

TencentDB for PostgreSQL

Operation Guide

Product Documentation



Copyright Notice

©2013-2024 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice



All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.

Contents

Operation Guide

Instance Management

- Instance Lifecycle

- Setting Instance Maintenance Time

- Modifying Instance Configuration

- Setting instance availability zone for disaster recovery or region disaster recovery

- Modifying AZs

- Terminating Instances

- Restoring Instances

- Eliminating Instances

- Restarting Instances

- Modifying Data Replication Mode

- Switch Source-Replica Instance

Upgrading Instance

- Kernel Minor Version Upgrade

- Upgrade the Major Version of the Database

Read-Only Instance

- Overview

- Managing RO Groups

- Removal Policy and Load Balancing

Account Management

- Database Privilege Overview

- Users and Permission Operations

- Console Operation Instructions

Database Optimization

- Slow Log Analysis

- Error Logs

Parameter Management

- Setting Instance Parameters

- Parameter Value Limits

- Parameter Template

Log management

- Execution Log Management

Backup and Restoration

- Backup Principles and Solutions

Backing up Data

Downloading Backup

Cloning Instance

Automatic Backup Settings

Restoring PostgreSQL Data on CVMs

Deleting Backup

Viewing Backup Space

Setting Backup Download Rules

Use Serverless Cloud Function to Transfer Historical Backups of PostgreSQL

Rollback to the Original Instance

Physical Migration

Configure Physical Migration Tasks

Physical Migration Check Items

Database Audit

Audit Service Description

View Audit Logs

Modify audit services

Audit Performance Description

Extension Management

Extension Overview

Supported Extensions

Overview

TencentDB for PostgreSQL 9.3

TencentDB for PostgreSQL 9.5

TencentDB for PostgreSQL 10

TencentDB for PostgreSQL 11

TencentDB for PostgreSQL 12

TencentDB for PostgreSQL 13

TencentDB for PostgreSQL 14

TencentDB for PostgreSQL 15 supported extensions

Plugins supported by PostgreSQL 16

pgAgent Extension

postgres_fdw Extension for Cross-database Access

pg_roaringbitmap Extension for Bitwise Operation

pg_cron Extension for Job Scheduling

Network Management

Overview

Modifying Network

Enabling Public Network Address

Access Management

Overview

Access Policy Syntax

Authorizable Resource Types

Console Examples

Data Encryption

TDE Overview

Enabling TDE

Tenant and Resource Isolation

Database Resource Isolation

Security Groups

Managing Security Groups

Associating Instances with Security Groups

Monitoring and Alarms

Monitoring Feature

Alarming Feature

Tag

Overview

Editing Tag

Operation Guide

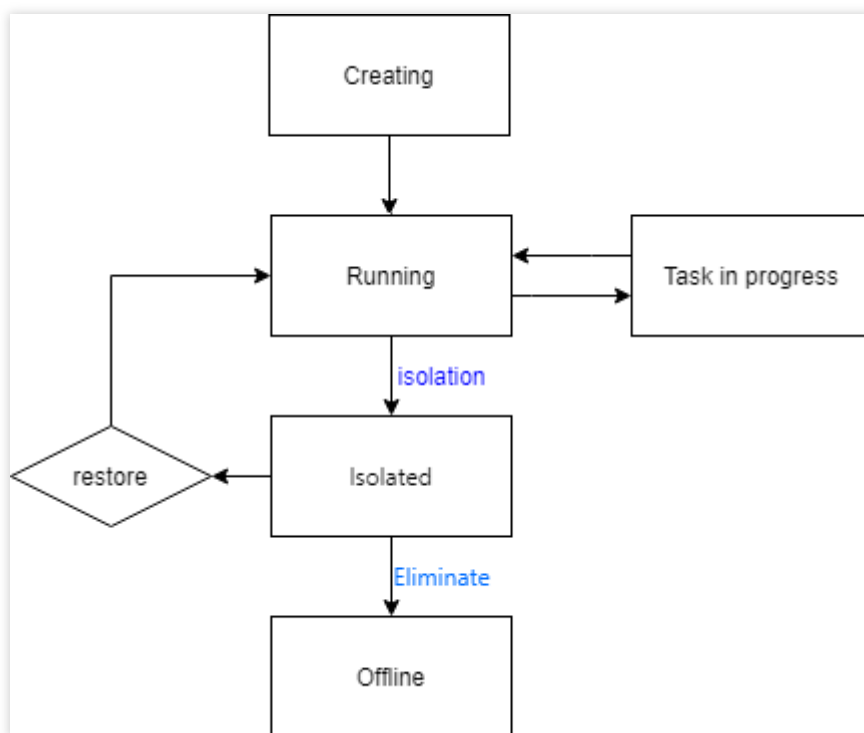
Instance Management

Instance Lifecycle

Last updated : 2024-01-24 11:16:51

TencentDB for PostgreSQL instances have many status. Instances in different status support different operations. This document describes the instance lifecycle.

TencentDB for PostgreSQL instances have the following status:



Creating is the initial status of instances, and instances can be used normally after creation.

Running and **Executing task** mean that the instance is running normally. Specifically, **Executing task** means that the instance is performing operations, such as modifying its configurations.

If a monthly subscribed instance expires or a pay-as-you-go instance has overdue payment or is terminated by users, it is **Isolated** and moved into the recycle bin.

You can manually **Restore** an instance from the recycle bin. After that, it becomes **Running** again.

A pay-as-you-go instance will be retained in the recycle bin for up to three days. Three days later, it expires and will be eliminated automatically. Once eliminated, it is deleted completely, and cannot be restored or viewed in the console.

Setting Instance Maintenance Time

Last updated : 2024-01-24 11:16:51

Overview

Maintenance time is a very important concept for TencentDB for PostgreSQL. To ensure the stability of your TencentDB for PostgreSQL instance, the backend system performs maintenance operations on the instance during the maintenance time. We highly recommend you set an acceptable maintenance time for your business instance, usually during off-peak hours, so as to minimize the potential impact on your business.

In addition, we also recommend you perform operations involving data migration during the maintenance time, such as instance specification adjustment. Currently, the maintenance time is supported by primary and read-only instances. Take the database instance specification upgrade as an example. As this operation involves data migration, after the upgrade is completed, a momentary disconnection from the database may occur. When the upgrade is initiated, the **Switch Time** can be set to **During maintenance time**, so that the instance specification will be switched during the next **maintenance time** after the instance upgrade is completed. Note that when you select **During maintenance time** for **Switch Time**, the switch will not occur immediately after the database specification upgrade is completed; instead, the sync will continue till the instance goes into the next **maintenance time** when the switch will be performed. As a result, the overall time it takes to upgrade the instance may be extended.

Note:

Before maintenance is carried out for TencentDB for PostgreSQL, notifications will be sent to the contacts configured in your Tencent Cloud account through SMS and email.

Instance switch is accompanied by a disconnection from the database lasting for just seconds. Make sure that your business has a reconnection mechanism.

Directions

Setting maintenance time

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance details page.
2. In the **Maintenance Info** section on the instance details page, click **Modify**.

Maintenance Info Modify

Maintenance Window

Mon, Tue, Wed, Thu, Fri, Sat, Sun

Maintenance Time

04:00 - 06:00

3. In the pop-up window, select **Maintenance Window** and **Maintenance Time** as needed and click **OK**.

Modify Maintenance Window and Time ×

Maintenance Window

☒ Mon

☒ Tue

☒ Wed

☒ Thu

☒ Fri

☒ Sat

☒ Sun

Maintenance Time

Start Time

04:00

🕒

Duration

2

▼

hr

OK

Cancel

Switching now

If a task is configured to be switched during the maintenance time, but you need to switch it urgently under special circumstances, you can click **Switch Now** in the **Operation** column.

Note:

Switching now is applicable to operations involving data migration such as instance specification upgrade.

©2013-2022 Tencent Cloud. All rights reserved.

Page 8 of 378

Modifying Instance Configuration

Last updated : 2024-01-24 11:16:51

This document describes how to modify the computing specification and storage capacity of a TencentDB for PostgreSQL instance.

Overview

When the current performance or storage capacity of an instance cannot meet the needs of business changes, the instance configuration can be adjusted to sustain the business growth.

Note:

Currently, instance configuration can be upgraded but not downgraded.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#), select the target instance in the instance list, and click **Adjust Configurations** in the **Operation** column.
2. In the pop-up window, select the instance configuration and storage capacity you want to adjust to. The specific configuration adjustment price will be displayed below. After confirming the price, click **Submit**.

Note:

As the instance configuration adjustment involves instance data migration, the adjustment process will be completed with a primary/secondary switch, which will cause an instantaneous disconnection and slightly affect business access to the database instance. Therefore, we recommend you set the switch time within off-peak hours.

If you need to control the switch time, please select **Specify Time** in **Switch Time**. Then, you can select a time range. After the instance data is migrated, the system will automatically check whether it is within the switch time range, and if not, the instance will enter the **Waiting for Switch** status until the time falls in the time range and then complete the switch.

The time range is calculated at the granularity of one day. If the time window on the current day is missed, switch will be performed in the time window on the next day.

When the instance is in the **Waiting for Switch** status, you can click **Switch Now** in the instance list to switch immediately.

Instance ID

Instance Name

kaylal

Network

[Default-VPC - Default-Subnet](#)

Current Specs

4 core 8 GiB, 100 GB storage, PostgreSQL12.4

Specification

4 core 16 GiB ▾

Disk

500GB1000GB1500GB

Backup Space

50% of purchased instance capacity is provided for free [Details](#)

Upgrade Time

It may take 16 minutes

This duration is only for reference. When the instance load is high or a large amount of data is written, a longer upgrade duration will be required to ensure

Switch Time

Upon upgrade completion

[Specify Time](#)

02:00:00 ~ 04:00:00

☒ In the process of adjusting instance configuration, data migration may occur but instance access is not affected. After the migration is completed, there ensure that your business has a reconnection mechanism.

Fees

USD/hour (After 15 days of use, it will be reduced to USD/hour. ⓘ)

Submit

Cancel

Setting instance availability zone for disaster recovery or region disaster recovery

Last updated : 2024-07-23 11:35:54

TencentDB for PostgreSQL allows cross-AZ disaster recovery and cross-region disaster recovery. The following explains them separately.

Setting Cross-AZ Disaster Recovery

This document describes how to modify the primary and standby AZs of an instance in the TencentDB for PostgreSQL console. Modifying AZs has no impact on the instance's properties, configurations, or connection addresses. The amount of time required to modify AZs depends on the data volume of the instance.

Overview

Compared with a single-AZ deployment scheme, a multi-AZ one has better disaster recovery capabilities and can protect your databases from being affected by database instance failures, AZ outages, and even IDC-level failures. Multi-AZ deployment provides database instances with high availability and failover support. Multiple AZ deployment is to combine several signal AZs in the same region into a physical zone based on single-AZ deployment.

Notes

The region where the instance is located include at least two AZs.

The target AZ has sufficient computing resources.

Because a read-only instance has only one node, it cannot use the multi-AZ deployment scheme. When the AZs of its primary instance are changed, the region where the read-only instance resides remains unchanged.

Fees

No additional charge needs to be paid for the time being.

Directions

To create an instance, select an AZ on the Purchase page

1. Log in to the [TencentDB for PostgreSQL console](#), and click **Create**.
2. On the Purchase page, select the corresponding region, and under this region, set the **Primary AZ** and **Secondary AZ** fields.

Database

Relational Database

- TDSQL-C
- MySQL
- SQL Server
- PostgreSQL
- Instance List**
- Serverless
- Parameter Templates
- Recycle Bin
- Data Migration
- Database Backup
- Database Audit

MariaDB

Enterprise Distributed DBMS

- TDSQL

NoSQL Database

- Redis
- Tendis

Configuration Info

Database Engine	PostgreSQL
Architecture	Dual-Server High-Availability (one-primary-one-standby)
Database Version	PostgreSQL 16.2
Kernel Version	v16.2_r1.4 Upgrade Kernel Minor Version
Used/Total	90.00MB/10GB
Specification	1-core, 2 GiB MEM
Billing Mode	Pay as You Go
Creation Time	2024-07-12 10:32:47
Expiration Time	--

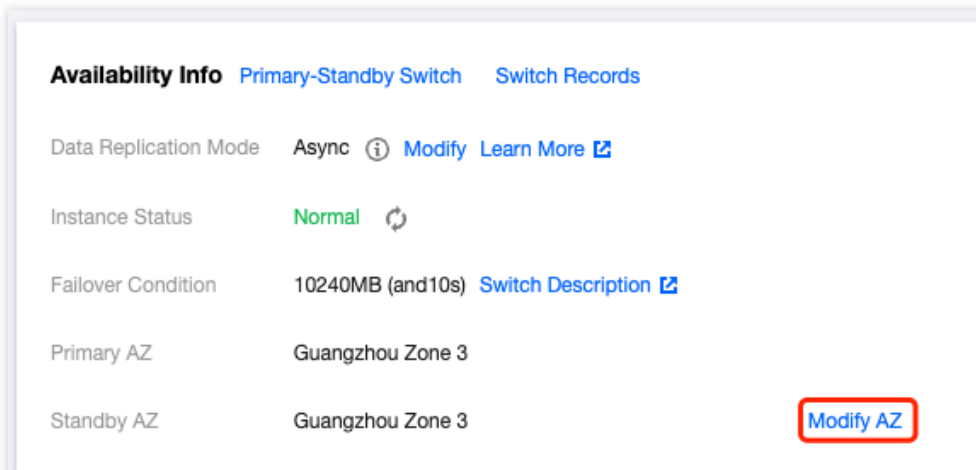
Availability Info [Primary-Standby Switch](#) [Switch Records](#)

Data Replication Mode	Async i Modify Learn More
Instance Status	Normal refresh
Failover Condition	10240MB (and 10s) Switch Description
Primary AZ	Guangzhou Zone 3
Standby AZ	Guangzhou Zone 3 Modify AZ

3. After the purchase, you can enter the **Instance Details** page to query the primary and secondary AZs in the **Availability Info** area.

Modifying an Availability Zone in the Console

1. Log in to the [TencentDB for PostgreSQL console](#), select the desired region, and click the name of the desired instance to enter the instance management page.
2. On the **Instance Details** page, click **Modify AZ** in the **Availability Info** area.



3. In the pop-up window for modifying deployment information, select the availability zone for either the primary or standby database.

Note:

Synchronous replication is used by default, which prioritizes data integrity. Therefore, the instance performance may be affected by the log transmission efficiency.

4. Select the switch time, and click **OK**.

Specify time: The switch will occur during the period of time you select.

Upon modification completion: The switch will occur right after the modification is completed.

Note:

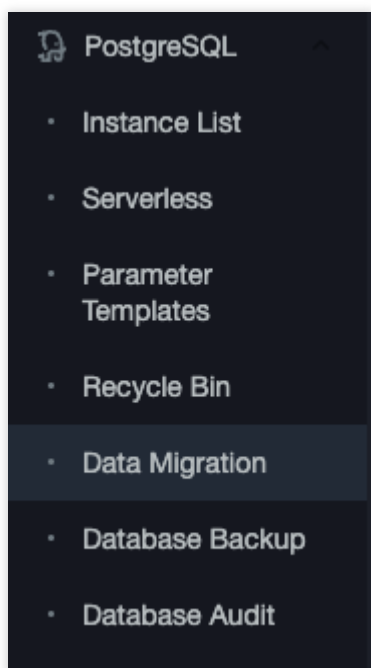
Modifying the primary AZ of an instance will trigger an instance switch, during which the database will be temporarily disconnected. Please be sure that your services can be reconnected. However, modifying the standby AZ has no impact on instance access.

5. Once the instance status changes from **Modifying AZ** to **Running**, the AZ switch is complete.

Setting Cross-Region Disaster Recovery

Directions

1. Log in to the [TencentDB for PostgreSQL console](#), and click **Create**. Purchase two TencentDB for PostgreSQL instances A and B.
2. In the [TencentDB for PostgreSQL console](#), click **Data Migration** to enter the data migration console.



3. Purchase a data migration task. Both source and target instances are of TencentDB for PostgreSQL type.

Service Type	Data Migration	Data Subscription	Data Sync		
Creation Mode	Create new task	Create similar task			
Billing Mode	Pay as you go				
Source Instance Type	MySQL	TDSQL-C MySQL	Redis	MongoDB	TDSQL MySQL
	PostgreSQL	Percona	SQL Server	Tendis	TDSQL-C PostgreSQL
Source Instance Region	<div>South China</div> <div>Guangzhou</div> <div>Shenzhen Finance</div> <div>Shenzhen</div>				
	<div>East China</div> <div>Shanghai</div> <div>Shanghai Finance</div> <div>Hangzhou</div> <div>Nanjing</div> <div>Hong Kong (China)</div>				
	<div>North China</div> <div>Beijing</div> <div>Tianjin</div> <div>Beijing Finance</div> <div>Singapore</div> <div>Bangkok</div>				
	<div>Southwest China</div> <div>Chengdu</div> <div>Chongqing</div> <div>Frankfurt</div> <div>Seoul</div> <div>Tokyo</div>				
	<div>South America</div> <div>Sao Paulo</div>				
Target Instance Type	PostgreSQL	TDSQL-C PostgreSQL			
Target Instance Region	<div>South China</div> <div>Guangzhou</div> <div>Shanghai</div> <div>Nanjing</div> <div>Hong Kong (China)</div> <div>Beijing</div>				

4. After the task is created, start configuring the parameters. Set **Access Type** to **Database**, select instance A as the source and instance B as the destination, and enter the accounts and passwords for connectivity tests.

1

Set source and target databases

>

2

Set migration options and select migration objects

>

3

Verify task

Task Configuration

Task Name *

dts-nispomg7

Running Mode *

Immediate execution

Scheduled execution

Source Database Settings

Source Database Type *

PostgreSQL

Region

South China(Guangzhou)

Access Type ⓘ *

Public Network

Self-Build on CVM

Direct Connect

VPN Access

Database

CCN

Access

Please add the DTS IP addresses to the security group allowlist in advance so that the connectivity test can be quickly passed. For

Cross-/Intra-Account *

Intra-account

Cross-account

Database Instance *

postgres-xxxxx (source)

Private IP:

Account *

postgres

Password *

✕

Test Connectivity

✔ Test passed

Target Database Settings

Target Database Type *

PostgreSQL

Region

North China (Beijing)

Access Type ⓘ *

Database


Database Instance *

postgres-xxxxx (dest)

Private IP:

xxxxxx2

5. Set **Migration Type** to **Full + incremental migration**, and **Migration Object** to **Entire instance**.

 **Configure a migration task**

✓ **Set source and target databases**

>

2 **Set migration options and select migration objects**

>

3 **Verify task**

Migration Method ⓘ *

Physical migration

Logical migration

Migration Type ⓘ *

Structural migration

Full migration

Full + incremental migration

Migration Object ⓘ *

Entire instance

Specify object

ⓘ For migration notes, see [Migration FAQs](#) ↗

Previous

Save

Next

The above steps can realize data synchronization from instance A to instance B in a different region, ultimately implementing cross-region disaster recovery for instance A.

Modifying AZs

Last updated : 2024-01-24 11:16:51

This document describes how to modify the primary and standby AZs of an instance in the TencentDB for PostgreSQL console. Modifying AZs has no impact on the instance's properties, configurations, or connection addresses. The amount of time required to modify AZs depends on the data volume of the instance.

Background

Compared with a single-AZ deployment scheme, a multi-AZ one has better disaster recovery capabilities and can protect your database from being affected by database instance failures, AZ outages, and even IDC-level failures. The multi-AZ deployment scheme guarantees the high availability and failover capability of database instances by combining multiple AZs in the same region into a single "multi-AZ".

Notes

The region where your instance resides should have at least two AZs.

The target AZ has sufficient computing resources.

Because a read-only instance has only one node, it cannot use the multi-AZ deployment scheme. The region where the read-only instance resides will not change if the AZs of its primary instance change.

Costs

There is no additional charge for the time being.

Directions

Selecting AZs on the instance purchase page

1. Log in to the [TencentDB for PostgreSQL console](#) and click **Create**.
2. On the displayed purchase page, select a region, **Primary AZ**, and **Standby AZ**.

Billing Mode **Pay as You Go**

Region **Shanghai** Guangzhou Nanjing Beijing Chengdu Hong Kong (China) Singapore Bangkok Seoul
Tokyo Silicon Valley Virginia Frankfurt Moscow

The classic network and VPC cannot communicate with each other. As the network type cannot be changed once purchased, please be cautious with your selection.

Primary AZ Shanghai Zone 1 **OUT** Shanghai Zone 2 **OUT** Shanghai Zone 3 **OUT** **Shanghai Zone 4** Shanghai Zone 5 Shanghai Zone 6
Shanghai Zone 7

Standby AZ **Shanghai Zone 4** Shanghai Zone 5

Network Default-VPC Default-Subnet 4093 subnet IP in total, with 4088 available

If the existing networks do not meet your requirements, go to [Create VPCs](#) or [Create Subnets](#).
In the current network environment, only devices in "Default-VPC" can access this database instance.

Fees Configuration Fees Traffic Fees
USD/hour (After 15 days of use, it will be reduced to JSD/hour) (Billing Details) **0.00**

Buy Now

3. After the instance is purchased, you can view its primary and standby AZs in the **Availability Info** block on the **Instance Details** tab.

Modifying AZs in the console

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, select a region and click an instance ID to access the instance management page.
2. Click **Modify AZ** in the **Availability Info** block on the **Instance Details** tab.

Availability Info

Data Replication Mode Sync

Primary AZ Shanghai Zone 4

Standby AZ Shanghai Zone 5 **Modify AZ**

3. In the pop-up **Modify Deployment Info** window, select AZs for the primary node and the standby node, respectively.

Note:

The default data replication mode is synchronous replication, which prioritizes data integrity. The instance performance is affected by the log transmission efficiency.

4. Select the switch time and click **OK**.

Specify time: The switch will occur during the period of time you select.

Upon modification completion: The switch will occur right after the modification is completed.

Note:

Modifying the primary AZ of an instance will trigger an instance switch, during which the database will be temporarily disconnected. Make sure that your business has a reconnection mechanism. However, modifying the standby AZ has no impact on instance access.

5. After the instance status changes from **Modifying AZ** to **Running**, the AZ modification is completed.

Terminating Instances

Last updated : 2024-01-24 11:16:51

This document describes how to terminate a TencentDB for PostgreSQL instance in the console.

Overview

You can manually terminate instances at any time as needed. Terminated instances will be moved to the recycle bin.

Notes

If a pay-as-you-go instance is terminated, it will be moved to the recycle bin and retained there for up to three days.

The instance in the recycle bin is in the "Isolated" status and cannot be accessed.

To use the instance again, you can restore it from the recycle bin.

If the instance in the recycle bin is no longer needed, you can eliminate it.

After the instance is eliminated, its data and backup files will also be deleted and cannot be restored in the cloud.

Please store your backup files safely elsewhere in advance.

If a primary instance has one or more read-only instances, terminating the primary instance won't affect the read-only instances, but eliminating the primary instance will eliminate the read-only instances at the same time. To prevent instances from being eliminated due to overdue payment, please pay attention to the instance expiration information.


If a pay-as-you-go instance is terminated, its billing stops.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#), locate the instance to be terminated in the instance list, and click **More > Terminate Instance** in the **Operation** column.
2. In the pop-up dialog box, indicate your consent and click **Terminate Now**.

Terminate Instance ✕

You've selected 1 instance [Show less](#) ▲

Instance ID / Name	Instance Type	Associate Instance
	Read-only Instance	--

After the instance is completely terminated, **the data will not be recovered**. Please back up the instance data in advance.

After the instance is completely terminated, the IP resources are reclaimed at the same time. If the instance has associated read-only instances:

The read-only instance will also be terminated.

We recommend that you terminate read-only instances before terminating the primary instance

☒ Confirm to terminate, instance data is not required or backup is done

Terminate Now Cancel

Restoring Instances

Last updated : 2024-01-24 11:16:51

This document describes how to restore an isolated TencentDB for PostgreSQL instance in the console.

Overview

If an instance is terminated by mistake, due to overdue payment, or when it expires, you can go to the recycle bin to restore it before it is eliminated.

Note:

After an instance is restored, it uses the same configurations as before.

An instance cannot be terminated, restored and terminated again in a short time.


Directions

1. Log in to the [TencentDB for PostgreSQL console](#), locate the instance to be restored in the recycle bin list, and click **Restore** in the **Operation** column.
2. In the pop-up dialog box, confirm the billing information, and click **Confirm**.

Restore Instance

You've selected 1 instance [Show less](#) ▲

NO.	Instance ID	Instance Name
1	pgro-g2xthbld	Unnamed

Total Fees  USD/hour

ConfirmCancel

3. After the restoration is completed, you can see the instance in the instance list.

Eliminating Instances

Last updated : 2024-01-24 11:16:51

This document describes how to eliminate an isolated TencentDB for PostgreSQL instance in the console.

Overview

You can manually eliminate an instance when you confirm that the instance is no longer needed.

Note:

After the instance is eliminated, its data and backup files will also be deleted and cannot be restored in the cloud.

Please store your backup files safely elsewhere in advance.


If a primary instance has one or more read-only instances which are running normally, the read-only instances will be eliminated along with the primary instance. To prevent instances from being eliminated due to overdue payment, please pay attention to the instance expiration information.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#), locate the desired instance in the recycle bin list, and click **Eliminate** in the **Operation** column.
2. In the pop-up dialog box, confirm that everything is correct and click **Confirm**.

Eliminate Instance ✕

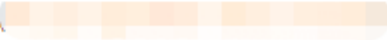
You've selected 1 instance [Show less](#) ▲

Instance ID / Name	Instance Type	Associate Instance
	Read-only Replica	--

When an instance is eliminated:

All its read-only replicas (if any) will be eliminated at the same time.

All data in the instance cannot be restored. Please back up instance data before elimination.

Are you sure you want to eliminate the instance ()?

Confirm Cancel

Restarting Instances

Last updated : 2024-01-24 11:16:51

Restart is indispensable to the maintenance of databases. Restarting a PostgreSQL instance is equivalent to restarting a database (service and process) on a local server.

Notes

Please exercise great caution when restarting a database, which plays a vital role in the business. Before the restart, it is recommended to disconnect the database from server and stop writing data.

Restarting an instance does not change its physical attributes, so the public IP, private IP, and any data stored on the instance will remain unchanged.

After the restart, reconnection to the database is needed. Please make sure your business has a reconnection mechanism.

Be sure to restart the instance during off-hours so as to ensure success and reduce impact on your business.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#), locate the desired instance in the instance list, and click **More > Restart** in the **Operation** column.

Note:

Generally, it takes a few seconds to minutes to restart an instance, during which the instance cannot be accessed and existing connections to it will be closed.

Restart will fail if there are a large number of writes and dirty pages during the restart. In this case, the instance will roll back to the status before the restart and can still be accessed.

There is a chance of failure in restarting a database. If it takes more than 10 minutes to restart, you can [submit a ticket](#) for help.

2. In the pop-up dialog box, indicate your consent and click **Confirm**.

Restart Instance ✕

You've selected 1 instance [Show less](#) ▲

NO.	Instance ID	Instance Name
1	postgre[REDACTED]	[REDACTED]

Restarting an instance will cause the database to be inaccessible for a period of time and bring about unpredictable risks. It takes 5 seconds to 5 minutes to complete the restart.

☒ I have read and agreed to [Database Instance Restart Instructions](#) [🔗](#)

Confirm Cancel

Modifying Data Replication Mode

Last updated : 2024-03-20 16:17:52

Supported Methods to Replicate Data

Database instance replication refers to the synchronization of data by configuring one or more backup databases for the server, distributing PostgreSQL data across multiple systems. TencentDB for PostgreSQL supports the following two data replication modes:

Asynchronous Replication

Asynchronous replication for TencentDB for PostgreSQL adopts the one-master-one-standby architecture.

After receiving a data update (including insert, update and delete operations) request from an application, the master performs the update operation. When the update is completed, the master immediately responds to the application and replicates the data to the slave.

In the process of data update, the master does not need to wait for the response of the slave. Therefore, the database instances of asynchronous replication usually have higher performance (for specific performance, see Test Results), and the unavailability of the slave does not affect the service provided by the master. However, because the data is not synchronized to the slave in real time, if a fault occurs on the master while the slave is delayed and a switch occurs, there is a slight chance of causing data inconsistency.

Note:

By default, TencentDB for PostgreSQL adopts asynchronous replication for data replication.

Semi-synchronous Replication

Semi-synchronous replication of TencentDB for PostgreSQL adopts a one-master-one-standby architecture.

An application initiates a data update request (including insert, update, delete operations). After the master completes the update operation, it immediately replicates the data to the slave. Only after the slave **receives the data and writes it to the WAL (without executing)** it returns a success message to the master. The master must only return a response to the application program after receiving the success message from the slave.

Only in the case of a data replication exception (the slave node is unavailable or there is a network issue impacting data replication), the master suspends (for approximately 10 seconds by default in PostgreSQL) its response to the application and downgrades the replication method to asynchronous replication. When data replication is restored to normal, semi-synchronous replication will be restored.

Degradation Description

Fault Degradation

If the current data replication mode of PostgreSQL is semi-synchronous, when data replication encounters an exception (either the slave node is unavailable or an exception occurs in the network used for data replication), the master suspends responding to the application (about 10 seconds for TencentDB for PostgreSQL by default), and downgrades the master-slave replication mode to asynchronous to ensure system availability. Once the high-availability system detects that data replication returns to normal, the master-slave replication mode will be restored to semi-synchronous replication.

Note:

Fault degradation in TencentDB for PostgreSQL's high-availability system is the default behavior. To ensure high availability for the system, currently, the setting cannot be changed.

Latency Degradation

If you have specific needs, you can enable latency degradation under semi-synchronous replication. Once delayed degradation is enabled, TencentDB for PostgreSQL's high-availability system will determine the master-slave replication latency based on the conditions you have set. Exceeding the latency will trigger semi-synchronous degradation to asynchronous. It is recommended that only this feature be enabled for highly latency-sensitive businesses.

The conditions for latency degradation are the size or time of master-slave synchronization. You can refer to the metrics of log write delay (bytes) and log write time delay (seconds) on the standby database. For more details, see [Master/Slave Latency Monitoring Metrics](#).

Failover Description

When the instance is in **asynchronous replication** mode or when **semi-synchronous is downgraded to asynchronous replication**, a master/slave switch is triggered when the master fails and cannot recover. As the data is not synchronized to the slave in real-time, there is a small probability that it may lead to data inconsistency. In TencentDB for PostgreSQL failover conditions can be flexibly configured. By default, the system allows switching when both conditions of **master/slave synchronous delay of 10240 MB** and **master/slave delay of 10 seconds** are met. It is recommended that changes be made only if you have special business requirements.

Modifying Data Replication Mode

1. Log in to the [PostgreSQL Console](#). In the instance list, click the Instance ID or **Operation** in the **Manage** column to open the instance details page.
2. In the **availability information** section of the instance details, detailed availability information of the instance is displayed.

2.1 When the data replication mode is asynchronous, the specific display information is as follows:

Information	Description
Data Replication Mode	For the data synchronization mode between the master and the slave, the current two-machine high availability (one master and one slave) architecture supports asynchronous replication and semi-synchronous replication .
Instance Availability Status	It displays the current accessibility status of the instance. When the status is normal, user requests are normally received. If the status is abnormal, the instance currently cannot accept application requests.
Failover Condition	When the master node fails and cannot recover, an automatic failover is required, with the slave providing the service. At this time, the system defines the failover conditions, which are the master-slave latency size and the master-slave latency time. The system's default conditions are 10240 MB and 10 seconds. For specific failover conditions, see Failover Description .
Master Availability Zone	It refers to the availability zone of the master node.
Slave Availability Zone	It refers to the availability zone of the slave node.

2.2 The specific display information is as follows when the data replication mode is semi-synchronous:

Information	Description
Data Replication Mode	For the data synchronization mode between the master and the slave, the current two-machine high availability (one master and one slave) architecture supports asynchronous replication and semi-synchronous replication .
Instance Availability Status	It displays the current accessibility status of the instance. When the status is normal, user requests are normally received. If the status is abnormal, the instance currently cannot accept application requests.
Fault Degradation Conditions	When the data replication mode of the instance is semi-synchronous replication, the system will automatically degrade the master and slave replication method to asynchronous to ensure system availability outside the range of user-set conditions. This degradation condition is the master and standby latency size or latency time. Among them, PostgreSQL instances with a large version number of 9 only support the condition of master and standby latency size. For details, see the regression explanation .
Failover Condition	When the master node fails and cannot be recovered, an automatic failover is required, to switch to the slave to provide service. At this time, the system has defined the failover conditions, which are the size or time of master-slave latency. Applications can modify the switch conditions based on special needs. For details, see Failover Description .

Master Availability Zone	It refers to the availability zone of the master node.
Slave Availability Zone	It refers to the availability zone of the slave node.


3. Click **Modify** to change the current instance's data replication mode.

Note:


Changes to the data replication mode are effective immediately, and modifying the data method may cause a master/slave switch. There will be a momentary disconnection during the master/slave switch, please ensure the application is reconnected.

Availability Info [Primary-Standby Switch](#) [Switch Records](#)


Data Replication Mode

Async ⓘ **Modify** [Learn More](#) 

Instance Status

Normal 

Failover Condition

10240MB (and10s) [Switch Description](#) 

Primary AZ

Guangzhou Zone 3

Standby AZ

Guangzhou Zone 3 [Modify AZ](#)

Modify Replication Mode and Configuration



⚠ Data replication mode takes effect upon modification, which may cause automatic primary-standby switch. Please proceed with caution.

Instance ID/Name postgres-b1xt1y1b / amy

Data Replication Mode

Async

Semi-sync

[Learn More](#)

Semi-sync mode is used for data replication, which supports the switch to the new primary instance when the original primary instance fails.

Delay Downgrade Condition



When delay size > 0 MB or delay time > 0 sec,
the downgrade will be performed.

You can set the sync delay threshold or delay time for the primary-standby node when the replication mode is switched from semi-sync to async.

Failover Condition

When delay size ≤ 10240 MB and delay time ≤ 10 sec,
the switch will be performed. [Switch Description](#)

To ensure the availability, after switching from semi-sync to async mode, you can set the delay threshold or delay time of the primary-standby sync for the system to automatically perform the primary-standby switch.

OK

Cancel

Switch Source-Replica Instance

Last updated : 2024-03-20 13:19:41

Reason for Master/Slave Switch

Interchange of master and slave node roles within an instance is called a master/slave switch. After the switch, the instance address remains the same, and the application automatically connects to the new master node, thereby ensuring high availability of the instance. The main reasons for the master/slave switch are:

Failover

The system automatically initiates a master/slave switch when it detects that the instance is abnormal and cannot be used normally. For specific switch conditions, see [Failover Description](#).

Manual Switch

It refers to the master/slave switch initiated manually by application Ops personnel or authorized Tencent Cloud technical experts. Manual switch includes the switches with normal master-slave latency and forced switches with latency exceeding the master-slave latency.

Forced Switch

When the instance's master-slave replication mode is asynchronous replication or when semi-synchronous replication is degraded to asynchronous replication, a master/slave switch will be triggered if the master fails and cannot recover. As the data is not synchronized to the slave in real time, there is a small chance that data inconsistency might occur. Conditions for allowing switches are configured in the system by default, but you can also set specific conditions based on the needs of your business. Therefore, a switch is only allowed when the switch conditions are met. To facilitate switches in emergencies, the system provides forced switch capability.

Note:

To prevent changes in switch conditions over time, when a force switch is performed, the switch must be performed immediately.

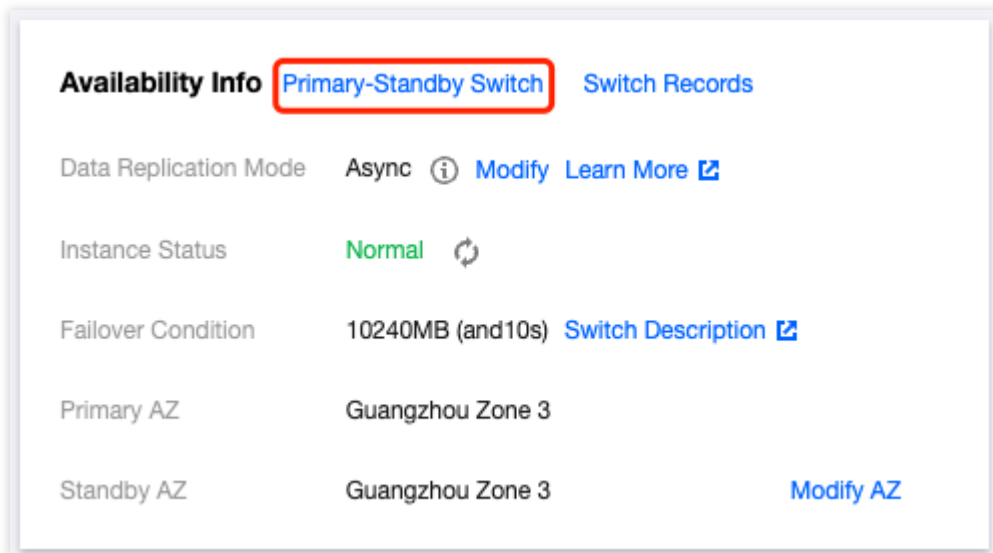
Impact of Master/Slave Switch

There will be a momentary disconnection during the master/slave switch process. Please ensure that your application has a reconnection mechanism.


If there are read-only instances mounted on the master instance, there will be a minute-level delay after the master/slave switch.

Manual Switch of Instances



1. Log in to the [PostgreSQL Console](#). In the instance list, click the **Instance ID** or **Operation** in the **Management** column to enter the instance detail page.
2. In the **Availability Info** section of the instance details, click on **Primary-Standby Switch**.



Primary-Standby Switch



Data replication mode takes effect upon modification, which may cause automatic primary-standby switch. Please proceed with caution.

Instance ID/Name	postgres-b1xt1y1b / amy
Data Replication Mode	Async 
Instance Status	Normal 
Automatic Async Switch Condition	10240MB (and10s)
Failover Condition	10240MB (and10s) Switch Description
Current Primary-Standby Sync Delay	0MB (0s)
Switch Time	<div><div>Switch Now</div><div>Specify time</div></div> <div><div>During maintenance time</div></div> <div>Switch Time Description</div>
Forced Switch	<div><input checked="" type="checkbox"/> Learn More</div> <div><input type="checkbox"/> A momentary disconnection will occur during the primary-standby switch. Please make sure that your business has a reconnection mechanism.</div>

OK

Cancel

View Change Record

1. Log in to the [PostgreSQL Console](#). In the instance list, click the **Instance ID** or **Operation** in the **Management** column to enter the instance detail page.
2. In the **Availability Info** section of the instance details, click **Switch Records**. The system will retain switch records for one year.

Availability Info
[Primary-Standby Switch](#)
[Switch Records](#)

Data Replication Mode Async ⓘ [Modify](#) [Learn More](#)

Instance Status Normal ↻

Failover Condition 10240MB (and10s) [Switch Description](#)

Primary AZ Guangzhou Zone 3

Standby AZ Guangzhou Zone 3 [Modify AZ](#)

Master/Slave Latency Monitoring Metrics

TencentDB for PostgreSQL provides detailed monitoring information to help you observe the synchronization latency between the master and slave nodes. The specific monitoring metrics are as follows:

Metric Name	Metric Description
Slave Log Write-to-Disk Latency (Bytes)	It refers to the difference in size between the slave log write-to-disk LSN and the current LSN of the master instance. For the master instance, this metric reflects the data loss size in the event of a failover.
Slave Log Write-to-Disk Time Latency (Seconds)	It refers to the time difference between the master instance sending the log to the slave instance and the slave instance receiving the log and writing it to disk. For the master instance, this metric shows the data loss size in the event of a failover. This metric is only available for instances in version 10.x and above.
Master-Slave Data Synchronization Latency (Bytes)	It refers to the difference in size between the slave replay LSN and the current LSN of the master instance. For the master instance, this metric reflects the RTO in the event of a failover. For read-only instances, this metric reflects the data latency size.
Master-Slave Data Synchronization Latency Time (Seconds)	It refers to the time difference between the master instance sending the log to the slave instance and the slave instance receiving and replaying the log. This metric is only available for instances in version 10.x and above.
Slave Log Send and Replay Position Difference (Bytes)	It refers to the size difference between the master instance sending the log to the slave instance and the completion of slave log replication. It reflects the speed of slave log application. You can mainly check the performance of the slave instance and the speed of network

transmission through this metric. This metric is not available for read-only instances.

Upgrading Instance Kernel Minor Version Upgrade

Last updated : 2024-01-24 11:16:51

TencentDB for PostgreSQL allows you to upgrade kernel minor version, so you can use new features, improve performance, or fix issues.

For details on the TencentDB for PostgreSQL kernel minor version, see [Kernel Version Release Notes](#).

Overview

You can manually upgrade kernel minor version in the console.

Upgrade Rules

If an instance to be upgraded is associated with other instances (e.g., read-only instances), these instances will be upgraded together to ensure data consistency.

TencentDB for PostgreSQL upgrade may involve data migration. The time it takes to migrate an instance depends on the data size of the instance. Your business will not be affected during the upgrade and can be accessed as per usual.

Note

Instance switch will be required after kernel minor version upgrade is completed, that is, the database may be disconnected for seconds. We recommend that you use applications configured with automatic reconnection feature and conduct the switch during the instance maintenance window.

The kernel minor version cannot be downgraded once upgraded.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance details page.

2. In the **Configuration Info** section, click **Upgrade Kernel Minor Version**.
3. In the pop-up window, set the target version to upgrade and click **OK**.

Upgrade Kernel Minor Version

Current Version v14.2_r1.5

Target Version

v14.2_r1.5

For the differences among the kernel minor versions, [see here](#).

Switch Time

During maintenance timeSpecify time

Upon upgrade completion

Switch Time Description

☐ During upgrade, data migration may occur but instance access is not affected. After the migration is completed, there will be a momentary disconnection due to primary-standby switch. Please ensure that your business has a reconnection mechanism. To ensure database replication consistency, the kernel minor versions of all associated read-only instances will also be upgraded.

OKCancel

Note:

As database kernel minor version upgrade may involve data migration, a momentary disconnection from the database may occur after the upgrade is completed. We recommend that you set the switch time to **During maintenance time**, so that the switch will be initiated within the next maintenance time after the upgrade is completed.

Upgrade the Major Version of the Database

Last updated : 2024-03-20 14:24:52

Overview and Advantages

There are new PostgreSQL versions continuously provided to customers. In new versions, more features, better performance, and higher stability and security are introduced. Therefore, it is recommended that you plan properly based on your business needs and perform upgrades to the major version of the database as soon as possible.

TencentDB for PostgreSQL supports major version upgrades, and this tool has the following advantages:

Supports cross-version upgrades among PostgreSQL 9 to 15. Supports upgrade rehearsals.

The upgrade is performed on a read-only instance pulled from the original instance, with no impact to the original instance.

The upgrade process has only a momentary disconnect and a short read-only impact on the business.

The read time depends on the number of original instance objects, regardless of data scale.

After the original instance upgrade is completed, connection addresses, tags, monitoring, backup sets, and other information are fully retained.

