB database, the updated data cannot be synchronized to the ApsaraDB for Redis instance. The following list shows the SSDB commands supported by the ssdb-port tool.
- If you want to use more commands, submit a suggestion.

- Commands supported by the ssdb-port tool
  - set
  - setx
  - setnx
  - expire
  - del
  - get
  - incr
  - qpop_front
  - qpush_front
  - qclear
  - qtrim_front
  - qtrim_back
  - zset
  - zdel
  - zincr
  - multi_zdel
  - multi_zset
  - hset
  - hdel
  - hclear
  - multi_hset
  - multi_hdel
  - hincr

Limits

If you run the hset or hget command and the keys in the destination ApsaraDB for Redis instance are in Chinese, the modified data cannot be synchronized. This limit does not apply to other supported commands.

Procedure

1. Run the following commands in the ECS instance to download and decompress the ssdb-port.tar.gz package:

```
# wget http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/attach/94155/cn_zh/1547
627852086/ssdb-port.tar.gz
# tar -xvf ssdb-port.tar.gz
# cd ssdb-port
```

2. Run the following command to modify the configuration file of the ssdb-port tool based on the example:

   ```
   vi ssdb_port.conf
   ```

   The following sample code shows the *ssdb_port.conf* file. Modify the connection information of the source SSDB database and the destination ApsaraDB for Redis instance based on the comments.

```
# ssdb-server config for replica
# MUST indent by TAB!
# relative to path of this file, directory must exists
work_dir = ./var_ssdb_port
pidfile = ./var_ssdb_port/ssdb.pid
# The connection information of the ssdb-port tool, which does not need to be modified.
server:
    ip: 127.0.0.1
    port: 8890
    #readonly: yes
replication:
    binlog: yes
        capacity: 100000000
    # Limit sync speed to *MB/s, -1: no limit
    sync_speed: -1
    slaveof:
        # to identify a master even if it moved(ip, port changed)
        # if set to empty or not defined, ip:port will be used.
        id: svc_1
        # sync|mirror, default is sync
        type: sync
        host: localhost # The connection address of the source SSDB database.
        port: 8888 # The port number of the source SSDB database.
        #auth: password
        redis_host: localhost # The connection address of the destination ApsaraDB for
Redis instance.
        redis_port: 6379 # The port number of the destination ApsaraDB for Redis instan
ce.
        redis_auth: password # The password of the destination ApsaraDB for Redis insta
nce.
logger:
    level: debug
    output: log_ssdb_port.txt
    rotate:
        size: 1000000000
leveldb:
    # in MB
    cache_size: 500
    # in MB
    write_buffer_size: 64
    # in MB/s
    compaction_speed: 1000
    # yes|no
    compression: yes
```

3. Run the `./ssdb-port-2.17 ssdb_port.conf` command to start synchronization.
4. Connect to the ApsaraDB for Redis instance to check whether data synchronization is complete.

> **Note** You can use the redis-cli tool or Data Management Service (DMS) to connect to the ApsaraDB for Redis instance. For more information about how to connect to the instance, see Quick Start.

2.3. Synchronize data between ApsaraDB for Redis instances

2.3.1. Configure one-way data synchronization between ApsaraDB for Redis instances

Data Transmission Service (DTS) supports one-way data synchronization between ApsaraDB for Redis instances. This feature is suitable for scenarios such as active geo-redundancy and geo-disaster recovery. This topic describes how to configure one-way data synchronization between ApsaraDB for Redis instances.

> **Warning** After you configure a data synchronization task, do not change the architecture type of the source or destination database. For example, if you change the master-replica architecture to the cluster architecture, data synchronization fails. For more information about the architecture types, see .

Prerequisites

- The engine versions of the source and destination instances are supported by DTS. For more information, see Overview of data synchronization scenarios.

  > **Note** The version of the destination instance must be the same as or later than that of the source instance. If you want to synchronize data between different versions of instances, make sure that the versions of the source and destination instances are compatible. For example, you can create a destination pay-as-you-go instance to verify the compatibility between the source and destination instances. Then, you can release this instance or change the billing method of the instance to subscription.

- The available storage space of the destination ApsaraDB for Redis instance is larger than the total size of the data in the source ApsaraDB for Redis instance.
- If the source instance is a persistent memory-optimized instance of the ApsaraDB for Redis Enhanced Edition (Tair), the appendonly parameter must be set to ON.
- The source instance cannot be a storage-optimized instance of the ApsaraDB for Redis Enhanced Edition (Tair). In this scenario, a storage-optimized instance of ApsaraDB for Redis Enhanced Edition (Tair) can be used only as the destination instance.

Precautions

- DTS uses the resources of the source and destination instances during initial full data synchronization. This may increase the loads of the database servers. If you migrate a large amount of data or if the server specifications cannot meet your requirements, database services may become unavailable. Before you synchronize data, evaluate the impact of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the

number of keys in the destination database.

> **Note** The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- If the engine version of the source instance is 2.8, incremental data synchronization is not supported.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command on the source instance during data synchronization. If you run one of the two commands, data inconsistency may occur between the source and destination instances.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction` , data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- During data synchronization, if the number of shards in the source instance is changed or if the database specifications are changed (for example, the memory capacity is scaled up), you must reconfigure the data synchronization task. To ensure data consistency, we recommend that you clear the data that has been synchronized to the destination instance before you reconfigure the task.
- During data synchronization, if the endpoint of the source instance is changed, you must submit a ticket to update the endpoint change. Instance operations that may cause endpoint changes include zone changes and network type changes from classic network to virtual private cloud (VPC). Otherwise, the append-only files (AOFs) of the source instance may be reset. For this reason, you must reconfigure the task.
- Limits on synchronizing a standalone ApsaraDB for Redis instance to an ApsaraDB for Redis cluster instance: Each command can be run only on a single slot in an ApsaraDB for Redis cluster instance. If you perform operations on multiple keys in the source database and the keys belong to different slots, an error occurs.

  ```
  CROSSSLOT Keys in request don't hash to the same slot
  ```

  We recommend that you perform operations on only one key during data synchronization. Otherwise, the synchronization task will be interrupted.

Supported synchronization topologies

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

For more information, see Synchronization topologies.

SQL operations that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM

- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, and PSETEX
- RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE
- SWAPDB and UNLINK (supported only if the engine version of the source instance is 4.0)

> Note
>
> - PUBLISH operations cannot be synchronized.
> - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
> - When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Permissions required for database accounts

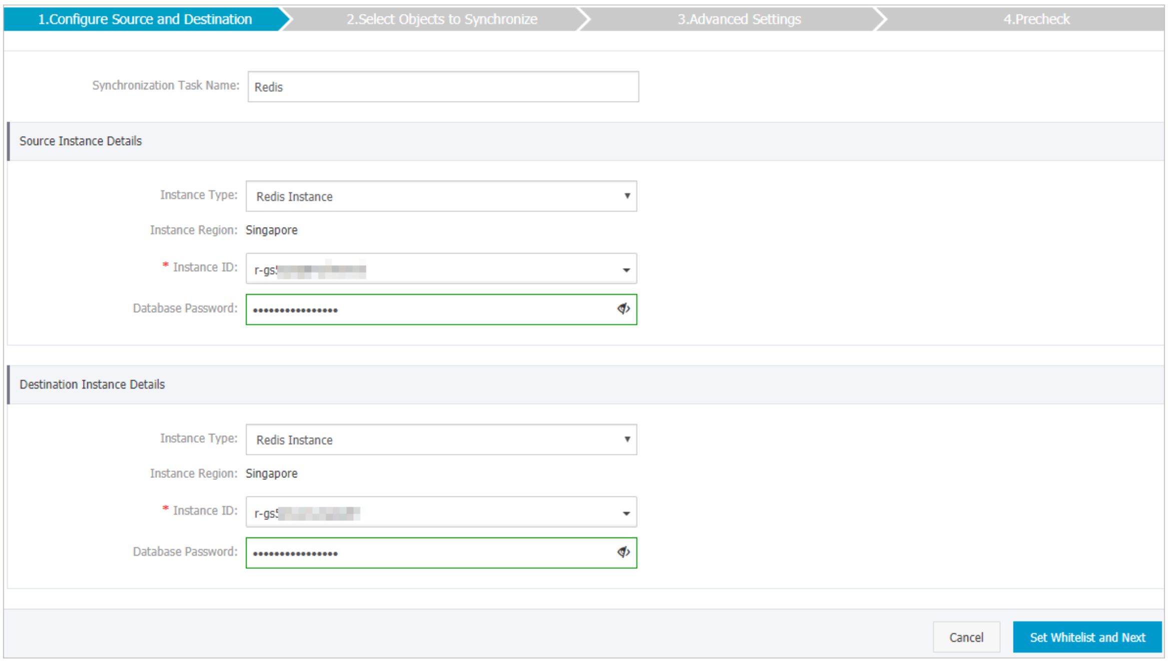
Database	Permission and authorization method
Source ApsaraDB for Redis instance	The database account must have read permissions. For more information about how to grant the required permissions to an account, see Create and manage database accounts.
Destination ApsaraDB for Redis instance	The database account must have read and write permissions. For more information about how to grant the required permissions to an account, see Create and manage database accounts.