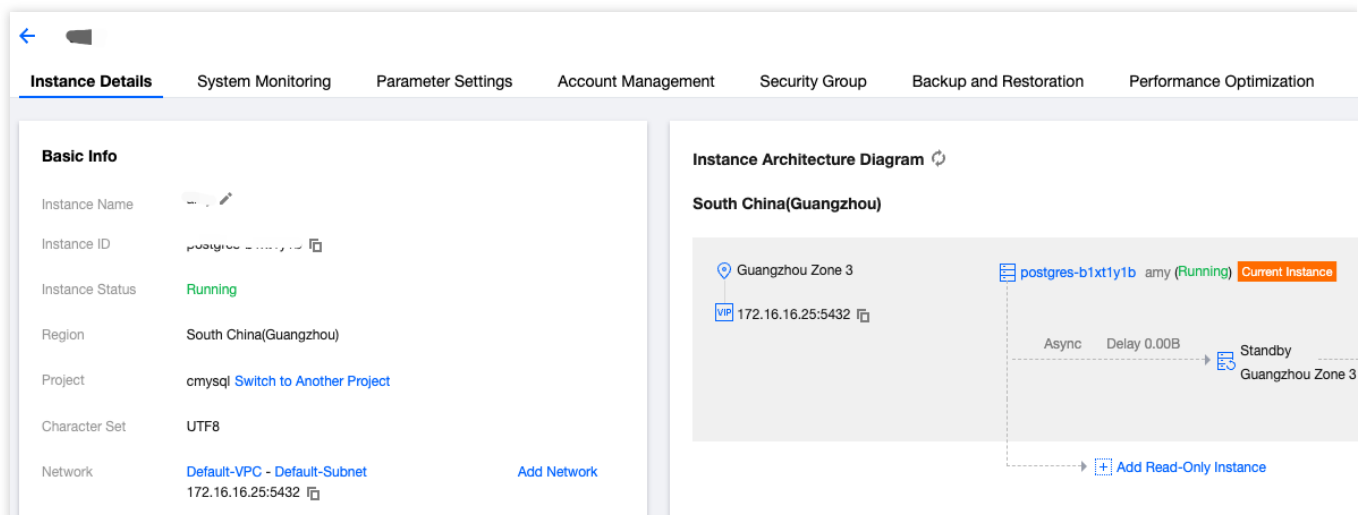
Note:

Currently, major version upgrades are not supported for instances with transparent data encryption enabled.

Directions

Formal Major Version Upgrade

1. Log in to the [PostgreSQL Console](#). In the instance list, click the **Instance ID** or **Operation** in the **Management** column to enter the instance detail page.
2. Click **Version Upgrade** to open the version upgrade operation page.



3. Click **Upgrade Major Version** to open the version upgrade page.

3.1 Target version

The current system will return all upgradable versions for you to choose, and the latest version number will be returned for each major version.

3.2 Upgrade time

There will be momentary disconnections and temporary read-only status during the upgrade process, therefore businesses need to evaluate the operational time window. You can choose to execute immediately, at a specific time, or within the maintenance time. For maintenance time settings, see [Setting Instance Maintenance Time](#).

When the upgrade is completed but the specified or maintenance time window is not reached, the instance status is **Waiting for Switch**. You can navigate to **Instance List > Operation > Switch Now** on the console to complete the upgrade process.

3.3 Collecting statistics

Accurate statistics can ensure that PostgreSQL query planner processes queries in the most optimal way. Lack of statistics may lead to incorrect query planning, which could hamper performance and consume too much memory. This operation mainly involves running ANALYZE on the master instance and updating system statistical information after the upgrade. The duration of generating statistics depends on the size of the instance data.

3.4 Plugin upgrade setting

This operation is to execute ALTER EXTENSION UPDATE in the database where extensions were created after the upgrade. There are three options:

Upgrade extensions prior to upgrade completion: As it requires checking and upgrading the extension list of all databases to the corresponding version, the execution time is proportional to the number of databases and extensions. The overall execution time of the major version upgrade will be extended. Therefore, please perform evaluation before choosing this option.

Upgrade extensions after upgrade completion: Read/write is immediately resumed after upgrade. You need to analyze the effect on your business before the extension upgrade is completed.

Do not upgrade extension version: Read/write is immediately resumed after upgrade. You need to perform the extension upgrade by yourself, and analyze the impact on business before the extension upgrade is completed.

3.5 Whether to perform backup before the upgrade starts

During a major version upgrade, to ensure the recoverability of data, the system by default conducts two backups. The type of backup is **Upgrade Backup**:

A full backup will be performed before the upgrade starts, which will be done immediately before the upgrade. You can restore the database instance to the state of its previous version using this backup.

A full backup will be performed after the upgrade is completed. This backup is created immediately after new writes are allowed on the upgraded database instance.

Backup options before the upgrade:

No: Choose this option when there are existing backups that can restore the instance to its pre-upgrade state.

Otherwise, this option is not recommended.

Yes: It is the default option. Charges may apply after backup commercialization. For details, see [Backup Space Billing](#). You can choose to delete it after the upgrade verification is done. The backup file deletion policy is also managed by User Setting's backup retention rules. If you want to extend the backup retention period, see the following figure:

File Name	Instance ID/Name	Task Start Time	Task End Time	Backup Expiration Time	Backup Size	Type	Backup Mode	Region
automat...09.tar.zst	pg-123456789	2024-03-19 03:31:09	2024-03-19 03:31:36	2024-03-26 03:31:36	2.89 MB	Automatic	Physical Cold Backup	Guangzhou
automat...13.tar.zst	pg-123456789	2024-03-18 03:30:13	2024-03-18 03:30:39	2024-03-25 03:30:39	2.89 MB	Automatic	Physical Cold Backup	Guangzhou
automat...12.tar.zst	pg-123456789	2024-03-17 03:31:12	2024-03-17 03:31:37	2024-03-24 03:31:37	2.89 MB	Automatic	Physical Cold Backup	Guangzhou
automat...36.tar.zst	pg-123456789	2024-03-16 03:30:36	2024-03-16 03:31:02	2024-03-23 03:31:02	2.89 MB	Automatic	Physical Cold Backup	Guangzhou
automat...09.tar.zst	pg-123456789	2024-03-15 03:31:09	2024-03-15 03:31:25	2024-03-22 03:31:25	2.89 MB	Automatic	Physical Cold Backup	Guangzhou

3.6 Task launch settings

Only perform check, and do not initiate task: A pre-upgrade check will be performed, including checking instance's running state, parameter setting validity, database connection, and so on. However, no task will be initiated. Users can use this operation to pre-check upgrade feasibility.


Perform check and initiate task: A pre-upgrade check will be performed, including instance's running state, parameter setting validity, database connection, and so on. If the check is passed, the task will be directly initiated.

Upgrade Major Version

Database version upgrade instructions:

- The upgrade process has no effect on existing business.
- There is a momentary disconnection during the upgrade process. To minimize the impact on your business, we recommend that you upgrade during maintenance time or off-peak hours.
- The upgrade time is subject to the number of database objects and the amount of data.
- The primary instance and its associated read-only instances cannot be upgraded at the same time. After the primary instance is upgraded, you can add read-only instances again.

Upgrade Instance Major Version

Instance ID     

Instance Name Unnamed

Instance Status Running

Instance Type Primary Instance

Current Version v11.12_r1.16

Target Version v16.0_r1.2

Upgrade Time During maintenance time Specify time Immediate execution [Upgrade Time Description](#)

Statistics Collection No collection Before upgrade After upgrade

The time of statistics collection depends on the data volume. You can collect statistics again to get a more accurate execution plan of the database

Extension Upgrade Settings No extension version upgrade Before upgrade After upgrade

Perform Backup Before Upgrade Yes No

Backing up the original instance before upgrade will increase the probability of data recoverability after the upgrade. The backup space beyond the

Upgrade Start Settings Check but not start Check and start

After the check task is passed, the upgrade task will be started.

[Submit](#)

4. Clicking **Commit** will create an upgrade task. The instance status will be set to **Kernel Version Upgrading**. You can check the task status and system logs in the task list.

实例详情 系统监控 参数设置 账号管理 安全组 备份恢复 性能优化 只读实例 数据加密 版本升级					
升级小版本 升级大版本 大版本升级演练					
升级任务列表					
任务 ID	创建时间	类型	状态	升级前备份	升级后备份
fd8a1550-b942-5f3c-a5f4-059f08b33b12	2023-08-03 11:01:41	大版本升级	成功	--	0a6ff00e-c517-54b7-...
2d3f8d40-ae2b-5823-9320-edf01c849644	2023-08-03 10:16:16	大版本升级检查	成功	--	--
a3137461-f35e-55cb-a60b-b66e55930c52	2023-08-03 09:34:43	大版本升级	成功	--	1f0469a9-6e84-5ba0...
a7037a53-9135-54ad-bbe3-25a74c7b25eb	2023-08-02 22:21:26	大版本升级	成功	4d0c1572-6cca-546c-8e77-3e253b2a68e3	2e5ddb66-344f-5a27...
ed845b7d-a4f7-51a4-82c4-d25326d11013	2023-08-02 20:29:08	大版本升级检查	成功	--	--
497adb99-e380-59f2-8de2-7c40b6c9c7b3	2023-08-02 18:55:43	大版本升级	成功	4eeb42b-2216-5106-a0a1-48ed58932675	9592312a-ba4f-5533...
32ce51a2-63d1-54e9-933f-567990789038	2023-08-02 18:50:36	大版本升级检查	成功	--	--
共 12 项					

日志详情

文件名 pg_upgrade_internal.log

内容

pg_upgrade run on Thu Aug 3 11:01:48 2023

Performing Consistency Checks on Old Live Server

```
Checking cluster versions ok
Checking database user is the install user ok
Checking database connection settings ok
Checking for prepared transactions ok
Checking for system-defined composite types in user tables
ok
Checking for reg* data types in user tables ok
Checking for contrib/isn with bigint-passing mismatch ok
Checking for user-defined encoding conversions ok
Checking for user-defined postfix operators ok
Checking for presence of required libraries ok
Checking database user is the install user ok
Checking for prepared transactions ok
Checking for new cluster tablespace directories ok
```

关闭

Drill Upgrade

To ensure the success rate of the upgrade, a drilling upgrade can be performed first for the businesses. In a drilling upgrade, a new instance is cloned first based on the current instance's backup set, the upgrade is performed on the new instance, and then a new instance is created after the upgrade is complete. You can choose to use the instance

directly or delete the instance. Instances generated through cloning will be billed normally in the pay-as-you-go billing mode. For specific prices, see [Pricing](#). The specific operations are as follows:

1. Log in to the [PostgreSQL Console](#). In the instance list, click the **Instance ID** or **Operation** in the **Management** column to enter the instance detail page.
2. Click **Version Upgrade** to open the version upgrade operation page.
3. Click **Major Version Upgrade Drill** to enter the rehearsal operation interface.

3.1 Cloned instance settings

For specific cloned instance settings, see [Cloning Instance](#).

1 Clone Instance > 2 Upgrade Configuration

Primary Instance Basic Info



Instance ID	[REDACTED]	Instance Name	Unnamed
Network	Default-VPC - Default-Subnet	Project	cmysql
Region	South China(Guangzhou)	AZ	Guangzhou Zone 3
Architecture	Dual-Server High-Availability (one-primary-one-standby)	Instance Specification	1 -core, 2 GiB MEM
Database Engine	PostgreSQL	Database Kernel Version	v11.12_r1.16

Clone Instance Configuration Info

Restoration Mode

[By time point](#)[By backup set](#)

Restoration Time Point

2024-03-19 19:17:16  

Database Kernel Version **v11.12_r1.16**

Billing Mode

[Pay as You Go](#)

Region

South China(Guangzhou)


Primary AZ


[Guangzhou Zone 3](#)[Guangzhou Zone 4](#)[Guangzhou Zone 6](#)[Guangzhou Zone 7](#)

Standby AZ

[Guangzhou Zone 3](#)[Guangzhou Zone 4](#)[Guangzhou Zone 6](#)[Guangzhou Zone 7](#)


Network

Default-VPC 


Default-Subnet 

4093 subnet IPs in total, with 4055 available
If the existing networks do not meet your requirements, go to [create a VPC](#) or [create a subnet](#).
In the current network environment, only devices in "Default-VPC" can access this database instance.

Instance Specification

[1 -core, 2 GiB MEM](#) 

Disk



10 GB500 GB1000 GB1500 GB

Local SSD, featuring powerful performance.

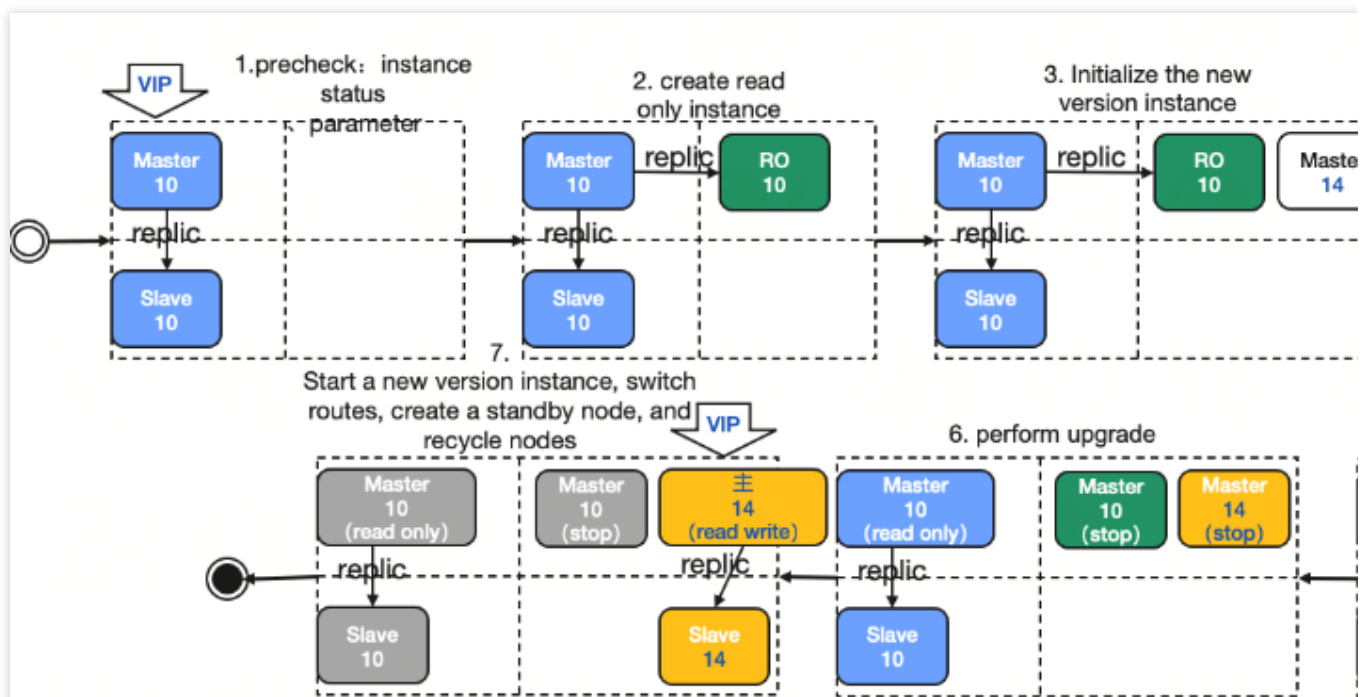
3.2 Upgrade settings

For upgrade settings, see [Formal Major Version Upgrade](#).

Click **Commit** to initiate a drill task. If the original instance is postgres-3e3ug2nj, the system will produce a new instance named postgres-3e3ug2nj-pre-upgrade-20230809154431, and complete the subsequent upgrade.

How It Works

To help you understand the system, and manage the major version upgrade time, this article shows the operations performed by the system during major version upgrade, as shown in the figure below:



The above figure uses the upgrade from PostgreSQL 10 to PostgreSQL 14 as an example, and detailed description of the backend system operating steps are as follows:

1. Precheck

Check if the instance state is running.

Check the validity of instance parameters.

Check if the target upgrade version is valid.

2. Create a read-only instance

Create a read-only instance of the source instance, to minimize the impact of upgrades on the original instance. Subsequent upgrade operations will be performed on this read-only instance.

3. Initialize the new version database

Create a blank directory, and initialize a new instance with the target version. This new instance only has a single node.

4. Stop the high version node, and set source instance master node to read-only.

Stop the new version instance, and set the source instance to read-only.

5. Promote the read-only instance to the master instance, and stop the process.

Promote the read-only instance to the master instance, and stop this instance.

6. Perform the upgrade

Perform the upgrade, export and import the metadata, and process data.

7. Launch the new version process, switch routes, build new standby machines, and reclaim the old nodes.

Launch the upgraded new version instance, switch the routing information of the original instance, and set up a secondary node for the new instance. In the end, clean up the environment to finish the task.

Note:

After the system upgrade is completed, carry out relevant verification to ensure the smooth operation of the business as follows:

Please run a business load test on the database after the upgrade.

Verify extension compatibility.

Verify parameter compatibility.

System Limits

If the instance has associated read-only instances, you need to delete the read-only instances before initiating the upgrade.

If the instance is overused in disk space, a major version upgrade cannot be initiated.

FAQs

Is the Major Version Downgrade supported?

The major version downgrade is not supported. The current TencentDB for PostgreSQL is forward compatible with major versions, a higher version is recommended.

Read-Only Instance

Overview

Last updated : 2024-01-24 11:16:51

TencentDB for PostgreSQL allows you to create one or more read-only instances. They are suitable for read/write separation and one-primary-multiple-standby application scenarios and capable of greatly enhancing the read load capacity of your database.

Unified read/write separation addresses (i.e., read and write requests are separated automatically) are not supported currently. You can access read-only instances with separate IPs and ports, or add them to a read-only instance group (RO group) to balance their loads.

Note:

For read-only instance pricing, see [Pricing](#).

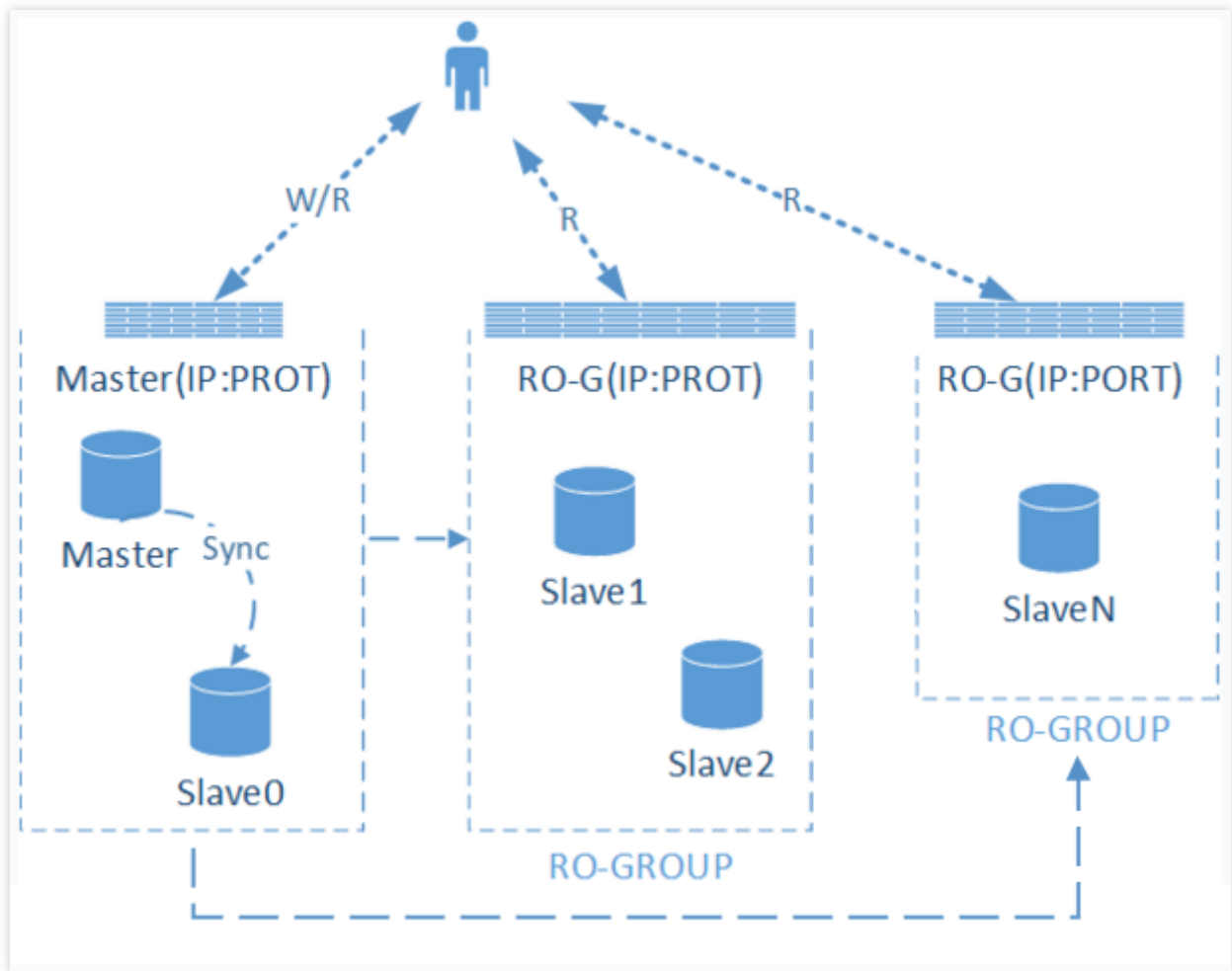
Concepts

RO group: It consists of one or more load balancing-enabled read-only instances. If there are multiple read-only instances in one RO group, read requests can be evenly distributed among the instances. RO groups provide IPs and ports for access to databases.

Read-only instance: It is a single-node (with no standby) instance that supports read requests. It cannot exist independently; instead, it must belong to a primary instance.

Architecture

Changes in the primary instance (source database) are synced to all read-only instances through PostgreSQL's streaming replication mechanism. Given the single-node architecture (with no standby) of read-only instances, repeated attempts to restore a failed read-only instance will be made. Therefore, we recommend you choose an RO group rather than a read-only instance for higher availability.



Feature Limits

The minimum disk capacity of a read-only instance must be greater than or equal to the storage capacity used by the primary instance.

Up to six read-only instances can be created for a primary instance.

Backup and rollback features are not supported.

Data cannot be migrated to read-only instances.

Databases cannot be created in or deleted from read-only instances.

Operations including account creation/deletion/authorization and account name/password modification are not supported.

Notes

There is no need to maintain accounts or databases for read-only instances, which are synced with those of the primary instance.

Data inconsistency between multiple read-only instances may occur due to the delay in data sync between the read-only instances and the primary instance. You can check the delay in the console and configure CM alarms.

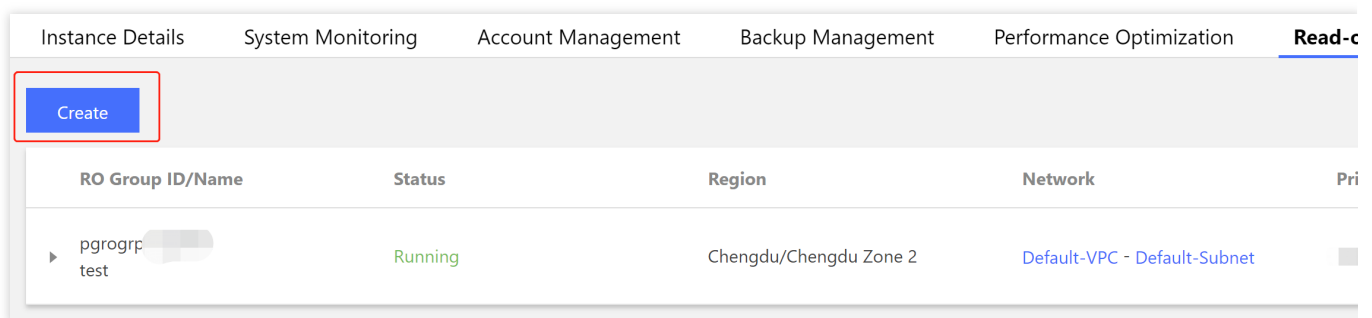
The specification of a read-only instance can be different from that of the primary instance, which makes it easier for you to upgrade the read-only instance based on your business load. We recommend you keep the same specifications of read-only instances in one RO group.

If the primary instance is written so frequently that the automatic log cleanup threshold is exceeded, logs will be automatically deleted. If the standby instance hasn't obtained the deleted logs yet, the primary-standby replication will be disconnected, and the read-only instance will be automatically rebuilt and become inaccessible.

Read-only instances don't have high availability. We recommend you use an RO group and configure at least two read-only instances to avoid business access failures caused by single points of failures.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. Click **Add Read-Only Instance** in the **Instance Architecture Diagram** section on the **Instance Details** tab or click **Create** on the **Read-Only Instance** tab.



3. On the purchase page, select the desired read-only instance configuration, confirm that everything is correct, and click **Buy Now**.

Specify RO Group: Select **Do not specify for now**, **Create RO group**, or **Existing RO group**.

Create RO group: If multiple read-only instances are purchased at a time, all of them will be assigned to the newly created RO group. The RO group automatically allocates read weights to each read-only instance and automatically distributes read requests among them based on their read weights. For more information, see [Managing RO Groups](#).

Existing RO group: Specify an existing RO group. If multiple read-only instances are purchased at a time, all of them will be assigned to the RO group.

Remove Delayed RO Instances: Remove a read-only instance from the RO group if the data sync log size difference between the primary instance and the read-only instance is greater than the specified threshold (in MB).

AZ: You can select any AZ in the same region as the primary instance.

4. After the purchase is completed, you will be redirected to the instance list. After the status of the instance changes to **Running**, the instance can be used normally.

Managing RO Groups

Last updated : 2024-01-24 11:16:51

TencentDB for PostgreSQL allows you to create one or more read-only instances to form an RO group, which is suitable for read/write separation and one-primary-multiple-standby application scenarios. This greatly improves the read load capacity of your database.

Prerequisites

You have created a primary instance. For more information, see [Purchase Methods](#).

You have created a read-only instance. For more information, see [Overview](#).

Creating RO Group

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click **More** > **Create Read-Only Instance** in the **Operation** column of the target instance to enter the instance purchase page.

PostgreSQL - Instance List

Guangzhou 2 Other regions 3

Create More Recycle Bin

<input type="checkbox"/> Instance ID/Type/Name	Monitoring/Status	AZ	Configuration	Database Version	Billing Mode
<input type="checkbox"/> <div></div>	<div><div></div><div>Running</div></div>	Guangzhou Zone 6	Dual-Server High-Availability Edition 1 core 2 GiB, 10 GB disk Network: <div></div>	PostgreSQL 13.3	Pay as You Go
<input type="checkbox"/> <div></div>	<div><div></div><div>Running</div></div>	Guangzhou Zone 6	Dual-Server High-Availability Edition 1 core 2 GiB, 10 GB disk Network: <div></div>	PostgreSQL 13.3	Pay as You Go

Note:

You can also click an instance ID or **Manage** in the **Operation** column to enter the instance management page. Click **Add Read-Only Instance** in the **Instance Architecture Diagram** section on the **Instance Details** tab to enter the read-only instance purchase page. Or, click **Create** on the **Read-Only Instance** tab to enter the read-only instance purchase page.

The screenshot displays the Tencent Cloud console interface for a PostgreSQL instance named 'Unnamed'. The top navigation bar includes tabs for 'Instance Details', 'System Monitoring', 'Account Management', 'Backup Management', 'Performance Optimization', and 'Read-only Instance'. The 'Instance Details' tab is active, showing a 'Basic Info' section with the following details:

- Instance Name: Unnamed
- Instance ID: postgres-XXXXXX
- Instance Status: Running
- Region: Southwest China (Chengdu)
- Availability Zone: Chengdu Zone 2
- Network: Default-VPC - Default-Subnet
- Project: Default Project
- Character Set: UTF8
- Private IPv4 Address: XXXXX
- Public IPv4 Address: Enable
- Tag: Modify

To the right, the 'Instance Architecture Diagram' shows the instance configuration for 'Southwest China (Chengdu)'. It illustrates a primary instance 'postgres-XXXXXX' in 'Chengdu Zone 2' connected to a 'Standby Instance' in the same zone via a 'Sync' connection with a 'Delay 0.00B'. Below this, an 'RO Group' is shown, containing a primary instance 'pgrogrp-XXXXXX' and a read-only instance 'pgro-XXXXXX' in 'Chengdu Zone 2', connected via an 'Async' connection with a 'Delay 0.00B'. A button 'Add Read-Only Instance' is visible at the bottom right of the diagram.

2. On the purchase page, select the desired read-only instance configuration, confirm that everything is correct, and click **Buy Now**.

Specify RO Group: Select **Create RO Group**. If multiple read-only instances are purchased at a time, all of them will be assigned to the newly created RO group. The RO group automatically allocates read weights to each read-only instance and automatically distributes read requests among them based on their read weights.

RO Group Name: The RO group name doesn't need to be unique and can contain up to 60 letters, digits, hyphens, and underscores.

Remove Delayed RO Instances: This option indicates whether to enable the removal policy. A read-only instance will be removed from the RO group when its delay exceeds the threshold, and will rejoin the RO group when its delay drops below the threshold. A removed read-only instance will become inactive, its weight will be set to 0 automatically, and warning notifications will be sent. For more information on how to configure read-only instance removal alarms and recipients, see [Alarming Feature](#).

No matter whether delayed read-only instance removal is enabled, a read-only instance that is removed due to instance failure will rejoin the RO group when it is repaired.

Delay: This sets the delay time for a read-only instance. When the threshold is exceeded, the instance will be removed from the RO group.

Delay Threshold: This sets the delay threshold for a read-only instance. When the data sync log size difference between the primary instance and the read-only instance is above the threshold, the read-only instance will be removed from the RO group.

Least RO Instances: This is the minimum number of instances that should be retained in the RO group. When there are fewer instances in the RO group, even if an instance exceeds the delay threshold, it will not be removed.

Specify RO Group

Create RO Group ▼

[Learn about RO Group](#)

Set RO Group Name

Up to 60 chars, supporting letters, digits, underscores, and dashes.

Eliminate Instances with Out-of-Limit Delay

☒ [What is elimination of instances with out-of-limit delay](#)

Whether or not enabled, read-only instances will be eliminated and recovered upon failure.

Delay

—

100

+

sec

Enter an integer greater than or equal to 5

Delayed Data

—

512

+

MB

Minimum RO Instances

—

1

+

3. Return to the instance list. The status of the created instance is **Delivering**. After the status changes to **Running**, the read-only instance has been successfully created.

Configuring RO Group

On the RO group configuration page, you can configure the basic information of the group such as name, removal policy, delay threshold, and least read-only instances.

Note:

Read-only instances in an RO group can have different specifications, and read weights are automatically assigned by the system according to instance specifications.

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click a primary instance ID to enter the instance management page.
2. On the instance management page, click the **Read-Only Instance** tab and click **Configure** in the **RO Group** column to enter the RO group configuration page.

Instance Details
System Monitoring
Account Management
Backup Management
Performance Optimization
Read-Only

Create

RO Group ID/Name	Status	Region	Network	Pr
test	Running	Chengdu/Chengdu Zone 2	Default-VPC - Default-Subnet	

Instance ID / Name	Status	Delay	Configuration	Billing Mode
pgre Unnamed	Running	Delay 0.00B	10 GB/2 GB	Pay as you go

3. On the RO group configuration page, configure the RO group and click **Submit**.

Assign Read Weight: The traffic of each read-only instance in an RO group will be automatically distributed according to its read weight, which can implement load balancing and reduce the difficulty in managing multiple read-only instance IP addresses. An RO group automatically assigns read weights to each read-only instance. The following table lists the read weights of read-only instances of different specifications:

Specification	Weight
2 GB memory	1
4 GB memory	2
8 GB memory	2
12 GB memory	4
16 GB memory	4
24 GB memory	8
32 GB memory	8
48 GB memory	10
64 GB memory	12
96 GB memory	14
128 GB memory	16
240 GB memory	26
480 GB memory	50

Rebalance Load:

If rebalancing is disabled, modifying weight will only affect new loads. The operation has no impact on read-only instances accessed by existing persistent connections and does not cause momentary database disconnections.

If rebalancing is enabled, when the configurations of read-only instances in the RO group are modified, all connections to the RO group will be disconnected, and new connections will be rebalanced according to instance weights.

RO Group ID

pgrogrp

RO Group Name

test

Up to 60 chars, supporting letters, digits, underscores, and dashes.

Eliminate Instances with Out-of-Limit Delay

☒ What is elimination of instances with out-of-limit delay [?](#)

Whether or not enabled, read-only instances will be eliminated and recovered upon failure.

Delay

—

100

+

sec

Enter an integer greater than or equal to 5

Delayed Data

—

512

+

MB

Minimum RO Instances

—

1

+

Assign Read Weight

Assigned by system [Read Weight Description](#) [?](#)

Rebalance Load

☒

If load rebalancing is disabled, read weight only takes effect for new loads after instance upgrade and will not affect t connection and not cause flash disconnection of database.

Submit

Cancel

Deleting RO Group

An RO group can be deleted if it has no read-only instances.

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click a primary instance ID to enter the instance management page.
2. On the instance management page, select the **Read-Only Instance** tab to view all RO groups. You can delete an RO group after confirming that there are no read-only instances in it.

Instance Details

System Monitoring

Account Management

Backup Management

Performance Optimization

Read-only

Create

RO Group ID/Name	Status	Region	Network	Priority
<div>▼ pgrogrp-<div>test</div></div>	Running	Chengdu/Chengdu Zone 2	Default-VPC - Default-Subnet	

Instance ID / Name	Status	Delay	Configuration	Billing Mode
<div>pgro-<div>Unnamed</div></div>	Running	Delay 0.00B	10 GB/2 GB	Pay as you go

Removal Policy and Load Balancing

Last updated : 2024-01-24 11:16:51

TencentDB for PostgreSQL allows you to create one or more read-only replicas to form a read-only replica group (RO group), which is suitable for read/write separation and one-primary-multiple-secondary application scenarios and capable of greatly enhancing the read load capacity of your databases. This document describes how to manage RO groups.

Rebalancing Traffic

If load rebalancing is disabled, modifying weight will take effect only for new loads but not affect the read-only replicas accessed by existing persistent connections or cause short disconnection from the database.

If load rebalancing is enabled, the read weights of upgraded read-only replicas will change, which causes all connections to the RO group to be disconnected after the upgrade is completed, and all new connections will be rebalanced according to the new weights.

If you are not satisfied with the connection distribution of each read-only replica in the RO group, you can also manually rebalance it: log in to the [console](#), click the primary instance ID to access the instance management page, select the **Read-only Replica** tab, locate the desired read-only replica in the RO group, and click **Rebalance** in the **Operation** column.

Note:

Make sure your business has an automatic reconnection mechanism. Neither enable automatic rebalancing nor manually rebalance if there is no automatic reconnection mechanism.

Removing Failed Read-only Replicas

When a read-only replica in a RO group becomes inaccessible due to an unexpected error, the RO group automatically removes the read-only replica. This rule is a default rule.

Removing Delayed Read-only Replicas

If this feature is enabled, a read-only replica will be removed from the RO group if the data sync log size difference between the primary instance and the read-only replica is greater than the specified threshold (MB).

Allocating Read Weights

The traffic of each read-only replica in an RO group will be automatically distributed according to its read weight, which can realize load balancing and reduce the difficulty of managing multiple read-only replica IP addresses. An RO group automatically allocates read weights to each read-only replica. The following table lists the read weights of read-only replicas of different specifications:

Specification	Weight
2 GB memory	1
4 GB memory	2
8 GB memory	2
12 GB memory	4
16 GB memory	4
24 GB memory	8
32 GB memory	8
48 GB memory	10
64 GB memory	12
96 GB memory	14
128 GB memory	16
240 GB memory	26
480 GB memory	50

References

For more information on how to create one or more read-only replicas, please see [Creating Read-only Replicas](#).
For more information on how to create one or more read-only replicas and add them to an RO group, please see [Managing RO Groups](#).

Account Management

Database Privilege Overview

Last updated : 2024-01-24 11:16:51

Account Privilege System

PostgreSQL adopts a role-based access control (RBAC) model to manage users, roles, and permissions.

In PostgreSQL, the concepts of users and roles are almost the same. The only difference is that a user has the login privilege, while a role has the nologin privilege.

PostgreSQL supports system permissions and database object permissions, and manages them using the concept of roles. Both categories of the permissions can be granted to a role, and this role can grant its own permissions to other roles or users.

You can grant system or object permissions to roles/users to manage databases.

System Permissions

System permissions are used to perform database operations. PostgreSQL manages system permissions using role attributes and default roles.

Role attributes

You can specify attributes when creating a role with `CREATE ROLE`, or modify them after creation with `ALTER ROLE`. Role attributes are stored in the `pg_authid` system table.

`CREATE ROLE` syntax:



```
CREATE ROLE name [ [ WITH ] option [ ... ] ]  
where option can be:  
    SUPERUSER | NOSUPERUSER  
    | CREATEDB | NOCREATEDB  
    | CREATEROLE | NOCREATEROLE  
    | INHERIT | NOINHERIT  
    | LOGIN | NOLOGIN  
    | REPLICATION | NOREPLICATION  
    | BYPASSRLS | NOBYPASSRLS  
    | CONNECTION LIMIT connlimit  
    | [ ENCRYPTED ] PASSWORD 'password' | PASSWORD NULL
```

```

| VALID UNTIL 'timestamp'
| IN ROLE role_name [, ...]
| IN GROUP role_name [, ...]
| ROLE role_name [, ...]
| ADMIN role_name [, ...]
| USER role_name [, ...]
| SYSID uid

```

A role with the superuser attribute can bypass all privilege checks and perform all database operations, because a superuser has the highest privilege in the database, which is similar to the root privilege in Linux.

Note:

TencentDB for PostgreSQL has disabled the superuser privilege due to security requirements. However, some operations must be performed by a superuser, so TencentDB for PostgreSQL provides the `tencentdb_superuser` role. For details, see [Roles and Permissions](#).

Default roles

PostgreSQL provides a set of default roles which provide access to certain, commonly needed, privileged capabilities and information. Administrators can grant these roles to users and/or other roles in their environment, providing those users with access to the specified capabilities and information. The following table lists the [default roles](#) supported in PostgreSQL 11.

Role	Allowed Access
<code>pg_execute_server_program</code>	Allow executing programs on the database server as the user the database runs as with COPY and other functions which allow executing a server-side program.
<code>pg_monitor</code>	Read/Execute various monitoring views and functions. This role is a member of <code>pg_read_all_settings</code> , <code>pg_read_all_stats</code> , and <code>pg_stat_scan_tables</code> .
<code>pg_read_all_settings</code>	Read all configuration variables, even those normally visible only to superusers.
<code>pg_read_all_stats</code>	Read all <code>pg_stat_*</code> views and use various statistics related extensions, even those normally visible only to superusers.
<code>pg_read_server_files</code>	Allow reading files from any location the database can access on the server with COPY and other file-access functions.
<code>pg_signal_backend</code>	Signal another backend to cancel a query or terminate its session.
<code>pg_stat_scan_tables</code>	Execute monitoring functions that may take ACCESS SHARE locks on tables, potentially for a long time.
<code>pg_write_server_files</code>	Allow writing to files in any location the database can access on the server with COPY and other file-access functions.

public

An implicitly defined group that always includes all roles. Any particular role will have the sum of permissions granted directly to public. PostgreSQL grants default permissions on some types of objects to public.

Database Object Permissions

PostgreSQL uses an access control list (ACL) to manage database object permissions. The following table lists all database object permissions in PostgreSQL and their abbreviations.

Permissions	Abbreviation	Supported Object
SELECT	r ("read")	LARGE OBJECT, SEQUENCE, TABLE (and table-like objects), table column
INSERT	a ("append")	TABLE, table column
UPDATE	w ("write")	LARGE OBJECT, SEQUENCE, TABLE, table column
DELETE	d	TABLE
TRUNCATE	D	TABLE
REFERENCES	x	TABLE, table column
TRIGGER	t	TABLE
CREATE	C	DATABASE, SCHEMA, TABLESPACE
CONNECT	c	DATABASE
TEMPORARY	T	DATABASE
EXECUTE	X	FUNCTION, PROCEDURE
USAGE	U	DOMAIN, FOREIGN DATA WRAPPER, FOREIGN SERVER, LANGUAGE, SCHEMA, SEQUENCE, TYPE

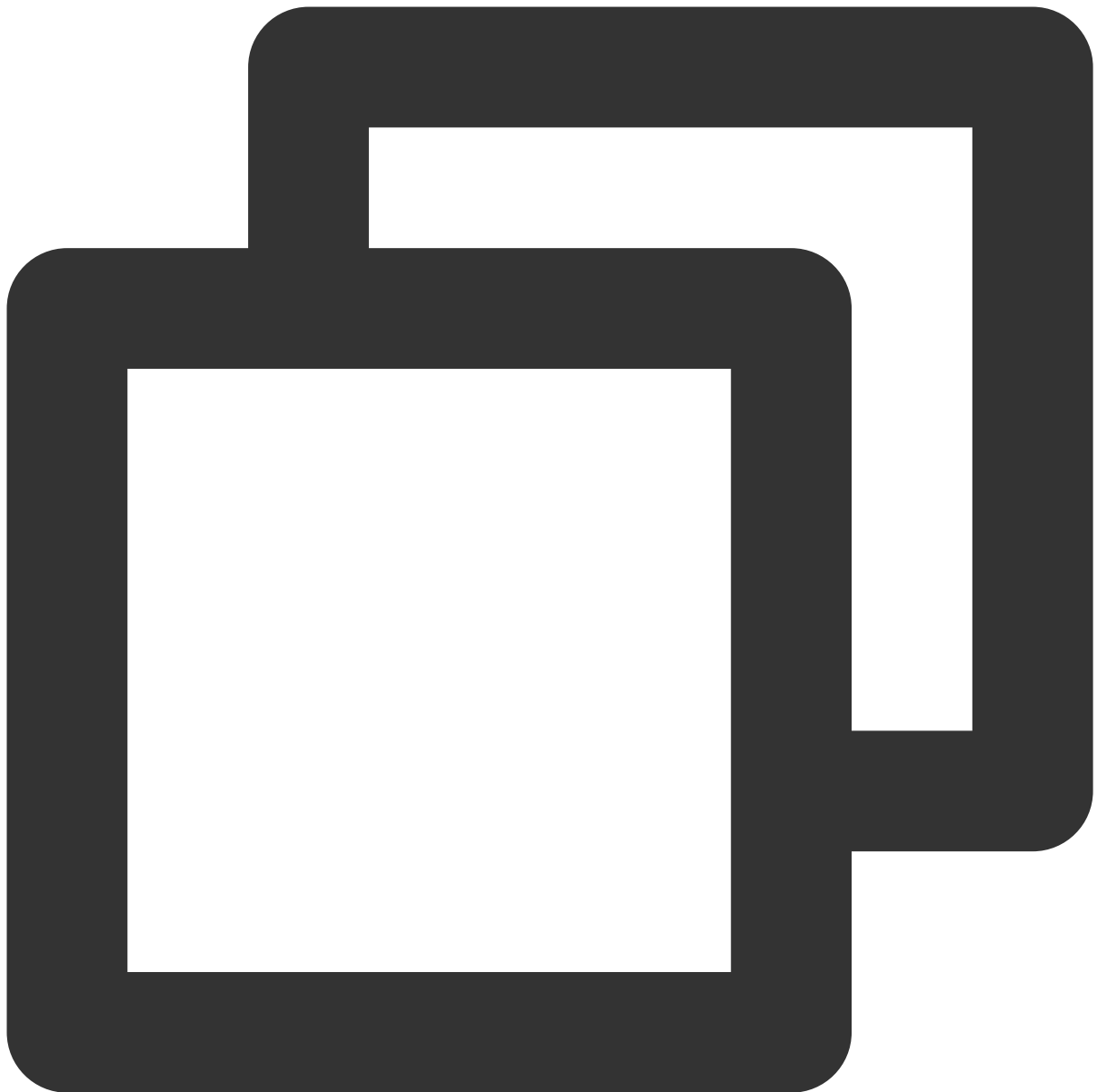
The following table lists the permissions owned by a type of objects and the psql command to query the permissions:

Object Type	Permissions	Permissions of Default Role (public)	psql Command to Query Permissions
DATABASE	CTc	Tc	\l
DOMAIN	U	U	\dD+

FUNCTION or PROCEDURE	X	X	\\df+
FOREIGN DATA WRAPPER	U	none	\\dew+
FOREIGN SERVER	U	none	\\des+
LANGUAGE	U	U	\\dL+
LARGE OBJECT	rw	none	-
SCHEMA	UC	none	\\dn+
SEQUENCE	rwU	none	\\dp
TABLE (and table-like objects)	arwdDxt	none	\\dp
Table column	arwx	none	\\dp
TABLESPACE	C	none	\\db+
TYPE	U	U	\\dT+

In PostgreSQL, the permissions granted for a particular object are displayed as a list of aclitem entries. The aclitem list of database and schema permissions is stored in `pg_database.dataacl` and `pg_namespace.nspacl`, that of permissions for tables, views, and other objects stored in `pg_class.relacl`, and that of column permissions stored in `pg_attribute.attacl`.