Procedure

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

   > Note On the buy page, set both Source Instance and Destination Instance to **Redis**.

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 Task** in the Actions column.
6. Configure the source and destination instances.



1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Task Name: Redis

Source Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs[redacted]

Database Password: ••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance

Instance Region: Singapore

* Instance ID: r-gs[redacted]

Database Password: ••••••••••••••••

Cancel
Set Whitelist and Next

Section	Parameter	Description
N/A	Synchronization Task Name	The task name that DTS automatically generates. We recommend that you specify a descriptive name that makes it easy to identify the task. You do not need to use a unique task name.
Source Instance Details	Instance Type	Select **Redis Instance**.
	Instance Region	The source region that you selected on the buy page. The value of this parameter cannot be changed.
	Instance ID	The ID of the source instance.
	Database Password	The database password of the source instance. For information about the permissions that are required for the database account, see Permissions required for database accounts. **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.
Destination	Instance Type	Select **Redis Instance**.
	Instance Region	The destination region that you selected on the buy page. The value of this parameter cannot be changed.
	Instance ID	The ID of the destination instance.

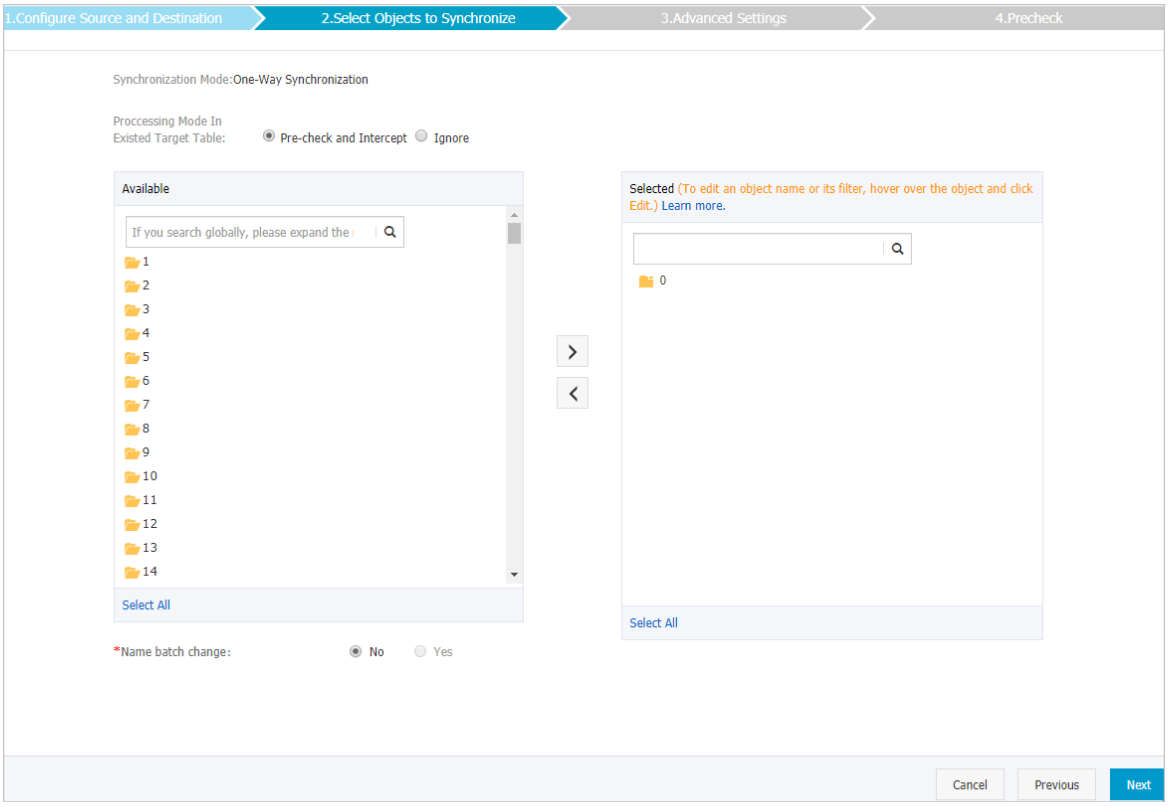
Section	Parameter	Description
Destination Instance Details	Database Password	The database password of the destination instance. For information about the permissions that are required for the database account, see Permissions required for database accounts. ⓘ **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

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

   ⓘ **Note**

   - You do not need to modify the security settings for ApsaraDB instances (such as and) and databases that are hosted on Elastic Compute Service (ECS). DTS automatically adds the CIDR blocks of DTS servers to the whitelists of ApsaraDB instances or the security group rules of ECS instances. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases.
   - After data synchronization is complete, we recommend that you remove the CIDR blocks of DTS servers from the whitelists or security groups.

8. Select the processing mode of conflicting tables and the objects to synchronize.



1.Configure Source and Destination > 2.Select Objects to Synchronize > 3.Advanced Settings > 4.Precheck

Synchronization Mode:One-Way Synchronization

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

Available

If you search globally, please expand the

1
2
3
4
5
6
7
8
9
10
11
12
13
14

Select All

Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.

0

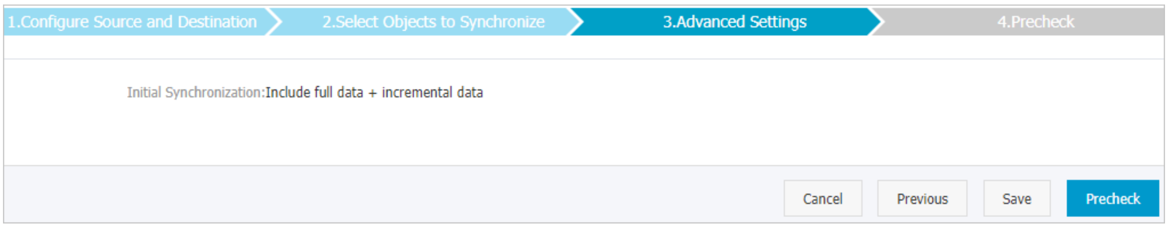
Select All

*Name batch change: ● No ○ Yes

Setting	Description
Select the processing mode of conflicting tables	○ **Pre-check and Intercept**: checks whether the destination instance is empty. If the destination instance is empty, the precheck is passed. If the instance is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ○ **Ignore**: skips the check for empty destination instances. **Warning** If you select **Ignore**, data records in the source instance overwrite the data records that have the same keys in the destination instance. Proceed with caution.
Select the objects to synchronize	○ Select one or more databases from the **Available** section and click the > icon to add the databases to the **Selected** section. ○ You can select only databases as objects to synchronize. Keys cannot be selected as objects to synchronize.
Rename Databases and Tables	In this scenario, you cannot rename objects.
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. Select initial synchronization types. The value is set to **Include full data + incremental data** and cannot be changed.



Initial Synchronization:Include full data + incremental data

1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Previous
Save
Precheck

Note

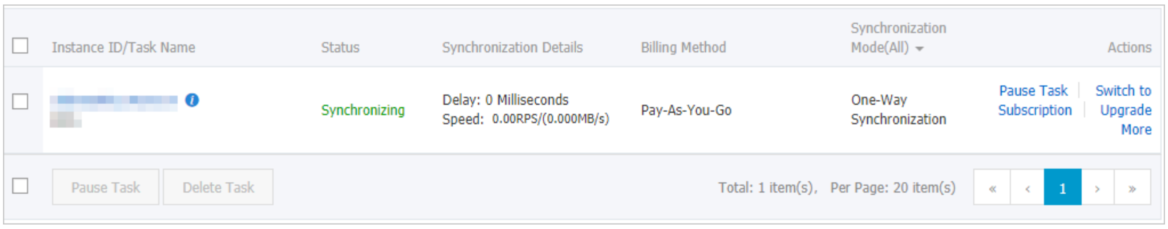
- DTS synchronizes historical data from the source instance to the destination instance. Then, DTS synchronizes incremental data.
- If a version-related error message is displayed, you must upgrade the source instance to a specified version. For more information about how to upgrade the version, see Upgrade the major version and Update the minor version.

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.
  - 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 **The precheck is passed.** message is displayed in the **Precheck** dialog box. Then, the data synchronization task starts.
13. Wait until initial synchronization is complete and the data synchronization task enters the **Synchronizing** state.