For example, "normal_user=a*r/test1" specifies that the user `normal_user` has the privilege `INSERT` with grant option (which gives the user the right to grant the privilege to others) and the privilege `SELECT`, both granted by `test1`.



```
postgres=# \dp
              Access privileges
 Schema | Name | Type | Access privileges | Column privileges | Policies
-----+-----+-----+-----+-----+-----
 public | t1   | table | test1=arwdDxt/test1 |                   |
(1 rows)
postgres=# grant select on t1 to normal_user;
GRANT
postgres=# grant insert on t1 to normal_user with grant option;
GRANT
postgres=# grant update on t1 to public;
```

```
GRANT
```

```
postgres=# grant select (a) on t1 to test2;
```

```
GRANT
```

```
postgres=# \dp
```

Access privileges

Schema	Name	Type	Access privileges	Column privileges	Policies
public	t1	table	test1=arwdDxt/test1 + a:		
			normal_user=a*r/test1+ test2=r/test1		
			=w/test1		

```
(1 rows)
```

```
-- Where, "=w/test1" specifies that test1 grants public the UPDATE privilege.
```

Users and Permission Operations

Last updated : 2024-01-24 11:16:51

TencentDB Default Role

TencentDB for PostgreSQL does not open the `superuser` role attribute and the `pg_execute_server_program`, `pg_read_server_files`, and `pg_write_server_files` roles for you to use. However, as some operations require the `superuser` role, TencentDB for PostgreSQL provides the `pg_tencentdb_superuser` to replace `superuser`.

pg_tencentdb_superuser Role

This role supports system permissions and database object permissions, as listed in the following tables.

System permissions

Permission	Description
CREATEDB	Create a database.
BYPASSRLS	Bypass all row-level security policy checks.
REPLICATION	Have the REPLICATION permission by default, and allow granting the REPLICATION permission to other users.
CREATEROLE	Have the same CREATEROLE permission as the community edition, except that the role cannot create the <code>pg_read_server_files</code> , <code>pg_write_server_files</code> , and <code>pg_execute_server_program</code> roles.

Object permissions

Object	Description
database	By default, have the permissions of all databases not owned by a superuser.
schema	By default, have the permissions of all schemas not owned by a superuser.
table/sequence	By default, have the permissions of all tables/sequences not owned by a superuser.
function	By default, have the permissions of all functions not owned by a superuser.

language	No permissions.
tablespace	No permissions.
FDW/foreign server	By default, have the permissions of all FDWs/foreign servers not owned by a superuser.
TYPE	By default, have the permissions of all TYPEs not owned by a superuser.

Other operations

Pub/Sub: the `tencentdb_superuser` role can implement the pub/sub messaging paradigm, create a publication for all tables, and create slots.

Extensions: the `tencentdb_superuser` role can create all supported extensions. When creating an extension, the `pg_tencentdb_superuser` is temporarily escalated to superuser and passes all permission checks.

The `load_file` permission only allows loading supported extension libraries.

The `tencentdb_superuser` role can use the `pgstat_get_backend_current_activity` function to view deadlock details, so that users can easily troubleshoot deadlocks themselves.

The use of the `pg_signal_backend` function is restricted, and processes of the `pg_tencentdb_superuser` role can only be killed by itself.

Permission Operations

For more information, see the official documents in the PostgreSQL community:

[Create a user:](#)



```
CREATE USER name [ [ WITH ] option [ ... ] ]
```

where option can be:

```
SUPERUSER | NOSUPERUSER  
| CREATEDB | NOCREATEDB  
| CREATEROLE | NOCREATEROLE  
| INHERIT | NOINHERIT  
| LOGIN | NOLOGIN  
| REPLICATION | NOREPLICATION  
| BYPASSRLS | NOBYPASSRLS
```

```
| CONNECTION LIMIT connlimit  
| [ ENCRYPTED ] PASSWORD 'password' | PASSWORD NULL  
| VALID UNTIL 'timestamp'  
| IN ROLE role_name [, ...]  
| IN GROUP role_name [, ...]  
| ROLE role_name [, ...]  
| ADMIN role_name [, ...]  
| USER role_name [, ...]  
| SYSID uid
```

[Create a role:](#)



```
CREATE ROLE name [ [ WITH ] option [ ... ] ]
```

where option can be:

```
    SUPERUSER | NOSUPERUSER  
| CREATEDB | NOCREATEDB  
| CREATEROLE | NOCREATEROLE  
| INHERIT | NOINHERIT  
| LOGIN | NOLOGIN  
| REPLICATION | NOREPLICATION  
| BYPASSRLS | NOBYPASSRLS  
| CONNECTION LIMIT connlimit  
| [ ENCRYPTED ] PASSWORD 'password' | PASSWORD NULL  
| VALID UNTIL 'timestamp'  
| IN ROLE role_name [, ...]  
| IN GROUP role_name [, ...]  
| ROLE role_name [, ...]  
| ADMIN role_name [, ...]  
| USER role_name [, ...]  
| SYSID uid
```

[Modify a role attribute:](#)



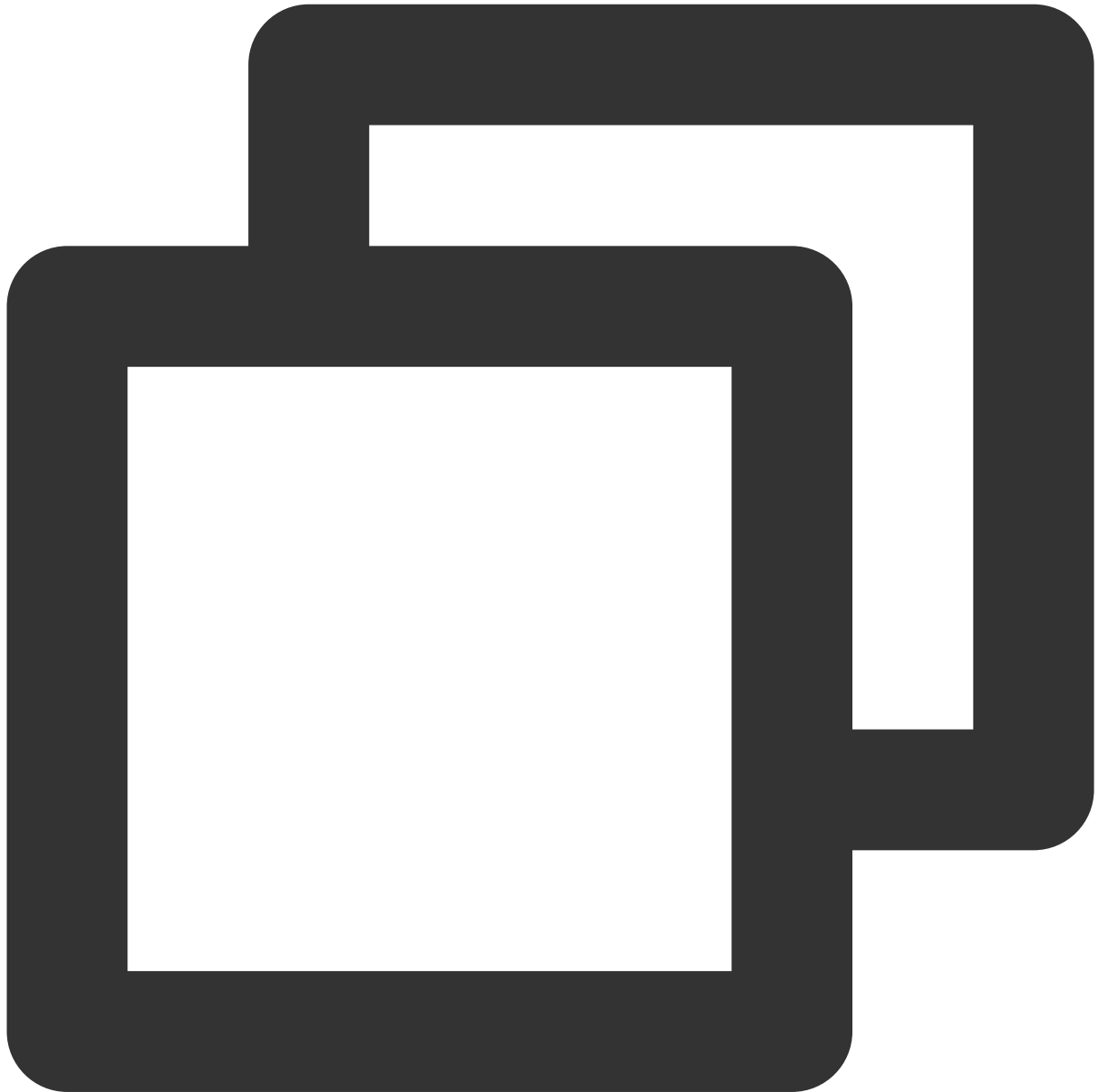
```
ALTER ROLE role_specification [ WITH ] option [ ... ]
```

where option can be:

```
SUPERUSER | NOSUPERUSER  
| CREATEDB | NOCREATEDB  
| CREATEROLE | NOCREATEROLE  
| INHERIT | NOINHERIT  
| LOGIN | NOLOGIN  
| REPLICATION | NOREPLICATION  
| BYPASSRLS | NOBYPASSRLS
```

```
| CONNECTION LIMIT connlimit  
| [ ENCRYPTED ] PASSWORD 'password' | PASSWORD NULL  
| VALID UNTIL 'timestamp'
```

[Grant an object privilege to a role:](#)



```
# Syntax example  
GRANT <privilege> on <object> to <role>;
```

[Revoke an object privilege from a role:](#)



```
# Syntax example  
REVOKE <privilege> ON <object> FROM <role>;
```

Grant a role to another role:



```
# Syntax example  
GRANT <role name> to <another role>;
```

Console Operation Instructions

Last updated : 2024-08-09 15:12:59

This document mainly focuses on interface-based database account operations on the console.

Creating an Account

TencentDB for PostgreSQL does not provide superuser role attributes and `pg_execute_server_program`, `pg_read_server_files` and `pg_write_server_files` roles for users to use. However, as some operations require superuser permissions, TencentDB for PostgreSQL offers the `pg_tencentdb_superuser` role as a substitute for the superuser role. For more information on `pg_tencentdb_superuser`, refer to [User and Permission Operations](#).

There are two types of console accounts: `pg_tencentdb_superuser` accounts and ordinary accounts. Any account that is a member of the `pg_tencentdb_superuser` role is a `pg_tencentdb_superuser` account; otherwise, it is an ordinary account.

You can log in to the [PostgreSQL console](#), and in the instance list, click **Instance ID** or **Manage** in the **Operation** column to go to the instance details page. Click **Account Management** > **Create account** to create an account.

Details are as follows:


Create account

Account Name *


Please enter your account name

The account name should contain 1-16 characters, including letters, digits, and underscores. It cannot be set to postgres, and can not start with a digit, pg_ or tencentdb_. All rules are case-insensitive.

Set Password *

Please enter account pa: 

Confirm Password *

Please enter account pa: 

Type

☒ General User ☐ pg_tencentdb_superuser [Account Type Introduction](#)

Remarks

Enter remarks

It can contain up to 60 letters, digits, or symbols (_-).

OK

Cancel

Note:

When an account is created on the console, the account name should meet the following criteria: 1-16 characters, consisting only of letters, numbers, or underscores; not being postgres; not beginning with a number, pg_, or tencentdb_; case insensitive for all rules.

The console only supports the creation of accounts for instance versions 9.5 and above. For instances whose version is below 9.5, upgrade their version first.

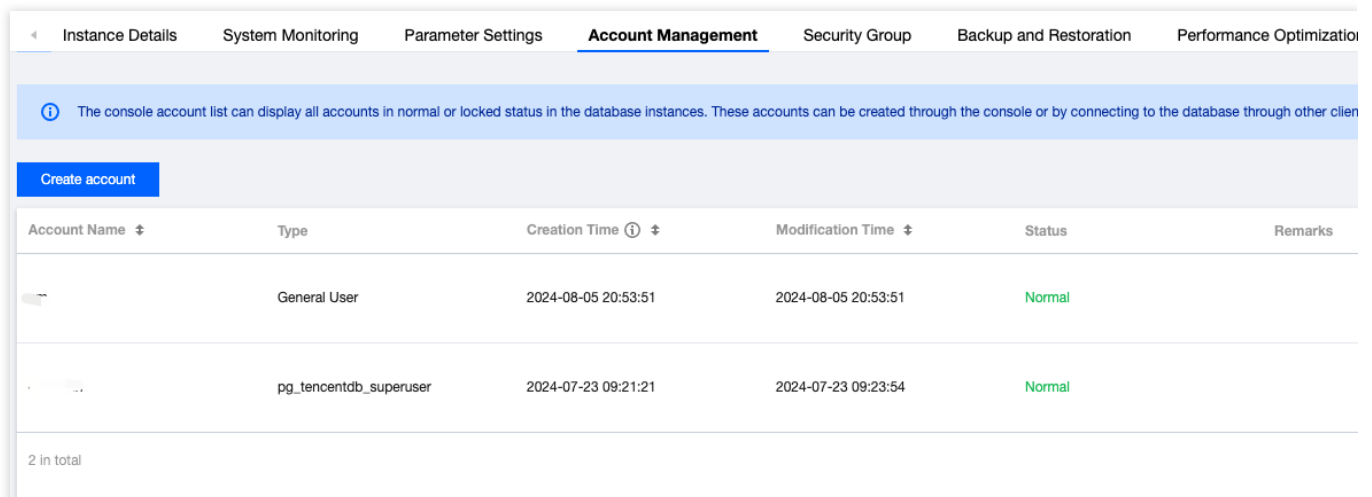
Account Display

Note:

1. The console account list can display all accounts in the database instance that are in normal or locked status. These accounts can be created through the console or other clients connected to the database.
2. The console only supports the display of users, not roles.
3. Only accounts created on the console will have their creation time recorded by the management system. Be aware of this.
4. When the system permissions of an account include nologin or CONNECTION = 0, the account is in the locked status.

5. The console only supports the display of database accounts that meet the account name requirements. Be aware of this.

Database accounts created through the console, other clients or programs can all be displayed in the account list. See the figure below for details:



Instance Details	System Monitoring	Parameter Settings	Account Management	Security Group	Backup and Restoration	Performance Optimization
<p>The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through other client.</p>						
Create account						
Account Name	Type	Creation Time	Modification Time	Status	Remarks	
	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal		
	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Normal		
2 in total						

Modifying an Account

Resetting the Password

You can reset the login password of an existing account on the console. If you have currently logged in to the database with the account, the reset password will be valid for subsequent connections. Click **Operation** > **Reset Password**, as shown in the figure below:

Instance DetailsSystem MonitoringParameter SettingsAccount ManagementSecurity GroupBackup and RestorationPerformance Optimization

The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through other client

Create account

Account Name	Type	Creation Time	Modification Time	Status	Remarks
	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Normal	

2 in total

The pop-up box is shown below:

Reset Password

Instance Name

Account Name

New Password

Please enter account password

Confirm Password

Please enter account password

OK

Cancel

Modifying Remarks

You can modify the account remarks on the console. Click **Operation** > **More** > **Modify Remarks**, as shown in the figure below:

Instance Details

System Monitoring

Parameter Settings

Account Management

Security Group

Backup and Restoration

Performance Optimization

The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through other clients

Create account

Account Name	Type	Creation Time	Modification Time	Status	Remarks
	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Normal	

2 in total

The pop-up box is shown below:

Modify Remarks

Account Name am

Remarks admin

It can contain up to 60 letters, digits, or symbols (._-).

OKCancel

Modifying the Type

There are two types of console accounts: pg_tencentdb_superuser accounts and ordinary accounts. Any account that is a member of the pg_tencentdb_superuser role is a pg_tencentdb_superuser account; otherwise, it is an ordinary account. For more information on pg_tencentdb_superuser, refer to [User and Permission Operations](#).

You can change the account type on the console. Click **Operation** > **More** > **Modify Type**, with details shown below:

Account Management					
The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through other clients					
Create account					
Account Name	Type	Creation Time	Modification Time	Status	Remarks
	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
pg_tencentdb_superuser	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Normal	
2 in total					

The pop-up box is shown below:

Modify Type

Account Name am

Type

☐ General User

☒ pg_tencentdb_superuser

Account Type

Introduction

OKCancel

Modifying Permissions

Description of Account Object Permissions

An account's object permissions consist of three parts:

Permissions inherited from roles: In PostgreSQL, a user can belong to one or more roles, and these roles can have specific permissions. For example, a role might have the permission to access a database or modify a table. If a user belongs to this role, then it will inherit all the permissions of the role.

Permissions inherited from PUBLIC: As mentioned earlier, PUBLIC is a special predefined group to which all users automatically belong. If a permission is granted to the PUBLIC group, then all users will have the permission.

Directly granted permissions: Besides the permissions inherited from roles and PUBLIC, a user can be directly granted permissions. For example, a database administrator can directly grant a user the permission to access a particular database or modify a table.

These three aspects of permissions can overlap with each other. To revoke a specific permission from a user, the permission should be revoked in all the three aspects. Granting and revoking object permissions do not apply to the OWNER. The OWNER has all permissions on an object.

Note:

The console's ability to modify permissions applies to directly granted permissions. If a permission of an account still exists after being revoked, you need to further check if the user has inherited the permission from the PUBLIC group or is the OWNER of the object.

On the console, you can grant or revoke multiple different permissions for multiple objects for an account, or grant or revoke multiple different permissions for multiple objects of the same category for an account in bulk.

Granting or Revoking Multiple Different Permissions for Multiple Objects for an Account

Click **Operation** > **Modify Permissions**. The pop-up box is as follows:

Instance Details

System Monitoring

Parameter Settings

Account Management

Security Group

Backup and Restoration

Performance Optimization

The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through other client.

Create account

Account Name	Type	Creation Time	Modification Time	Status	Remarks
--	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
pg_tencentdb_superuser	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Normal	

2 in total

The pop-up box is as follows:

Note:

Since account permissions can be modified through multiple clients or programs, it is strongly recommended that you click the **Refresh** button each time before operating permissions on the Console to retrieve the latest permissions.

Set Database Object Permission

You have selected 1 account(s). [Show Less](#) ▲

Account Name	Type
am	General User

Database permissions

[Batch Authorize/Deauthorize](#) [Reset](#) [Refresh](#)

Enter keywords for filtering.

Object-level permissions

postgres

public

☒ CONNECT ⓘ

☒ TEMPORARY ⓘ

☐ CREATE ⓘ

Public permissions ⓘ [Documentation](#)

☒ CONNECT

☒ TEMPORARY

☐ CREATE

☐ Full Authorization

[View Permission Description Documentation](#)

You can select up to 10 database(s) objects at a time to modify the permissions.

Each database object supports a maximum of 50 permission object(s) for modification.

OK

Cancel

Note:

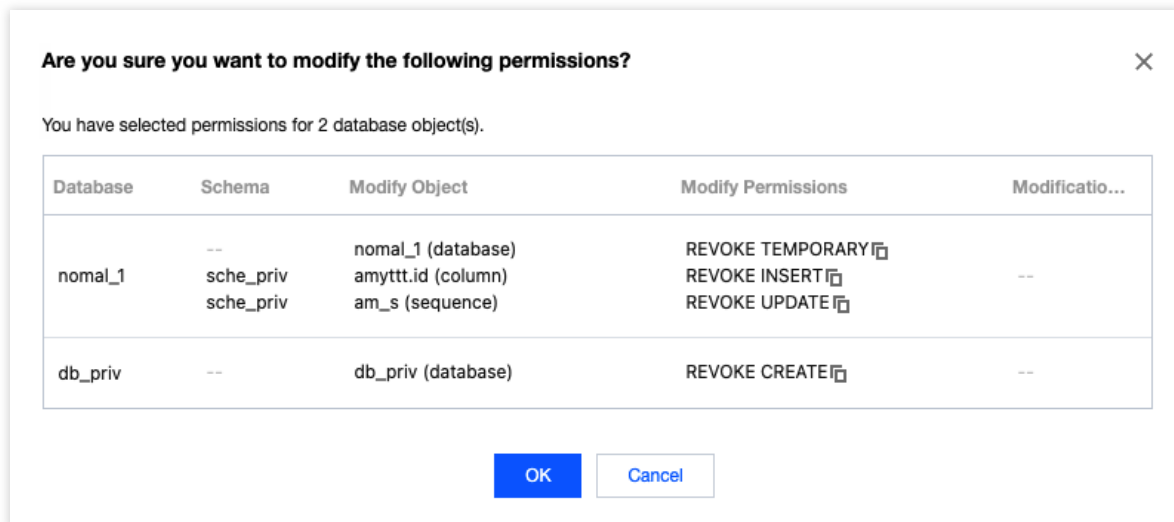
PUBLIC is a special predefined group to which all users automatically belong. When permissions are granted to the PUBLIC group, they are actually granted to all users, including existing and future users.

If a certain permission of an object is already possessed by the PUBLIC group, to revoke the permission, you should remove it from the PUBLIC group for the revocation to take effect.

Once a permission is revoked from the PUBLIC group, all users will lose the permission, so operate with caution.

To help you quickly revert to the initialization state after an erroneous click, click the **Reset** button. This will restore all interface operations to the current account permission status, allowing you to reoperate on this basis.

Click OK. The system will summarize all the permission operations you are about to initiate for the currently selected account, so as to facilitate your second confirmation. Details are shown in the figure below:



As shown above, this account's permission modification operations involve two databases: nomal_1 and db_priv, with details shown below:

1. Operations on the nomal_1 database involve the following three aspects:
 - 1.1 A REVOKE TEMPORARY operation is performed on the nomal_1 database.
 - 1.2 A REVOKE INSERT operation is performed on the id field of the amyttd table in the sche_priv schema.
 - 1.3 A REVOKE UPDATE operation is performed on the am_s sequence in the sche_priv schema.
2. A REVOKE CREATE operation is performed on the db_priv database.

Granting or Revoking Multiple Different Permissions for Multiple Objects of the Same Category for an Account

When you need to handle multiple permissions for objects of the same category for an account simultaneously, you can use the **Batch Authorize/Deauthorize** feature. The pop-up box is as follows:

Batch Authorize/Deauthorize ✕

You have selected 1 account(s). [Show More](#) ▼

☒ Authorize ☐ Deauthorize

Database permissions Reset

▼ Object-level permissions

☐ postgres

☐ nomal_1

☒ db_priv

☐ CONNECT ⓘ

☐ TEMPORARY ⓘ

☐ CREATE ⓘ

☐ All

ⓘ All permissions (including the granted ones) are deselected by default on this page.

You can select up to 10 database(s) objects at a time to modify the permissions.
Each database object supports a maximum of 50 permission object(s) for modification.

OK

Cancel

Note:

The Bulk Authorization/Revocation feature is for bulk operations on objects of the same category for an account. It does not support performing a bulk operation that includes the database and schema.

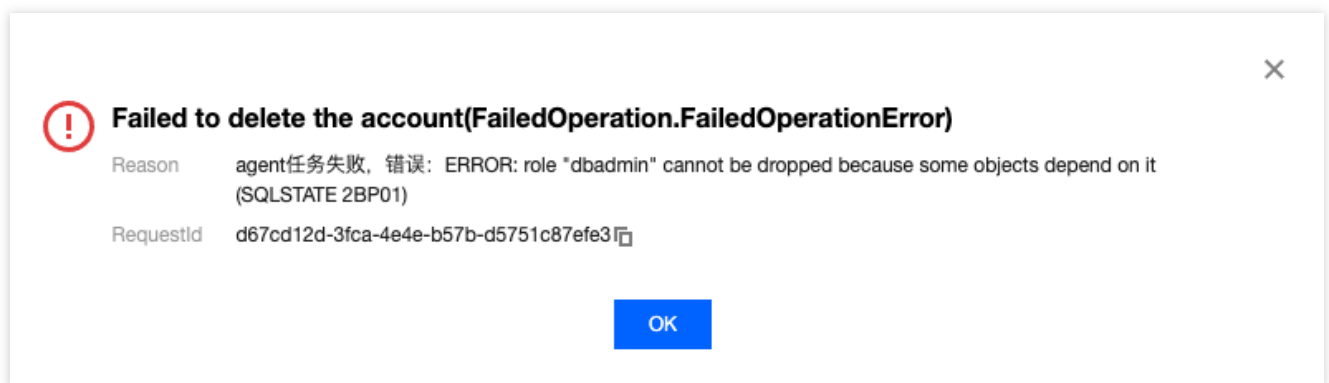
In bulk mode, the display of authorized status is not supported because permissions may vary. Operate with caution.

Deleting an Account

When you no longer need a certain database account, the console supports the deletion operation. Click **Operation > More > Delete Account**, with details shown below:

Instance Details	System Monitoring	Parameter Settings	Account Management	Security Group	Backup and Restoration	Performance
<div><div></div>The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database throu</div>						
<div>Create account</div>						
Account Name	Type	Creation Time	Modification Time	Status	Remarks	
	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal		
	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Normal		
2 in total						21

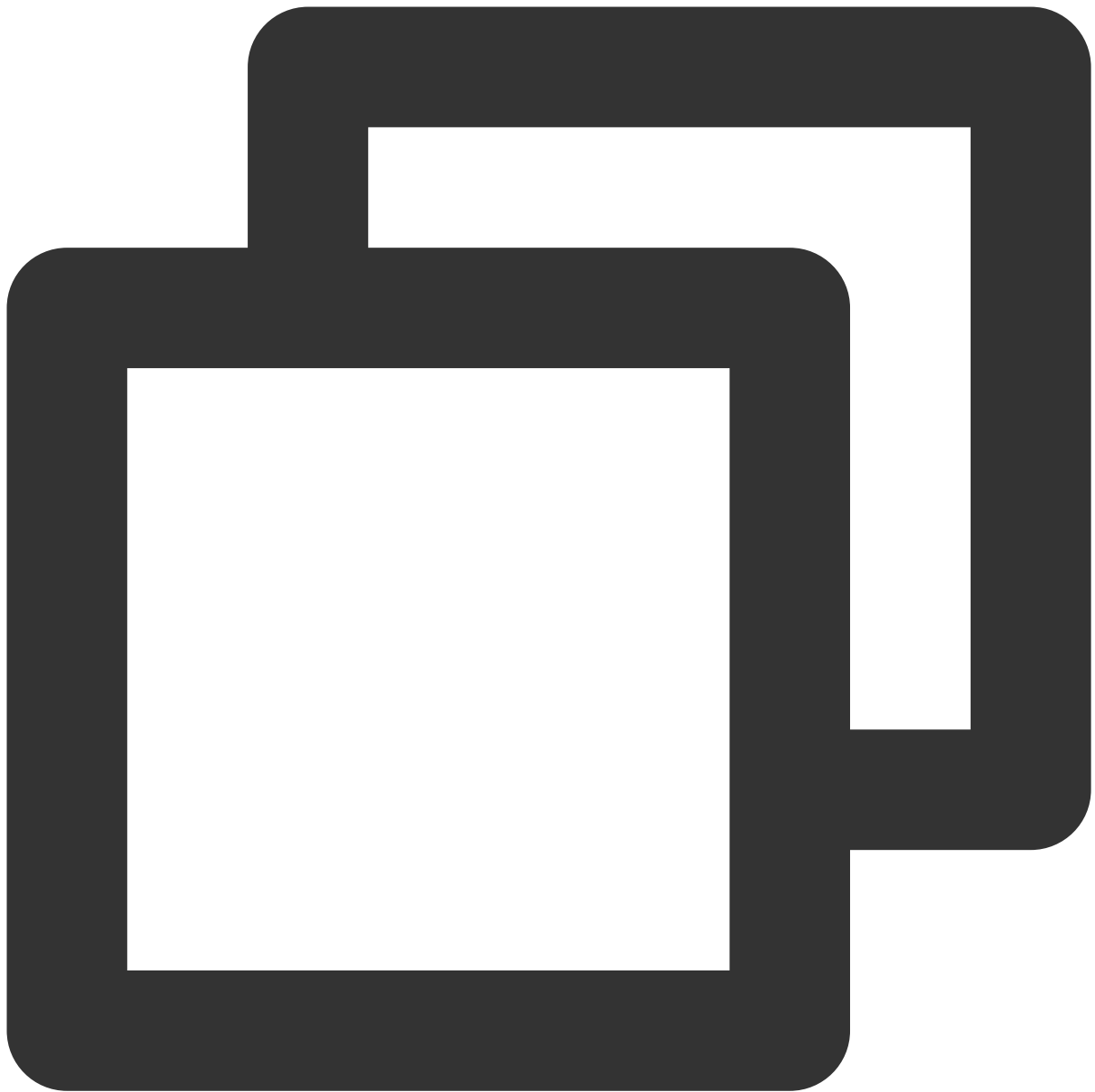
An account can be deleted only when it has no object permissions; otherwise, an error will be reported when the account is deleted, as shown in the figure below:



If an error is reported when a database account is deleted, we can check whether the account has permissions on certain objects or is the OWNER of certain objects. If so, revoke the corresponding permissions or replace the object's OWNER before re-authorizing the account. In PostgreSQL, aclitem is used to represent permissions on a specific database object. For a detailed explanation, refer to the [Overview of Database Permissions](#).

Methods to check whether an account has permissions on certain objects are as follows:

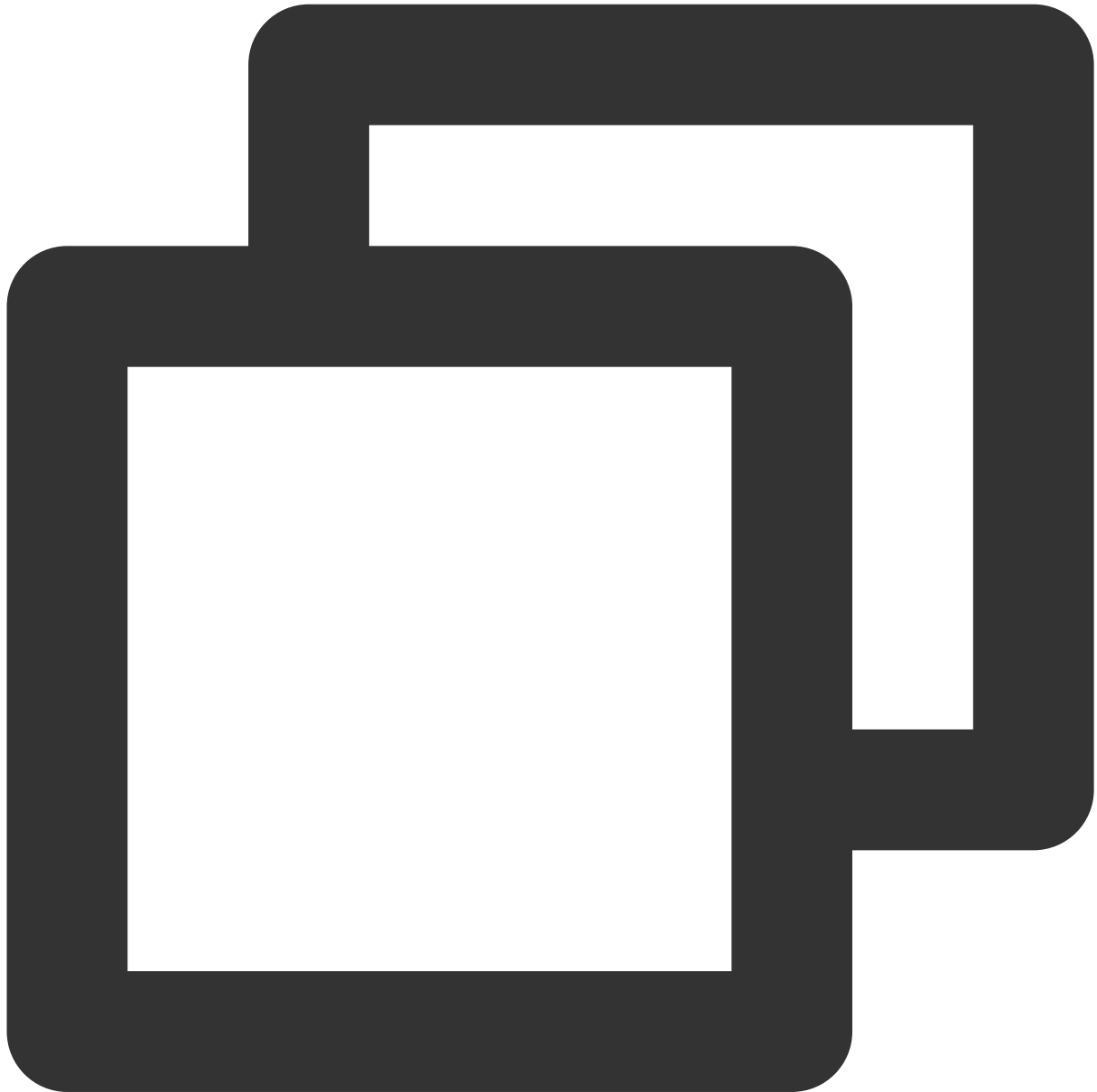
(1) To check whether the account has permissions on certain databases, refer to the following command:



```
nomal_1=> select datname,datacl from pg_database;
datname  | datacl
-----+-----
template1 | {=c/postgres,postgres=CTc/postgres}
template0 | {=c/postgres,postgres=CTc/postgres}
postgres  | {=Tc/postgres,postgres=CTc/postgres,pg_tencentdb_superuser=C/postgres}
nomal_1    | {=Tc/nomal_usr1,nomal_usr1=CTc/nomal_usr1}
db_priv    | {=Tc/dbadmin,dbadmin=Tc/dbadmin}
(5 rows)
```

As shown in the figure above, the datacl field clearly describes the account's permissions on the database. We find the corresponding account's permissions, revoke them, and then delete the account.

(2) To check whether the account has permissions on certain schemas, you can query under each database. Refer to the following command:

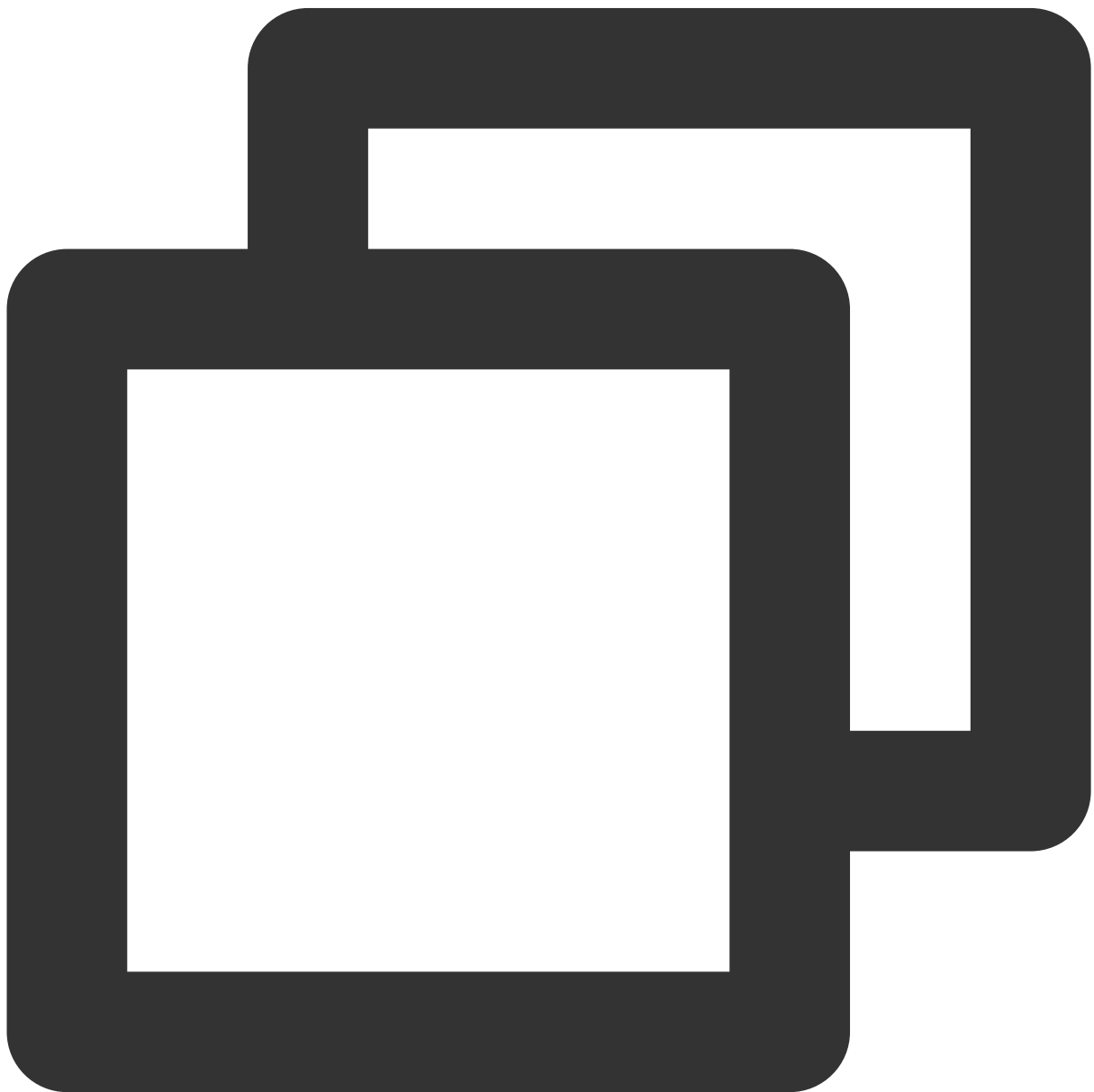


```
nomal_1=> select nspname,nspacl from pg_namespace;
          nspname          |          nspacl
-----+-----
 pg_toast                  |
 pg_temp_1                  |
 pg_toast_temp_1           |
```

```
pg_catalog      | {postgres=UC/postgres,=U/postgres}  
public         | {postgres=UC/postgres,=UC/postgres}  
information_schema | {postgres=UC/postgres,=U/postgres,nomal_usr1=C/postgres}  
sche_priv      | {dbadmin=UC/dbadmin}  
(7 rows)
```

As shown in the figure above, the datacl field clearly describes the account's permissions on the schema. We find the corresponding account's permissions, revoke them, and then delete the account.