□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All) ▾	Actions
□	[redacted]	Synchronizing	Delay: 0 Milliseconds Speed: 0.00RPS/(0.000MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task Switch to Subscription Upgrade More

□ Pause Task Delete Task   Total: 1 item(s), Per Page: 20 item(s) « ‹ 1 › »

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

2.3.2. Configure two-way data synchronization between ApsaraDB for Redis Enhanced Edition (Tair) instances

This topic describes how to configure two-way data synchronization between ApsaraDB for Redis Enhanced Edition (Tair) instances by using Data Transmission Service (DTS). The data synchronization feature is applicable to scenarios such as active geo-redundancy and geo-disaster recovery.

Prerequisites

The source and destination instances are ApsaraDB for Redis Enhanced Edition (Tair) instances (version 5.0).

> Note
>
> - If the ApsaraDB for Redis instance is an Enhanced Edition instance (storage-optimized instance), it can be used only as the destination database.
> - If the source ApsaraDB for Redis instance is an Enhanced Edition instance (persistent memory-optimized instance), you must set the appendonly parameter to yes.
> - ApsaraDB for Redis Enhanced Edition (Tair) supports the **cluster**, **standard**, and **read/write splitting** architectures.

Precautions

- During two-way data synchronization, the data synchronization task in the forward direction performs and . The data synchronization task in the reverse direction performs only incremental data synchronization.

  > Warning To ensure data consistency, do not modify or write data to the same key in the source and destination databases when the two-way data synchronization tasks are running.

  full data synchronizationincremental data synchronization

- DTS uses the resources of the source and destination databases during full data synchronization.

This may increase the loads of the database servers. If you synchronize a large volume of data or the server specifications cannot meet your requirements, the database services may become unavailable. Before you synchronize data, evaluate the impacts of data synchronization on the performance of the source and destination instances. We recommend that you synchronize data during off-peak hours.

- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command in the source instance during data synchronization. Otherwise, data may become inconsistent between the source and destination instances.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction` , data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

  Note The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- If direct connection is disabled for the destination ApsaraDB for Redis instance, DTS uses the proxy forwarding mode to write data to the destination instance.

  Note For more information about how to enable direct connection, see Enable the direct connection mode.

- During data synchronization, if the number of shards in the source or destination ApsaraDB for Redis instance is increased or decreased, or if the database specifications are changed (for example, the memory capacity is scaled up), you must reconfigure the task. To ensure data consistency, we recommend that you clear the data that has been synchronized to the source and destination Redis databases before you reconfigure the task.
- During data synchronization, if the endpoint of the source or destination ApsaraDB for Redis instance is changed (for example, the zone of the instance is changed or the network type is changed from classic network to VPC), you must submit a ticket to update the change. Otherwise, the append-only files (AOF) of the source or destination ApsaraDB for Redis instance may be reset. In this case, you must reconfigure the task.
- If the source or destination instance of a two-way data synchronization task resides in a region outside the Chinese mainland, data can be synchronized only within this region. Cross-region two-way synchronization is not supported. For example, if the source instance resides in the Japan (Tokyo) region, data can be synchronized only within the Japan (Tokyo) region and cannot be synchronized to the Germany (Frankfurt) region.
- Limits on synchronizing a standalone ApsaraDB for Redis instance to an ApsaraDB for Redis cluster instance: Each command can be run only on a single slot in an ApsaraDB for Redis cluster instance. If you perform operations on multiple keys in the source database and the keys belong to different slots, an error occurs.

  ```
  CROSSSLOT Keys in request don't hash to the same slot
  ```

  We recommend that you perform operations on only one key during data synchronization. Otherwise, the synchronization task will be interrupted.

Operations that can be synchronized

- APPEND
- BITOP, BLPOP, BRPOP, and BRPOPLPUSH
- DECR, DECRBY, and DEL
- EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT
- GEOADD and GETSET
- HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX
- INCR, INCRBY, and INCRBYFLOAT
- LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM
- MOVE, MSET, MSETNX, and MULTI
- PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, and PSETEX
- RENAME, RENAMENX, RPOP, RPOPLPUSH, RPUSH, and RPUSHX
- SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE
- UNLINK, ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE
- SWAPDB (This operation is not supported if the source or destination ApsaraDB for Redis instance is deployed in the cluster architecture.)

Notice

- PUBLISH operations cannot be synchronized.
- If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination database. During incremental data synchronization, the destination database does not explicitly return the execution results of Lua scripts.
- When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data. As a result, the destination instance may contain duplicate data records.

Permissions required for database accounts

Database	Permissions and authorization method
Source ApsaraDB for Redis instance	The database accounts of the source and destination instances must have the read and write permissions. For more information about how to authorize a database account, see Create and manage database accounts.
Destination ApsaraDB for Redis instance	

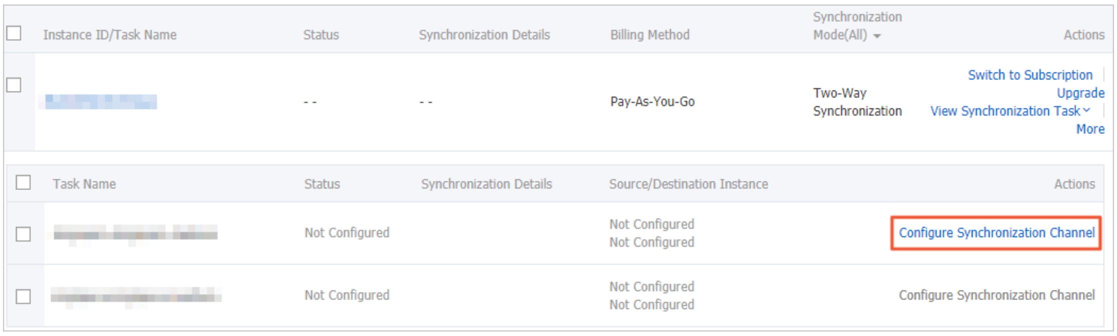
Procedure

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

   Notice On the buy page, set Source Instance to **Redis**, set Destination Instance to **Redis**, 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. Configure the data synchronization task in the forward direction.
   i. Find the data synchronization instance, and click **Configure Task** in the **Actions** column of the first data synchronization task.

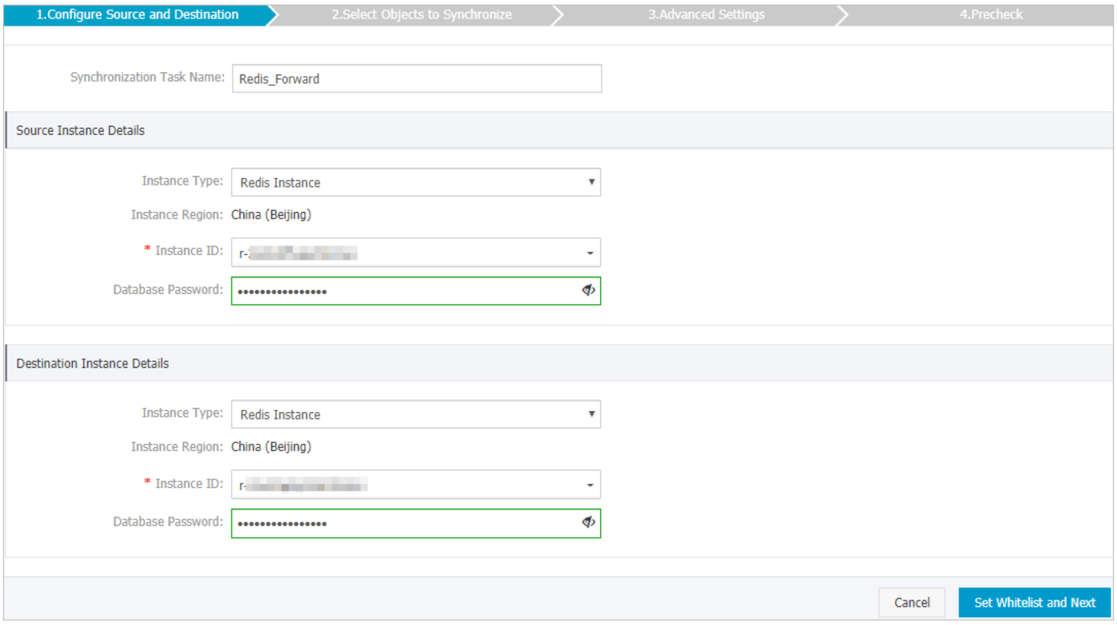
> **Notice** A two-way data synchronization instance contains two data synchronization tasks. You must set parameters for each task. When you configure the second data synchronization task, find the task and click **Configure Task** in the **Actions** column.