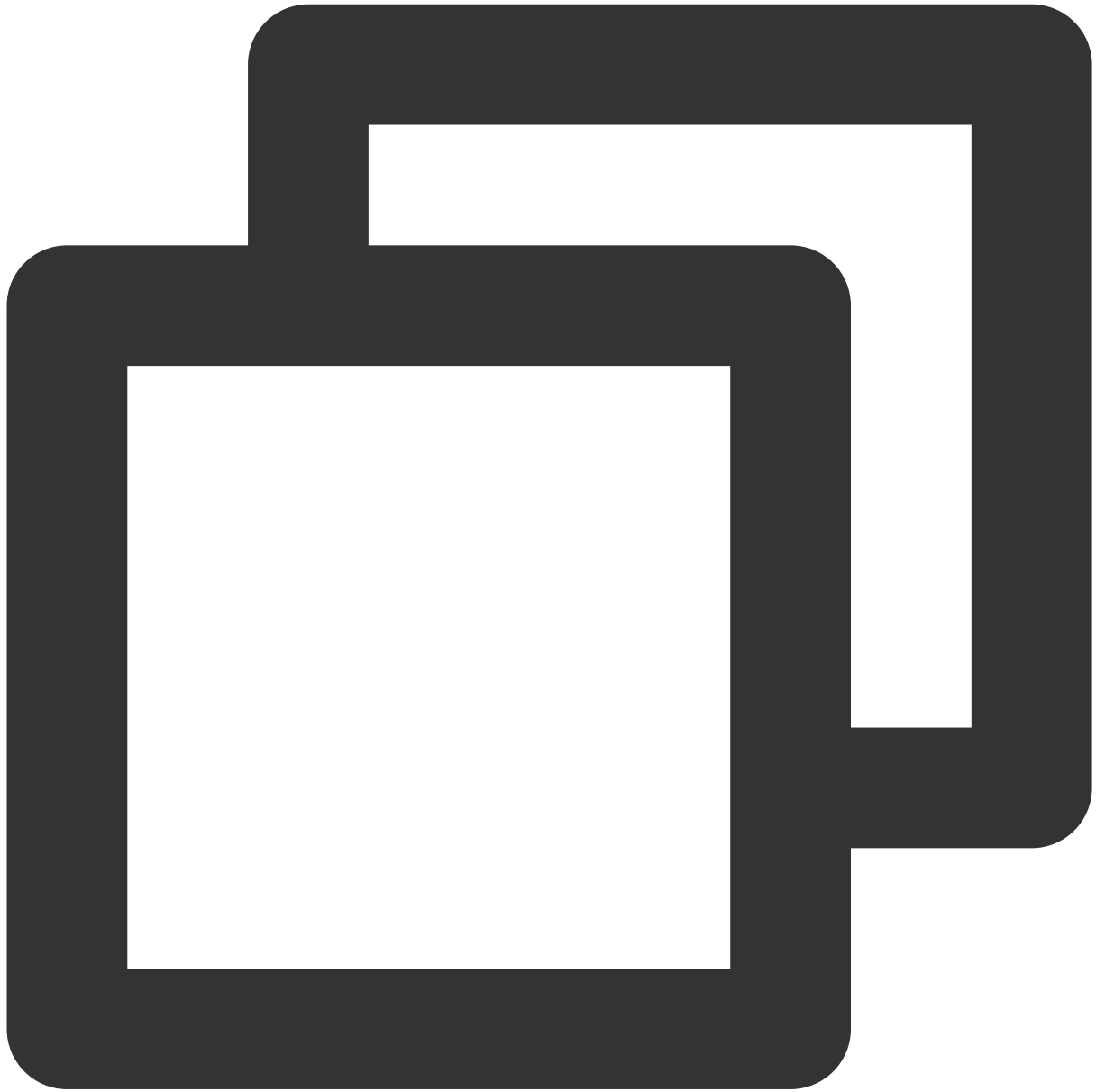
(3) To check whether the account has permissions on certain objects, you can query under each database. Refer to the following command:



```
nomal_1=> select relname,a.typname as reltype ,relacl from pg_class join pg_type a
```

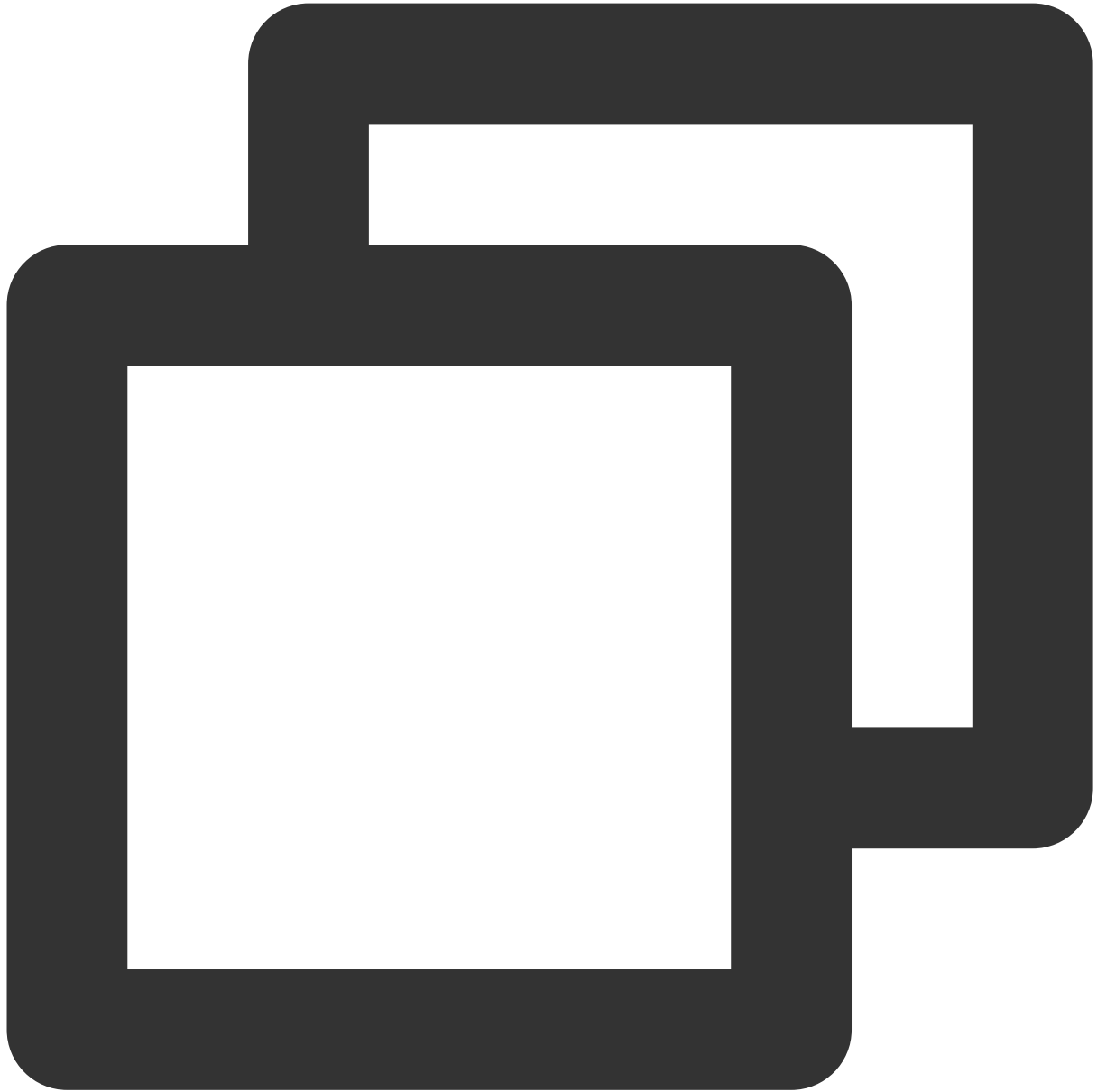
Examples of checking whether an account is the OWNER of a certain object are as follows:

(1) To check whether an account is the OWNER of a certain database, refer to the following example:



```
nomal_1=> SELECT r.rolname,d.datdba,datname AS database_name FROM pg_database d le
rolname | datdba | database_name
-----+-----+-----
dbadmin | 16398 | db_priv
nomal_usr1 | 16401 | nomal_1
```

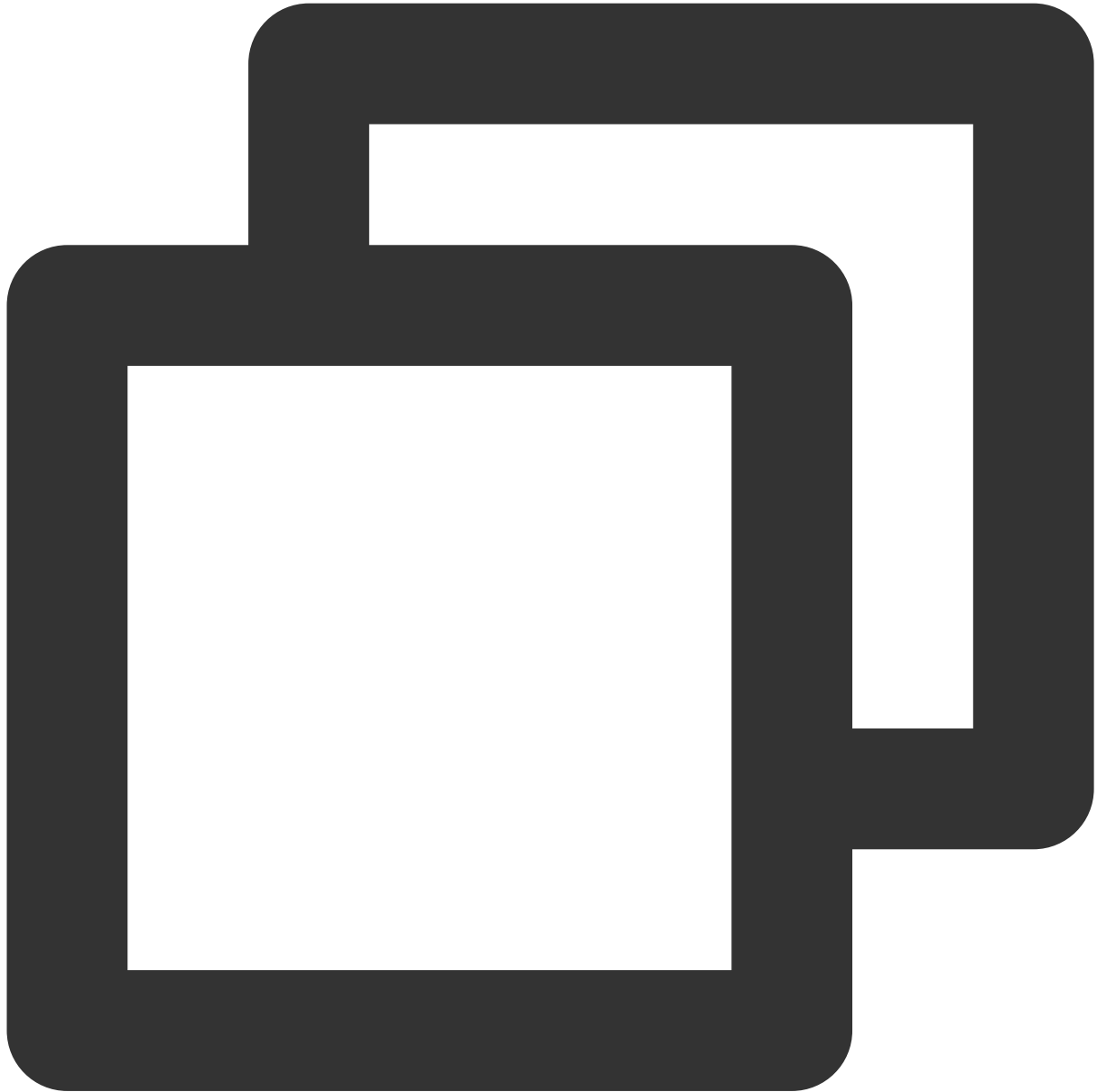
(2) To check whether an account is the OWNER of a certain schema, refer to the following example:



```
nomal_1=> SELECT n.nspname as schema_name, pg_catalog.pg_get_userbyid(n.nspowner)
           schema_name      | schema_owner
-----+-----
information_schema | postgres
pg_catalog         | postgres
pg_temp_1          | postgres
pg_toast           | postgres
pg_toast_temp_1    | postgres
public             | postgres
```

```
sche_priv      | dbadmin
```

(3) To check whether the account is the OWNER of a certain object, refer to the following example:



```
nomal_1=> select * from (select relname,relnamespace as schema_name ,a.typname as r
      relname      | schema_name |      reltype      | owner
-----+-----+-----+-----
pg_toast_16488 |          99 | pg_toast_16488 | dbadmin
amyttd          |        16480 | amyttd          | dbadmin
am              |         2200 | am              | dbadmin
am_s            |        16480 | am_s            | dbadmin
bug_id_seq      |         2200 | bug_id_seq      | dbadmin
```


bug		2200		bug		dbadmin
pg_toast_16507		99		pg_toast_16507		dbadmin

Locking an Account

Note:

When an account is locked, the management system will change the system permission to NOLOGIN and set CONNECTION to 0.

When an account is locked, any existing connection will take effect immediately.

You can lock an existing account by clicking **Operation** > **Lock account**, as shown below:

Instance DetailsSystem MonitoringParameter SettingsAccount ManagementSecurity GroupBackup and RestorationPerformance

The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through the client.

Create account

Account Name	Type	Creation Time	Modification Time	Status	Remarks
...	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
pg_tencentdb_superuser	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Normal	

2 in total20

The pop-up box is shown below:

Lock account

You have selected 1 account(s). [Show Less](#)

Account Name	Type
dbadmin	pg_tencentdb_superuser

OK

Cancel

After an account is locked, its status will become locked, as shown below:

Instance DetailsSystem MonitoringParameter SettingsAccount ManagementSecurity GroupBackup and RestorationPerformance

The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database thro

Create account

Account Name	Type	Creation Time	Modification Time	Status	Remarks
	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Locked	

2 in total

2

After an account is locked, you can unlock it on the console by clicking **Operation > More > Unlock Account**, as shown below:

<div>Instance DetailsSystem MonitoringParameter SettingsAccount ManagementSecurity GroupBackup and RestorationPerformance</div>					
<div><div></div>The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through the database client.</div>					
<div>Create account</div>					
Account Name	Type	Creation Time	Modification Time	Status	Remarks
General User	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
pg_tencentdb_superuser	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Locked	
2 in total					20

Note:

When an account is unlocked, the management system will change the account's system permission to LOGIN and set CONNECTION to -1.

Exporting a List

The console supports two types of account list export: downloading current page data and downloading all data, as shown below:

<div>Instance DetailsSystem MonitoringParameter SettingsAccount ManagementSecurity GroupBackup and RestorationPerformance</div>					
<div><div></div>The console account list can display all accounts in normal or locked status in the database instances. These accounts can be created through the console or by connecting to the database through the database client.</div>					
<div>Create account</div>					
Account Name	Type	Creation Time	Modification Time	Status	Remarks
General User	General User	2024-08-05 20:53:51	2024-08-05 20:53:51	Normal	
pg_tencentdb_superuser	pg_tencentdb_superuser	2024-07-23 09:21:21	2024-07-23 09:23:54	Locked	
2 in total					20

Database Optimization

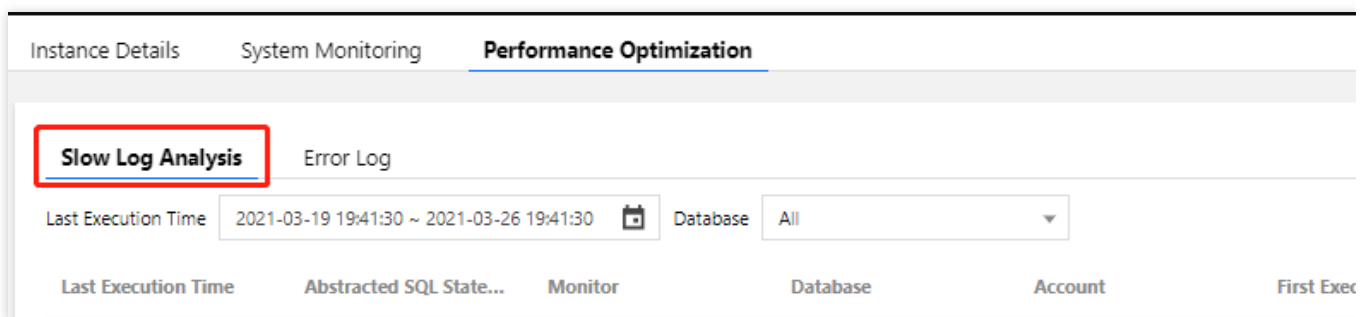
Slow Log Analysis

Last updated : 2024-01-24 11:16:51

Overview

By default, a SQL query that takes more than one second is a slow query, and the corresponding statement is a slow query statement. The process where a database administrator (DBA) analyzes slow query statements and finds out the reasons why slow queries occur is known as slow query analysis.

You can log in to the [TencentDB for PostgreSQL console](#), click an instance ID in the instance list to enter the instance management page, and select the **Performance Optimization** tab to analyze slow queries, as shown below:



Monitoring Views

There are two monitoring views in the console, visually and conveniently illustrating the monitoring data of database slow queries.

Combined View (Slow Log and Other Metrics): This view shows and compares the monitoring data of the slow query metric and another metric in the same chart. Supported metrics include CPU utilization, QPS, requests, read requests, write requests, other requests, buffer cache hit rate, and average execution latency.

Slow SQL Execution Time Distribution: This view shows in what time period slow queries mainly occur.

Slow SQL List

The slow SQL list shows slow query statements of the database in real time. The list is arranged in descending order by time, that is, the latest slow query statement is automatically displayed in the first row.

The slow SQL list has the following fields: the execution time, the slow SQL statement, the total time, the client IP, the database name, and the account executing the statement.

Note:

By default, the slow SQL list displays slow SQL data over the past seven days. The slow SQL data is stored in a log, and the oldest data is automatically deleted from the log to ensure that the log only stores data within the past seven days and the log size does not exceed 50 GiB.

Slow SQL queries larger than 20 KB in size cannot be viewed in the console. To view them, [submit a ticket](#).

Slow SQL Statistics and Analysis

The slow SQL statistics and analysis page shows the slow query statements with abstract parameter values within the specified time range and their aggregated statistical analysis results. The page has the following fields:

Last Execution Time: The time when the abstract statement is executed for the last time within the specified time range. As some statements may take a long time to execute, the `begin_time` of statement execution is logged as the last execution time.

Abstract SQL Statement: A slow query statement whose constants are removed. The abstract statement can be used for summary statistics of similar statements to facilitate your analysis.

Database: The database queried by the statement.

Account: The account executing the statement.

Client IP: The clients executing the statement.

First Execution Time: The time when the abstract slow query statement is executed for the first time within the specified time range (there may be many records after abstraction).

Total Execution Time: The total time consumed by the abstract slow query statement within the specified time range.

Avg Execution Time: The average time is calculated by dividing the total time consumed by the abstract slow query statement by the total number of its executions.

Min Execution Time: The minimum among all execution time of the abstract slow query statement. This parameter is used to determine whether the statement is sporadic.

Max Execution Time: The maximum among all execution time of the abstract slow query statement. This parameter is used to determine whether the statement is sporadic.

Total Time (%): The ratio in percentage of the total time consumed by the abstract slow query statement to the total time consumed by all abstract slow query statements within the specified time range.

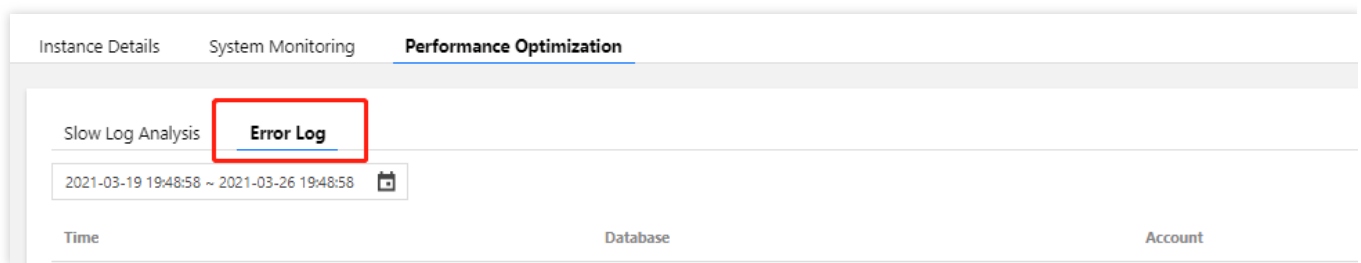
Error Logs

Last updated : 2024-01-24 11:16:51

Overview

Error logs refer to logs generated due to operation, SQL, and system errors during database running. Error logs are usually used by developers to find out the causes of errors in business systems or databases.

You can log in to the TencentDB for PostgreSQL console, click an instance ID/name in the instance list to access the instance management page, and select the **Performance Optimization** tab to view error logs as shown below.



Error Log Default Configurations

Error log feature: enabled by default.

Error log level: `log_min_error_statement=ERROR` .

Analyzed data output delay: 1–5 minutes.

Log retention period: 7 days (up to the last 10,000 records).

Parameter Management

Setting Instance Parameters

Last updated : 2024-01-24 11:16:51

You can view and modify certain parameters and query parameter modification logs in [TencentDB for PostgreSQL console](#).

Notes

To ensure instance stability, only some parameters can be modified in the console. These parameters are displayed on the **Parameter Settings** page.

If the modified parameter requires instance restart to take effect, the system will ask you if you wish to restart. We recommend that you do so during off-peak hours and ensure that your application has a reconnection mechanism.

Note:

Some are key parameters, modifying which may affect your use or the normal running of the instance. Therefore, exercise caution when modifying key parameters.

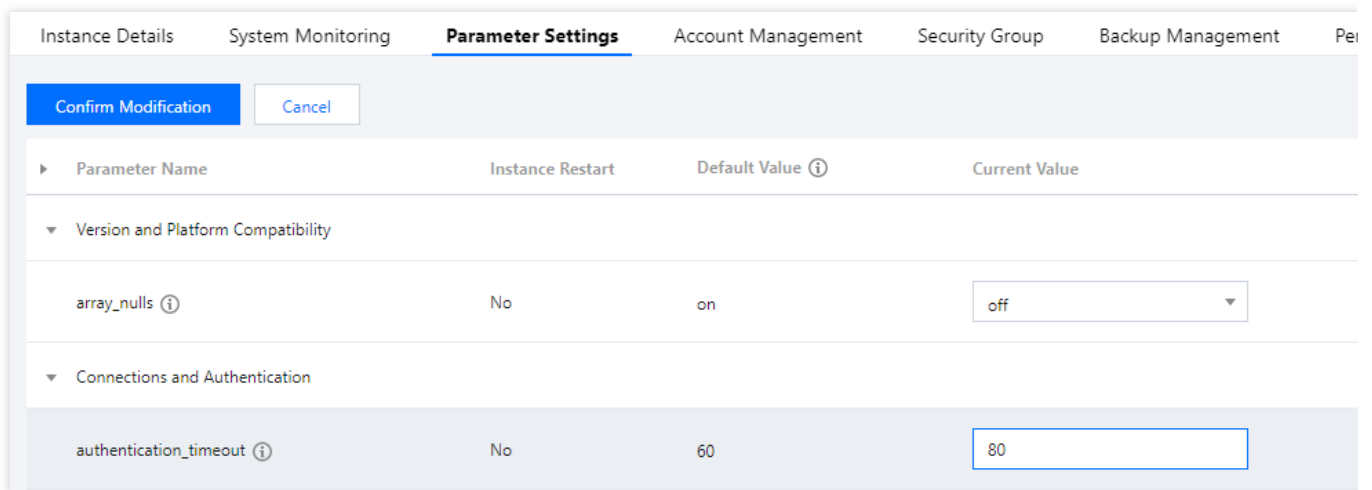
Modifying Parameters in the Parameter List

Modifying parameters in batches

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to access the instance management page.
2. On the **Parameter Settings** tab, click **Batch Modify Parameters**.

Instance Details	System Monitoring	Parameter Settings	Account Management	Security Group	Backup Manage
<div>Confirm Modification Cancel</div>					
Parameter Name	Instance Restart	Default Value ⓘ	Current Value		

3. Locate the desired parameters, and modify their values in the **Current Value** column. After confirming that everything is correct, click **Confirm Modification**.



Instance Details System Monitoring **Parameter Settings** Account Management Security Group Backup Management Per

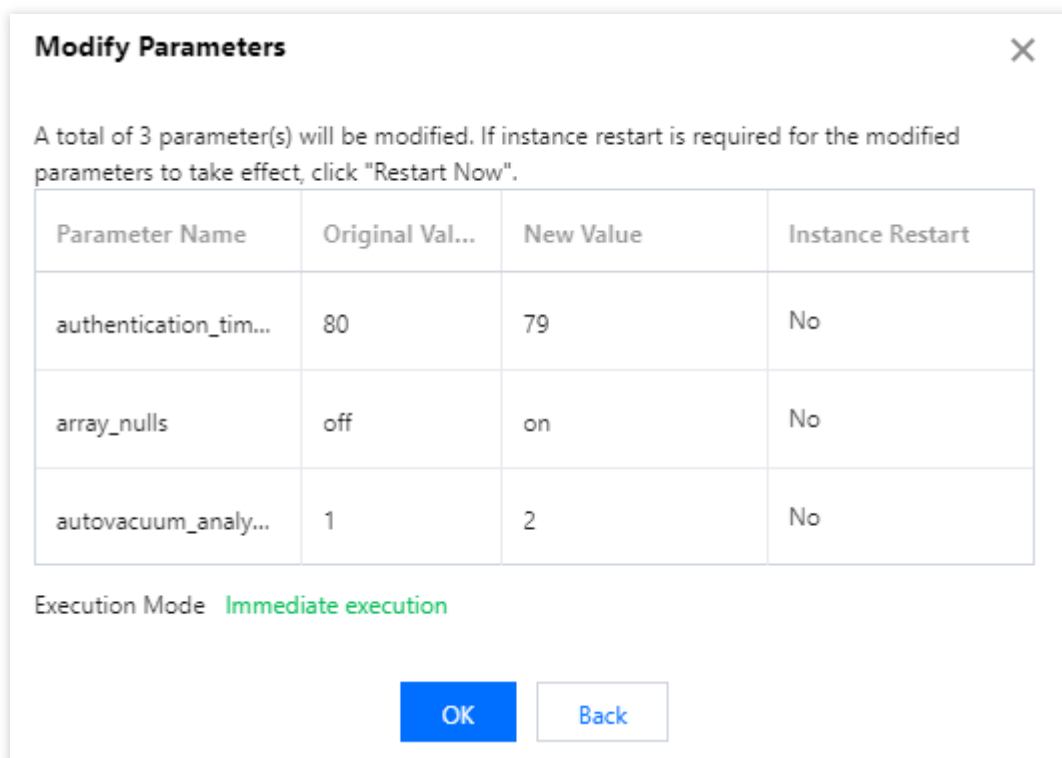
Confirm Modification Cancel

Parameter Name	Instance Restart	Default Value ⓘ	Current Value
▼ Version and Platform Compatibility			
array_nulls ⓘ	No	on	off ▼
▼ Connections and Authentication			
authentication_timeout ⓘ	No	60	80

4. In the pop-up dialog box, confirm that all parameter values are correctly modified, and click **OK**.

Note:

If there are modified parameters requiring instance restart to take effect, click **Restart Now** and then the instance will be restarted. The modified parameters take effect only after the restart is completed.



Modify Parameters ✕

A total of 3 parameter(s) will be modified. If instance restart is required for the modified parameters to take effect, click "Restart Now".

Parameter Name	Original Val...	New Value	Instance Restart
authentication_tim...	80	79	No
array_nulls	off	on	No
autovacuum_analy...	1	2	No

Execution Mode **Immediate execution**

OK Back

Modifying one parameter

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to access the instance management page.
2. On the **Parameter Settings** tab, locate the desired parameter in the parameter list and click



in the **Current Value** column.

Parameter Name	Instance Restart	Default Value ⓘ
Version and Platform Compatibility		
Connections and Authentication		
authentication_timeout ⓘ	No	60

3. Modify the value within the restrictions stated in the **Acceptable Values** column and click



to save the modification. You can click



to cancel the operation.

4. In the pop-up dialog box, confirm that the parameter value is correctly modified, and click **OK**.

Note:

If the modified parameter requires instance restart to take effect, click **Restart Now** and then the instance will be restarted. The modified parameter takes effect only after the restart is completed.

Viewing Parameter Modification Logs

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to access the instance management page.
2. On the **Parameter Settings** tab, click **Recent Modifications** on the right.
3. You can view the recent parameter modification records here.

← Recently Modified Parameters

Parameter Name	Original Value	Modified to	Modification Status	Modification Time
authentication_timeout	60	80	Successful	2021-10-14
autovacuum_analyze_scale_factor	0.1	1	Successful	2021-10-14
autovacuum_analyze_threshold	50	51	Successful	2021-10-14
array_nulls	on	off	Successful	2021-10-14

Parameter Value Limits

Last updated : 2024-01-24 11:16:51

This document describes instance parameters that cannot be modified due to instance specification limits and their required values.

Notes

Setting the following parameters will use system resources. To prevent the database use from being affected by parameter values, configure them as needed.

Note:

Some are key parameters, and changing them may affect your use or the normal running of the instance. Therefore, exercise caution when modifying key parameters.

Limits on Parameter Values Restricted by Specification

Specification\\Parameter	max_replication_slots	max_wal_senders	max_worker_processes	max_log
1C2GiB	[10-100]	[27-150]	[4-300]	[4-150]
2C4GiB	[10-100]	[27-150]	[4-300]	[4-150]
2C6GiB	[10-150]	[27-200]	[4-400]	[4-200]
4C8GiB	[10-150]	[27-200]	[4-400]	[4-200]
4C16GiB	[10-150]	[27-200]	[4-400]	[4-200]
6C24GiB	[10-200]	[27-250]	[4-500]	[4-250]
8C32GiB	[10-200]	[27-250]	[4-500]	[4-250]
8C48GiB	[10-200]	[27-250]	[4-500]	[4-250]
12C64GiB	[10-400]	[27-450]	[4-900]	[4--450]
16C96GiB	[10-400]	[27-450]	[4-900]	[4--450]
20C128GiB	[10-500]	[27-450]	[4-900]	[4--450]
28C240GiB	[10-600]	[27-650]	[4-1300]	[4--650]

48C480GiB	[10-600]	[27-650]	[4-1300]	[4--650]
-----------	----------	----------	----------	----------

Parameter Template

Last updated : 2024-05-11 14:27:59

Through the TencentDB for PostgreSQL console, you can create custom parameter templates to configure parameters in batches according to your business scenarios.

You can use the database parameter template to manage the configuration of database engine parameters. A parameter group acts like a container for engine configuration values, which can be applied to one or more database instances.

Parameter templates support the following features. Users can log in to the [PostgreSQL console](#), select the **Parameter Template** page from the left navigation to view parameters:

Support the creation of new templates to generate custom parameter optimization schemes.

Support importing templates from PostgreSQL configuration files `**.conf`.

Save parameter configurations as templates.

Import parameters from templates to apply to one or more instances.

Support comparing between two parameter templates.

Notes

Database instances that use a parameter template will not automatically update with the template; manual bulk instance updates are required.

You can apply the parameter changes to single or multiple instances by importing a template.

Creating a Parameter Template

To use your own database parameter template, you can create a new template, modify the necessary parameters and apply it to your instances.

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation, and click **Create Template**.

2. In the dialog box that appears, configure the following, and click **Create and Set Parameters**.

Template Name: Enter a unique template name.

Engine: Select the appropriate database engine, such as PostgreSQL or SQL Server compatible.

Database Version: Select a database version.

Template Description: Enter a brief description of the parameter template.

Create Parameter Template

1 Create Template

>

2 Set Template Parameters

Template Name *

Enter a template name

It can contain up to 60 letters, digits, or symbols (-_./[]+=:@).

Engine *

PostgreSQL

The engine is required.

Database Version *

16

Database version is required.

Template Description

Describe the template

Create and Set Parameters

Cancel

3. After creation, you can modify, import, and export parameters on the template details page.

Applying a Parameter Template to Instances

Note:

Before applying a parameter template to multiple instances, make sure that the instances do support those parameters.

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation.
2. In the Parameter Template list, find the template you need to apply, click **Apply to Instance**.

Tencent Cloud

OverviewProductsAPI Explorer

Database

Relational Database

TDSQL-CMySQLSQL ServerPostgreSQLInstance ListServerlessParameter Templates

PostgreSQL - Parameter Template

GuangzhouOther regions

Create Template

Template ID/Name	Database Version	Engine	Template Description
3677a24f-5a89-50e1-8787-4b3309b67885 high_performance	PostgreSQL 15	PostgreSQL	--
fb61e9e6-ce08-524b-a9aa-feef6c81c4e2 hight_level	PostgreSQL 15	PostgreSQL	--
5ab16721-73e4-5d73-b0e9-336b3c13c8df 1	PostgreSQL 14	PostgreSQL	1

3. On the pop-up page, select the execution mode and instances, ensure all parameter modifications are correct, then click **Submit**.

PostgreSQL Instance: Select the instances that need to apply the parameter template in the specified region.

Parameter Comparison: View the changed parameter values of the selected instance.

← Apply to Instance

Template ID/Name

9b2c90c7-0ef5-5001-bf82-a41413dc63d4 (1222)

Database Engine

PostgreSQL

Database Version

12

Region

Guangzhou 2

Other regions 0 ▾

PostgreSQL Instance

Available Instance

Search by instance ID

Instance ID/Name

Status

postgres-d8sc3vdb (Unnamed)

Running

Hold the Shift key down to select multiple items

Selected Instance (1)

Instance ID/Name

postgres-d8sc3vdb (Unnamed)

↔

Compare Again

Remove All Instances

Parameter Comparison ⓘ

☐

Only preview changed parameters

Parameter Name

postgres-d8sc3 (Unnamed)

No changed parameters

Back

Submit

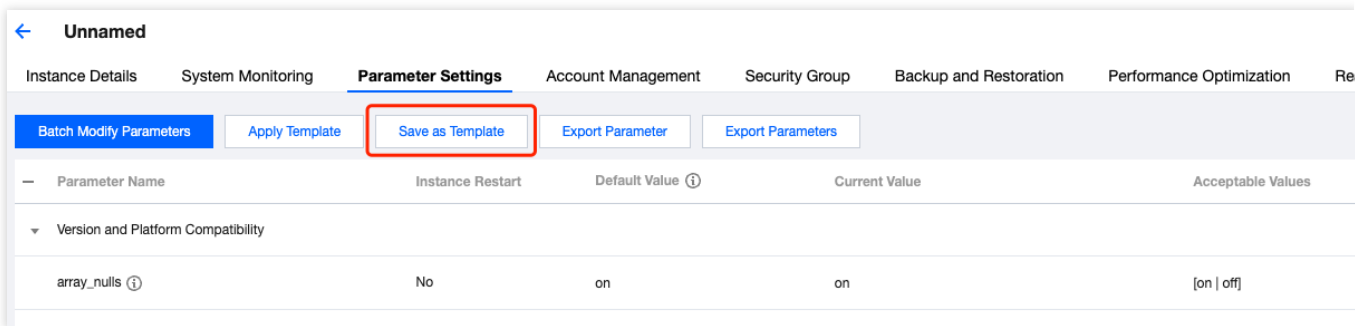
Copying a Parameter Template

To include most of the custom parameters and values of an existing parameter template in a new template, you can copy the existing template.

Option 1. Copying an existing parameter template

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation, click on the Template ID or the **Operation** column's **View Details** to enter the template details page.

2. On the template details page, click **Save as Template**.



3. In the pop-up dialog box, specify the following configurations:

Template Name: Enter a unique template name.

Template Description: Enter a brief description of the parameter template.

The dialog box is titled 'Save as Parameter Template'. It has two input fields: 'Template Name' and 'Template Description'. The 'Template Name' field contains the text 'new' and has a red asterisk next to it. Below it is a message: 'The template name is required.' The 'Template Description' field is empty and has a placeholder text: 'Describe the template'. At the bottom are 'OK' and 'Cancel' buttons.

4. After confirming that everything is correct, click **OK** to save the current parameter template as a new one, completing the copying operation.

Option 2. Saving parameters of an instance as a parameter template

1. Sign in to the [PostgreSQL console](#), select **Instance List** on the left navigation, click an instance ID to enter the management page.

2. Select the **Parameter Settings** tab, click **Save as Template**.

3. In the pop-up dialog box, specify the following configurations:

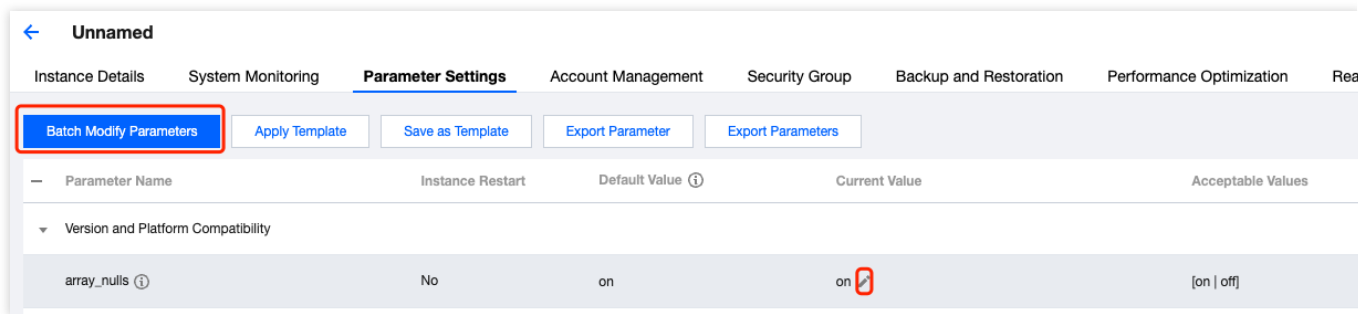
Template Name: Enter a unique template name.

Template Description: Enter a brief description of the parameter template.

4. After confirming that everything is correct, click **OK** to save the current parameter template as a new one, completing the copying operation.

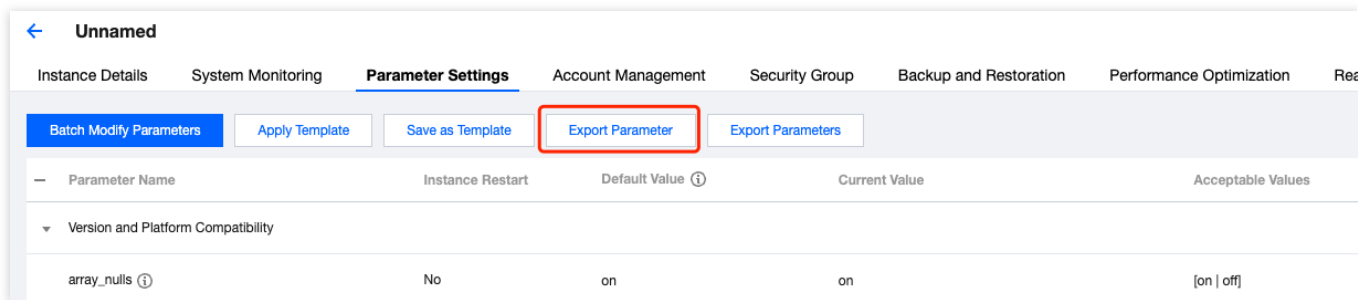
Modifying Parameter Values in a Parameter Template

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation, and click on the Template ID to enter the template details page.
2. On the template details page, click **Modify Parameters in Batches**, or in the running value column, click the "Modify Parameter Value" icon.



Importing a Parameter Template

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation, and click on the Template ID to enter the template details page.
2. On the template details page, click **Import Parameters**.



Note:

When choosing to import parameters, on the local file import page, when selecting a parameter configuration file, it is important to ensure that the format of the configuration file matches the configuration file format of the PostgreSQL database server, or use the file template of exported parameters, otherwise, an import failure message will appear.

3. In the pop-up window, select a file and click **Import and Overwrite Original Parameters**.

Exporting a Parameter Template

Method 1

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation.

2. In the parameter template list, in the **Operation** column of the desired template, click **Export**.

Method 2

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation, and click on the Template ID to enter the template details page.
2. At the top of the template details page, click **Export Parameters**.

Deletes parameter template

If a parameter template is created redundantly or no longer needed, it can be easily deleted.

1. Sign in to the [PostgreSQL console](#), select **Parameter Template** on the left navigation.
2. In the parameter template list, in the **Operation** column of the desired template, click **Delete**.

PostgreSQL - Parameter Template			
<div>Create Template</div>			
Template ID/Name	Database Version	Engine	Template Description
3677a24f-5a89-50e1-8787-4b3309b67885 high_performance	PostgreSQL 15	PostgreSQL	--
fb61e9e6-ce08-524b-a9aa-feef6c81c4e2 hight_level	PostgreSQL 15	PostgreSQL	--

模板 ID / 名称	数据库版本	引擎	模板描述
高性能	PostgreSQL 14	PostgreSQL	--

3. In the pop-up window, click **OK**.

Log management

Execution Log Management

Last updated : 2024-03-20 14:27:25

This document introduces how to manage the retention duration of pg_log.

Introduction to PG_LOG

pg_log typically records the status information of the database, such as error information, slow log SQL, database startup and shutdown information, etc. This log will be automatically segmented by size and time, and the pg_log of TencentDB for PostgreSQL is retained by default for 30 days. pg_log will occupy the storage space of the database instance, and you can modify the retention duration based on your needs.

Note:

The slow log and error log of the database instance are retained for 7 days by default. Modifying the pg_log retention duration does not affect the slow log and error log retention duration.

Modifying PG_LOG Retention Duration

You can modify the pg_log retention duration based on your needs. The system currently supports two options: 7 days and 30 days. The detailed steps are as follows:

1. Sign in [TencentDB for PostgreSQL Console](#).
2. Find the instance you need to modify in the instance list, click **Operation** > **Management** to enter the instance details.
3. Find **Parameter Setting** in the Instance detail page, search for the parameter log_filename in the upper right corner and update it.

The introduction for the log_filename parameter is as follows:

Parameter Value	Description
postgresql_%a_%H.log	Selecting this value will keep logs for 7 days.
postgresql_%d_%H.log	Selecting this value allows logs to be retained for 30 days, which is the system default.

Backup and Restoration

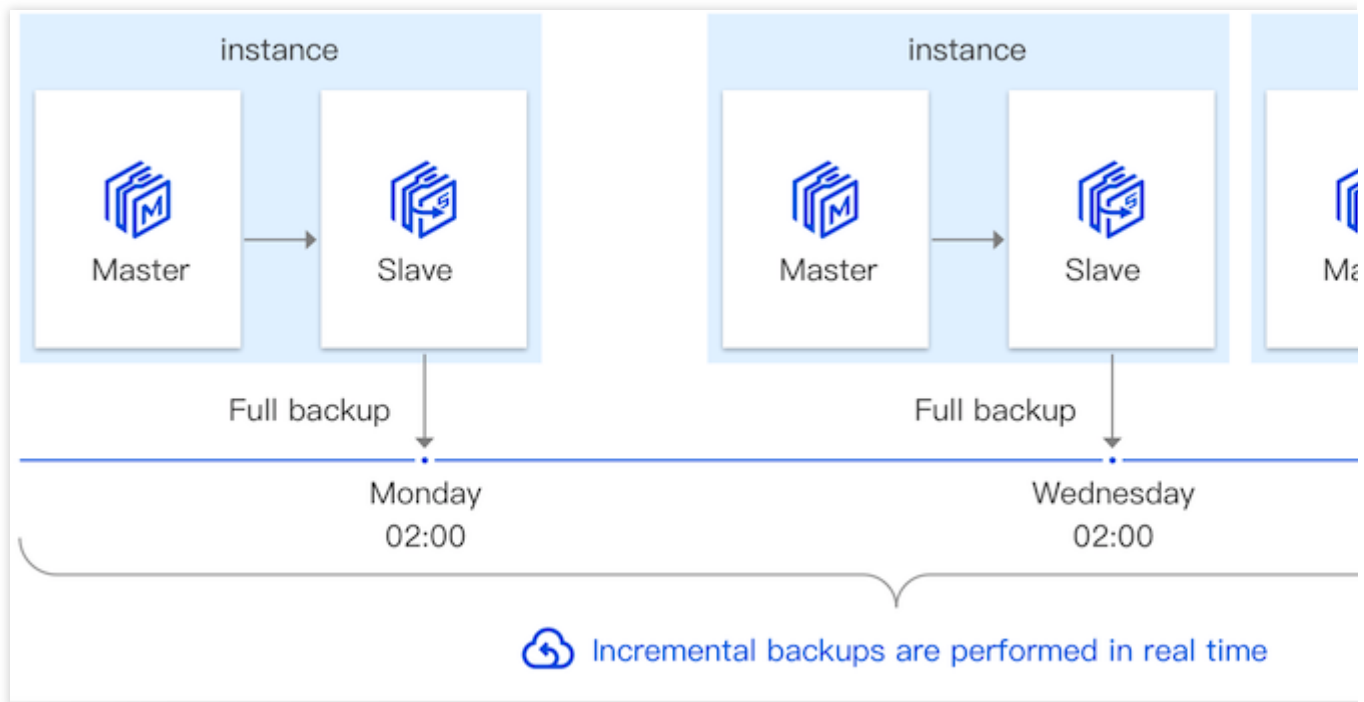
Backup Principles and Solutions

Last updated : 2024-03-20 14:30:33

TencentDB for PostgreSQL supports **Data Backup**, **Manual Backup**, and **Log Backup**. You can schedule regular database backups through automatic backup settings. In the event of a database failure or accidental deletion, you can restore the database from the stored backup files. TencentDB for PostgreSQL supports storing database backups in a compressed format. The size of a compressed backup file is approximately 30% of its original size (the exact compression ratio depends on the amount of duplicated data stored in the instance, the more duplicate data, the greater the compression ratio).

Backup Principles

For the double-availability (primary-standby) architecture, when a backup task is triggered, the system pulls data from the instance's standby node and compresses it before uploading it to the object storage service. Backup space does not occupy the disk space of the instance. If a database failure or accidental deletion occurs, you can recover the database from the stored backup files. When restoring data, you can use the cloning method, please refer to [Cloning Instances](#), or you can download the backup files for recovery, please refer to [Restoring PostgreSQL Data on CVMs](#). The principle is shown in the following figure:



Backup Plans

Operation Type	Backup Type	Operation Details
Data Backup	Data Backup	<p>The system will trigger a full data backup according to your automatic backup setting within the specified time. The backup method is physical backup, which offers fast backup speed and higher recovery efficiency.</p> <p>The generated full automatic backup data is retained based on the data backup retention time you set. The backup data will not be deleted when the instance is terminated. It will be automatically deleted when it expires. This can meet your need to extend the retention of backup data to prevent serious impact due to accidental deletion of instances. Since full backup uses backup storage space, you can delete it promptly if necessary.</p> <p>The automatic backup data within one week cannot be deleted, while data older than one week can be flexibly deleted according to your needs. Please be cautious when deleting backup data as it cannot be recovered after deletion.</p>
	Manual Backup	<p>You can manually initiate a full instance backup based on application needs through the console. After the manual backup task is initiated, the system will perform a full instance backup using the physical backup method within 1 minute.</p>

		The expiration time of manual backups is one week after they are initiated. Since manual backups occupy backup storage space, you can delete manual backups promptly if necessary.
	Incremental Backup	Incremental backup is WAL log backup, which is automatically enabled by default and cannot be disabled. The incremental backup is stored based on the data backup retention time you set. The system will perform real-time backup based on the WAL log generated by the database. Since incremental backup occupies backup storage space, you can delete it promptly if necessary. Incremental backup data within a week is not allowed to be deleted, while data older than one week can be flexibly deleted according to your needs. After the incremental backup data is deleted, point-in-time recovery cannot be performed, so exercise caution when deleting incremental backup data.
Backup Files Download	Download Full Backup	Supports local browser download and download by address.
	Download Incremental Backup	Supports local browser download and download by address.
Transfer Historical Backup Through Serverless Cloud Function	Transfer through Serverless Cloud Function	You can transfer historical backup data using the Serverless Cloud Function. For more information, please see Use Serverless Cloud Function to transfer historical PostgreSQL backup .

Backup Fees

TencentDB for PostgreSQL Backup files are stored in the form of compressed files, and the compressed backup file size is about 30% of the original file size. The single instance backup space is given according to the purchased capacity and most instances don't need to pay. For the specific billing rules, please refer to [Backup Space Billing](#).

Backing up Data

Last updated : 2024-04-09 10:47:26

This document describes how to set backup parameters and download backup files via the TencentDB for PostgreSQL console.

Operation Scenarios

Currently, TencentDB for PostgreSQL High Availability Edition only supports physical backup:

Full backup: once a day at 01:00.

Incremental backup: for xlog files, backup will be performed once every 15 minutes or when the number of files reaches 60.

Data file retention period: Range: 7 to 1830 days.

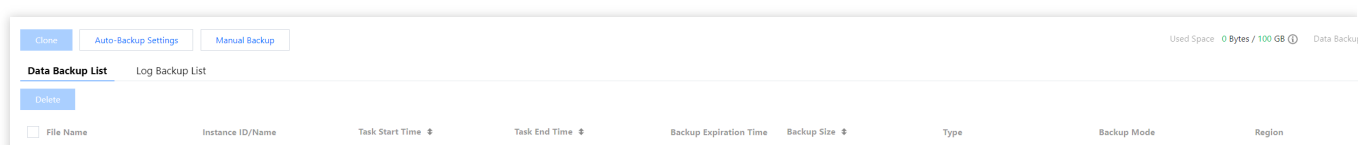
Manual Backup: If you initiate a backup task manually, the task will start within one minute after you click **OK**.

Directions

1. Log in to the [TencentDB for PostgreSQL Console](#) and click an instance ID to enter the instance details page.
2. On the **Backup Management** tab, click **Date Backup List** or **Log Backup List**, select the desired backup, and click **Download** in the "Operation" column.

Note:

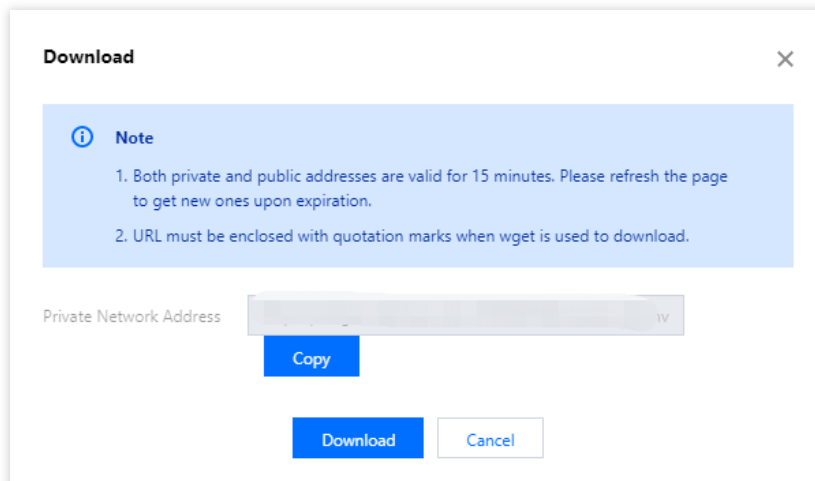
Backup data in the **Date Backup List** is full backup, while that in the **Log Backup List** is incremental backup.



3. In the pop-up window, you can choose a VPC address or public network address to download the backup file.

Note:

To ensure data security, an address will be valid for 12 hours. After it expires, please refresh the page to get a new address. Please access a VPC address in a VPC.



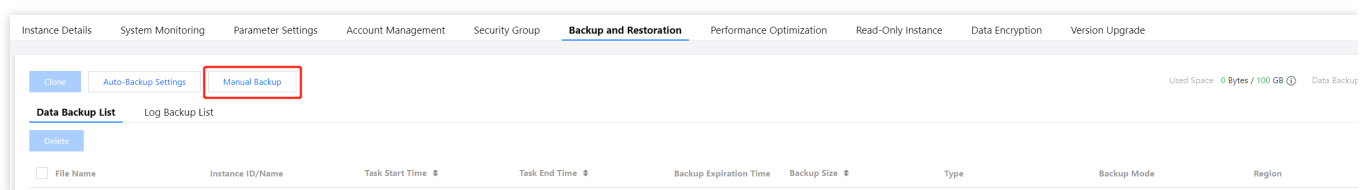
4. You can initiate backup tasks as needed. On the **Backup and Restoration** tab, click **Manual Backup** to start a manual backup task.

Note:

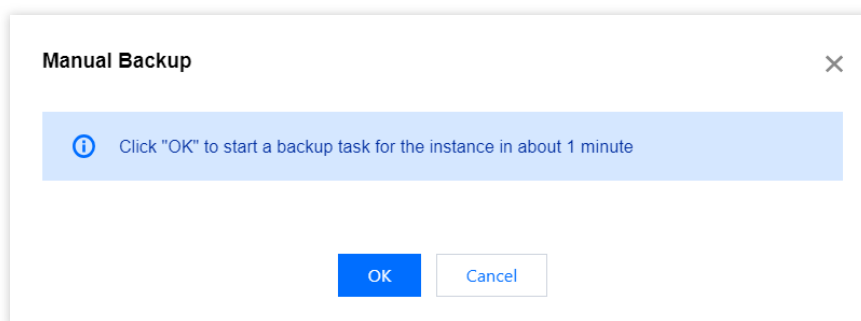
You can not initiate a manual backup during the execution of the routine auto backup tasks.

You can delete manual backups in the backup list to free up backup space, avoiding space waste and occupation.

Otherwise, these manual backups will be retained until the database instance goes offline.



5. In the **Manual Backup** dialog, click **OK**.



Note:

The creation time of a manual backup relates to the actual capacity of the instance. Larger capacity results in longer creation time.

FAQs

Can backups exceeding the retention period still be downloaded or restored?

Expired automatic backup sets will be automatically deleted and cannot be downloaded or restored. You can manually backup instance data in the console, and manual backups are always saved.

Can backups be manually deleted?

Automatic backups within 7 days cannot be deleted. Those beyond 7 days can be deleted as needed, and the system will also delete automatic backups based on their corresponding retention durations. Manual backups can be manually deleted from the backup list; otherwise, they will be retained permanently.

Can data and log backups be disabled?

No, they cannot be disabled. However, you can lower the backup frequency and delete unnecessary to reduce the backup space usage.

How can I reduce backup space overhead?

Delete unnecessary manual backup data.

Lower the automatic backup frequency for non-core service data (at least twice a week).

Downloading Backup

Last updated : 2024-04-09 10:54:27

This document describes how to download backup files in the TencentDB for PostgreSQL console.

Note

Both private address and local download address are valid for 12 hours. Refresh the page to get new ones upon expiration.

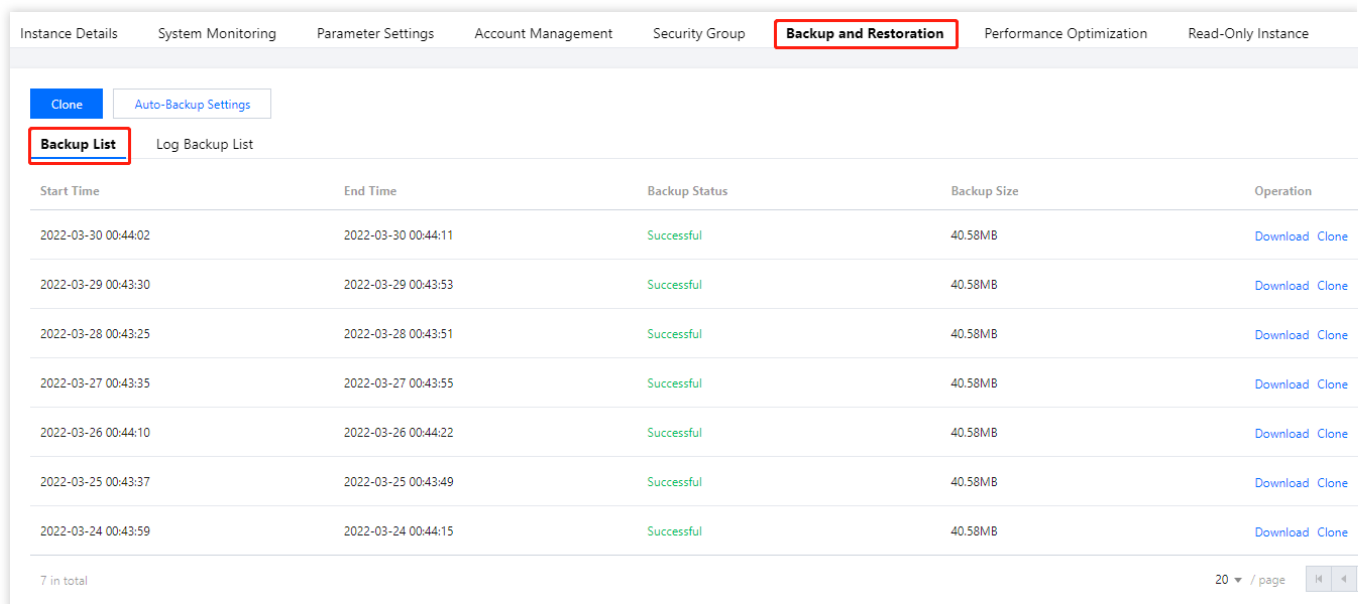
The URL must be enclosed with quotation marks when the `wget` command is used to download.

After the file is downloaded, you need to decompress it using `zstd`. If this tool is not available, install it.

Directions

Data backup download

1. Log in to the [TencentDB for PostgreSQL console](#), select a region, and click an instance ID to access the instance management page.
2. On the instance management page, select the **Backup and Restoration** tab and click **Backup List**.



The screenshot shows the TencentDB for PostgreSQL console interface. The 'Backup and Restoration' tab is selected and highlighted with a red box. Below it, the 'Backup List' button is also highlighted with a red box. The table displays a list of backups with columns for Start Time, End Time, Backup Status, Backup Size, and Operation. All backups shown are 'Successful' and 40.58MB in size. The 'Operation' column contains 'Download' and 'Clone' links for each backup.

Start Time	End Time	Backup Status	Backup Size	Operation
2022-03-30 00:44:02	2022-03-30 00:44:11	Successful	40.58MB	Download Clone
2022-03-29 00:43:30	2022-03-29 00:43:53	Successful	40.58MB	Download Clone
2022-03-28 00:43:25	2022-03-28 00:43:51	Successful	40.58MB	Download Clone
2022-03-27 00:43:35	2022-03-27 00:43:55	Successful	40.58MB	Download Clone
2022-03-26 00:44:10	2022-03-26 00:44:22	Successful	40.58MB	Download Clone
2022-03-25 00:43:37	2022-03-25 00:43:49	Successful	40.58MB	Download Clone
2022-03-24 00:43:59	2022-03-24 00:44:15	Successful	40.58MB	Download Clone

7 in total

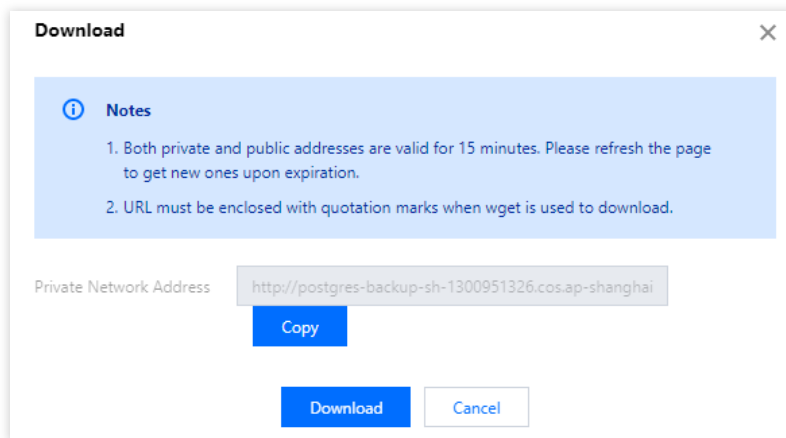
20 / page

3. Select the backup you want to download from the backup list and click **Download** in its **Operation** column.

4. In the download window, both VPC address and local download address are provided for download.

Note:

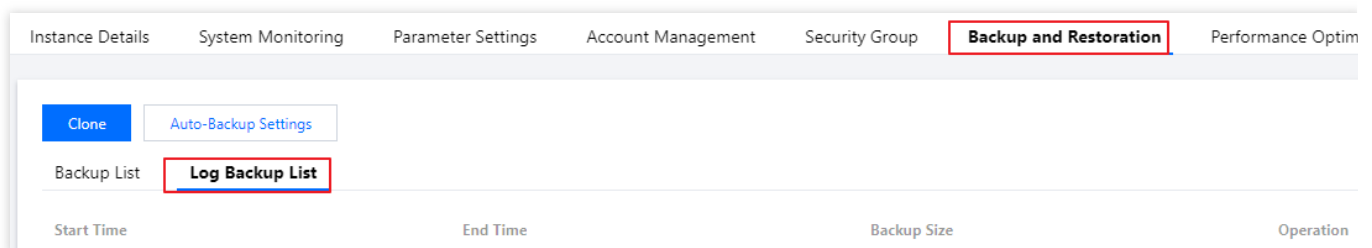
We recommend that you copy the download address, log in to a Linux CVM instance in the same VPC as the TencentDB instance, and run the `wget` command for fast download over the private network. For more information, see [Customizing Linux CVM Configurations](#).



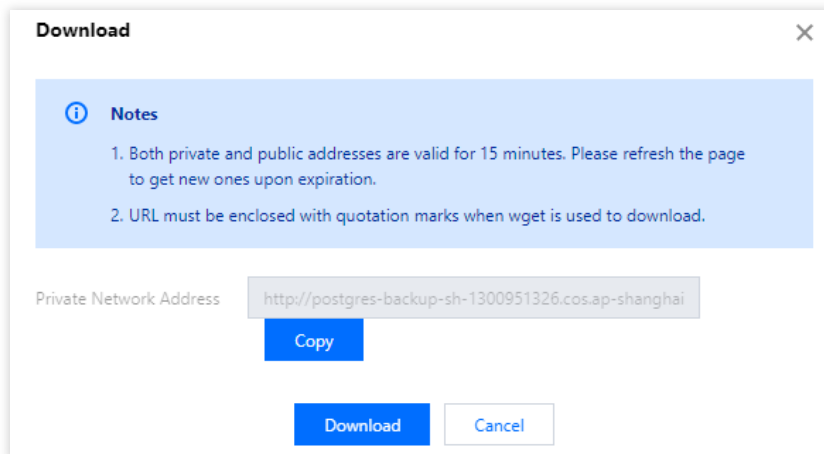
5. You can also click **Database Backup** on the left sidebar to enter the **Backup List** page, and click **Data Backup List** to download the data backup.

Log backup download

1. Log in to the [TencentDB for PostgreSQL console](#), select a region, and click an instance ID to access the instance management page.
2. On the instance management page, select the **Backup and Restoration** and click **Log Backup List**.



3. Select the backup you want to download from the log backup list and click **Download** in its **Operation** column.
4. In the download window, both VPC address and local download address are provided for download.



5. You can also click **Database Backup** on the left sidebar to enter the **Backup List** page, and click **Log Backup List** to download the data backup.

Cloning Instance

Last updated : 2024-01-24 11:16:51

This document describes how to clone a TencentDB for PostgreSQL instance in the console. This feature enables you to quickly restore original instance data from a backup to a newly purchased instance.

Overview

You can restore a TencentDB for PostgreSQL instance to any time point within the log backup retention period or from a specific physical backup set through instance clone. The clone is a new instance created from the backup data according to the restoration time point you specify. After the clone is verified, you can migrate its data back to the original instance with [DTS](#) or directly use the clone.

Clone mode

Clone an instance and restore the clone to any time point within the log backup retention period you specify.

Clone an instance and restore the clone from a specific physical backup set within the data backup retention period you specify.

Clone billing

You can select a billing mode for the clone during the clone process in the same way as during instance purchase.

The clone will not be billed until the clone process is completed.

Prerequisites

The original instance must be in the **Running** status.

If the clone mode is set to **By backup set**, the original instance must have created at least one physical backup. You can log in to the [console](#), select **Database Backup** on the left sidebar, and view backup status on the **Backup List** tab.

Your account balance must be positive.

Notes

The hard disk space of the clone must be larger than the amount of the data to be cloned from; otherwise, the clone task may fail.

The database version of the clone must be the same as that of the original instance.

For instances with a used capacity greater than 6 TB, [submit a ticket](#) for data restoration.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. On the instance management page, select **Backup and Restoration** > **Backup List** and click **Clone** in the top-left corner, or locate the target backup and click **Clone** in the **Operation** column.

The screenshot shows the TencentDB for PostgreSQL console interface. The top navigation bar includes tabs for Instance Details, System Monitoring, Parameter Settings, Account Management, Security Group, **Backup and Restoration** (highlighted with a red box), and Performance Center. Below the navigation bar, there are two buttons: **Clone** (highlighted with a red box) and Auto-Backup Settings. The main content area is divided into two tabs: **Backup List** (selected) and Log Backup List. The Backup List tab displays a table with the following columns: Start Time, End Time, Backup Status, and Backup Size. The table contains seven rows of backup data, all with a status of 'Successful' and a size of 40.58MB. At the bottom left of the table, it indicates '7 in total'.

Start Time	End Time	Backup Status	Backup Size
2022-03-30 00:44:02	2022-03-30 00:44:11	Successful	40.58MB
2022-03-29 00:43:30	2022-03-29 00:43:53	Successful	40.58MB
2022-03-28 00:43:25	2022-03-28 00:43:51	Successful	40.58MB
2022-03-27 00:43:35	2022-03-27 00:43:55	Successful	40.58MB
2022-03-26 00:44:10	2022-03-26 00:44:22	Successful	40.58MB
2022-03-25 00:43:37	2022-03-25 00:43:49	Successful	40.58MB
2022-03-24 00:43:59	2022-03-24 00:44:15	Successful	40.58MB

3. On the displayed purchase page, specify the clone mode and other configurations and click **Buy Now**.

By time point: You can restore an instance to any time point within the past seven days.

By backup set: You can restore data from a backup set to a new instance. The available backup sets depend on the data backup retention period.

Note:

You can log in to the [console](#), select **Database Backup** on the left sidebar, and view backup retention period on the **Backup List** tab.

Clone TencentDB for PostgreSQL Instance

You are restoring the original instance to a new instance (the clone). The new instance will be deployed in the **sa** use the **default** database parameters.

The original instance is accessible during the process of cloning.

Original Instance Info

Instance ID		Instance Name	Unnamed	Project
Network		Region	East China (Shanghai)	AZ
Architecture	Dual-Server High Availability (one-primary-one-secondary)	Instance Specs	1 core 2 GiB, 10 GB storage	Database Version

Restoration

By time point

By backup set

Mode

Restoration Time

2022-03-24 00:44:16



Point

Billing Mode

Pay as You Go

4. After successful purchase, you can view the details of the clone on the instance list page.

FAQs

Will the access to the original instance be affected during the clone process?

The original backup set and log files uploaded to COS are used for restoration, which will not affect the access to the original instance.

Automatic Backup Settings

Last updated : 2024-04-09 11:04:58

This document describes how to modify the automatic backup settings in the TencentDB for PostgreSQL console. TencentDB for PostgreSQL will automatically back up data based on the default or configured backup settings.

Notes

Back up your data during off-peak hours.

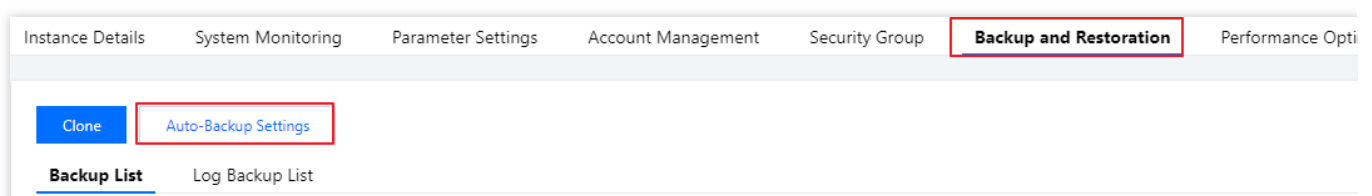
Backup may take a long time if the data volume is large.

Download required backup files to your local file system promptly before they expire, or [utilize cloud functions to transfer PostgreSQL historical backups](#).

The backup mode is physical backup, while logical backup is not supported currently.

Directions

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, select a region and click an instance ID to enter the instance management page.
2. On the instance management page, select the **Backup and Restoration** tab and click **Auto-Backup Settings**.



3. In the **Backup Settings** pop-up window, complete the data backup settings and click **OK**.

Backup Start Time: You can select the default time (i.e., when the resources are idle) or specify a time. Backups are initiated within this time window. If a backup fails to start as configured, it will not be retried, and another backup will be initiated in the next time window.

Data Backup Retention Period: You can specify a time range of 3–7 days after which the backup set will be automatically deleted. Data can only be restored to a specific point in time within the retention period.

Backup Cycle: Any day between Monday and Sunday can be selected.

Backup Settings

Only physical backups are supported.

Data Backup Settings

Backup Start Time

Default time

Custom

02:50 ~ 05:50

🕒

Data Backup Retention Period

–

7

+

day(s)

3-7 days. Backup sets are automatically deleted after expiration. You can only restore an instance to a point in time within the retention period.

Backup Schedule

☒ Mon

☒ Tue

☒ Wed

☒ Thu

☒ Fri

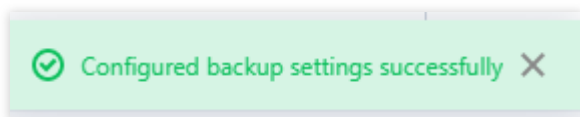
☒ Sat

☒ Sun

OK

Cancel

4. When "Configured backup settings successfully" is displayed in the top-right corner, the automatic backup settings are completed.



Description of Automatic Backup Settings

Parameter	Note
Backup Start Time	<p>The default time is the moment when an automatic backup is initiated. You can customize this parameter. It is recommended that you set the backup start up when the traffic is low. The backup initiation time only indicates when the backup procedure starts, but does not represent the end time.</p> <p>For example, if you decide to start a backup task between 02:00~06:00, the backup task will be triggered at a certain point within the 02:00~06:00 interval, depending on the backup policy at the backend and the conditions of the backup system.</p>
Data Backup Retention Time	Range: 7 to 1,830 days, default: 7 days. Expired backup sets will be automatically deleted.
Backup Schedule	Monday to Sunday (7 days) are selected by default. You can also custom the specific days. However, to ensure your data's security, please set to back up data at least twice a week.

Restoring PostgreSQL Data on CVMs

Last updated : 2024-01-24 11:16:51

When data is lost or corrupted, you can use the [instance clone](#) feature to restore data to a specific time point within the log retention period.

Downloading Backup in Console for Restoration

1. Install PostgreSQL

In the CVM instance where data is to be restored to, install PostgreSQL on the same version as that of the backup data. If PostgreSQL has already been installed, skip this step.

Note:

This document shows you how to install PostgreSQL 10 and restore data in a CentOS 7 CVM instance.

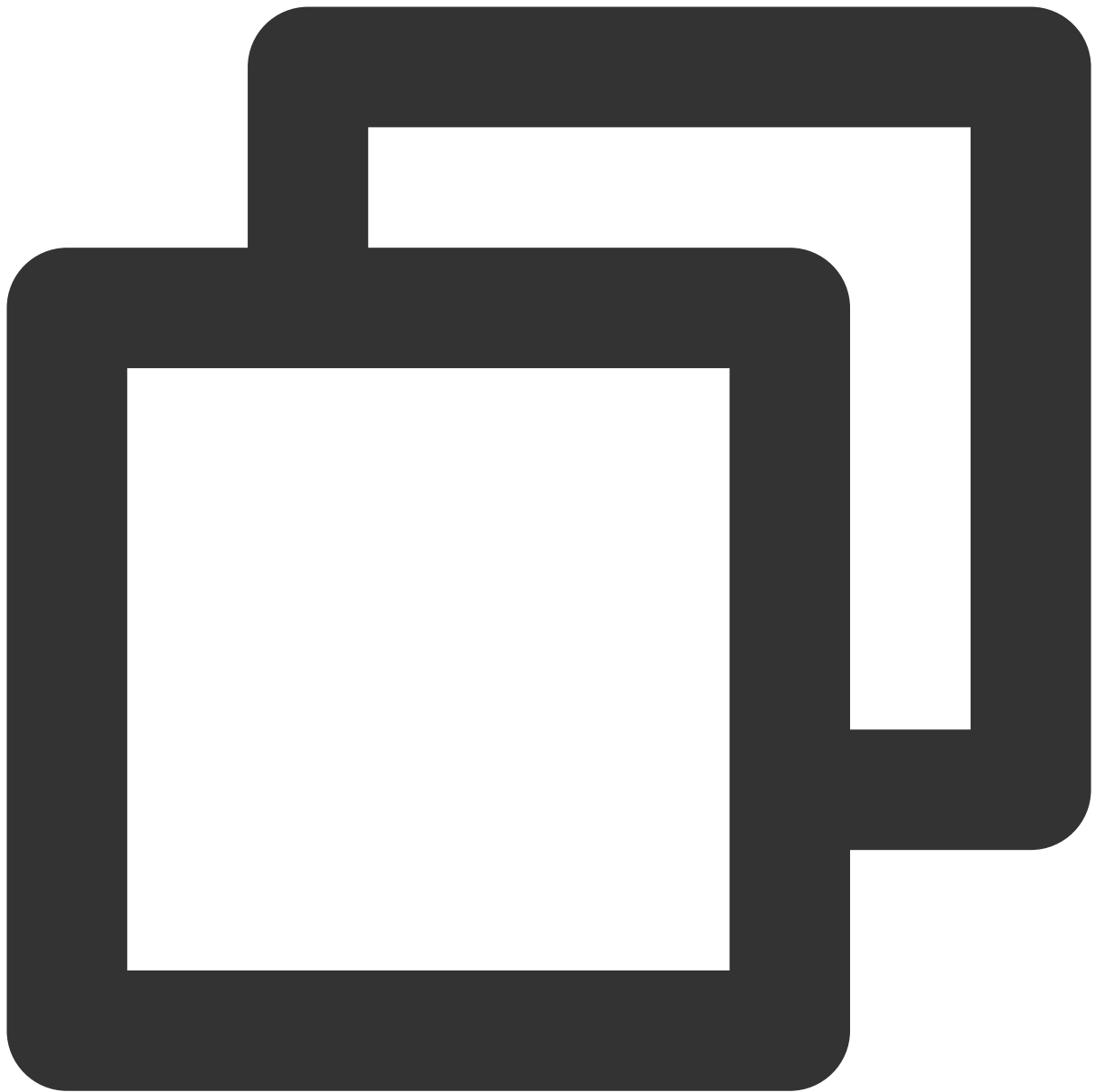
1. Log in to the Linux CVM instance. For more information, see [Customizing Linux CVM Configurations](#).
2. Install PostgreSQL. The yum repository method is used in this document. You can click [here](#) to find the needed yum repository.

Note:

To restore backup data of PostgreSQL v11.8 or 12.4, you need to modify the version number in the installation package name to install PostgreSQL on the same version as that of the backup data. For example, replace

`postgresql10-server` with `postgresql11-server` or `postgresql12-server` .

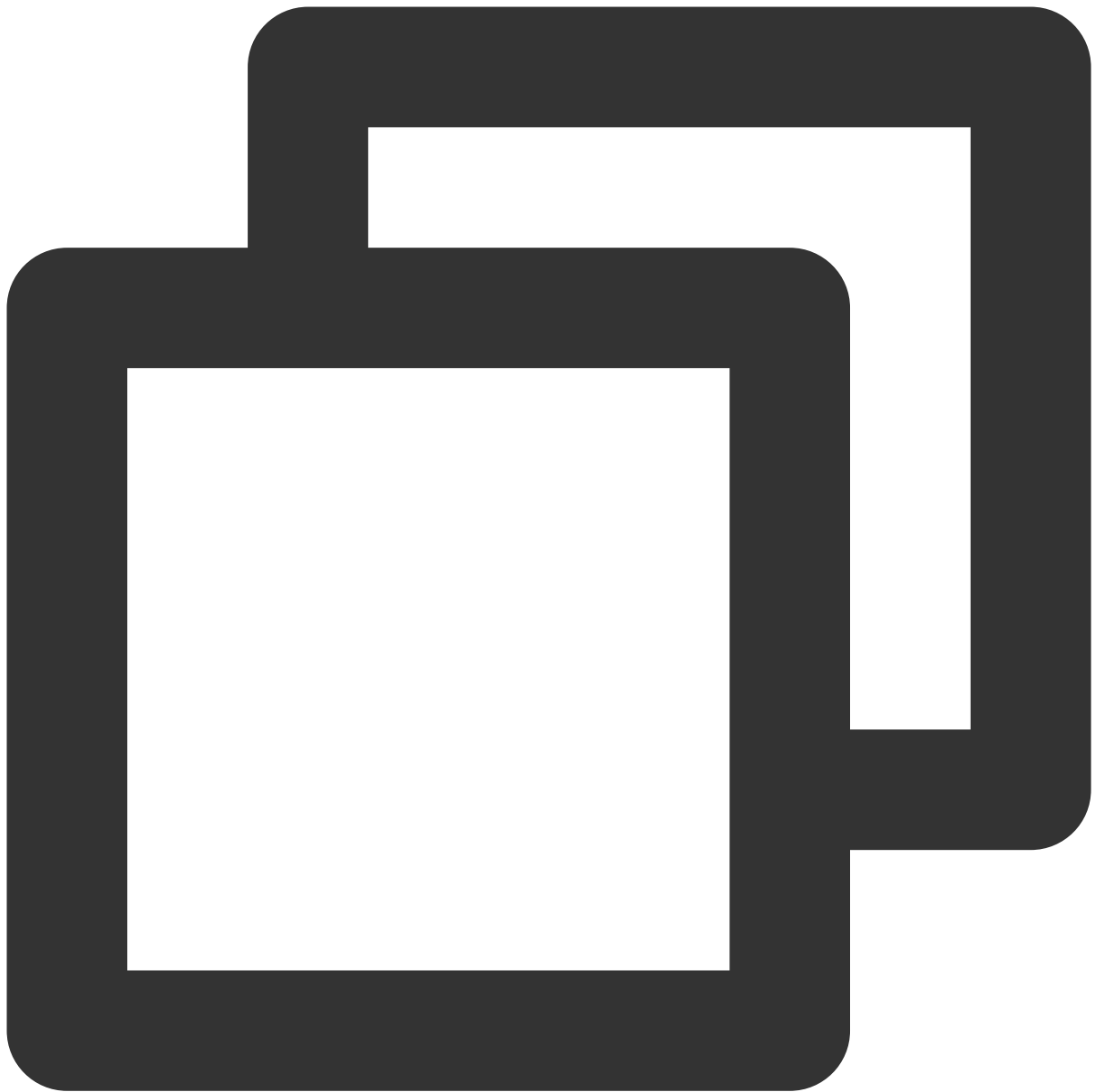
Run the following command to install PostgreSQL 10:



```
yum install https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/pgdg  
yum install postgresql10-server postgresql10-contrib postgresql10 postgresql10.x86_
```

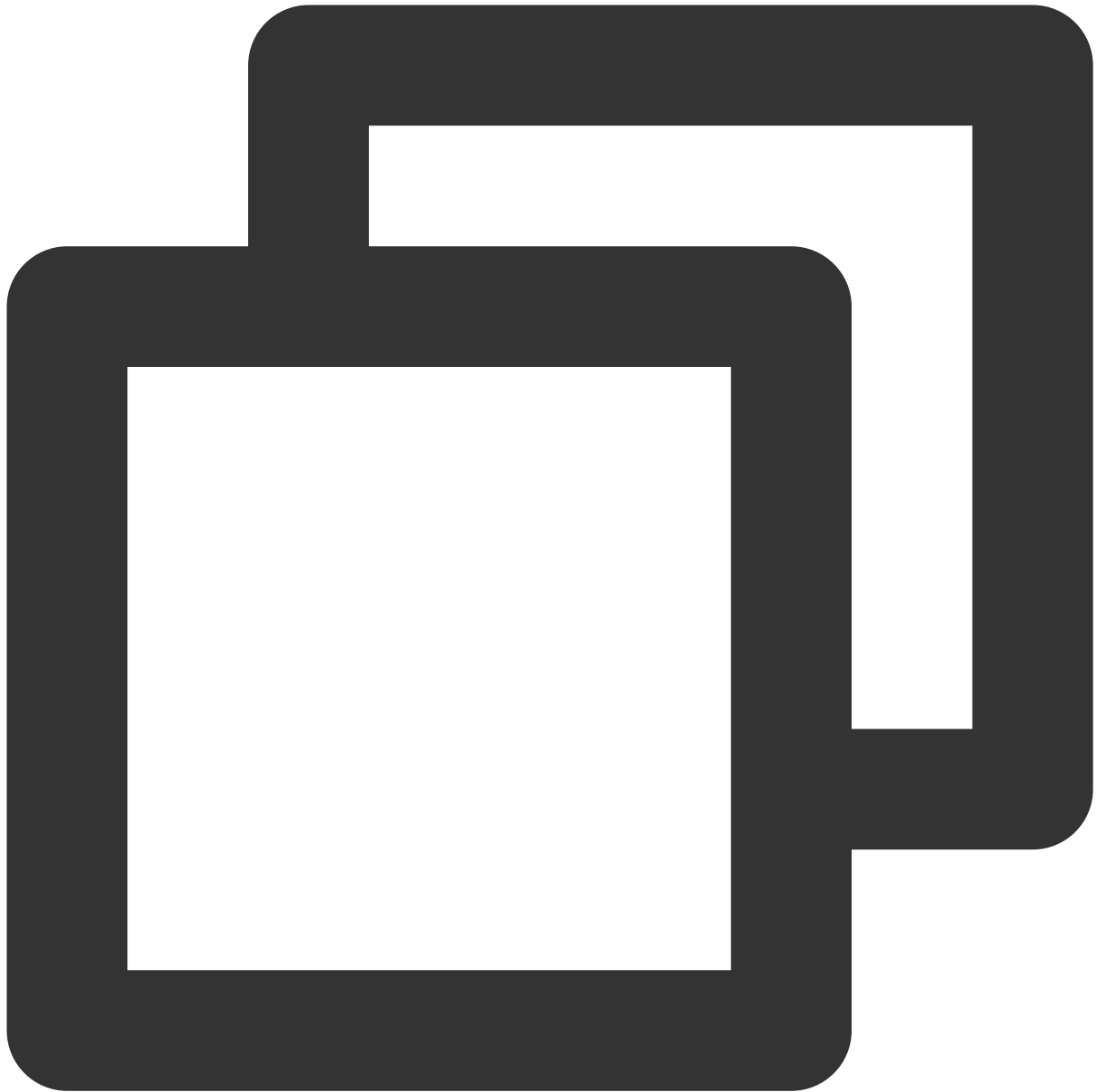
Note:

The command for installing PostgreSQL 9.5 is as follows:



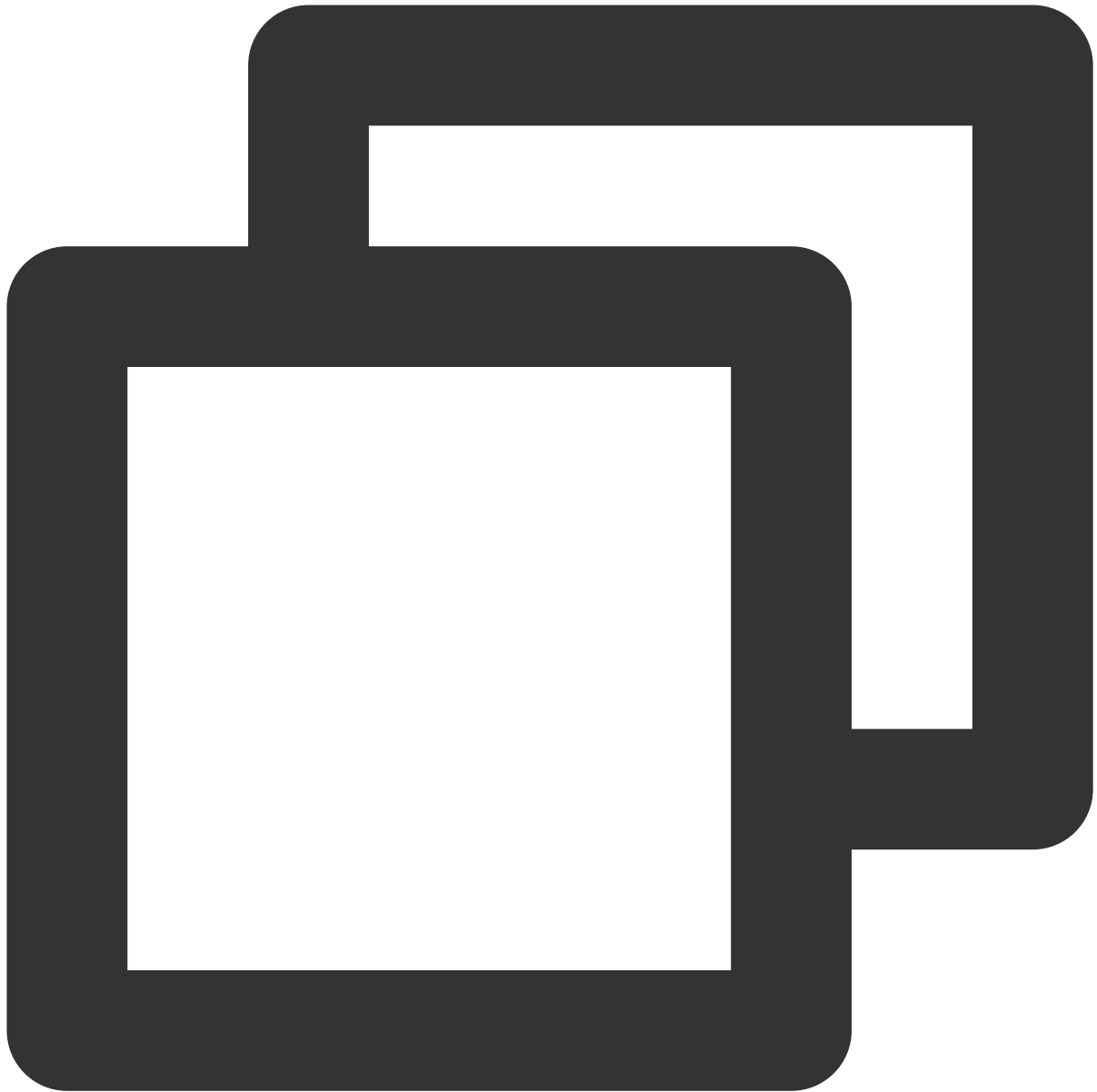
```
yum install https://yum.postgresql.org/9.5/redhat/rhel-7.6-x86_64/pgdg-centos95-9.5
yum install postgresql95-server postgresql95-contrib postgresql95
```

3. Run the following command to check the installation result:



```
rpm -aq | grep postgres
```

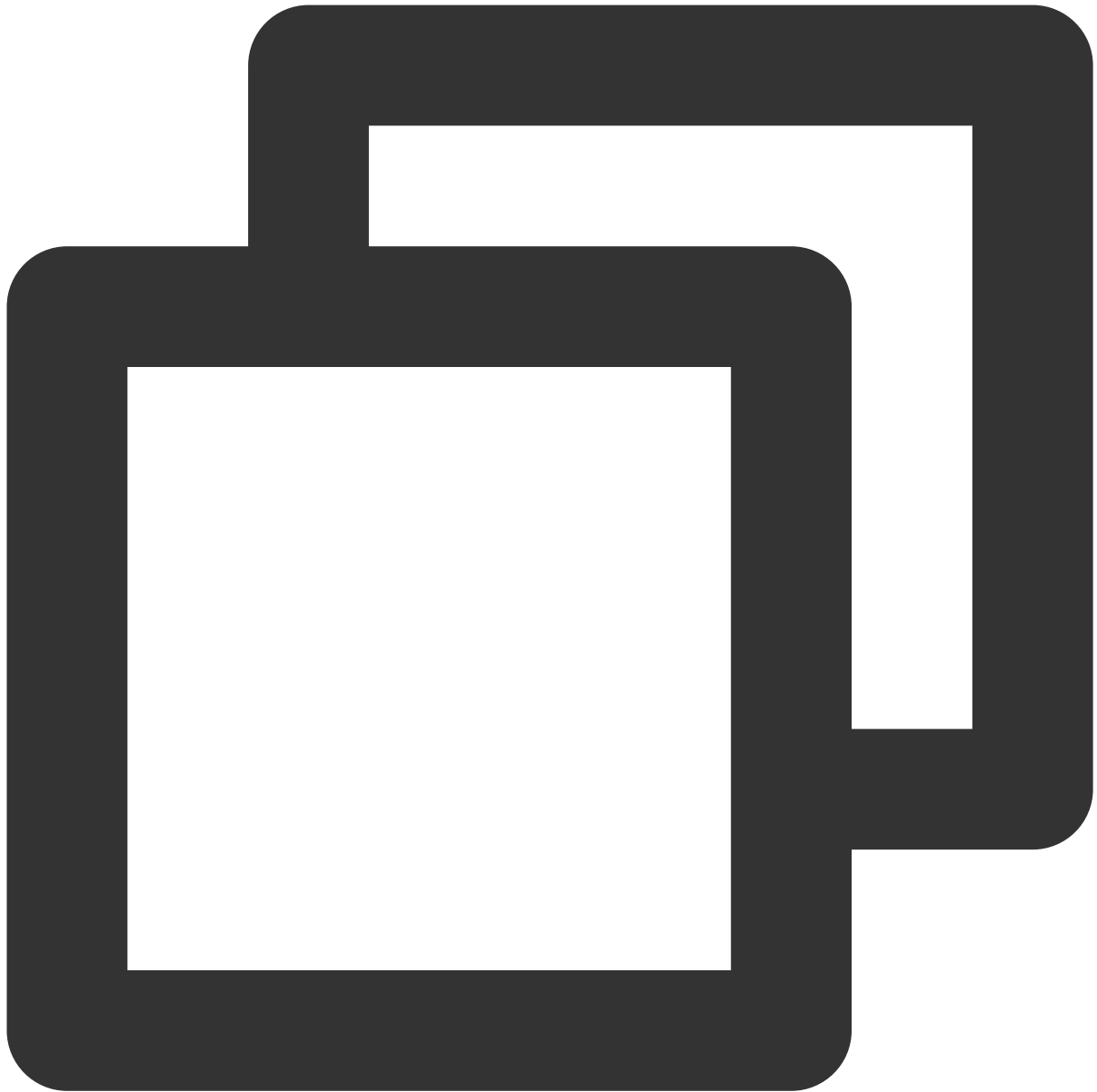
A message similar to the following will be returned:



```
[root@i-87-575-VM vmuser]# rpm -aq| grep postgres
postgresql10-libs-10.11-2PGDG.rhel7.x86_64
postgresql10-server-10.11-2PGDG.rhel7.x86_64
postgresql10-contrib-10.11-2PGDG.rhel7.x86_64
postgresql10-10.11-2PGDG.rhel7.x86_64
```

2. Create a restoration directory as the postgres user

Switch to the postgres user and create a restoration directory in the CVM instance.