□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
□	[redacted]	- -	- -	Pay-As-You-Go	Two-Way Synchronization	Switch to Subscription Upgrade View Synchronization Task More

□	Task Name	Status	Synchronization Details	Source/Destination Instance	Actions
□	[redacted]	Not Configured		Not Configured Not Configured	Configure Synchronization Channel
□	[redacted]	Not Configured		Not Configured Not Configured	Configure Synchronization Channel

   ii. Configure the source and destination instances.



1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Synchronization Task Name: Redis_Forward

Source Instance Details

Instance Type: Redis Instance
Instance Region: China (Beijing)
* Instance ID: r-[redacted]
Database Password: ••••••••••••••••

Destination Instance Details

Instance Type: Redis Instance
Instance Region: China (Beijing)
* Instance ID: r-[redacted]
Database Password: ••••••••••••••••

Cancel
Set Whitelist and Next

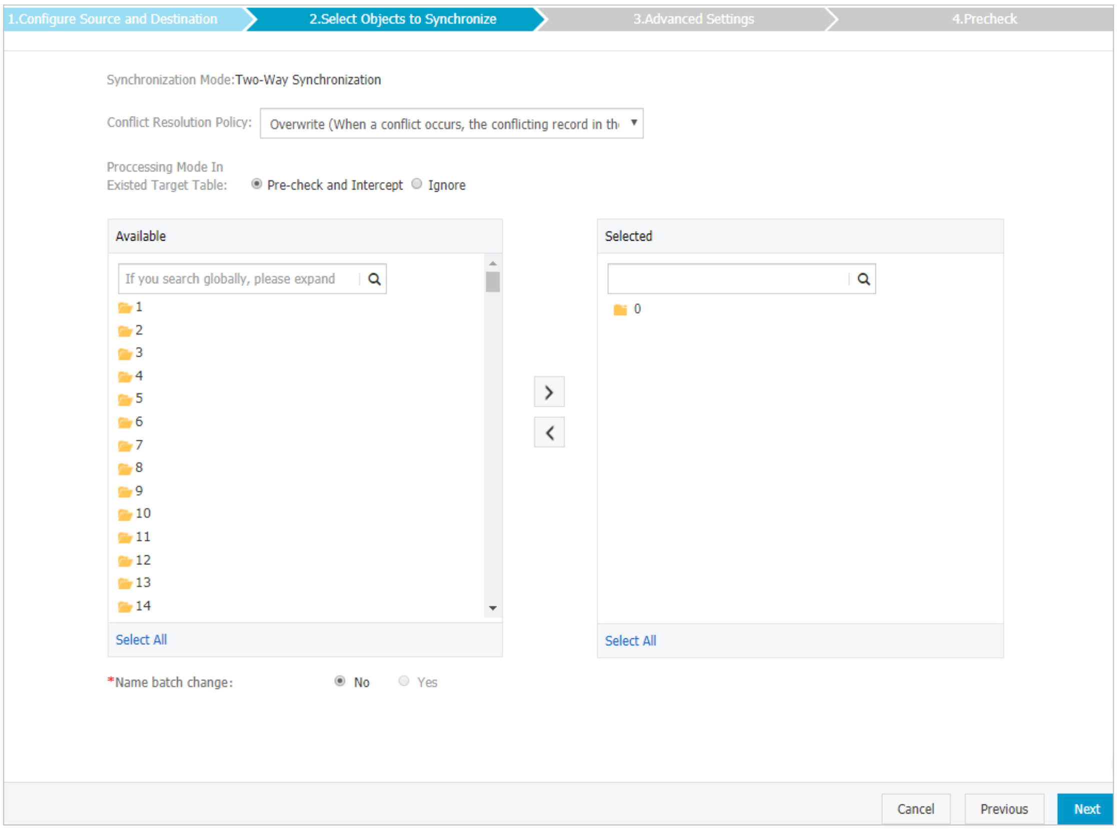
Section	Parameter	Description
N/A	Synchronization Task Name	DTS automatically generates a task name. We recommend that you specify an informative name to identify the task. You do not need to use a unique task name.

Section	Parameter	Description
Source Instance Details	Instance Type	Select **Redis 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 ApsaraDB for Redis instance. **Notice** When you configure the data synchronization task in the reverse direction, select the ID of the destination ApsaraDB for Redis instance.
	Database Password	Enter the database password of the ApsaraDB for Redis instance. For information about the permissions that are required for the account, see Permissions required for database accounts. **Notice** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.
Destination Instance Details	Instance Type	Select **Redis 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 ApsaraDB for Redis instance. **Notice** When you configure the data synchronization task in the reverse direction, select the ID of the source ApsaraDB for Redis instance.
	Database Password	Enter the database password of the ApsaraDB for Redis instance. For information about the permissions that are required for the account, see Permissions required for database accounts. **Notice** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

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

> **Note**
>
> - You do not need to modify the security settings for ApsaraDB instances (such as and) and ECS-hosted databases. DTS automatically adds the CIDR blocks of DTS servers to the whitelists of ApsaraDB instances or the security group rules of Elastic Compute Service (ECS) instances. For more information, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases.
> - After data synchronization is completed, we recommend that you remove the CIDR blocks of DTS servers from the whitelists or security groups.

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



1.Configure Source and Destination › 2.Select Objects to Synchronize › 3.Advanced Settings › 4.Precheck

Synchronization Mode:Two-Way Synchronization

Conflict Resolution Policy: Overwrite (When a conflict occurs, the conflicting record in th ▾

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

*Name batch change: ● No ○ Yes

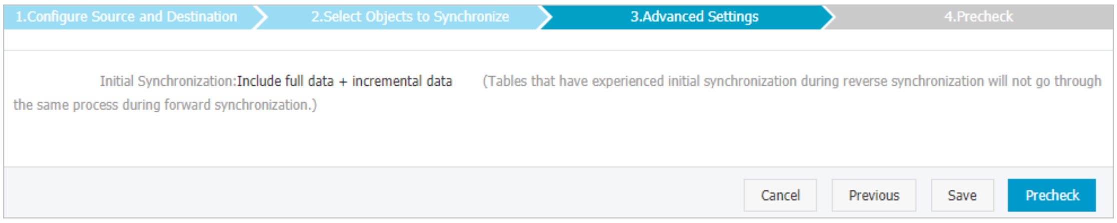
Setting	Parameter	Description
Select	Conflict Resolution Policy	**Overwrite** (Conflicting records in the destination instance are overwritten) During data synchronization, if data records have the same key but different values, the data record with the latest key value overwrites the conflicting records.

Setting	Parameter	Description
the synchronization policy	Select the processing mode of conflicting tables	■ **Pre-check and Intercept**: checks whether the destination instance is empty. If the destination database is empty, the precheck is passed. If the database is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ■ **Ignore**: skips the check for empty destination databases. **Warning** If you select **Ignore**, the data records in the source database overwrite the data records with the same keys in the destination database. Proceed with caution.
Select the objects to be synchronized	N/A	■ Select one or more databases from the **Available** section and click > icon to move the databases to the **Selected** section. ■ You can select only databases as the objects to be synchronized. You cannot select keys as the objects to be synchronized.
Rename Databases and Tables	N/A	In this scenario, you cannot rename objects.
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.

v. In the lower-right corner of the page, click **Next**.

vi. Select the initial synchronization types.



1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck

Initial Synchronization:Include full data + incremental data (Tables that have experienced initial synchronization during reverse synchronization will not go through the same process during forward synchronization.)

Cancel
Previous
Save
Precheck

The value is set to **Include full data + incremental data**. DTS synchronizes historical data from the source ApsaraDB for Redis instance to the destination ApsaraDB for Redis instance. Then, DTS synchronizes incremental data.

Notice

- If all the required objects have been synchronized from the source instance to the destination instance, the data synchronization task in the reverse direction synchronizes only incremental data.
- If a version-related error message appears, you can upgrade the source ApsaraDB for Redis instance to a specified version. For more information, see Upgrade the major version and Update the minor version.

vii. 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.

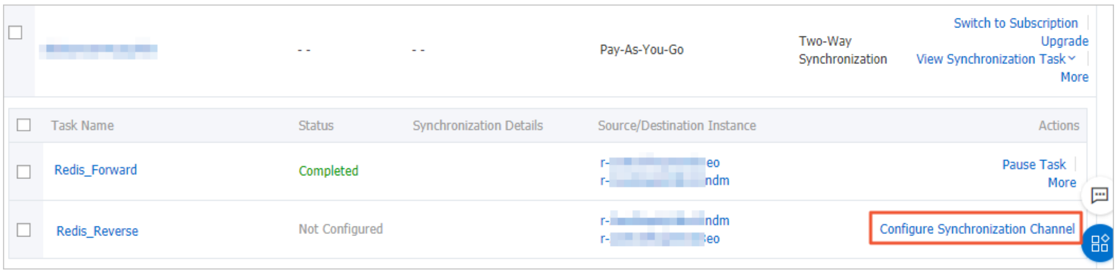
viii. Close the **Precheck** dialog box after the following message is displayed: **Precheck Passed**. Then, the data synchronization task in the forward direction starts.

6. 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.

7. Configure the data synchronization task in the reverse direction.

   i. Find the second data synchronization task, and click **Configure Synchronization Channel** in the Actions column.



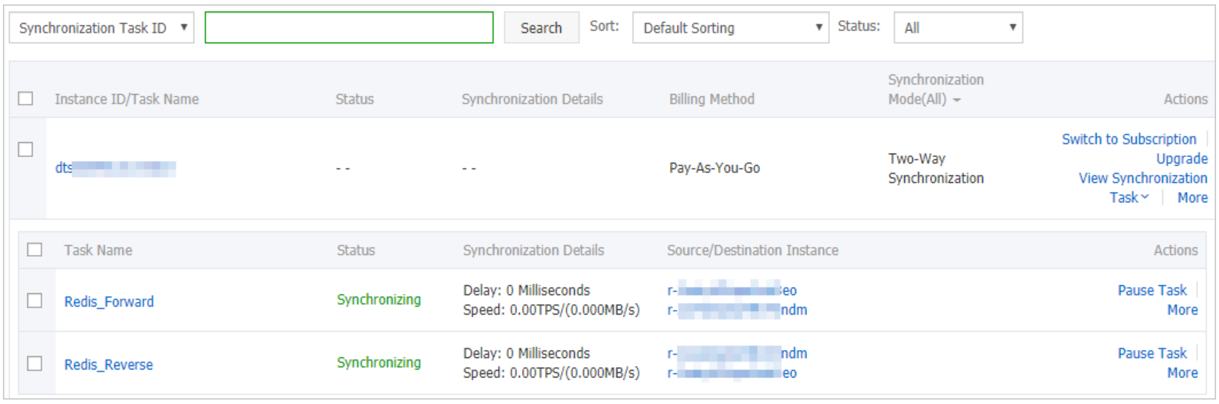
☐	[redacted]	- -	- -	Pay-As-You-Go	Two-Way Synchronization	Switch to Subscription Upgrade View Synchronization Task More

☐	Task Name	Status	Synchronization Details	Source/Destination Instance	Actions
☐	Redis_Forward	Completed		r-[redacted]eo r-[redacted]ndm	Pause Task More
☐	Redis_Reverse	Not Configured		r-[redacted]ndm r-[redacted]eo	Configure Synchronization Channel

   ii. Repeat substeps ii to viii that are described in Step 5.

Result

After a period of time, both data synchronization tasks are in the **Synchronizing** state.



☐	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All)	Actions
☐	dts[redacted]	- -	- -	Pay-As-You-Go	Two-Way Synchronization	Switch to Subscription Upgrade View Synchronization Task More

☐	Task Name	Status	Synchronization Details	Source/Destination Instance	Actions
☐	Redis_Forward	Synchronizing	Delay: 0 Milliseconds Speed: 0.00TPS/(0.000MB/s)	r-[redacted]eo r-[redacted]ndm	Pause Task More
☐	Redis_Reverse	Synchronizing	Delay: 0 Milliseconds Speed: 0.00TPS/(0.000MB/s)	r-[redacted]ndm r-[redacted]eo	Pause Task More

2.3.3. Synchronize data between ApsaraDB for Redis cluster instances across Alibaba Cloud accounts

You can use Data Transmission Service (DTS) to perform one-way synchronization between ApsaraDB for Redis cluster instances (Redis instances in short) that belong to different Alibaba Cloud accounts. The data synchronization feature is applicable to scenarios such as resource migration or resource merging across Alibaba Cloud accounts and business architecture adjustment.

Prerequisites

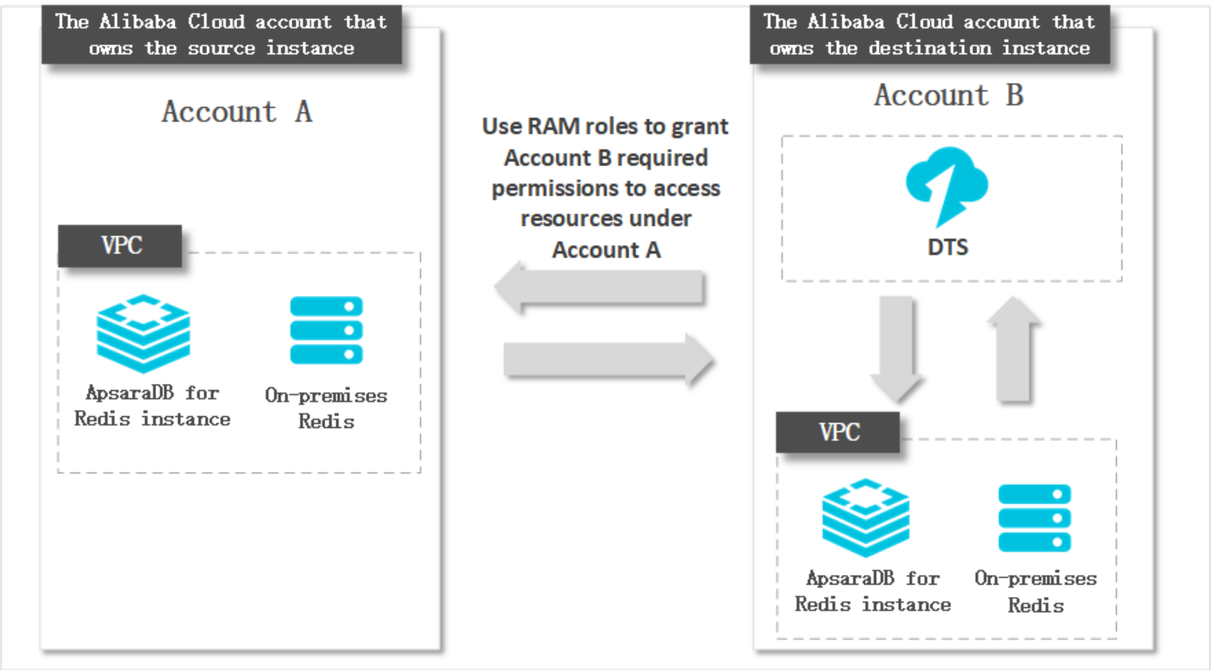
- If the source Redis instance is a Community Edition instance, the engine version must be 4.0 or 5.0. If the source Redis instance is an Enhanced Edition instance, the engine version must be 5.0.

  > **Note** The engine version of the destination Redis instance can be 4.0 or 5.0. The engine version of the destination Redis instance cannot be earlier than the engine version of the source Redis instance. Before you synchronize data between two Redis instances that use different engine versions, check the compatibility of the two versions. For example, you can create a destination Redis instance based on the pay-as-you-go billing method to verify the compatibility between the source and destination instances. Then, you can release this instance or change the billing method of the instance to subscription.

- The source Redis instance is deployed in a virtual private cloud (VPC). If the source instance is deployed in the classic network, you can change the network type to VPC. For more information, see Change the network type from classic network to VPC.
- Secure Sockets Layer (SSL) encryption is disabled for the source Redis instance. For more information, see Configure SSL encryption.
- The available storage space of the destination Redis instance is larger than the total size of the data in the source Redis instance.
- ApsaraDB for Redis Enhanced Edition (Tair) is integrated with more Redis modules than ApsaraDB for Redis Community Edition. For more information, see Integration of multiple Redis modules. To ensure compatibility between the source and destination instances, the edition of the destination instance must be Enhanced if the edition of the source instance is Enhanced.
-
-

Context

Two Redis instances are created by different Alibaba Cloud accounts. You need to migrate data from the instance of Account A to the instance of Account B. The following figure shows the architecture of the migration solution.


The Alibaba Cloud account that owns the source instance
Account A
VPC
ApsaraDB for Redis instance
On-premises Redis
Use RAM roles to grant Account B required permissions to access resources under Account A
The Alibaba Cloud account that owns the destination instance
Account B
DTS
VPC
ApsaraDB for Redis instance
On-premises Redis

> **Notice** To synchronize data between Redis instances of different accounts, note that the source and destination instances must be deployed in the cluster or standard architecture. In this scenario, the read/write splitting architecture is not supported.

DTS provides the following data migration solution:

You cannot use the data migration feature to migrate data between ApsaraDB for Redis cluster instances. In this solution, the data synchronization feature of DTS is used to synchronize data to the destination instance. The following table describes how to configure a data synchronization task.