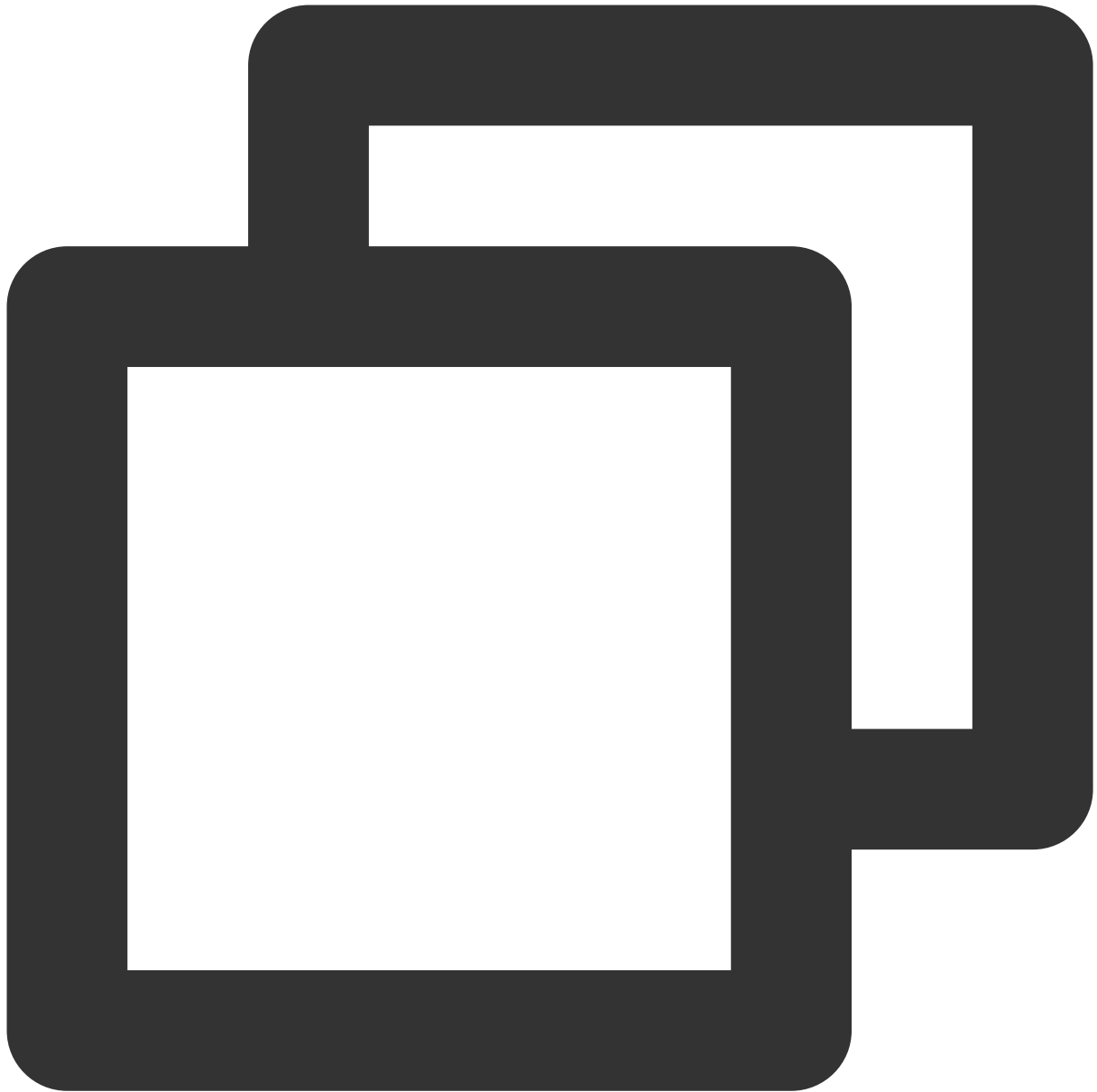


```
mkdir /var/lib/pgsql/10/recovery
```

`recovery` is a sample directory name, which can be modified as needed. In the following examples, the directory names will be the same for one major version. For example, the directory will be `/var/lib/pgsql/10` for PostgreSQL 10.x and `/var/lib/pgsql/9.5` for PostgreSQL 9.5.x.

Note:

The command for PostgreSQL 9.5 is as follows:



```
mkdir /var/lib/pgsql/9.5/recovery
```

3. Download the full backup file

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click **Manage** in the **Operation** column to enter the management page.
2. On the **Backup Management** tab, locate the backup to be restored based on backup time in the backup list and click **Download** in the **Operation** column.
3. Download the backup file from the provided VPC address or public network address.

Note:

If a VPC address is to be used, the TencentDB instance and CVM instance should be in the same VPC, and the backup needs to be downloaded to the `/var/lib/pgsql/10/recovery` directory.

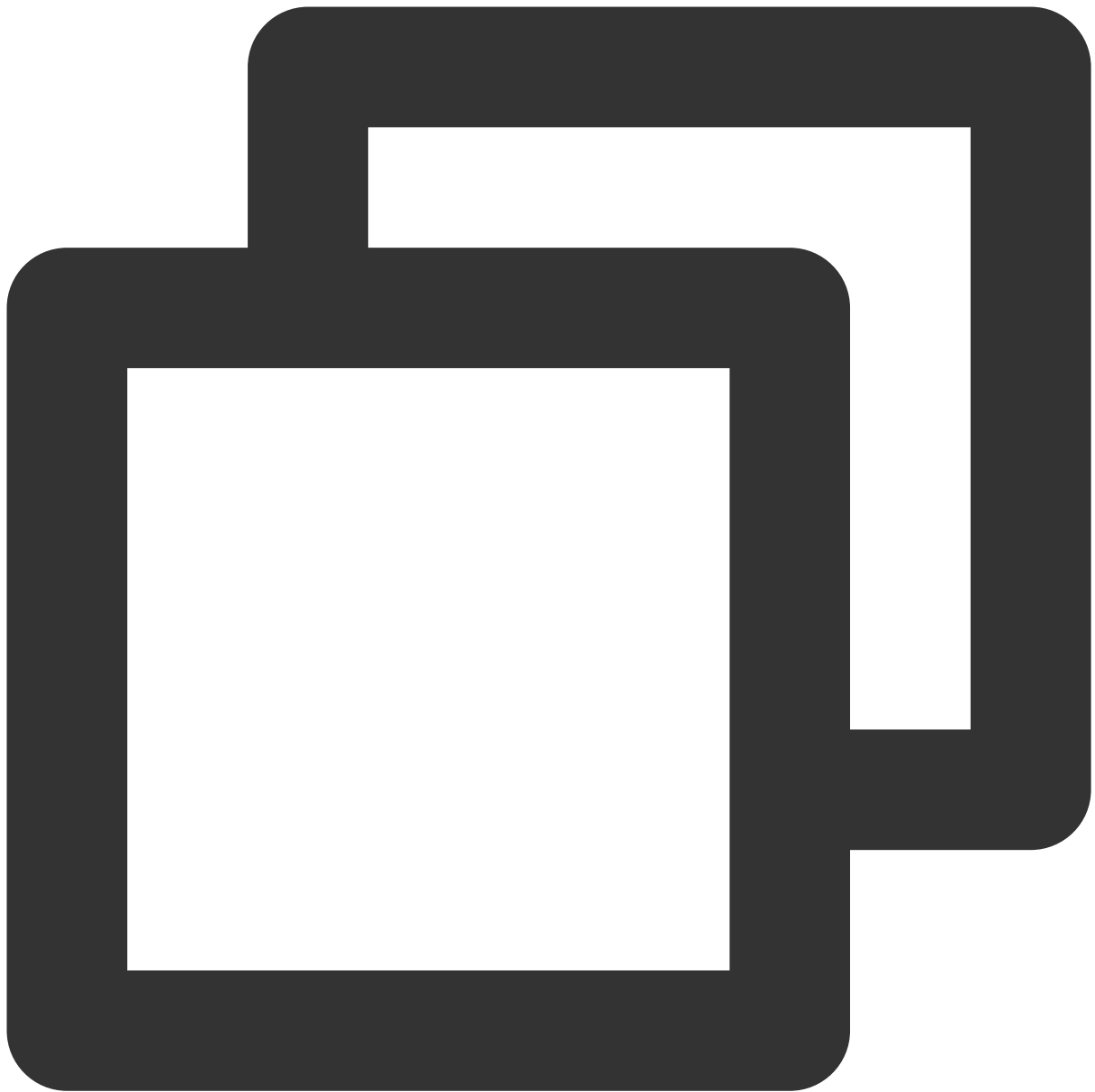
If a public network address is to be used, the downloaded backup file needs to be uploaded to the `/var/lib/pgsql/10/recovery` directory in the CVM instance. For more information, see [Copying Local Files to CVMs](#).

After upload, the following information will be displayed:

```
[root@VM_0_12_centos recovery]# ls -lh
total 6.3M
-rw-r--r-- 1 root root 6.3M Dec 23 11:45 20191221010146.tar.gz
```

4. Decompress the full backup file

Run the following command to decompress the full backup file:



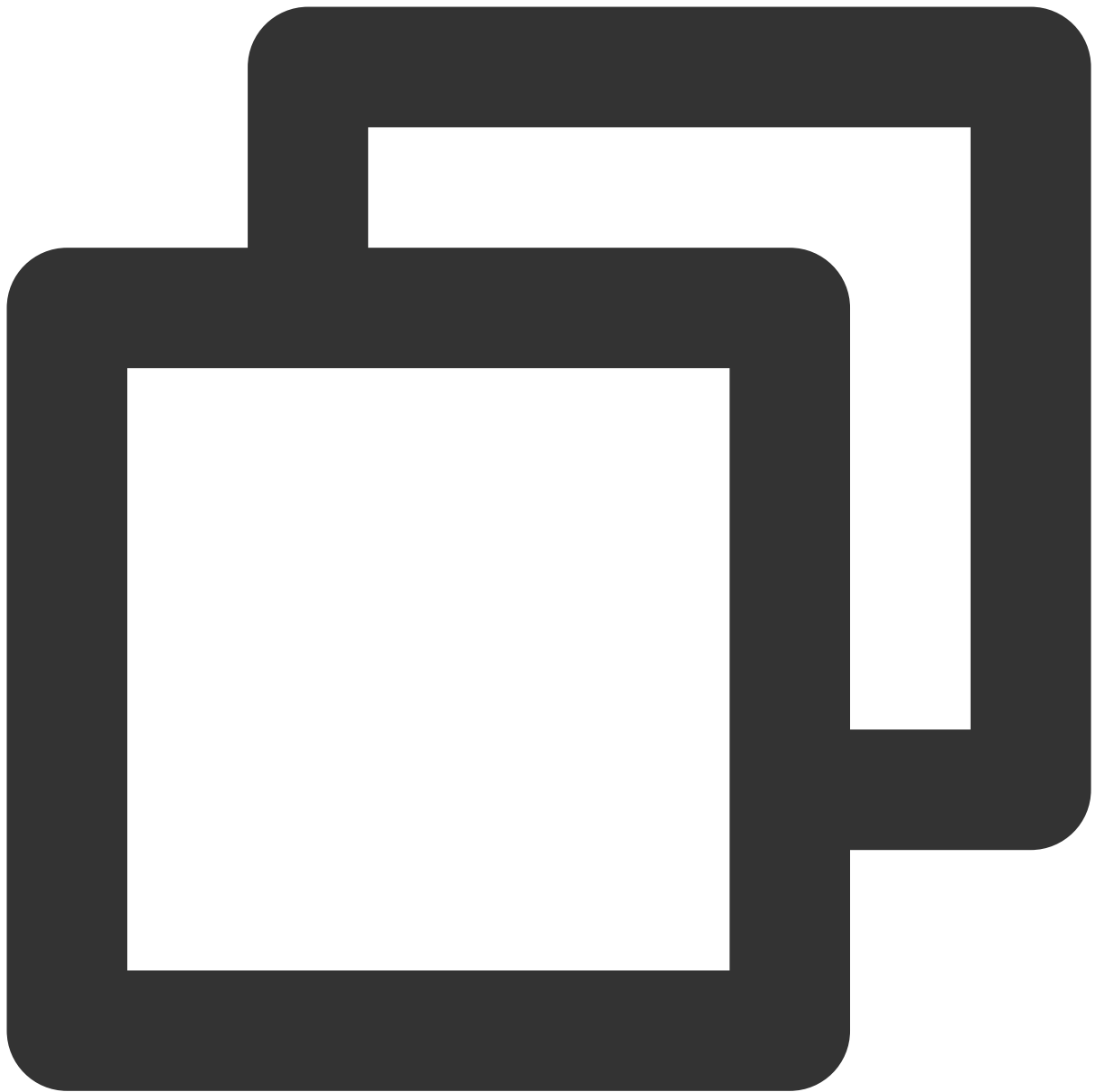
```
cd /var/lib/pgsql/10/recovery  
tar -xf 20191221010146.tar.gz
```

After decompression, the following information will be displayed:

```
[root@VM_0_12_centos recovery]# ls -lh
total 6.4M
-rw-r--r-- 1 root root 6.3M Dec 23 11:45 20191221010146.tar.gz
-rw----- 1 1003 users 215 Dec 21 01:01 backup_label
drwx----- 7 1003 users 4.0K Dec 13 17:37 base
-rw----- 1 1003 users 35 Dec 21 00:00 current_logfiles
drwx----- 2 1003 users 4.0K Dec 5 20:12 global
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_commit_ts
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_dynshmem
-rw----- 1 1003 users 4.8K Dec 2 21:59 pg_hba.conf
-rw----- 1 1003 users 1.6K Dec 2 21:59 pg_ident.conf
drwx----- 4 1003 users 4.0K Dec 21 01:01 pg_logical
drwx----- 4 1003 users 4.0K Dec 2 21:59 pg_multixact
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_notify
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_replslot
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_serial
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_snapshots
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_stat
drwx----- 2 1003 users 4.0K Dec 21 01:01 pg_stat_tmp
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_subtrans
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_tblspc
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_twophase
-rw----- 1 1003 users 3 Dec 2 21:59 PG_VERSION
drwx----- 2 1003 users 4.0K Dec 2 21:59 pg_xact
drwxr-xr-x 2 1003 users 4.0K Dec 3 01:01 pg_xlog
-rw----- 1 1003 users 11 Dec 20 12:02 pg_xlog_archive.tmp
-rw----- 1 1003 users 88 Dec 2 21:59 postgresql.auto.conf
-rw----- 1 1003 users 24K Dec 2 21:59 postgresql.conf
```

5. Remove unnecessary temporary files

Run the following command to remove unnecessary temporary files:

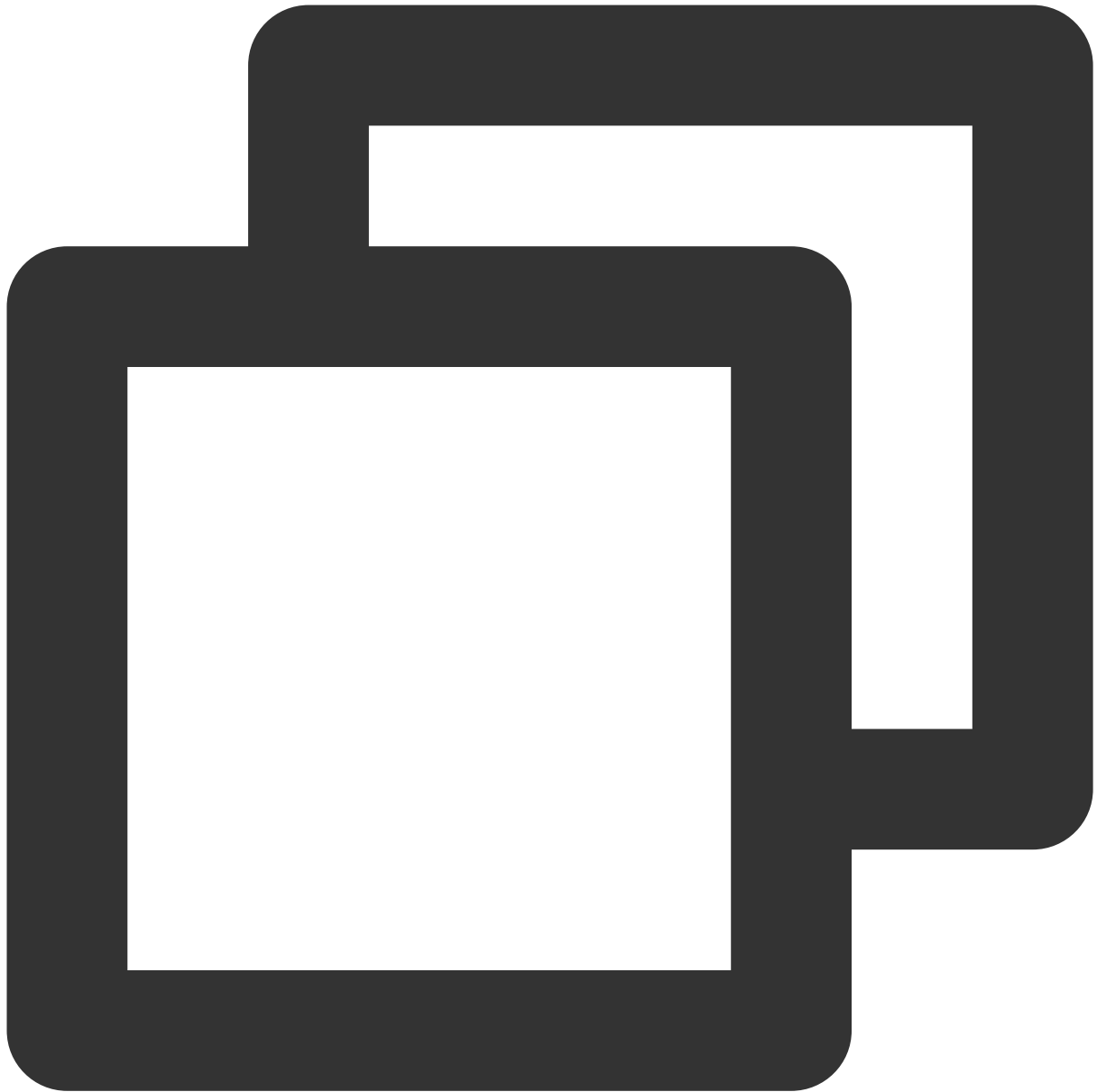


```
rm -rf backup_label
```

6. Modify the configuration file

1. Use # at the beginning of a line to comment out the following options in the `postgresql.conf` configuration file.

Comment all out if there is more than one such option.

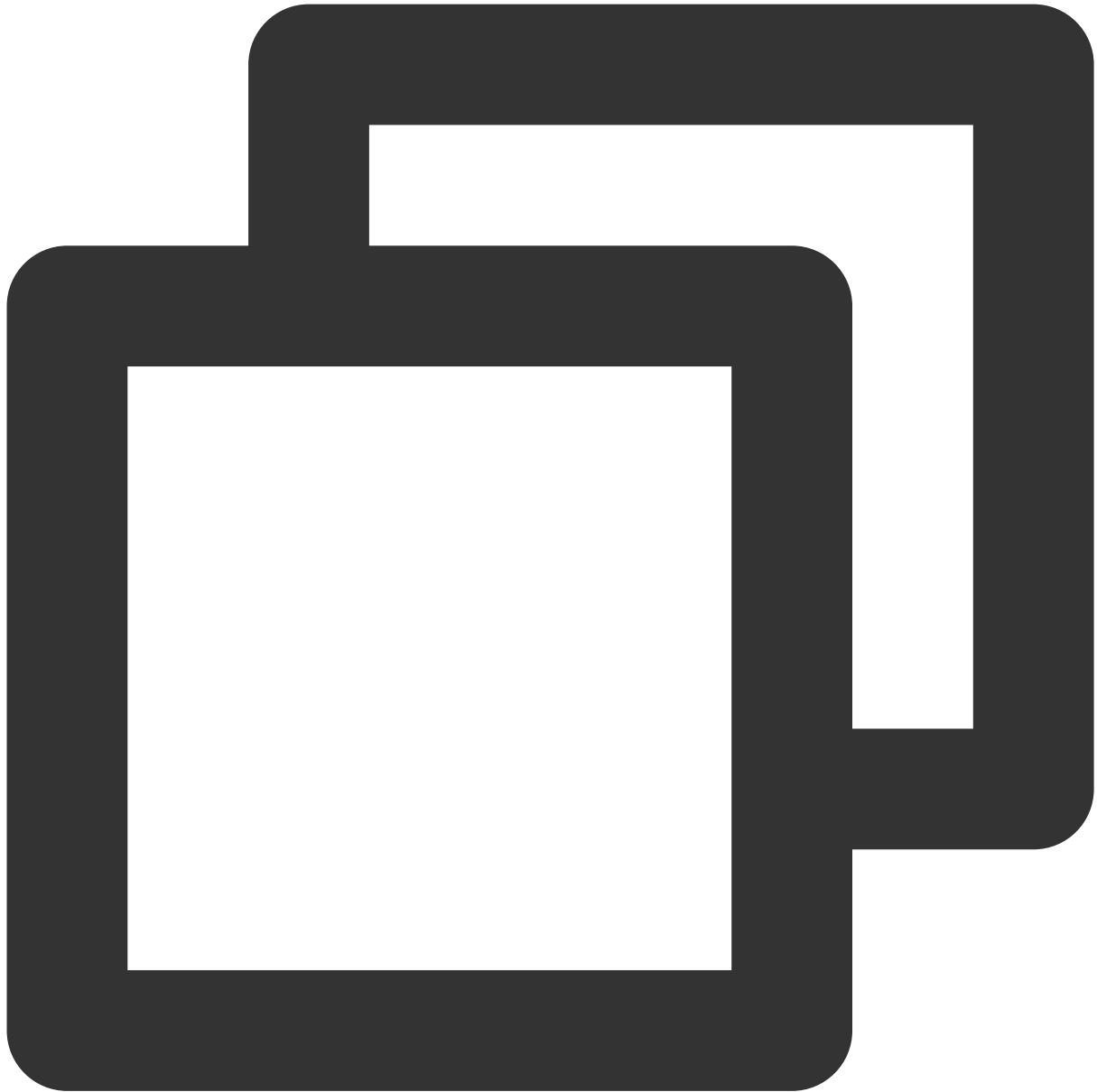


```
shared_preload_libraries
local_preload_libraries
pg_stat_statements.max
pg_stat_statements.track
archive_mode
archive_command
synchronous_commit
synchronous_standby_names
```

Note:

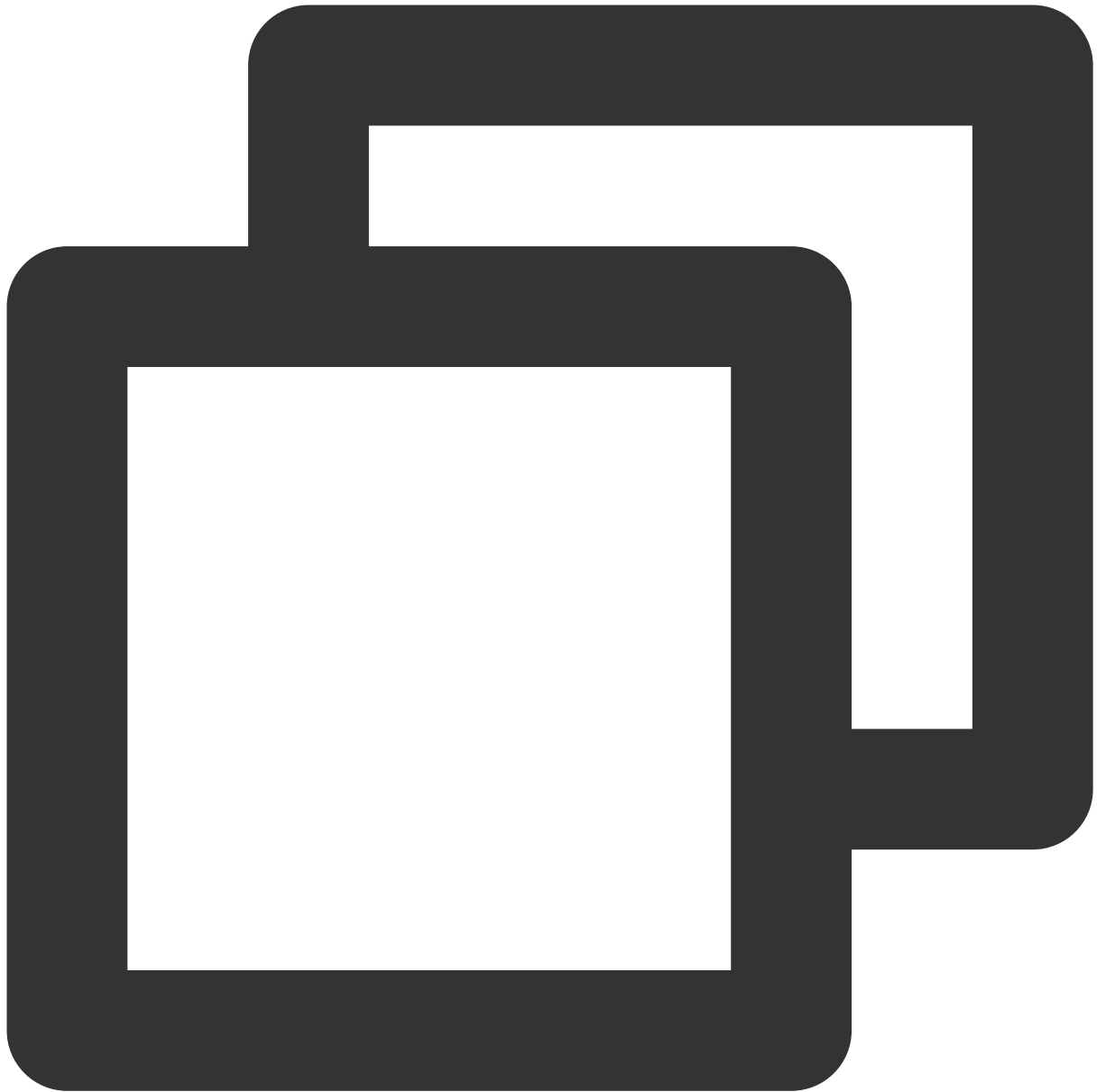
To restore backup data of PostgreSQL v12.4, `include = 'standby.conf'` also needs to be commented out.

2. Modify the `postgresql.conf` configuration file.



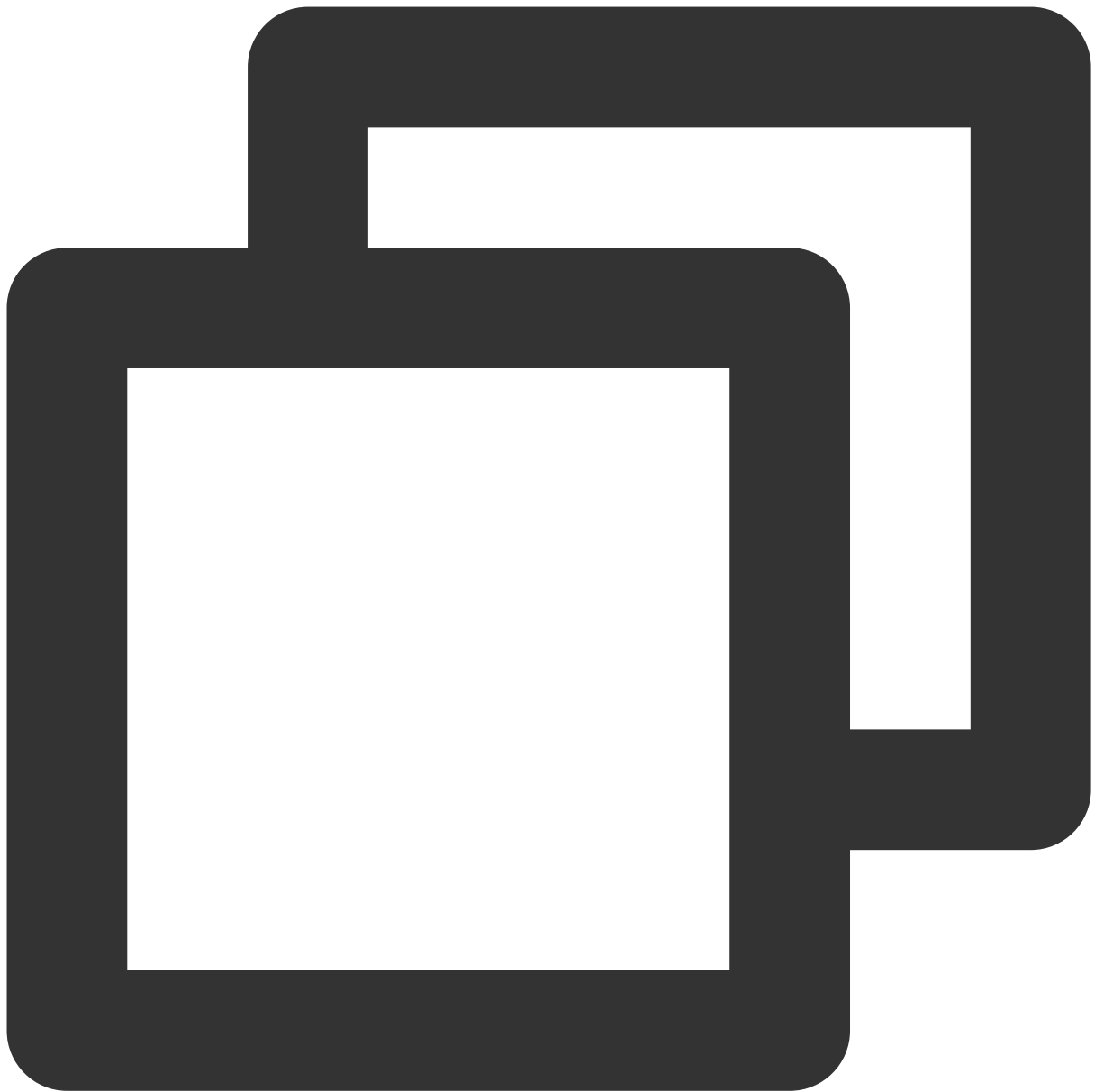
```
port = '5432'      ## Change the value of the `port` parameter to 5432
unix_socket_directories = '/var/run/postgresql/'  ## Change the value of `unix_socket_directories` to '/var/run/postgresql/'
```

2. Append configurations to the `postgresql.conf` configuration file, indicating that the strong sync mode will no longer be used.



```
synchronous_commit = local  
synchronous_standby_names = ''
```

7. Modify folder permissions as the root user



```
chmod 0700 /var/lib/pgsql/10/recovery  
chown postgres:postgres /var/lib/pgsql/10/recovery -R
```

After modification, the following information will be displayed:

```
[root@VM_0_12_centos recovery]# ls -al
total 6528
drwx----- 19 postgres postgres 4096 Dec 23 11:50 .
drwx----- 6 postgres postgres 4096 Dec 23 11:44 ..
-rw-r--r-- 1 postgres postgres 6546935 Dec 23 11:45 20191221010146.tar
-rw----- 1 postgres postgres 215 Dec 21 01:01 backup_label
drwx----- 7 postgres postgres 4096 Dec 13 17:37 base
-rw----- 1 postgres postgres 35 Dec 21 00:00 current_logfiles
drwx----- 2 postgres postgres 4096 Dec 5 20:12 global
drwx----- 2 postgres postgres 4096 Dec 2 21:59 pg_commit_ts
drwx----- 2 postgres postgres 4096 Dec 2 21:59 pg_dynshmem
-rw----- 1 postgres postgres 4858 Dec 2 21:59 pg_hba.conf
```

8. Use the incremental backup file (optional)

If this step is skipped, the content of the database will be that when the full backup was started.

Put the xlog files in the `/var/lib/pgsql/10/recovery/pg_wal` folder. If the downloaded backup does not contain the `pg_wal` directory, modify `pg_xlog` to `pg_wal`, and PostgreSQL will automatically replay the xlog files.

For example, if a full backup is started at 12:00 and all xlog files between 12:00 and 13:00 are put in the `pg_wal` folder, then data can be restored to 13:00.

Note:

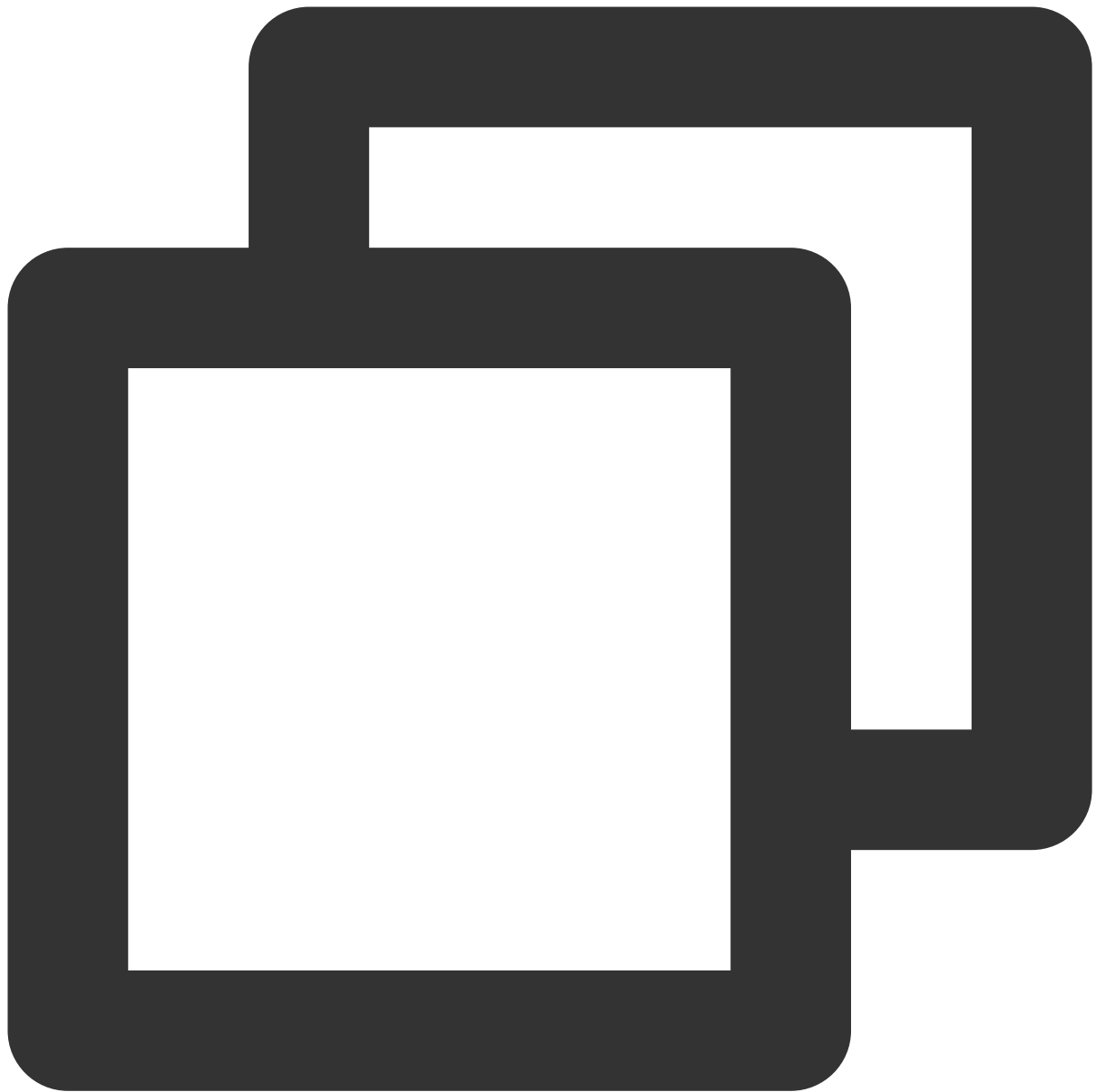
For PostgreSQL 9.x, the folder is `/var/lib/pgsql/9.x/recovery/pg_xlog`.

1. On the **Backup Management** page in the console, get the xlog download address and download the incremental backup file (xlog).

After download, the following information will be displayed:

```
[root@VM_0_12_centos recovery]# mv pg_xlog pg_wal
[root@VM_0_12_centos recovery]# cd pg_wal/
[root@VM_0_12_centos pg_wal]# ls -lh
total 64K
-rw-r--r-- 1 root root 31K Dec 23 11:44 20191221010157_20191221010157.ta
-rw-r--r-- 1 root root 31K Dec 23 11:44 20191221013157_20191221013157.ta
```

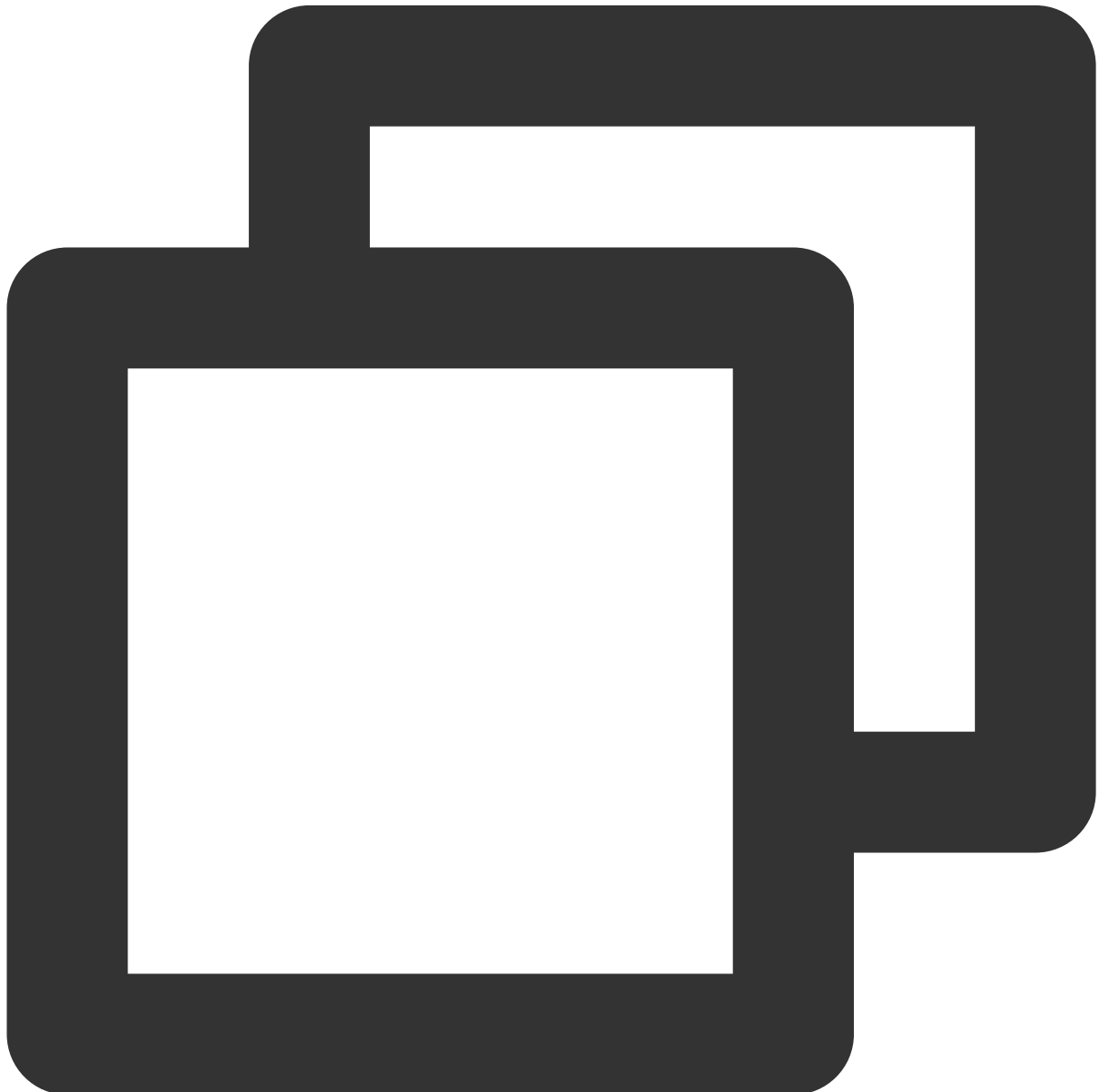
2. Decompress the log to the `pg_wal` folder.



```
tar -xf 20170904010214_20170905010205.tar.gz
```

```
[root@VM_0_12_centos pg_wal]# ll
total 32836
-rw----- 1 postgres postgres 16777216 Dec 21 01:01 00000001000000000000
-rw----- 1 postgres postgres    312 Dec 21 01:01 00000001000000000000
-rw----- 1 postgres postgres 16777216 Dec 21 01:31 00000001000000000000
-rw-r--r-- 1 postgres postgres   31319 Dec 23 11:44 20191221010157_2019
-rw-r--r-- 1 postgres postgres   30966 Dec 23 11:44 20191221013157_2019
```

9. Start PostgreSQL as the postgres user

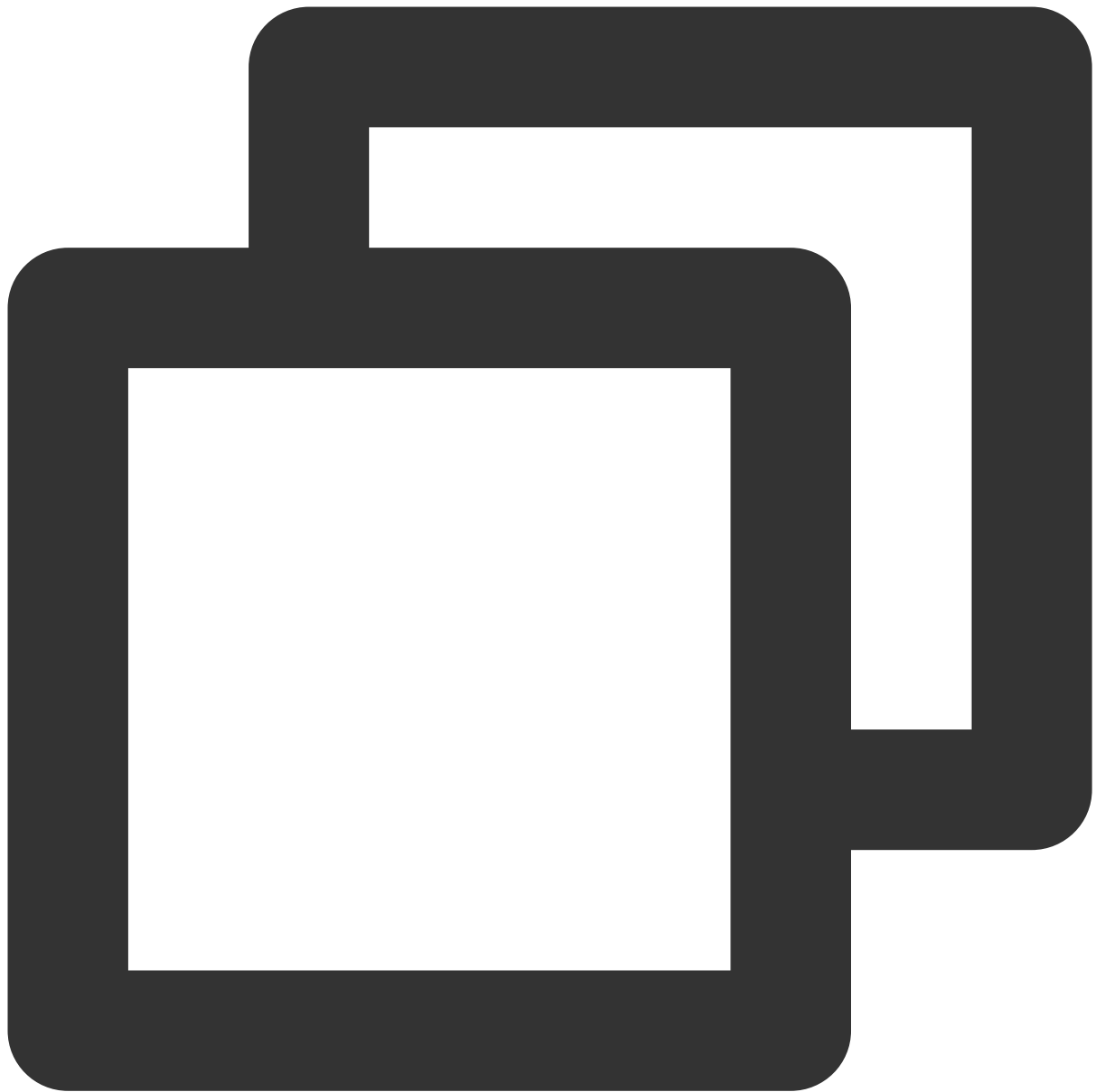


```
/usr/pgsql-10/bin/pg_ctl start -D /var/lib/pgsql/10/recovery
```

```
-bash-4.2$ /usr/pgsql-10/bin/pg_ctl start -D /var/lib/pgsql/10/recovery
waiting for server to start....2019-12-23 11:59:42.654 CST [14061] LOG:
.0.0", port 5432
2019-12-23 11:59:42.654 CST [14061] LOG:  listening on IPv6 address ":::
2019-12-23 11:59:42.664 CST [14061] LOG:  listening on Unix socket "/var
2019-12-23 11:59:42.686 CST [14061] LOG:  listening on Unix socket "/tmp
2019-12-23 11:59:42.840 CST [14061] LOG:  redirecting log output to logg
2019-12-23 11:59:42.840 CST [14061] HINT:  Future log output will appear
done
server started
```