Step	Description
1. Use Account A to log on to the Alibaba Cloud Management Console and grant the required permission to a RAM role. For more information, see Step 1 in Before you begin.	When you configure the RAM role, set Account B as the trusted account, and authorize the RAM role to access the resources of Account A.
2. Use Account A to log on to the Alibaba Cloud Management Console and prepare the environment that is required for the source Redis instance. For more information, see Steps 2 to 4 in Before you begin.	Add the CIDR blocks of DTS servers to the whitelist of the source Redis instance. Apply for a private endpoint for the source Redis instance, and create a database account that is authorized to **replicate** the data of the source Redis instance. **Note** By default, you are not allowed to create a database account that is authorized to **replicate** the data of an ApsaraDB for Redis cluster instance. To do this, submit a ticket.

Step	Description
3. Use Account B to log on to the Alibaba Cloud Management Console and configure the data synchronization task. For more information, see Procedure.	DTS cannot read the information of source Redis instance across Alibaba Cloud accounts. When you configure the data synchronization task, you must specify the source instance as a self-managed database that is connected to DTS over Express Connect.

Precautions

- DTS uses the resources of the source and destination databases during full data synchronization. This may increase the loads of the database servers. If you synchronize a large volume of data or the server specifications cannot meet your requirements, the database services may become unavailable. 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.
- We recommend that you do not run the `FLUSHDB` or `FLUSHALL` command on the source instance during data synchronization. If you run one of the two commands, data may become inconsistent between the source and destination instances.
- If the data eviction policy (`maxmemory-policy`) of the destination instance is not set to `noeviction` , data inconsistency may occur between the source and destination instances. For more information about data eviction policies, see How does ApsaraDB for Redis evict data by default?
- If an expiration policy is enabled for some keys in the source database, these keys may not be deleted in a timely manner after they expired. Therefore, the number of keys in the destination database may be less than that in the source database. You can run the info command to view the number of keys in the destination database.

  Note The number of keys that do not have an expiration policy or have not expired is the same in the source and destination databases.

- During data synchronization, if the number of shards in the source instance is changed or if the database specifications are changed (for example, the memory capacity is scaled up), you must reconfigure the data synchronization task. To ensure data consistency, we recommend that you clear the data that has been synchronized to the destination instance before you reconfigure the task.
- During data synchronization, if the endpoint of the source instance is changed, you must submit a ticket to update the endpoint change. Instance operations that may cause endpoint changes include zone changes and network type changes from classic network to virtual private cloud (VPC). Otherwise, the append-only files (AOFs) of the source instance may be reset. For this reason, you must reconfigure the task.

Operations that can be synchronized

Edition	Operation

Edition	Operation
ApsaraDB for Redis Community Edition	• APPEND • BITOP, BLPOP, BRPOP, and BRPOPLPUSH • DECR, DECRBY, and DEL • EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT • GEOADD and GETSET • HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX • INCR, INCRBY, and INCRBYFLOAT • LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM • MOVE, MSET, MSETNX, and MULTI • PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, and PSETEX • RENAME, RENAMENX, RESTORE, RPOP, RPOPLPUSH, RPUSH, and RPUSHX • SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE • ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE • SWAPDB and UNLINK (supported only if the engine version of the source Redis instance is 4.0)
ApsaraDB for Redis Enhanced Edition (Tair)	• APPEND • BITOP, BLPOP, BRPOP, and BRPOPLPUSH • DECR, DECRBY, and DEL • EVAL, EVALSHA, EXEC, EXPIRE, and EXPIREAT • GEOADD and GETSET • HDEL, HINCRBY, HINCRBYFLOAT, HMSET, HSET, and HSETNX • INCR, INCRBY, and INCRBYFLOAT • LINSERT, LPOP, LPUSH, LPUSHX, LREM, LSET, and LTRIM • MOVE, MSET, MSETNX, and MULTI • PERSIST, PEXPIRE, PEXPIREAT, PFADD, PFMERGE, and PSETEX • RENAME, RENAMENX, RPOP, RPOPLPUSH, RPUSH, and RPUSHX • SADD, SDIFFSTORE, SELECT, SET, SETBIT, SETEX, SETNX, SETRANGE, SINTERSTORE, SMOVE, SPOP, SREM, and SUNIONSTORE • UNLINK, ZADD, ZINCRBY, ZINTERSTORE, ZREM, ZREMRANGEBYLEX, ZUNIONSTORE, ZREMRANGEBYRANK, and ZREMRANGEBYSCORE

> Note
> - PUBLISH operations cannot be synchronized.
> - If you run the EVAL or EVALSHA command to call Lua scripts, DTS cannot identify whether these Lua scripts are executed on the destination instance. This is because the destination instance does not explicitly return the execution results of Lua scripts during incremental data synchronization.
> - When DTS runs the SYNC or PSYNC command to transfer data of the LIST type, DTS does not clear the existing data in the destination instance. As a result, the destination instance may contain duplicate data records.

Before you begin

Use the Alibaba Cloud account that owns the source Redis instance to log on to the Alibaba Cloud Management Console. Then, perform the following steps:

1. 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.
2. Apply for a private endpoint for the source Redis instance. For more information, see Enable the direct connection mode.
3. Select the CIDR blocks of the DTS servers that are deployed in the region of the source Redis instance. Add these CIDR blocks to the whitelist of the source Redis instance. For more information, see Step 2: Set IP address whitelists.

   > Note For more information about the CIDR blocks of the DTS servers in each region, see Add the CIDR blocks of DTS servers to the security settings of on-premises databases.

4. Create a database account for the source Redis instance.
   1. Submit a ticket to create a database account that is authorized to replicate the data of your ApsaraDB for Redis cluster instance.

      > Note By default, you are not allowed to create a database account that is authorized to **replicate** the data of an ApsaraDB for Redis cluster instance. To do this, submit a ticket.

   2. Create a database account that is authorized to **replicate** data of the source instance. For more information, see Create and manage database accounts.

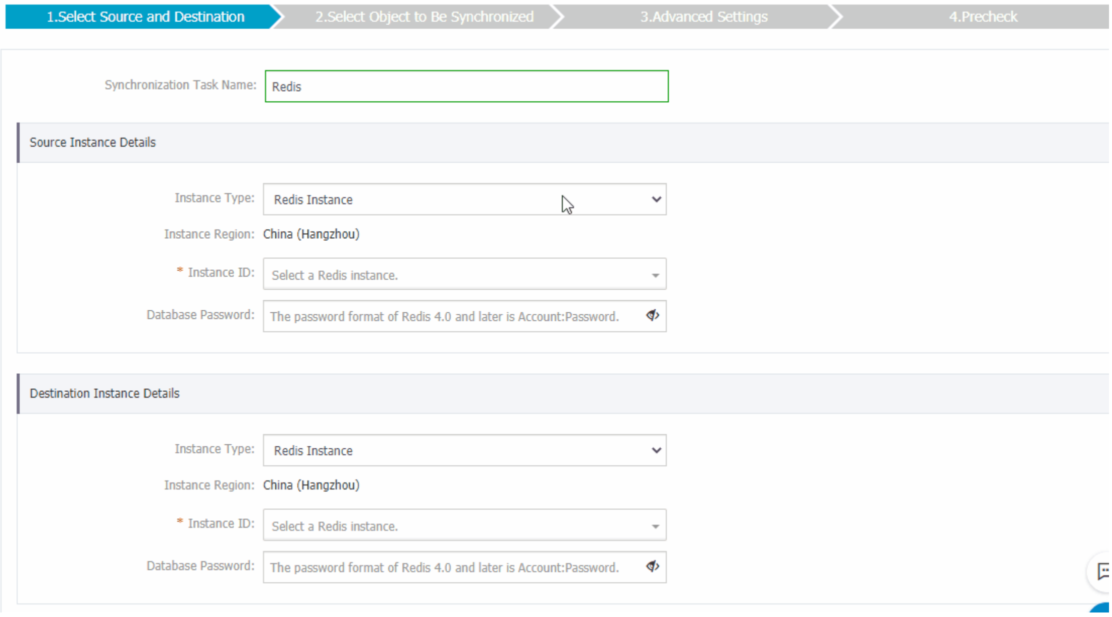
Procedure

1. Use the Alibaba Cloud account that owns the destination Redis instance to log on to the Alibaba Cloud Management Console. Then, purchase a data synchronization instance. For more information, see Purchase procedure.

> **Note** On the buy page, set the following parameters:
>
> - Set Source Instance to **Redis**, set Destination Instance to **Redis**, and set Synchronization Topology to **One-way Synchronization**.
> - Set the Source Region parameter based on the region where the source Redis instance resides, and set the Destination Region parameter based on the region where the destination Redis instance resides.

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.
   i. (Optional)Specify an informative task name for easy identification. You do not need to specify a unique task name.
   ii. 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.

> **Note** DTS cannot read the information of source Redis instance across Alibaba Cloud accounts. When you configure the data synchronization task, you must specify the source instance as a self-managed database that is connected to DTS over Express Connect.


1.Select Source and Destination
2.Select Object to Be Synchronized
3.Advanced Settings
4.Precheck

Synchronization Task Name: Redis

Source Instance Details

Instance Type: Redis Instance

Instance Region: China (Hangzhou)

* Instance ID: Select a Redis instance.

Database Password: The password format of Redis 4.0 and later is Account:Password.

Destination Instance Details

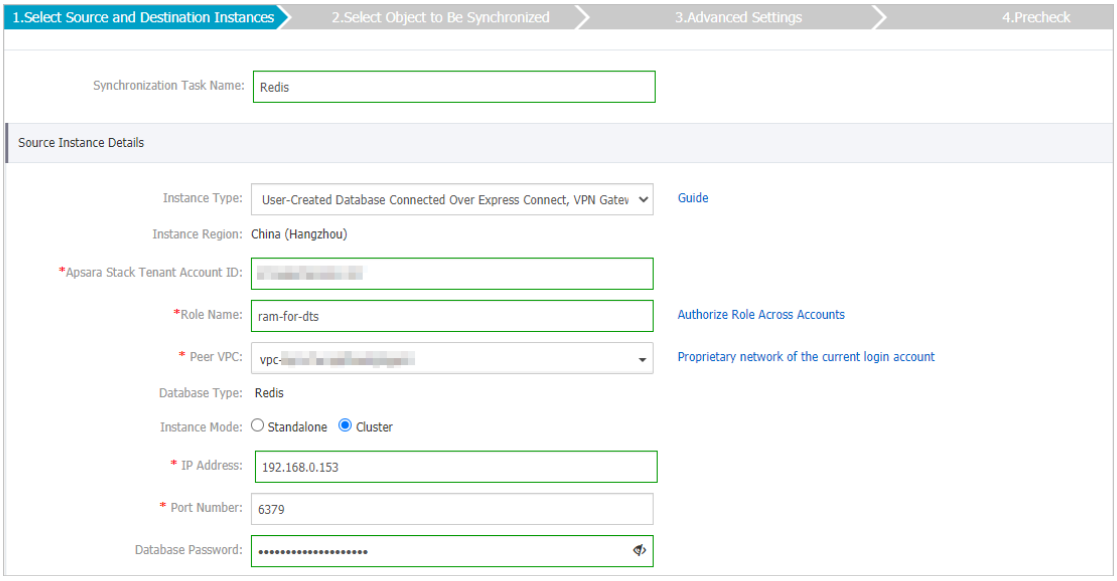
Instance Type: Redis Instance

Instance Region: China (Hangzhou)

* Instance ID: Select a Redis instance.

Database Password: The password format of Redis 4.0 and later is Account:Password.

   iii. Configure the source instance.



1.Select Source and Destination Instances > 2.Select Object to Be Synchronized > 3.Advanced Settings > 4.Precheck

Synchronization Task Name: Redis

Source Instance Details

Instance Type: User-Created Database Connected Over Express Connect, VPN Gatev ⌄ Guide

Instance Region: China (Hangzhou)

*Apsara Stack Tenant Account ID: [redacted]

*Role Name: ram-for-dts Authorize Role Across Accounts

* Peer VPC: vpc-[redacted] Proprietary network of the current login account

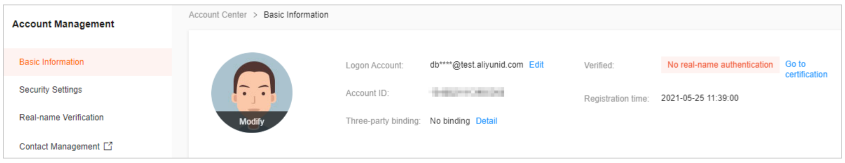
Database Type: Redis

Instance Mode: ○ Standalone ● Cluster

* IP Address: 192.168.0.153

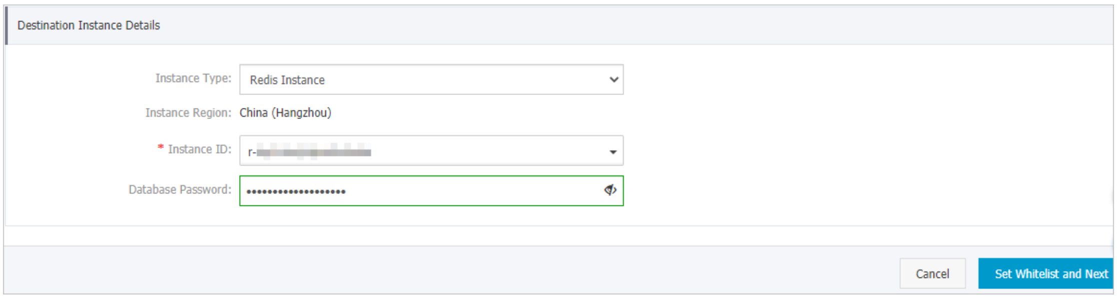
* Port Number: 6379

Database Password: ••••••••••••••••••••

Parameter	Description
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.
Alibaba Cloud Account ID	Enter the ID of the Alibaba Cloud account that owns the source Redis instance. **Note** To obtain the ID of the Alibaba Cloud account that owns the source Redis instance, you must log on to the Account Management console by using this account. The account ID is displayed on the Security Settings page.  Account Management Basic Information Security Settings Real-name Verification Contact Management Account Center > Basic Information Modify Logon Account: db****@test.aliyunid.com Edit Account ID: [redacted] Three-party binding: No binding Detail Verified: No real-name authentication Go to certification Registration time: 2021-05-25 11:39:00
Role Name	Enter the name of the RAM role that you created in Step 1 of Before you begin.
Peer VPC	Select the VPC where the source Redis instance is deployed.
Database Type	Select **Redis**.
Instance Mode	Select **Cluster**.

Parameter	Description
IP Address	Enter the IP address corresponding to the private endpoint that you obtained for the source Redis instance in Step 2 of Before you begin. In this example, enter *192.168.0.153*. **Note** To obtain the IP address corresponding to the private endpoint of your source Redis instance, run the ping command, for example, `ping r-********.redis.rds.aliyuncs.com` .
Port Number	Enter the service port number of the source Redis instance.
Database Password	Enter the password of the account that you created in Step 4 of Before you begin. This account is authorized to **replicate** the data of the source Redis instance. **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

iv. Configure the destination instance.



Destination Instance Details

Instance Type: Redis Instance

Instance Region: China (Hangzhou)

* Instance ID: r-[redacted]

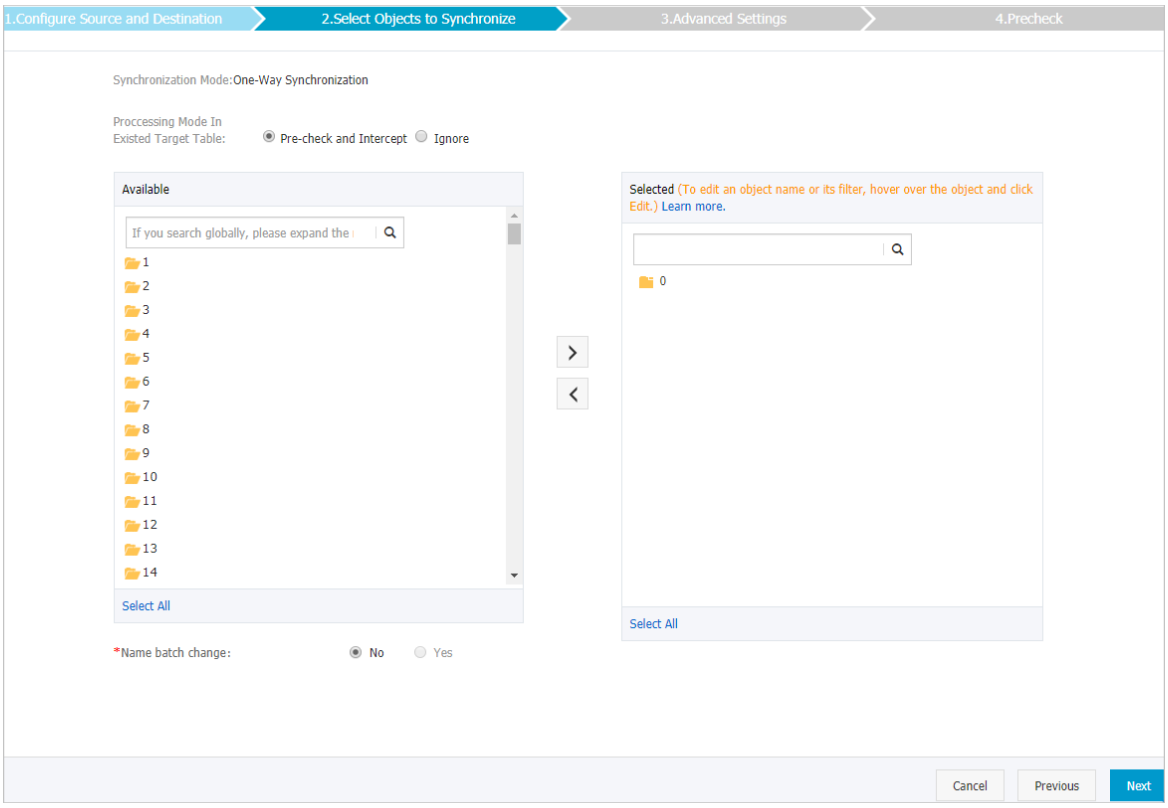
Database Password: ••••••••••••••••••

Parameter	Description
Instance Type	Select **Redis 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 Redis instance.
Database Password	Enter the database password of the destination Redis instance. The account must have the **read and write** permissions on the destination database. **Note** The database password is in the <user>:<password> format. For example, if the username of a custom account is admin and the password is Rp829dlwa, the database password is admin:Rp829dlwa.