10. Log in to PostgreSQL

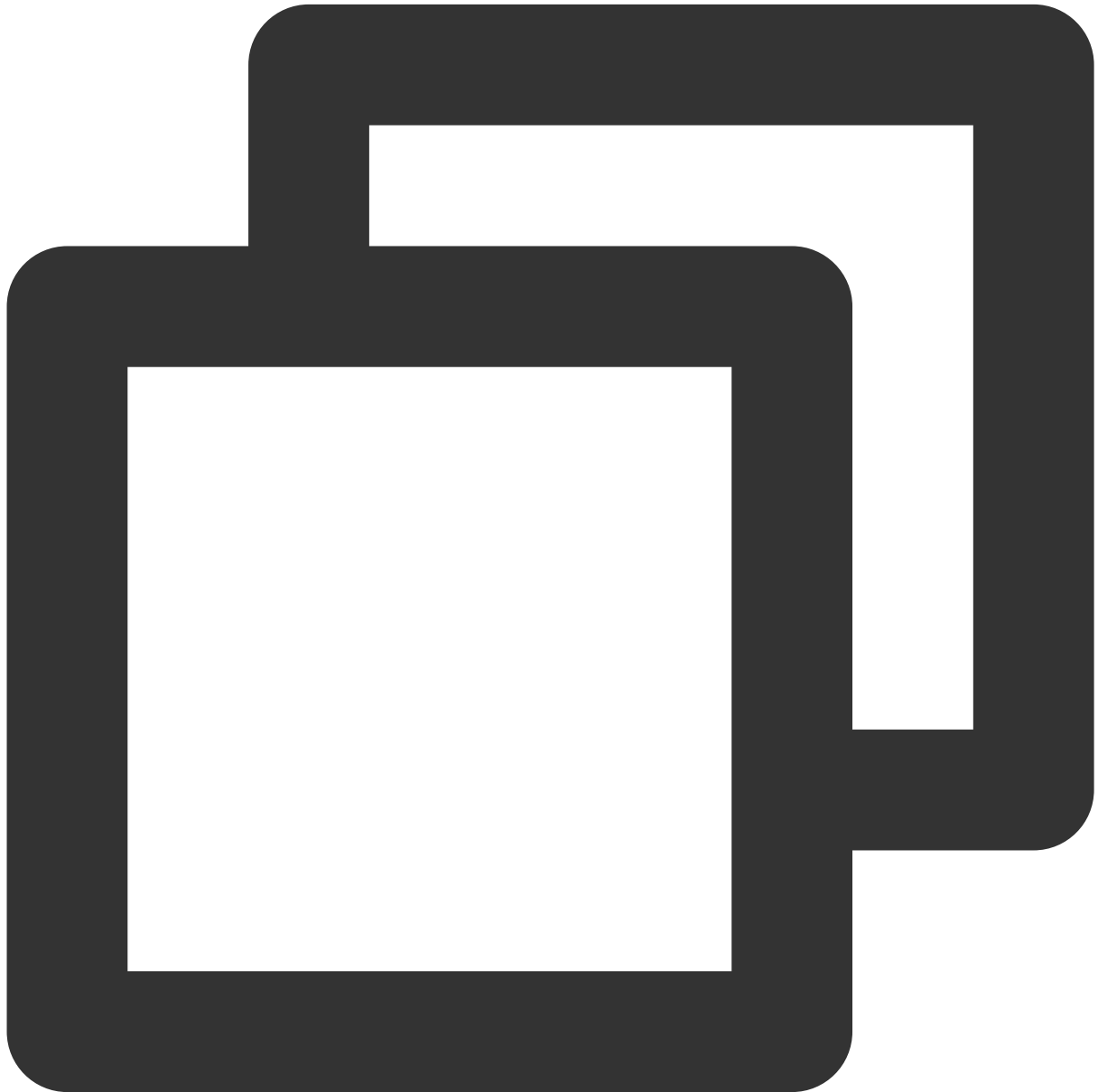
1. Log in to PostgreSQL.



```
export PGDATA=/var/lib/pgsql/10/recovery  
psql
```

```
-bash-4.2$ export PGDATA=/var/lib/pgsql/10/recovery
-bash-4.2$ psql
psql (9.2.24, server 10.11)
WARNING: psql version 9.2, server version 10.0.
         Some psql features might not work.
Type "help" for help.
```

2. Check whether the database is running.



```
/usr/pgsql-10/bin/pg_ctl status -D /var/lib/pgsql/10/recovery
```

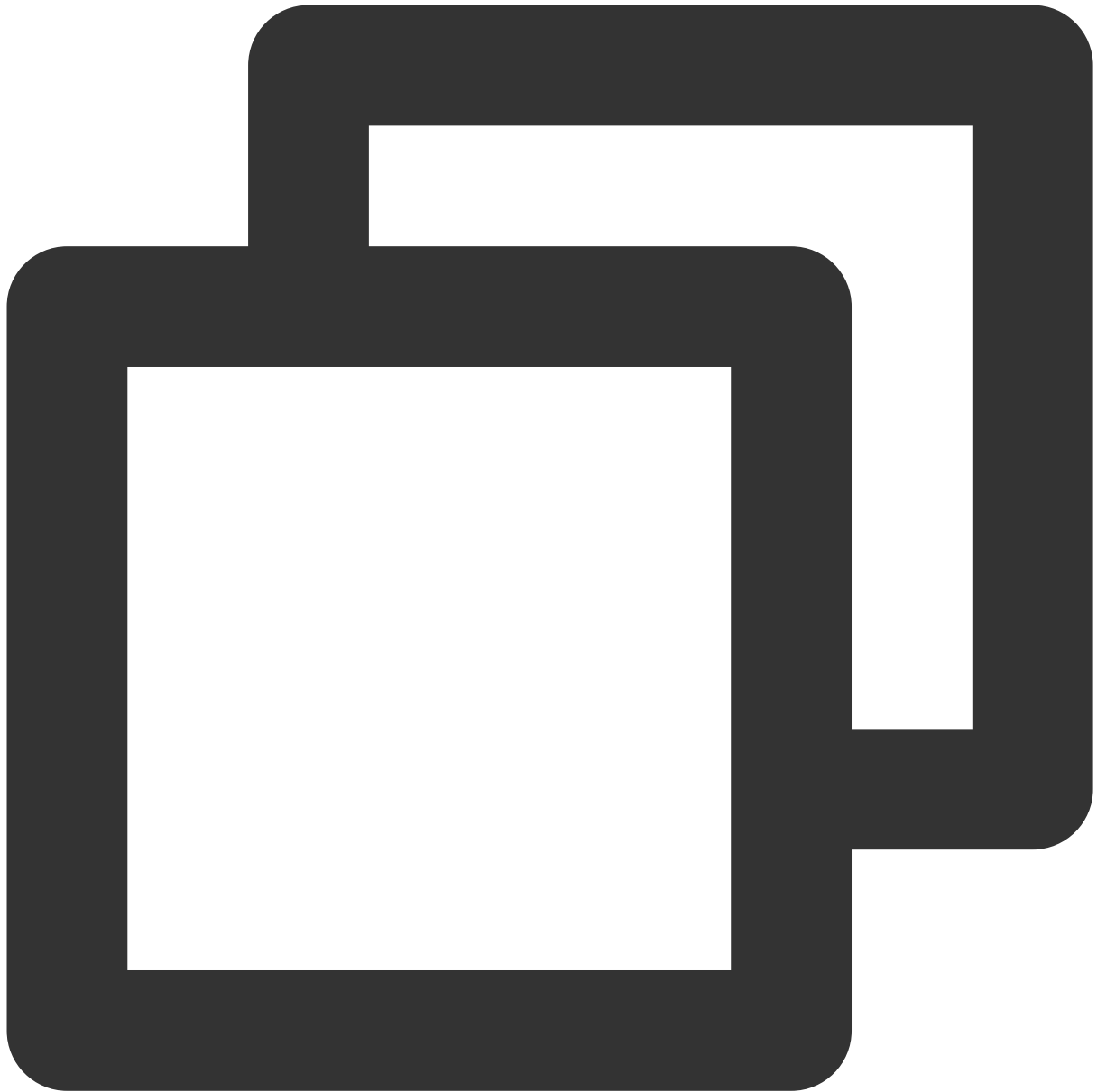
If the prompt is "server is running", the database is running.

```
-bash-4.2$ /usr/pgsql-10/bin/pg_ctl status -D /var/lib/pgsql/10/recovery
pg_ctl: server is running (PID: 14061)
/usr/pgsql-10/bin/postgres "-D" "/var/lib/pgsql/10/recovery"
```

Manually Exporting Data for Restoration

You can also manually export backup data and then restore it in the CVM instance. This scheme is applicable to both Windows and Linux regardless of the file system where physical files reside.

1. Dump the data from the CVM instance as shown below:

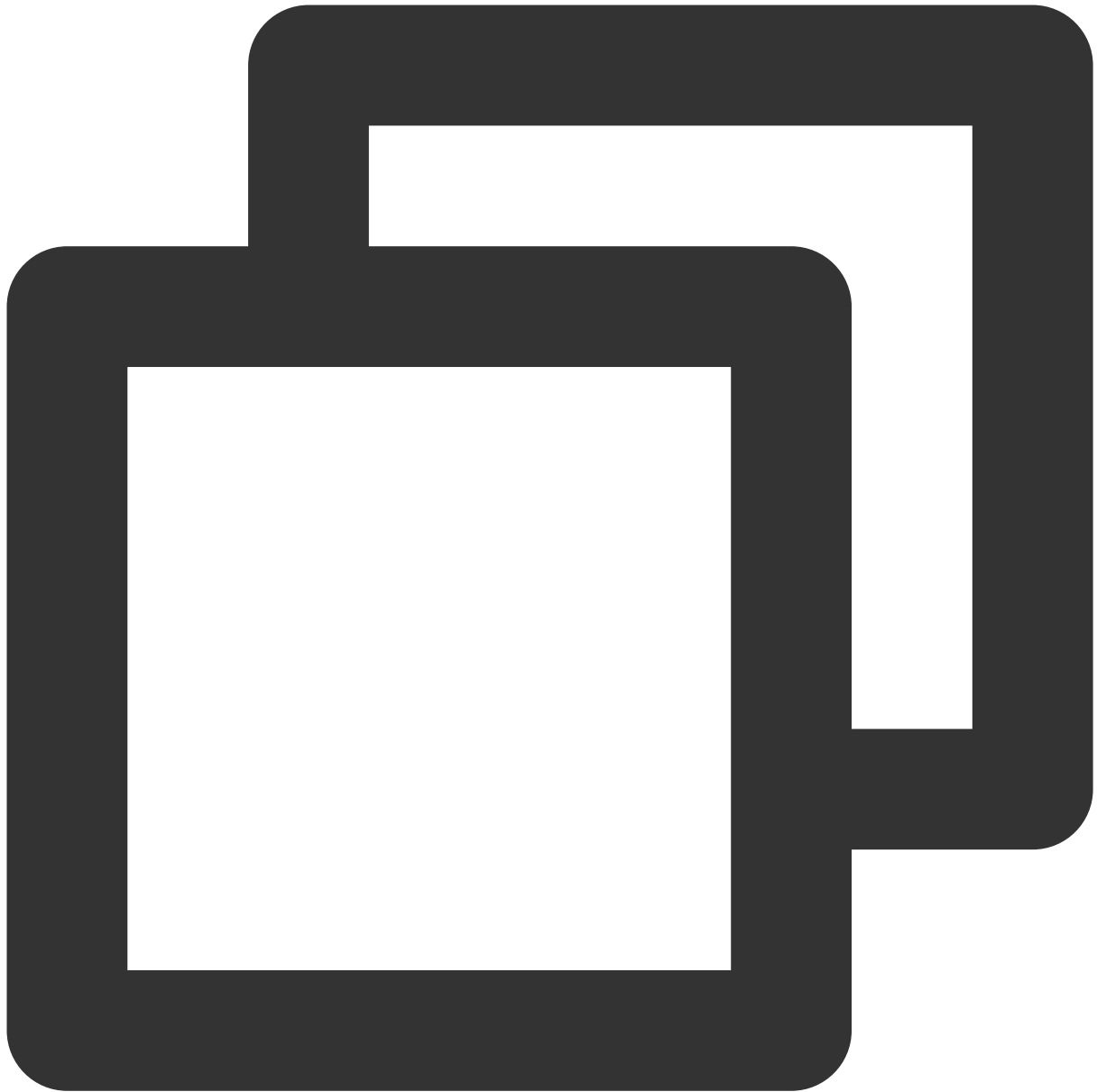


Command format: `pg_dump -h <access IP> -U <accessing user> -f <full path of the backup file>`

Example:

```
/usr/pgsql-10/bin/pg_dump -h 192.168.0.16 -U testroot -f backup.sql -c -C postgres
```

If no file format is specified, a text file will be exported by default as shown below:



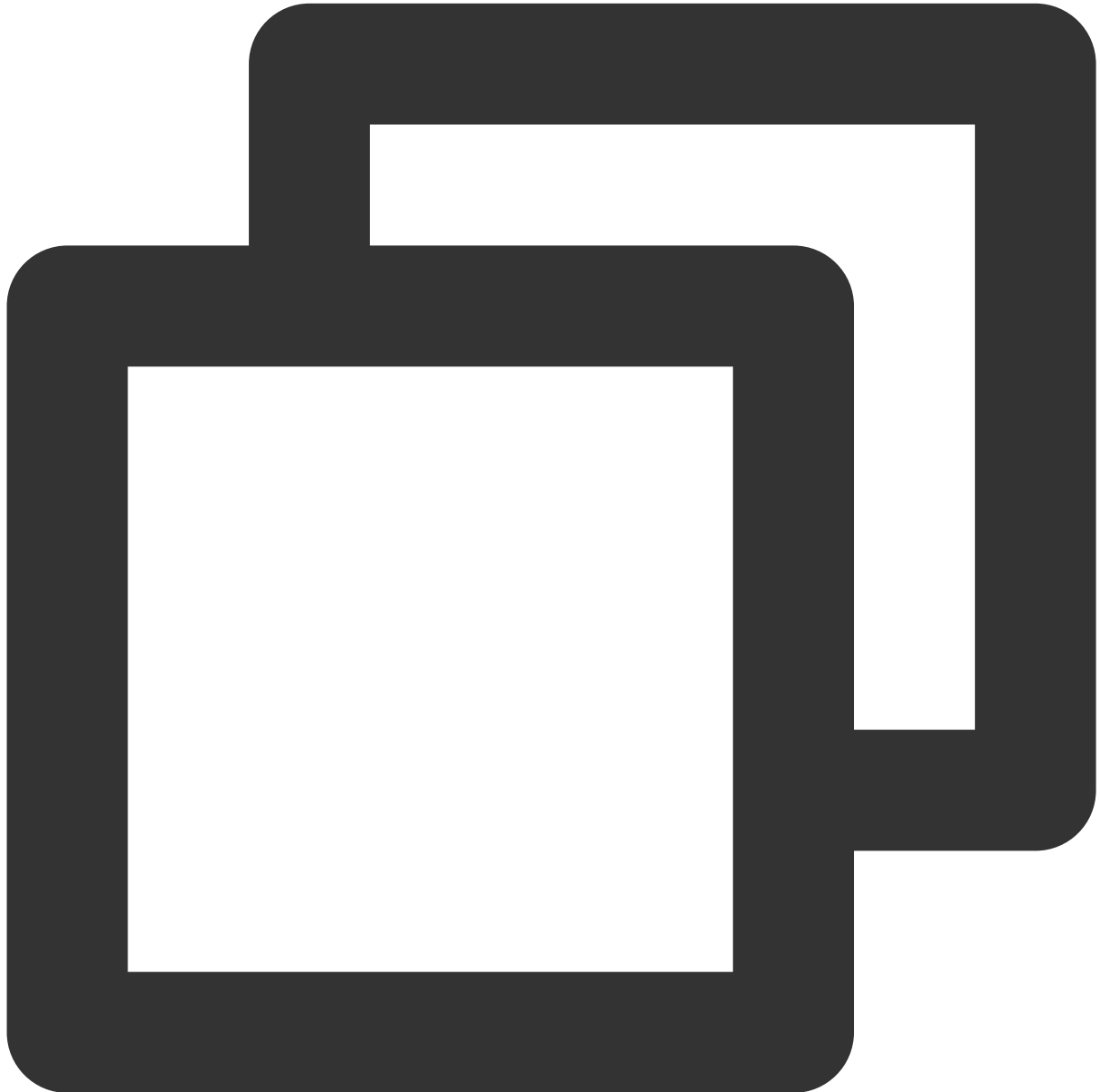
```
-- PostgreSQL database dump
--
-- Dumped from database version 9.5.4
-- Dumped by pg_dump version 9.5.19
SET statement_timeout = 0;
SET lock_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SELECT pg_catalog.set_config('search_path', '', false);
SET check_function_bodies = false;
SET xmloption = content;
```

```
SET client_min_messages = warning;  
SET row_security = off;
```

If there is a massive amount of data, specify the file format as binary file by using `-Fc` .

2. Restore the data in the CVM instance.

For text files, data can be restored by running the following SQL statement:



```
psql -U postgres <./backup.sql
```

Note:

Because there are extensions like `pg_stat_error` , an error may occur, but that does not affect data import.

For binary files, data needs to be restored by using `pg_restore` .

Deleting Backup

Last updated : 2024-01-24 11:16:51

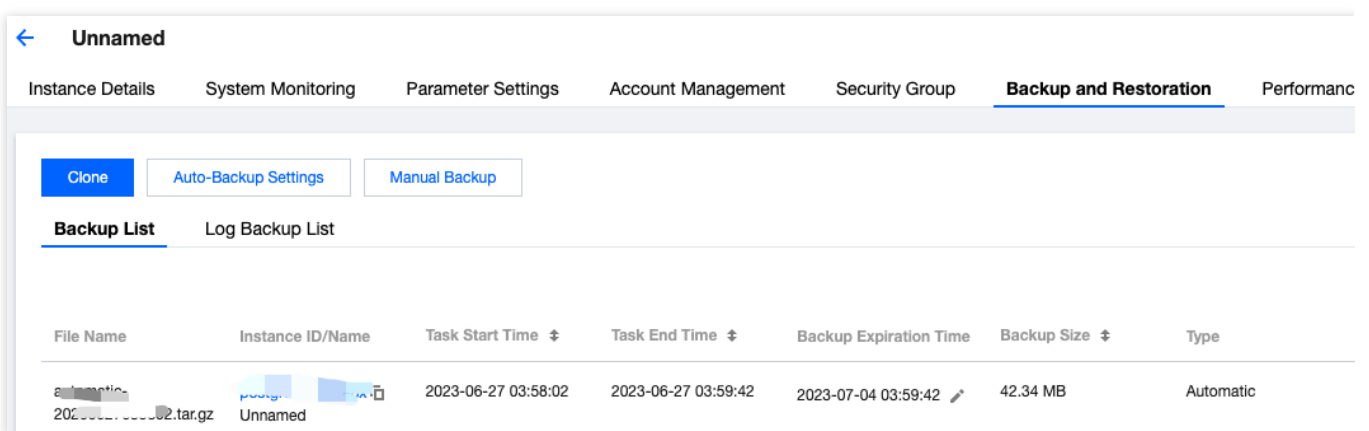
You can delete database backups to reduce backup space costs.

Note:

Manual backups, automatic backups, and log backups can all be deleted. After the backup is deleted, the data cannot be recovered. This is particularly true for automatic backups and log backups, where PITR recovery is impossible due to discontinuous backup data.

Directions

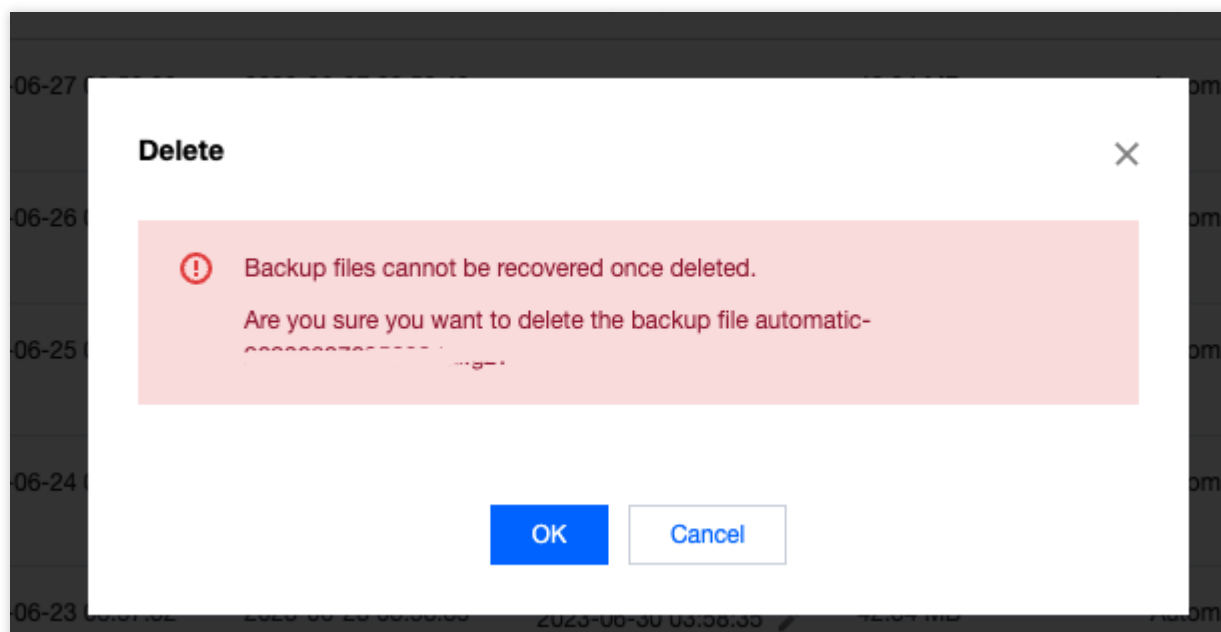
1. Log in to the [TencentDB for PostgreSQL console](#), select a region, and click an instance ID to access the instance management page.
2. On the **Backup and Restoration** tab, locate the backup you want to delete in the backup list, and click **Delete** in the **Operation** column.



The screenshot shows the TencentDB for PostgreSQL console interface. The top navigation bar includes tabs for Instance Details, System Monitoring, Parameter Settings, Account Management, Security Group, Backup and Restoration (selected), and Performance. Below the navigation bar, there are buttons for Clone, Auto-Backup Settings, and Manual Backup. The Backup List tab is active, showing a table with backup information.

File Name	Instance ID/Name	Task Start Time	Task End Time	Backup Expiration Time	Backup Size	Type
20230627035802.tgz	Unnamed	2023-06-27 03:58:02	2023-06-27 03:59:42	2023-07-04 03:59:42	42.34 MB	Automatic

3. In the pop-up dialog box, confirm the backup file to be deleted and click **OK**.



Viewing Backup Space

Last updated : 2024-01-24 11:16:51

Overview

For instances of dual-server high-availability edition, the backup space occupied by TencentDB for PostgreSQL instance backup files is allocated by region. It is equivalent to the total storage capacity used by all database backups in a region, including automatic data backups, manual data backups, and log backups. Increasing manual backup frequency will use more database backup space.

Viewing the backup space

1. Log in to the [TencentDB for PostgreSQL console](#) and select **Database Backup** on the left sidebar.
2. Select a region at the top to view its backup information on the **Overview** tab, including total backup and backup statistics.

Total Backup: This section displays the size and quantity of all backups, data backups, and log backups as well as used free/paid space.

Note:

Green: The total backup space used does not exceed the free tier.

Orange: The total backup space used has exceeded the free tier and incurred fees. For more information, see [Backup Space Billing](#).

Backup Statistics: This section displays the names/IDs/status of instances in the current region, backup space, as well as the size and quantity of data, log, automatic, and manual backups.

3. Select **Backup List** at the top, where the backup list is divided into data backups and log backups. In the instance list, click the instance ID to enter the **Instance Details** page. The backup list supports filtering by time period and fuzzy search by instance ID/name.

Data Backup List

Backups can be sorted by backup time, task start time, task end time, or backup size.

Click **Details** in the **Operation** column to enter the **Backup and Restoration** tab, where you can click **Download** to download backups.

Note:

When a monthly subscribed instance or a pay-as-you-go instance is terminated, the system will **provide** you with an additional "final" full physical backup to avoid nonrecovery events due to misoperation. The "final" backup capacity is

not included in the backup space statistics, hence no fees are charged. You can download it in the data backup list. The "final" physical backups are automatically deleted seven days after the instance is terminated.

Log Backup List

Backups can be sorted by log data start time or end time and backup size.

Click **Details** in the **Operation** column to enter the **Backup and Restoration** tab, where you can click **Download** to download logs and click **Delete** to delete logs.

Note:

A log cannot be recovered once deleted. Proceed with caution.

Free tier

TencentDB for PostgreSQL will start billing for backup space soon. During the beta test of backup billing, the free backup space is the sum of the storage space of all the primary instances in the corresponding region multiplied by 700%. After the backup billing officially starts, the free backup space will be the sum of the storage space of all the primary instances in the corresponding region multiplied by 100%. For more information, see [Backup Space Billing](#).

FAQs

How will I be charged for backup space beyond the free tier? How can I reduce the costs of backup space?

For more information, see [Backup Space Billing](#).

Setting Backup Download Rules

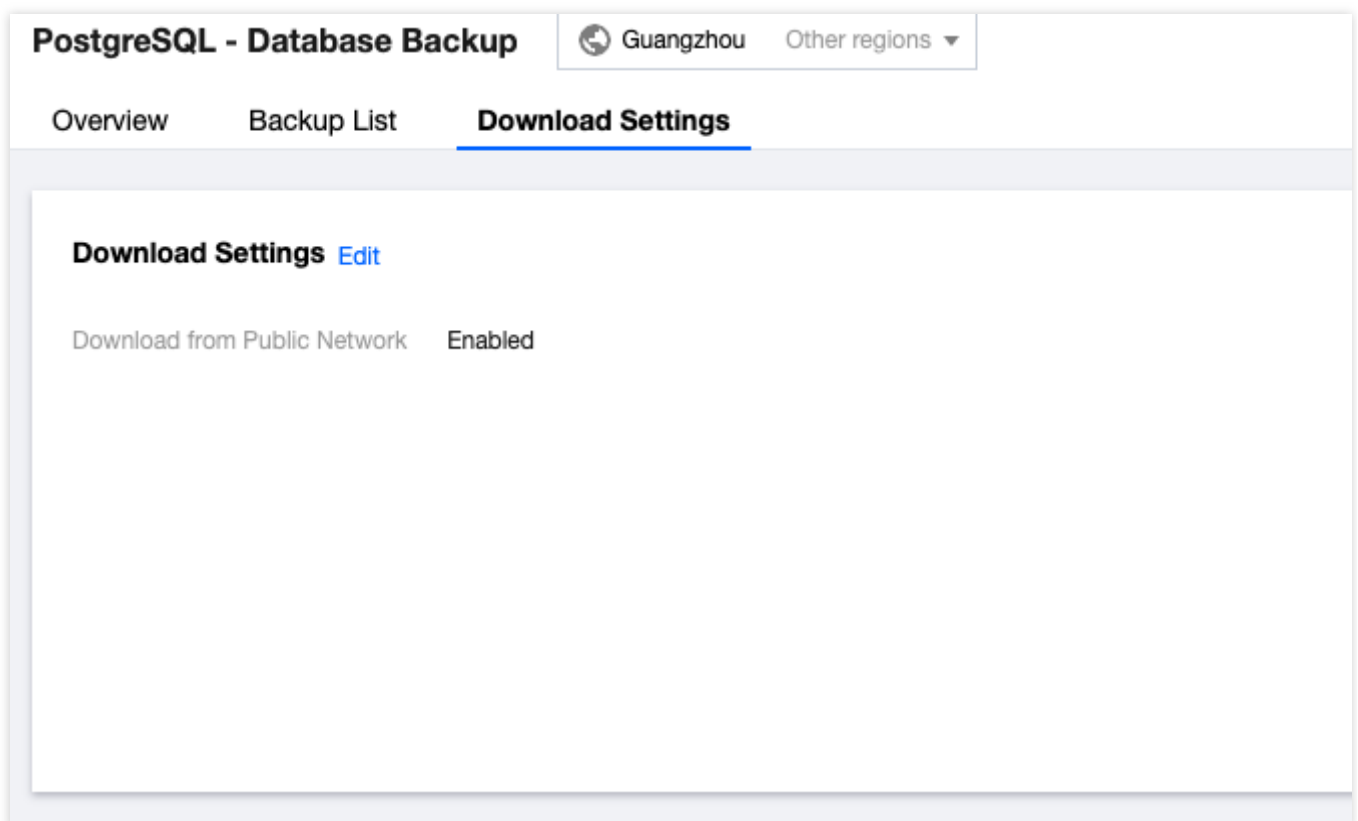
Last updated : 2024-01-24 11:16:51

By default, you can download backup files of TencentDB for PostgreSQL instances over public or private network. To limit the download, you can adjust backup download settings. This document describes how to do so in the console.

1. Log in to the [TencentDB for PostgreSQL console](#), select **Database Backup** on the left sidebar, and select a region at the top.
2. On the **Download Settings** tab, view the backup download settings and click **Edit** to modify them.

Note:

Download over public network is enabled by default and when it is enabled, download over private network is also allowed.



3. On the displayed page, set the download rules and click **OK**.

Download over public network:

Enabled: You cannot set any download rule.

Disabled: You can set the download rules by allowing or blocking specific IPs and VPCs.

Set the download rules:

If you don't specify any value, the condition won't take effect. If no IP and VPC requirements are set, there will be no limit on download over private network.

PostgreSQL - Database Backup

Guangzhou

Other regions ▼

Overview

Backup List

Download Settings**Download Settings**

Download from Public Network

Enable

Disable

Download Conditions

Field	Operator	Value ⓘ
IP	Include ▼	<input type="text" value="Enter IPs and separate them by comma"/>
VPC	Include ▼	<input type="text" value="Please select"/>

OK

Cancel

PostgreSQL - Database Backup

Guangzhou

Other regions ▼

Overview

Backup List

Download Settings**Download Settings** [Edit](#)

Download from Public Network

Disabled

Download Conditions

Field	Operator	Value
IP	Include	<input type="text" value="10.10.10.10"/>
VPC	Include	<input type="text" value="vpc-12345678"/>

Use Serverless Cloud Function to Transfer Historical Backups of PostgreSQL

Last updated : 2024-03-20 15:43:50

Currently, TencentDB for PostgreSQL supports retaining backup data for 3 to 7 days. You can also transfer backup files of TencentDB for PostgreSQL to Cloud Object Storage for long-term preservation through Serverless Cloud Function.

Note:

Please ensure data pull success through strict configuration and monitoring in the [Cloud Object Storage Console](#). Otherwise, if the backup data in the database is automatically deleted after expiration, it may result in backup data loss.

Directions

1. Sign in to the [Cloud Object Storage](#) console. Go to **Application Integration > Data Backup > PostgreSQL backup**, as shown in the figure below:

The screenshot displays the Tencent Cloud console interface, specifically the 'Data Backup' section under 'Cloud Object Storage'. The left sidebar contains navigation options: Overview, Bucket List, Bookmark path, Statistic, Package Manage, Storage+, Data Processing Workflow, Batch Operation, Storage Security, Sensitive Content Moderation, Ecological Service, Application Integration, Data Backup (highlighted), Extended Function, Data Export, Data Migration, and Tools. The main content area is titled 'Data Backup' and features a grid of backup services, each with a 'Configure Backup Rule' button. A notice at the top states: 'Each application in Application Integration is built through SCF, SCF free quota and billing method have been officially adjusted on June 1, 2022, please see S...'. The services listed are: MySQL Backup, TDSQL-C for MySQL ba..., SQL Server Backup, Redis Backup, CKafka Message Backup, TDMQ Message Backup, CLS Log Backup, and Bill Delivery. Each service card includes a brief description of the backup functionality and a link to 'View Documentation'.

2. Click **Configure Backup Rules** to enter the PostgreSQL Backup configuration page. Click **Add Function** to configure the Serverless Cloud Function to pull the data backup and log backup files of PostgreSQL instances within the specified period.

← PostgreSQL backup Nanjing

Each application in Application Integration is built through SCF, SCF free quota and billing method have been officially adjusted on June 1, 2022, please see [SCF free quota description](#)

Note: The size of a single backup file cannot exceed 2TB. Exceeding 2TB may cause backup failure.

[Add Function](#)

Function Name

Create PostgreSQL backup function

Function Name -postgres-
Beginning with a letter, support a-z, A-Z, 0-9, -, _, up to 10 characters

Associated Bucket Please select
You don't have buckets, please create in bucket list page

Trigger Period be reset daily be reset daily00:00

Cron Expression 0 0 0 * * *
Cron follows China Standard Time (UTC+08:00). For detailed information, please see [Cron Expression](#)

Database Instance Please select
There is no available PostgreSQL instance in the current region

Delivery Path ⓘ ☒ Root directory ☐ Specified prefix

SCF Authorization ☐ Authorize SCF Service
SCF needs to be granted a third-party role to access cloud resources

3. After configuration, you can view the list of configured backup functions on the PostgreSQL Backup Function page, as shown in the figure below:

← PostgreSQL backup Guangzhou (1)

Each application in Application Integration is built through SCF, SCF free quota and billing method have been officially adjusted on June 1, 2022, please see [SCF free quota description](#)

Note: The size of a single backup file cannot exceed 2TB. Exceeding 2TB may cause backup failure.

[Add Function](#)

Function Name	Associated Bucket	Trigger Period	Database Instance	Delivery Path
bck-postgres-241e9618303e45536a516...	amy1-1312520572	be reset daily01:00 Cron expression: 0 0 1 * * *	Region: Guangzhou Instance: postgres-b1xt1y1b amy	Root directory

4. Once the function is created, you can click **More** > **Trigger** > **Confirm** to trigger the function execution. This function can pull the backup data of TencentDB for PostgreSQL, accessing data for up to the last 7 days.

← PostgreSQL backup Guangzhou (1)

Each application in Application Integration is built through SCF, SCF free quota and billing method have been officially adjusted on June 1, 2022, please see SCF free

Note: The size of a single backup file cannot exceed 2TB. Exceeding 2TB may cause backup failure.

Add Function

Function Name	Associated Bucket	Trigger Period	Database Instance
bck-postgres-241e9618303e45536a516...	amy1-1312520572	be reset daily01:00 Cron expression: 0 0 1 * * *	Region: Guangzhou Instance: postgres-b1xt1y1b amy

Trigger Cloud Function ×

Function Name bck-postgres-241e9618303e45536a516a94017399f1

Specified Range 2024-03-19 To 2024-03-19 📅

Specify the range of database backup files that need to be copied to the storage bucket, based on the file creation time.

Confirm Cancel

You can also use the default settings for the system to automatically trigger function execution.

5. Once the backup data pulling is successful, click the corresponding bucket address in the **Associated Bucket** column to view the saved data.

← PostgreSQL backup Guangzhou (1)

Each application in Application Integration is built through SCF, SCF free quota and billing method have been officially adjusted on June 1, 2022, please see SCF free

Note: The size of a single backup file cannot exceed 2TB. Exceeding 2TB may cause backup failure.

Add Function

Function Name	Associated Bucket	Trigger Period	Database Instance
bck-postgres-241e9618303e45536a516...	amy1-1312520572	be reset daily01:00 Cron expression: 0 0 1 * * *	Region: Guangzhou Instance: postgres-b1xt1y1b amy

Note:

The running log of Serverless Cloud Function can be viewed in the [Serverless Cloud Function Console](#). It is not recommended to directly modify the function for pulling backup from the Serverless Cloud Function Console. Pulling historical backups of TencentDB for PostgreSQL may incur related costs. For specific charging details, please refer to [Pay-As-You-Go](#), [Pay-As-You-Go](#), and [Product Pricing](#).

Rollback to the Original Instance

Last updated : 2024-08-09 15:15:34

Note:

Rollback to the original instance supports the operation of database-level objects.

For instances of SQL Server compatible engines, the rollback of mssql_compatible databases is currently not supported.

Up to 100 databases can be selected for a single rollback task.

Rollback to the original instance allows you to restore existing database objects in the current instance to a specific point in time or backup set. It can also solve the problem of recovering accidentally deleted databases. A detailed description is provided below.

Selecting the Recovery Method and Time Point

Note:

The selectable time for database table recovery is strongly related to your current backup retention policy. For backup settings, refer to [Instructions](#).

You can choose to recover to a specific backup set or to any point in time when the system detects data.

Database Table Restoration

Select Restoration Type

Current Instance

Restoration Mode

By time pointBy backup set

Restoration Time Point

2024-07-30 02:19:33

Database Kernel Version

v16.2_r1.4

Select Database to Restore

Select All | Select Database to Restore

Ente

bill

postgres

Database Selected for Restoration(0)

Name After RestorationDetails

A maximum of 100 database tables can be selected for restoration at a time for a single instance.

OK

Cancel

Selecting the Database to Recover

You can select an existing database pulled by the current system for recovery. Details are shown in the figure below:

Database Table Restoration

Select Restoration Type

Current Instance

Restoration Mode

By time pointBy backup set

Restoration Time Point

2024-07-30 02:19:33

Database Kernel Version

v16.2_r1.4

Select Database to Restore

Select All | Select Database to Restore

Enter

Q

bill

postgres

Add

Database Selected for Restoration(1)

Name After RestorationDetails

bill

postgres

A maximum of 100 database tables can be selected for restoration at a time for a single instance.

OKCancel

Name after Recovery

The name of the recovered database will have a `*_bak_timestamp` suffix. The timestamp is a Unix timestamp when a backend task is initiated. After you submit a database table rollback task on the console or via API, the backend will initiate a rollback task within 5 minutes. For example, if the database name selected by you is `dbone` and the backend initiates the task at 2024-05-30 11:26:25, the new database name after the recovery task is completed will be `dbone_bak_1717039585`.

Handling Accidental Database Deletion

If a database is deleted due to misoperation, the database table rollback feature can resolve this problem. Since the deleted database cannot be retrieved when a task is initiated, you can start the task by adding a new database. Click **Add** in the figure below.

Database Table Restoration

Select Restoration Type

Current Instance

Restoration Mode

By time point

By backup set

Restoration Time Point

2024-07-30 02:19:33

Database Kernel Version v16.2_r1.4

Select Database to Restore

Select All | Select Database to Restore

Enter

Q

bill

postgres

Add

Database Selected for Restoration(1)

bill

postgres

A maximum of 100 database tables can be selected for restoration at a time for a single instance.

OKCancel

Note:

If the added database name does not exist in the selected backup set or PITR time point, an empty database will be recovered.

Add Database

If the newly added database name does not exist in the selected backup set or the PITR time point, then its corresponding empty database will be recovered.

Instance ID/Name postgres-1 / bill

Database Name

Enter a database name.

OKCancel

Physical Migration

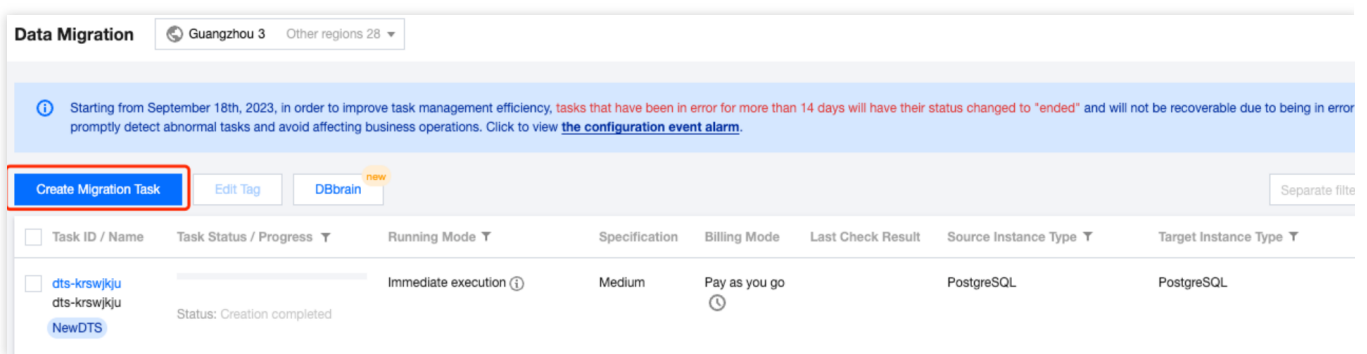
Configure Physical Migration Tasks

Last updated : 2024-07-23 11:56:42

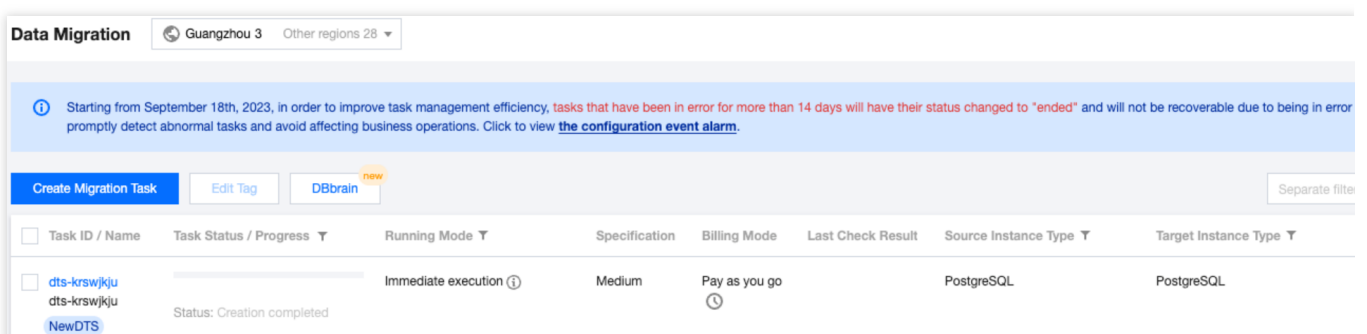
Physical migration employs TencentDB for PostgreSQL physical streaming replication, which is faster and more stable. This document describes how to configure a physical migration task and related precautions.

Configuring a Task

1. Log in to the [TencentDB for PostgreSQL Console](#).
2. In the left sidebar, select **Data Migration**.
3. Go to the Data Migration task list. Click **Create Migration Task**, to set up a migration task with the source and target instance type as TencentDB for PostgreSQL.



4. After creating the migration task, go to the migration task list, click **Operation** > **Configure**, to configure the newly created task.



5. The configuration task involves three steps. Note that in the second step, the migration method needs to set to **Physical migration**. Select default for both migration type and migration object. Proceed to task verification afterward.

Modify Migration Task

1 Set source and target databases > **2 Set migration options and select migration objects** > 3 Verify task

Migration Method ⓘ • Physical migration Logical migration

Migration Type ⓘ • Full + incremental migration

Migration Object ⓘ • Entire instance

ⓘ For migration notes, see [Migration FAQs](#)

Previous Save

6. For system verification details and repair methods, please see [Physical migration check items](#).

Note:

If the source instance is a self-hosted TencentDB for PostgreSQL instance, be sure that the TencentDB for PostgreSQL data directory does not contain any other self-hosted files or directories. Otherwise, the procedure failure may occur.

Modify Migration Task

1 Set source and target databases > 2 Set migration options and select migration objects > **3 Verify task**

ⓘ Please configure the following monitoring alarms in time to quickly detect task errors or abnormal metrics.

1. [Configure event alarms](#)

2. [Configure migration metric alarms](#)

Task ID / Name	Running Mode	Source Instance Type	Target Instance Type	Source Access Type	Target Access Type	Address
dts-krswjkju dts-krswjkju	Immediate execution ⓘ	PostgreSQL	PostgreSQL	Database	Database	Source: postgres-b1 Target: postgres-j5w

Migration Method Logical migration

Migration Type Full + incremental migration

Migration Object Entire instance

• **Create Verification Task**

○ **Query Verification Result 0%** ↻

↻ Querying verification result...