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

> **Note** In this step, DTS adds the CIDR blocks of DTS servers to the whitelist of the destination Redis instance. This ensures that the DTS servers can connect to the source and destination Redis instances.

8. Select the processing mode of conflicting tables, and the objects to be synchronized.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Available
If you search globally, please expand the
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Select All
>
<
Selected (To edit an object name or its filter, hover over the object and click Edit.) Learn more.
0
Select All
Cancel
Previous
Next

Synchronization Mode:One-Way Synchronization

Proccessing Mode In Existed Target Table: ● Pre-check and Intercept ○ Ignore

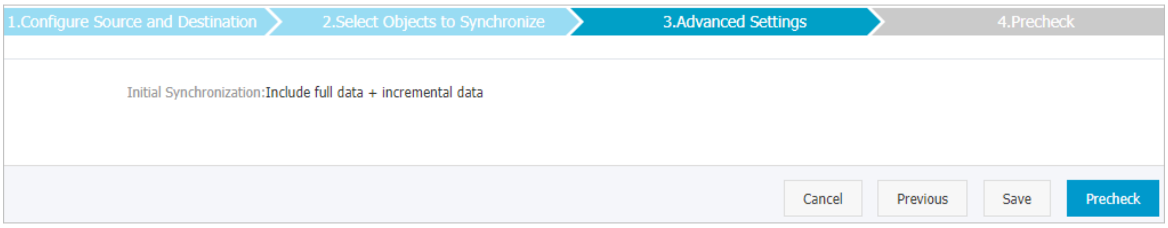
*Name batch change: ● No ○ Yes

Setting	Description
Select the processing mode of conflicting tables	○ **Pre-check and Intercept**: checks whether the destination instance is empty. If the destination instance is empty, the precheck is passed. If the instance is not empty, an error is returned during the precheck and the data synchronization task cannot be started. ○ **Ignore**: skips the check for empty destination instances. **Warning** If you select **Ignore**, data records in the source instance overwrite the data records that have the same keys in the destination instance. Proceed with caution.

Setting	Description
Select the objects to synchronize	○ Select one or more databases from the **Available** section and click the > icon to add the databases to the **Selected** section. ○ You can select only databases as objects to synchronize. Keys cannot be selected as objects to synchronize.
Rename Databases and Tables	In this scenario, you cannot rename objects.
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.


1.Configure Source and Destination
2.Select Objects to Synchronize
3.Advanced Settings
4.Precheck
Cancel
Previous
Save
Precheck

Initial Synchronization:Include full data + incremental data

Note

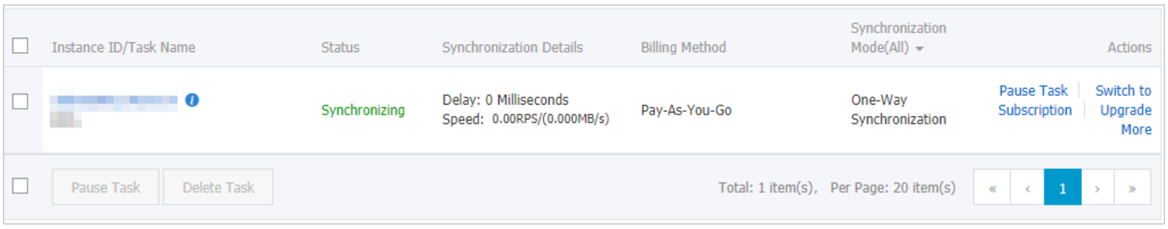
- The value is set to **Include full data + incremental data**. DTS synchronizes historical data of the source Redis instance to the destination Redis instance and then synchronizes incremental data.
- If a version-related error message appears, you can upgrade the source ApsaraDB for Redis instance to a specified version. For more information, see Upgrade the major version and Update the minor version.

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: **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.



□	Instance ID/Task Name	Status	Synchronization Details	Billing Method	Synchronization Mode(All) ▾	Actions
□	[redacted] ⓘ	Synchronizing	Delay: 0 Milliseconds Speed: 0.00RPS/(0.000MB/s)	Pay-As-You-Go	One-Way Synchronization	Pause Task Switch to Subscription Upgrade More

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

3.FAQ about data migration and data synchronization

This topic provides answers to some frequently asked questions about data migration and data synchronization.

- Q: What are data migration and data synchronization?
- Q: Which one can I choose for data migration or data synchronization, Data Transmission Service (DTS) or redis-shake?
- Q: How do data migration and data synchronization work?
- Q: What is the difference between data migration or data synchronization by using DTS?
- Q: How can I resolve precheck errors when I use DTS?
- Q: How can I troubleshoot redis-shake errors?

Q: What are data migration and data synchronization?

- A: Data migration allows you to migrate data stored in key-value pairs from one Redis database to another Redis database. In most cases, the migration task stops after all data is migrated. The migration task is a one-time task.
- Data synchronization allows you to synchronize data stored in key-value pairs from one Redis database to another Redis database in real time. In most cases, the data synchronization task continuously runs to maintain data consistency between the source and destination databases.

> **Note** Data migration and data synchronization replicate data from the source database to the destination database. The data of the source database remains unaffected.

Q: Which one can I choose for data migration or data synchronization, Data Transmission Service (DTS) or redis-shake?

A: We recommend that you use DTS. You can manage data migration or data synchronization tasks in a visualized console. No additional resources are required to deploy software. This simplifies the configuration process of data migration and data synchronization tasks. To solve issues when you use DTS, submit a ticket.

For more information about how to configure data migration or data synchronization tasks, see Overview and Overview.

Q: How do data migration and data synchronization work?

A: For more information, see System architecture and design concepts.

Q: What is the difference between data migration or data synchronization by using DTS?

Item	Data migration	Data synchronization

Item	Data migration	Data synchronization
Scenario	The data migration feature is used to migrate data from on-premises databases, self-managed databases that are hosted on Elastic Compute Service (ECS) instances, and databases on third-party cloud platforms to Alibaba Cloud.	Data synchronization is used for real-time data synchronization between two Redis databases, for example, from Instance A to Instance B. Data synchronization can be used in scenarios such as active geo-redundancy, disaster recovery, and data analytics.
Feature support	Support for data migration.	Support for data synchronization.
Features	You can read and migrate data across self-managed databases that are deployed in virtual private clouds (VPCs) owned by different Alibaba Cloud accounts.	● You can specify how duplicate keys are handled. ● You can configure two-way data synchronization between ApsaraDB for Redis instances ● You can read and synchronize data across self-managed databases that are deployed in VPCs owned by different Alibaba Cloud accounts.
Billing method	Only the pay-as-you-go billing method is supported.	The pay-as-you-go and subscription billing methods are supported.
Billing rules	You are billed when incremental data migration is in progress. This includes the period when incremental data migration is paused. You are not billed for schema migration and full data migration.	● If you use the pay-as-you-go billing method, billing starts after the task is configured and started. You are also billed when the task is paused. ● If you use the subscription billing method, the fee is deducted at the time of purchase based on the selected quantity and configurations.

Note The data synchronization feature of DTS is suitable for more scenarios. We recommend that you use the data synchronization feature of DTS to migrate data. For more information, see Overview.

Q: How can I resolve precheck errors when I use DTS?

A: Before you perform a data migration or data synchronization task, DTS runs a precheck on the source and destination databases to ensure that the task can run as expected. For more information about how to troubleshoot the errors, see Troubleshoot precheck failures and Troubleshoot precheck failures.

Q: How can I troubleshoot redis-shake errors?

A: For more information about how to troubleshoot the errors, see FAQ.