Previous Start Task

7. After all verifications pass, the task officially starts, and you can view the task details in the task list. The entire task involves seven steps, as shown in the figure:

<div> <div>Create Migration Task</div> <div>Edit Tag</div> <div>DBbrain</div> <div>Separate filter</div> </div>									
Task ID / Name	Task Status / Progress	Running ...	Specification	Billing Mode	Last Check Result	Source Ins...	Target Insta...	Source Acce...	Target Acce..
<input type="checkbox"/> dts-5be1reps dts-5be1reps NewDTS	<div> <div>(5 / 7)</div> <div> <div></div> <div>Waiting for switch</div> </div> </div> <div> Current step: Waiting for switch Status: Prepared Start: 2024-07-19 10:07:52 End: -- Source-target data gap: 0 KB Target database last updated: 2024-07-19 10:09:52.012274 +0800 CST Position of WAL logs last sent from source database: 0/8000060 Position of WAL logs last received and stored to target database: 0/8000060 </div>	Immediate execution	Medium	Pay as you go		PostgreSQL	PostgreSQL	Database	Database

For detailed descriptions of the migration steps, see [Migration steps explanation](#).

Migration Steps Explanation

1. Precheck

Before initiating the migration task, the system performs multiple environmental checks on the source and target instances, including the following items:

Target instance session checks: Checks if there are any user sessions on the target instance, and if yes, a system report error is raised. User sessions mainly refer to sessions initiated by accounts other than the following three types of database accounts.

postgers

repluser

Accounts starting with tencentdb_

Target instance database and table check: Checks if there are any user databases or objects on the target instance, and if yes, a system error is raised.

Target instance associated instance checks: Checks if the target instance is already associated with any RO instances, and if yes, an error is raised.

2. Data Backup

To ensure the recoverability of data on the target instance, the system first performs a full automatic backup of the target instance. By using the native TencentDB for PostgreSQL tool `pg_basebackup`, it sets up a standby for the source instance in the Tencent Cloud PostgreSQL environment. Then, it is ready to start the primary-standby data synchronization. TencentDB for PostgreSQL Backup space is already billed, and costs incurred beyond the instance's complimentary space will be charged. For more details on charges, please see [Backup space billing explanation](#).

3. Full Migration

Use the TencentDB for PostgreSQL physical streaming replication to implement full synchronization between the source instance and the standby instance on the Tencent Cloud PostgreSQL.

4. Incremental Sync

Use the TencentDB for PostgreSQL physical streaming replication to implement incremental synchronization between the source instance and the standby instance on the Tencent Cloud PostgreSQL. The master-standby synchronization progress in the task list needs to be observed in time. Once synchronization is achieved, be sure that the source instance needs to stop writing.

<div> <div>Create Migration Task</div> <div>Edit Tag</div> <div>DBbrain</div> <div>Separate filter</div> </div>									
Task ID / Name	Task Status / Progress	Running ...	Specification	Billing Mode	Last Check Result	Source Ins...	Target Insta...	Source Acce...	Target Acce...
<div> <div>dts-5be1reps</div> <div>dts-5be1reps</div> <div>NewDTS</div> </div>	<div> <div>(5 / 7)</div> <div> <div></div> <div></div> <div></div> </div> <div>Current step: Waiting for switch</div> <div>Status: Prepared</div> <div>Start: 2024-07-19 10:07:52</div> <div>End: --</div> <div> <div>Source-target data gap: 0 KB</div> <div>Target database last updated: 2024-07-19 10:09:52.012274 +0800 CST</div> <div>Position of WAL logs last sent from source database: 0/8000060</div> <div>Position of WAL logs last received and stored to target database: 0/8000060</div> </div> </div>	Immediate execution	Medium	Pay as you go		PostgreSQL	PostgreSQL	Database	Database

5. Waiting for Switch

When the difference in the master-standby incremental synchronization between the source instance and the standby instance on the Tencent Cloud PostgreSQL is 0, the system is waiting for a switch. You need to manually click **Complete** to initiate the switch.

<div> <div>Create Migration Task</div> <div>Edit Tag</div> <div>DBbrain</div> <div>Separate filter</div> </div>									
Task ID / Name	Task Status / Progress	Running ...	Specification	Billing Mode	Last Check Result	Source Ins...	Target Insta...	Source Acce...	Target Acce...
<div> <div>dts-5be1reps</div> <div>dts-5be1reps</div> <div>NewDTS</div> </div>	<div> <div>(5 / 7)</div> <div> <div></div> <div></div> <div></div> </div> <div>Current step: Waiting for switch</div> <div>Status: Prepared</div> <div>Start: 2024-07-19 10:07:52</div> <div>End: --</div> <div> <div>Source-target data gap: 0 KB</div> <div>Target database last updated: 2024-07-19 10:09:52.012274 +0800 CST</div> <div>Position of WAL logs last sent from source database: 0/8000060</div> <div>Position of WAL logs last received and stored to target database: 0/8000060</div> </div> </div>	Immediate execution	Medium	Pay as you go		PostgreSQL	PostgreSQL	Database	Database

6. Switching

After you click **Complete**, the system will switch the standby machine on the Tencent Cloud PostgreSQL to the master.

Note:

The system can execute the switch after detecting that the instant LSNs are the same on the source and target instances, but this does not mean that the source instance has stopped writing. Please be sure that the source instance has ceased writing before proceeding with the switch.

7. Completion

After the switch, the standby machine acts as the master. The Tencent Cloud PostgreSQL management system will take over the new master and adapt to the corresponding management operations.

Notes

Instances must be master instances, read-only, and do not support physical migration.

Physical replication does not support data transfer to or from sources or targets with Transparent Data Encryption (TDE) enabled.

Physical Migration Check Items

Last updated : 2024-04-09 10:09:51

The following describes the pre-task check items of TencentDatabase for PostgreSQL physical migration:

Database Connection Check

For detailed check information, reasons, and solutions, see [Source/Target Instance Connection Check](#).

Version Check

The major version of both the source and target instances needs to be consistent, and the kernel version of the target instance cannot be lower than the following versions.

Major Version	Kernel Version
10	v10.17_r1.5
11	v11.12_r1.5
12	v12.7_r1.5
13	v13.3_r1.4
14	v14.2_r1.11
15	v15.1_r1.4

Source Instance Permission Check

Check Details

The account for the source instance migration needs to have LOGIN and REPLICATION permissions.

The configuration of the pg_hba.conf file of the source instance must meet system requirements.

Fix

Users may not have the operation permissions. Please grant users permissions based on the permission requirements in the check details and run the verification task again.

If an error similar to `connect source with replication failed: pq: no pg_hba.conf entry for replication connection from host "xx.xx.xxx.xx", user "xxx"` occurs, the configuration of the `pg_hba.conf` file of the source instance does not meet migration requirements. You can add `host replication all 0.0.0.0/0 md5` to the `pg_hba.conf` file of the source instance, and then restart the instance or execute `select pg_reload_conf();` to reload the configuration.

Note:

When there are permission and password conflicts between source and target instance database accounts, the system uses the source instance account configuration by default.

If the source instance is a PostgreSQL instance from another cloud provider, please use the account with the highest permissions to migrate.

Target Instance Table Existence Check

Check Details

Check if the target instance is empty.

Note:

The have user-created database account can be used on the target instance, but it cannot contain created databases or objects.

Fix

Existing databases or objects on the target instance need to be deleted.

Target Database Space Check

Check Details

The disk space of the target instance needs to be at least larger than 110% of the disk space occupied by the source instance. Otherwise, an error will be reported.

Fix

To expand the disk space on the target end, see [Modifying instance configuration](#).

Key Parameter Check for the Instance

Check Details

`wal_level` of the source instance must be logical.

block_size of the source and target instances must be the same.

For instance of PostgreSQL lower than version 13, the value of wal_keep_segments in the source instance must be greater than or equal to 256.

For PostgreSQL 13 and later versions, the value of wal_keep_size/wal_segment_size in the source instance must be greater than or equal to 256.

wal_block_size of the source and target instances must be the same.

segment_size of the source and target instances must be the same.

max_connections of the target instance must be greater than or equal to max_connections of the source instance.

max_wal_senders of the target instance must be greater than or equal to the max_wal_senders of the source.

max_worker_processes of the target instance must be greater than or equal to max_worker_processes of the source.

Fix

Modify the parameters of the source or target instance according to the requirements in the check details.

Plugin Conflict Check

Check Details

Check if the plugins installed on the source instance are supported by the target instance. If not, the system will report an error. Moreover, if the plugin versions on the source and target instances are inconsistent, the system will report a warning.

Fix

Ensure consistency of plugins between source and target instances by installing or upgrading plugin versions.

Parameter Conflict Check

Check Details

In principle, the parameter values of the source and target instances need to be consistent. When the system detects inconsistent parameter values between the source and target, the usual approach is that **use the target instance's parameter values** and **do not notify the user**. Different approaches apply to the following parameters:

Parameter Name	Default Handling for Migration	Verification Action
data_checksums	Use the parameter value	There is no alert when the parameter values are different.

	of the source instance.	
enable_partitionwise_aggregate	Use the parameter value of the source instance.	There is no alert when the parameter values are different.
enable_partitionwise_join	Use the parameter value of the source instance.	There is no alert when the parameter values are different.
lc_ctype	Use the parameter value of the source instance.	There is no alert when the parameter values are different.
max_locks_per_transaction	Use the parameter value of the source instance.	There is no alert when the parameter values are different.
max_prepared_transactions	Use the parameter value of the source instance.	There is no alert when the parameter values are different.
random_page_cost	Use the parameter value of the source instance.	There is no alert when the parameter values are different.
max_connections	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
max_wal_senders	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
max_worker_processes	Use the parameter value	There is an alert for the user to modify the parameter when parameter values are

	of the target instance.	different. If the parameter is not modified, the default handling method for the migration will be adopted.
array_nulls	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
authentication_timeout	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_analyze_scale_factor	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_analyze_threshold	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_freeze_max_age	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_multixact_freeze_max_age	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_naptime	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_vacuum_cost_delay	Use the	There is an alert for the user to modify the

	parameter value of the target instance.	parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_vacuum_cost_limit	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_vacuum_insert_scale_factor	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_vacuum_insert_threshold	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_vacuum_scale_factor	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
autovacuum_vacuum_threshold	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
bytea_output	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
check_function_bodies	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.

constraint_exclusion	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
cursor_tuple_fraction	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
DateStyle	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
deadlock_timeout	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
default_statistics_target	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
default_transaction_isolation	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
exit_on_error	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
extra_float_digits	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.

from_collapse_limit	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
geqo	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
geqo_effort	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
geqo_generations	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
geqo_pool_size	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
geqo_seed	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
geqo_selection_bias	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
geqo_threshold	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the

		default handling method for the migration will be adopted.
idle_in_transaction_session_timeout	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
idle_session_timeout	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
IntervalStyle	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
jit	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
jit_above_cost	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
jit_inline_above_cost	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
jit_optimize_above_cost	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
join_collapse_limit	Use the parameter value	There is an alert for the user to modify the parameter when parameter values are

	of the target instance.	different. If the parameter is not modified, the default handling method for the migration will be adopted.
lc_monetary	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
lc_numeric	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
lc_time	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
local_preload_libraries	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
log_filename	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
max_logical_replication_workers	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
max_parallel_workers	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
max_replication_slots	Use the	There is an alert for the user to modify the

	parameter value of the target instance.	parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
max_standby_archive_delay	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
max_standby_streaming_delay	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
recursive_worktable_factor	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
search_path	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
statement_timeout	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
stats_fetch_consistency	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
TimeZone	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.

vacuum_cost_delay	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
vacuum_cost_limit	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
vacuum_cost_page_dirty	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
vacuum_cost_page_hit	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
vacuum_cost_page_miss	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
vacuum_freeze_min_age	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
vacuum_freeze_table_age	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
vacuum_multixact_freeze_min_age	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.

<code>vacuum_multixact_freeze_table_age</code>	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.
<code>wal_level</code>	Use the parameter value of the target instance.	There is an alert for the user to modify the parameter when parameter values are different. If the parameter is not modified, the default handling method for the migration will be adopted.

Fix

Modify the parameter values as needed.

Target Instance Status Check

Check Details

TDE data encryption cannot be enabled for the target instance

In the target instance, read-only groups are allowed, but associated read-only instances are not allowed.

No user connections are allowed for the target instance.

Note:

During physical migration, ensure that the source instance is not an encrypted instance.

Fix

Currently, TDE instances and read-only instances are not supported.

Glibc Version Compatibility Check

Check Details

Check the glibc version compatibility issue between the source instance and target instance.

Note:

The sorting rules for some UTF8 characters of glibc 2.28 have been changed. When versions are incompatible, the data sorting rules are different, resulting in a risk of unexpected sorting results.

Fix

Upgrade the glibc version.

Database Audit

Audit Service Description

Last updated : 2024-07-23 12:30:06

Audit Classification

TencentDB for PostgreSQL supports two audit types: **Ultrafast audit** and **Refined audit**. The details are as follows:

- Ultrafast audit**, has the minimal impact on performance, and the same effect as the native community PostgreSQL with log_statement=all enabled. In addition, it also records the affected rows and execution time.
- Refined audit** enables full audit using the pgaudit plugin. The audit logs are more detailed, covering SQL types and object names. However, compared to the Express Edition, it has a higher learning curve, so it is suitable for developers with specific needs.

Note:

For a single SQL entry, if there are subqueries or function calls, the Audit Detailed Edition generates multiple logs, each containing information about the objects being called. To avoid multiple printing of the same statement, from the second printing onwards, the statement is shown as `<previously logged>`, with the SQL type as `???`.

Below are the comparison of the logs generated by **Ultrafast audit** and **Refined audit** in several different scenarios.

Function Call

The specific SQL statements are as follows:

```
CREATE FUNCTION a_t(integer, integer) RETURNS integer
AS 'select $1 + $2;'
LANGUAGE SQL;
select a_t(2,3);
```

The Refined audit log is shown in the following figure:

时间	客户端 IP	账户名称	SQL 类型	数据库名	执行语句	对象类型	会话 ID	对
2023-10-13 11:14:28		n	SELECT	amyt	SELECT a_t(2, 3) LIMIT 11 OFFSET 0	--		
2023-10-13 11:14:24		n	SELECT	amyt	<previously logged>	TABLE		
2023-10-13 11:14:24		n	SELECT	amyt	<previously logged>	TABLE		

The Ultrafast audit log is shown in the following figure:

Audit Instance

Audit Log

Audit

Instance

Last 1 hour

Last 3 hours

Last 24 hours

Last 7 days

2024-07-23 08:30:56 ~ 2024-07-23 09:30:56

Separate keywords with "|"; press Enter to separate filter tags

Search Tips

Time ↕	Client IP	Database Account	Database Name	Command
2024-07-23T09:17:56.427Z	10.10.10.10	dbadmin	postgres_bak_1721206338	SELECT a.t(2, 3) LIMIT 11 OFFSET 0
2024-07-23T09:17:56.423Z	10.10.10.10	dbadmin	postgres_bak_1721206338	;
2024-07-23T09:17:56.419Z	10.10.10.10	dbadmin	postgres_bak_1721206338	;
2024-07-23T09:17:21.950Z	10.10.10.10	dbadmin	postgres_bak_1721206338	CREATE FUNCTION a_t(integer, integer) RETURNS inte ...

Table Association

The specific SQL statements are as follows:

```
create table a(id integer,name varchar);
create table b(id integer,age int);
insert into a(id,name)values(1, 'anne'), (2, 'bob');
insert into b(id,age)values(2, 30);
select a.id,name,age from a,b where a.id=b.id;
```

The Refined audit log is shown in the following figure:

Audit Instance

Audit Log

Audit

Instance

Last 1 hour

Last 3 hours

Last 24 hours

Last 7 days

2024-07-23 06:37:00 ~ 2024-07-23 09:37:00

Separate keywords with "|"; press Enter to separate filter tags

Search Tips

Time ↕	Client IP	Database Account	SQL Type ▼	Database Name	Command
2024-07-23T09:36:22.009Z	10.10.10.10	dbadmin	SELECT	postgres	<previously logged>
2024-07-23T09:36:22.009Z	10.10.10.10	dbadmin	SELECT	postgres	SELECT a.id, name, age FROM a, b WHERE a.id =

The Ultrafast audit log is shown in the following figure:

Audit Instance

Audit Log

Audit

Instance

Last 1 hour

Last 3 hours

Last 24 hours

Last 7 days

2024-07-23 06:37:00 ~ 2024-07-23 09:37:00

Separate keywords with "|"; press Enter to separate filter tags

Search Tips

Time	Client IP	Database Account	Database Name	Command
2024-07-23T09:36:26.591Z		dbadmin	postgres_bak_1721206338	SELECT a.id, name, age FROM a, b WHERE a.id = b.id ...

Subquery

The Refined audit log is shown in the following figure:

Audit Instance

Audit Log

Audit

Instance

Last 1 hour

Last 3 hours

Last 24 hours

Last 7 days

2024-07-23 08:41:18 ~ 2024-07-23 09:41:18

Separate keywords with "|"; press Enter to separate filter tags

Search Tips

Time	Client IP	Database Account	SQL Type	Database Name	Command
2024-07-23T09:40:21.421Z		dbadmin	SELECT	postgres	SELECT c.id FROM (SELECT a.id, name, age FR
2024-07-23T09:40:21.421Z		dbadmin	SELECT	postgres	<previously logged>

The Ultrafast audit log is shown in the following figure:

Audit Instance

Audit Log

Audit

Instance

Last 1 hour

Last 3 hours

Last 24 hours

Last 7 days

2024-07-23 08:40:56 ~ 2024-07-23 09:40:56

Separate keywords with "|"; press Enter to separate filter tags

Search Tips

Time	Client IP	Database Account	Database Name	Command
2024-07-23T09:40:29.205Z		dbadmin	postgres_bak_1721206338	SELECT c.id FROM (SELECT a.id, name, age FROM a, b ...

Storage Procedure

The definition and invocation of the storage procedure are as follows:

```
CREATE OR REPLACE PROCEDURE update_m(  
  p_city in integer,  
  p_ldate in date,  
  p_id in integer)  
AS $$  
BEGIN
```



```
call update_m(4, '2023-02-05', 4);
```

The Ultrafast audit log is shown in the following figure:

Log Description

- Page 189 of 378

protocols are logged. For any extended query protocol (extended query), statements that fail before the execution stage (i.e., during parse, analysis, or planning) will not be logged.

2. In the **Ultrafast audit** and **Refined audit**, the default SQL statement length is 8,192 bytes. Statements exceeding this limit will be truncated, and the object type, object name, execution time, and affected rows of such SQL statements will also be unable to be displayed. To custom the statement length, please modify the `tencentdb_audit_message_truncate_length` parameter in the console. When an SQL statement is truncated, if the statement is a slow SQL statement or a SQL statement that fails to be executed, you can go to [TencentDB for PostgreSQL console](#)'s **Performance Optimization > Slow Query Analysis** or **Error Log** to check its details.

3. Instances of TencentDB for PostgreSQL of the current major version 11 do not support the statistics of affected rows.

4. Due to differences in the timing systems used by audit and slow query, there may be millisecond-level differences in the recorded SQL execution times.

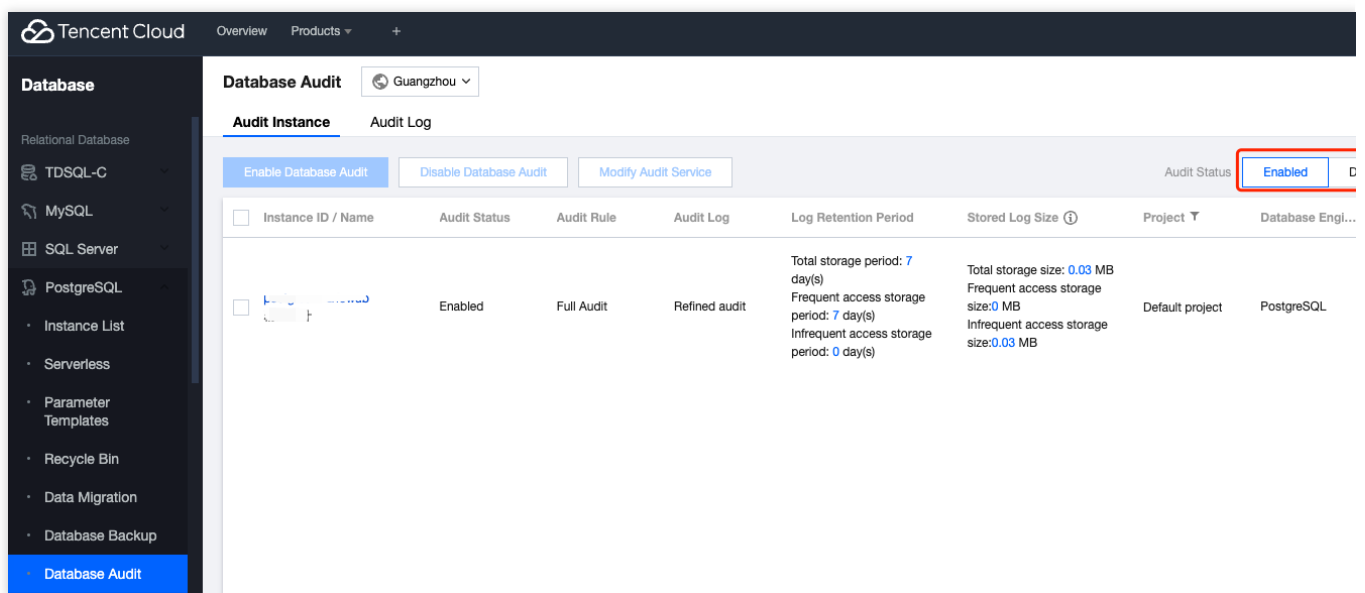
View Audit Logs

Last updated : 2024-07-23 12:31:24

This document describes how to view database audit logs and related fields in the audit log list.

Viewing Audit Logs

1. Log in to the [TencentDB for PostgreSQL Console](#).
2. On the left sidebar, select **Database Audit**.
3. Select **region** at the top, then on the **Audit Instance** page, click **Audit Status** and select the **Enabled** option to filter instances with audit enabled.



4. In the **Audit Instance** list, find the **Target Instance** (you can also quickly search by resource attributes in the search box), click **View Audit Log** in its action column to jump to the **Audit Log** page to view the corresponding log. For the difference between Rapid Audit and Detailed Audit logs, please see [Audit service description](#).

Database Audit

Guangzhou

Audit Instance

Audit Log

Audit

Instance

Last 1 hour

Last 3 hours

Last 24 hours

Last 7 days

2024-07-23 07:12:07 ~ 2024-07-23 10:12:07

Separate keywords with "|"; press Enter to separate filter tags

Search Tips

Time	Client IP	Database Account	SQL Type	Database Name	Command
2024-07-23T09:46:52.069Z	10.10.10.10	dbadmin	CALL	postgres	call update_m(4,'2023-02-05',4)
2024-07-23T09:46:52.069Z	10.10.10.10	dbadmin	UPDATE	postgres	update m set city = p_city, ldate = p_ldate where .
2024-07-23T09:46:52.069Z	10.10.10.10	dbadmin	EXECUTE	postgres	call update_m(4,'2023-02-05',4)
2024-07-23T09:46:31.437Z	10.10.10.10	dbadmin	CREATE PROCEDURE	postgres	CREATE OR REPLACE PROCEDURE update_m(
2024-07-23T09:45:24.953Z	10.10.10.10	dbadmin	SELECT	postgres	SELECT * FROM m WHERE ldate > '2023-01-03'
2024-07-23T09:43:36.895Z	10.10.10.10	dbadmin	SELECT	postgres	SELECT * FROM m WHERE ldate > '2023-01-03'

37 data entries matched(A maximum of the first 60,000 records are displayed. To view complete data, use the log download feature.)

Search Tool Description

In the **Audit Instance Filter**, you can select to switch between other audit instances with the audit service enabled.

In the **time box**, select a time period (log retention period) to peek at the relevant audit logs within the selected period.

Note:

You can select any time period with data for search. Up to the first 60,000 eligible records can be displayed.

In the **search box**, select search items (SQL Details, Client IP, User Account, Database Name, SQL Type, Execution Time (ms), Affected Rows) to look up relevant audit results. Multiple search items are separated by carriage return keys .

All search items are in "include" search mode.

Search Item	Description
Command	Enter SQL Details. Multiple keywords are separated by line breaks. SQL command details search is case-insensitive. When the match type is either "include" or "exclude", it only supports fuzzy search at the token level, instead of wildcard fuzzy search.
Client IP	Enter Client IP. Multiple keywords are separated by line breaks. IP addresses can be filtered using * as a condition.
Database Account	Enter user account name. Multiple keywords are separated by line breaks.

Database Name	Enter the database name. Multiple keywords are separated by line breaks.
SQL Type	Select one or more SQL types from the drop-down list (ALTER, ANALYZE, BEGIN, CALL, CHECKPOINT, CLOSE, COMMENT, COMMIT, COPY, CREATE, DEALLOCATE, DECLARE, DISCARD, DO, DROP, EXECUTE, EXPLAIN, FETCH, GRANT, IMPORT, LISTEN, LOAD, LOCK, MOVE, NOTIFY, PREPARE, REASSIGN, REFRESH, REINDEX, RELEASE, RESET, REVOKE, ROLLBACK, SAVEPOINT, SECURITY, SELECT, SET, SHOW, START TRANSACTION, TRUNCATE, UNLISTEN, UPDATE, VACUUM). Multiple selections are supported.
Execution Time (us)	Enter execution time in the format of number N. Support filtering for matches above N milliseconds.
Affected Rows	Enter the number of affected rows in the format of number N. Support filtering for matches above N rows.

Downloading a Log

You can generate an audit log file first and then download it. The steps are as follows:

1. On the audit log page, click



on the right to generate an audit log file.

Database Audit

Guangzhou

Audit Instance

Audit Log

Audit

Instance

Last 1 hour

Last 3 hours

Last 24 hours

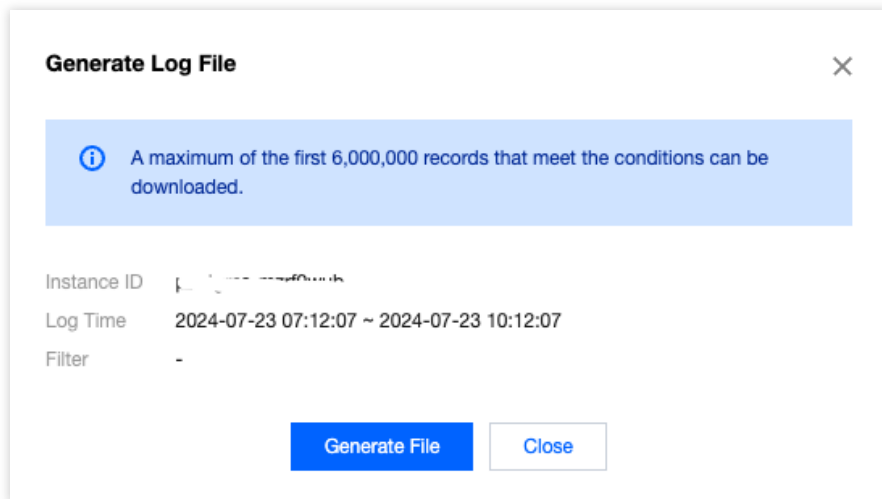
Last 7 days

2024-07-23 07:12:07 ~ 2024-07-23 10:12:07

Separate keywords with "|"; press Enter to separate filter tags

Search Tips

Time	Client IP	Database Account	SQL Type	Database Name	Command
2024-07-23T09:46:52.069Z	10.10.10.10	dbadmin	CALL	postgres	call update_m(4,'2023-02-05',4)
2024-07-23T09:46:52.069Z	10.10.10.10	dbadmin	UPDATE	postgres	update m set city = p_city, ldate = p_ldate where .
2024-07-23T09:46:52.069Z	10.10.10.10	dbadmin	EXECUTE	postgres	call update_m(4,'2023-02-05',4)



2. After the audit log file is generated, you can download it from the file list.

Note:

An audit log file can save up to the first 6,000,000 records of the search results.

Currently, you can download log files only at the Tencent Cloud private network addresses. Please download them through the Tencent cloud service servers in the same region (for example, to download the audit logs of TencentDB for PostgreSQL instances in the Beijing region, please download them with a CVM instance in the Beijing region).

Log files are valid for 24 hours, so please download them in a timely manner.

The wget command format is `wget -c '<log file download URL>' -O <custom file name>.tar.gz`.

Each database instance contains no more than 30 log files. Please download files and delete them promptly to clear space.

Modify audit services

Last updated : 2024-04-09 10:26:14

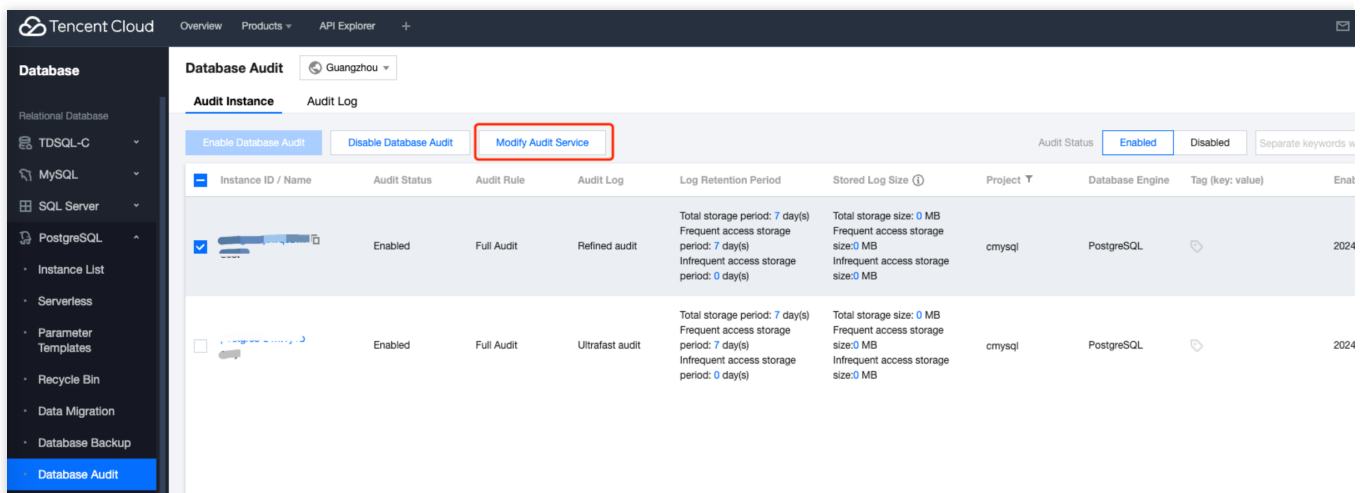
This document describes how to modify the audit service in the console.

Setting the Audit Service

For specific operations, refer to the following two methods for setting the audit service.

Method 1:

1. Log in to the [TencentDB for PostgreSQL Console](#).
2. In the left sidebar, select **Database Audit**.
3. Select **Region** at the top, then on the **Audit Instance** page, click **Audit Status**, and select the **Enabled** option to filter instances with audit enabled.
4. You can modify the audit service for one or more instances in the audit instance list, as shown below:



Modify Audit Service

1. If you choose to extend the log retention period, the change will take effect immediately; if you choose to shorten the log retention period, expired logs will be cleared immediately.

2. If you configure to store the data of the last n days in frequent access storage, older data will be automatically transitioned to infrequent access storage. After the frequent access storage period is extended, the audit data that falls in the period will be automatically migrated from infrequent access storage to frequent access storage. For more information, see [Documentation](#).

Configure Audit

Log Retention Period (day)

Frequent Access Storage Period (day)

Infrequent Access Storage Period (day)

Audit Rule Settings

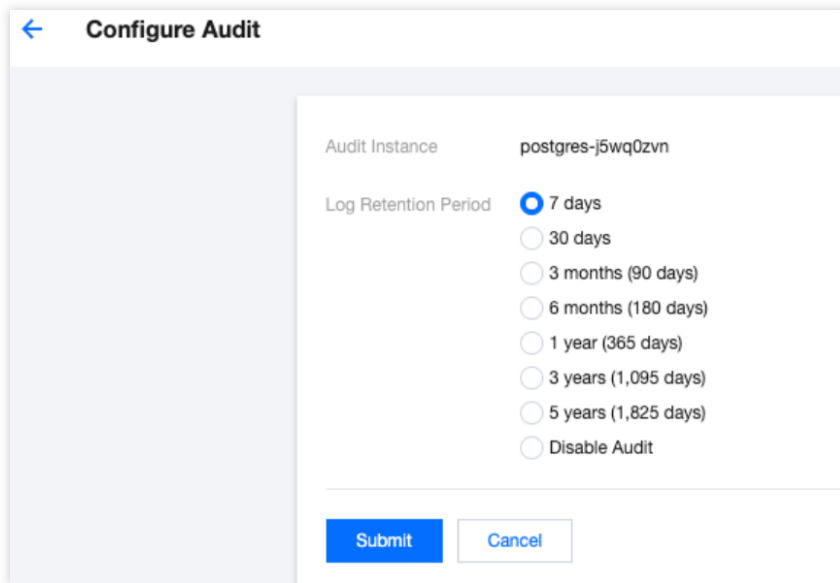
Audit Log

☐ I agree to [Tencent Cloud Terms of Service](#)

OK Cancel

Method 2:

1. Log in to the [TencentDB for PostgreSQL Console](#).
2. In the left sidebar, select **Database Audit**.
3. Select **region** at the top, then on the **Audit Instance** page, click **Audit Status**, and select the **Enabled** option to filter instances with audit enabled.
4. Click **Instance ID** to enter the audit log list of that instance, then click **Service Settings** to configure the audit service for the instance.



← **Configure Audit**

Audit Instance postgres-j5wq0zvn

Log Retention Period ☒ 7 days

☐ 30 days

☐ 3 months (90 days)

☐ 6 months (180 days)

☐ 1 year (365 days)

☐ 3 years (1,095 days)

☐ 5 years (1,825 days)

☐ Disable Audit

Submit **Cancel**

Disabling the Audit Service

Note:

1. After the audit service is disabled, audit for the instance will be stopped and historical audit logs will be cleared. Please proceed with caution.

2. Switching between fast audit and detailed audit will not clear historical audit logs.

You can disable the audit service for one or more instances in the audit instance list, or set the audit service for one or more instances in the audit log list.

1. Log in to the [TencentDB for PostgreSQL Console](#).

2. In the left sidebar, select **Database Audit**.

3. Select **Region** at the top, then on the **Audit Instance** page, click **Audit Status**, and select the **Enabled** option to filter instances with audit enabled.

4. You can modify the audit service for one or more instances in the audit instance list, as shown below:

The screenshot shows the Tencent Cloud Database Audit console. The left sidebar contains a navigation menu with options like 'Database', 'Relational Database', 'TDSQL-C', 'MySQL', 'SQL Server', 'PostgreSQL', 'Instance List', 'Serverless', 'Parameter Templates', 'Recycle Bin', 'Data Migration', and 'Database Backup'. The main panel is titled 'Database Audit' and shows a table of audit instances. The 'Disable Database Audit' button is highlighted with a red rectangle. The table lists two instances: 'postgres-j5wq0zvn' and 'postgres-b1xt1y1b'. The first instance is selected and has its audit status set to 'Enabled'.

Instance ID / Name	Audit Status	Audit Rule	Audit Log	Log Retention Period	Stored Log Size	Project	Database Engine	Tag (key: value)
<input checked="" type="checkbox"/> postgres-j5wq0zvn	Enabled	Full Audit	Refined audit	Total storage period: 7 day(s) Frequent access storage period: 7 day(s) Infrequent access storage period: 0 day(s)	Total storage size: 0 MB Frequent access storage size: 0 MB Infrequent access storage size: 0 MB	cmysql	PostgreSQL	
<input type="checkbox"/> postgres-b1xt1y1b	Enabled	Full Audit	Ultrafast audit	Total storage period: 7 day(s) Frequent access storage period: 7 day(s) Infrequent access storage period: 0 day(s)	Total storage size: 0 MB Frequent access storage size: 0 MB Infrequent access storage size: 0 MB	cmysql	PostgreSQL	

5. You can also click **Instance ID** to enter the audit log details, and then set or disable the audit service for an individual instance.

The 'Configure Audit' dialog box is shown. It has a title bar with a back arrow and the text 'Configure Audit'. The main content area shows the 'Audit Instance' as 'postgres-j5wq0zvn'. Below this, the 'Log Retention Period' is set to '7 days'. A list of options is shown: '7 days', '30 days', '3 months (90 days)', '6 months (180 days)', '1 year (365 days)', '3 years (1,095 days)', '5 years (1,825 days)', and 'Disable Audit'. The 'Disable Audit' option is selected with a blue radio button. At the bottom, there are 'Submit' and 'Cancel' buttons.

Audit Performance Description

Last updated : 2024-04-19 21:58:54

Testing Tool

Sysbench is an open-source, modular, and cross-platform multi-threaded benchmark tool tailored for Online Transaction Processing (OLTP) scenarios. It allows you to evaluate and test the performance of database core parameters under high loads.

In a standard OLTP read/write scenario of SysBench, a transaction contains 18 read/write SQL statements.

In a standard OLTP read-only scenario of SysBench, a transaction contains 14 read SQL statements (ten primary key point queries and four range queries).

In a standard OLTP write-only scenario of SysBench, a transaction contains four write SQL statements (two UPDATE, one DELETE, and one INSERT).

This stress test uses SysBench version 1.1.0. For more information, please see [Sysbench official documentation](#).

Testing Environment

This document describes the environment used for the TencentDB for PostgreSQL performance test.

Region/AZ: Beijing - Beijing Zone 7.

Client: Standard cloud server S6 (16-core 32 GB, SSD volume) 5Mbps.

Client OS: TencentOS Server 2.6 (Final) 64-bit.

Network: Both the CVM and TencentDB for PostgreSQL instances use the Virtual Private Cloud (VPC) network and are in the same subnet.

The information on the TencentDB for PostgreSQL instances tested is as follows:

Storage type: General purpose - local high-performance SSD disk, 8-core 32GB.

Instance architecture: Dual-machine high availability (one primary and one standby) - read/write instance.

Instance version: V14.2.

Primary-standby replication mode: Asynchronous replication.

Test Metrics

This document describes the metrics of the TencentDB for PostgreSQL performance test.

Metrics	Definition
Queries Per Second (QPS)	Number of SQL statements executed per second by the database, including

	INSERT, SELECT, UPDATE, DELETE, COMMIT, etc.
Concurrency	Number of concurrent requests initiated by the client during performance test.

Parameter Description

pgsql-host: private network address of the TencentDB for PostgreSQL instance.

pgsql-port: port number of the TencentDB for PostgreSQL instance.

pgsql-user: username of the TencentDB for PostgreSQL instance.

pgsql-password: password of the above username.

pgsql-db: database name.

table-size: data volume in a single table.

tables: total number of tables.

threads: number of concurrent threads.

time: running time.

Test Method

Note:

Please replace XXX in the following commands with the actual private network address, port number, username, user password, and database name of the TencentDB for PostgreSQL instance, and the single table data volume and total number of tables in the corresponding test scenario.

1. Prepare the data.

```
sysbench /usr/local/share/sysbench/oltp_write_only.lua --db-driver=pgsql --pgsql-host=XXX --pgsql-port=XXX --pgsql-user=XXX --pgsql-password=XXX --pgsql-db=XXX --table-size=XXX --tables=XXX prepare
```

2. Run the command to conduct stress testing.

```
sysbench /usr/local/share/sysbench/oltp_read_write.lua --db-driver=pgsql --pgsql-host=XXX --pgsql-port=XXX --pgsql-user=XXX --pgsql-password=XXX --pgsql-db=XXX --table-size=XXX --tables=XXX --threads=XXX --time=XXX --report-interval=1 run
```

3. Clear the data.

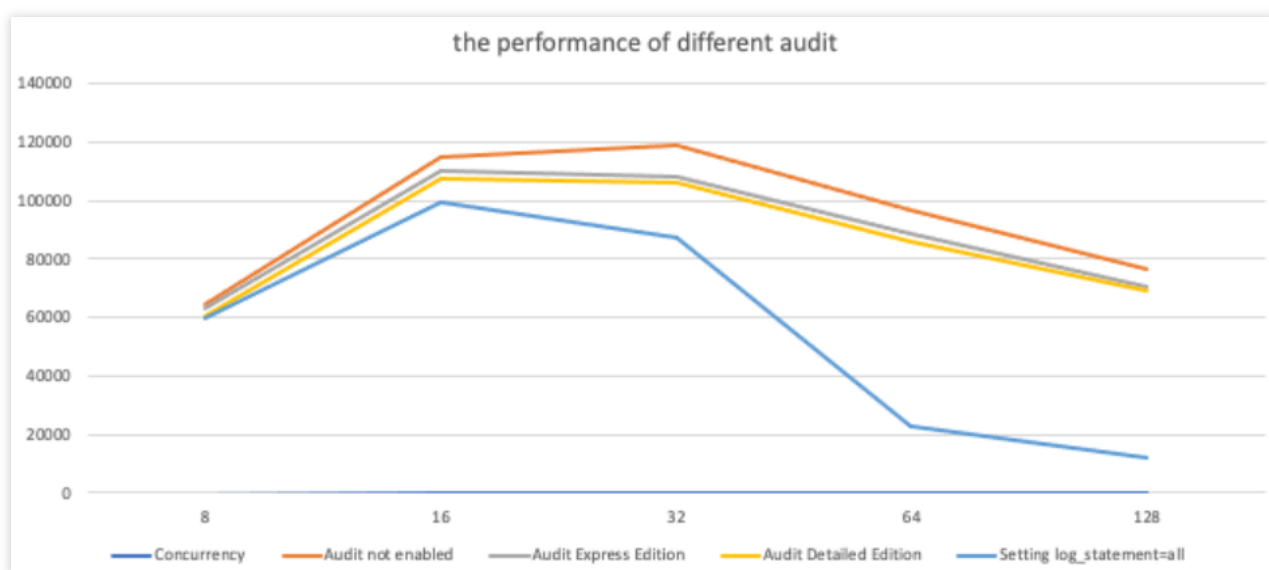
```
sysbench /usr/local/share/sysbench/oltp_write_only.lua --db-driver=pgsql --pgsql-host=XXX --pgsql-port=XXX --pgsql-user=XXX --pgsql-password=XXX --pgsql-db=XXX --table-size=XXX --tables=XXX cleanup
```

Test Results

The table below compares the QPS performance in different scenarios under the test conditions with a single table data volume (table_size) of 25,000 and a total of 64 tables.

Concurrency	Audit not enabled	Audit Express Edition	Audit Detailed Edition	Set log_statement=all
8	64435.35	63150.59	60371.58	59686.77
16	114649.43	110425.06	107427.92	99278.01
32	118850.71	108233.97	106368.01	87540.78
64	97012.02	88828.3	85892.76	23017.53
128	76381.21	70592.04	69241.45	11799.92

As shown below.



Extension Management

Extension Overview

Last updated : 2024-01-24 11:16:51

This document describes the extensions supported by TencentDB for PostgreSQL.

Extension Overview

TencentDB for PostgreSQL supports multiple open-source and proprietary extensions. They help you perform instance OPS easier and improve the query and write performance as well as various capabilities such as token query, data retrieval, and incremental data migration.

Using Extension

Currently, TencentDB for PostgreSQL supports most common extensions for direct use. However, to enable certain extensions, you need to use specified versions or get special permissions. In this case, [submit a ticket](#) and provide the instance ID and extension name to enable it.

Creating Extension

When creating an extension, the `pg_tencentdb_superuser` is temporarily escalated to superuser and passes all permission checks.

TencentDB for PostgreSQL extensions are managed at the database level. You can create different extensions for different databases, but databases cannot use extensions in other databases.

To create an extension, access the database with the client tool and run the following statements:

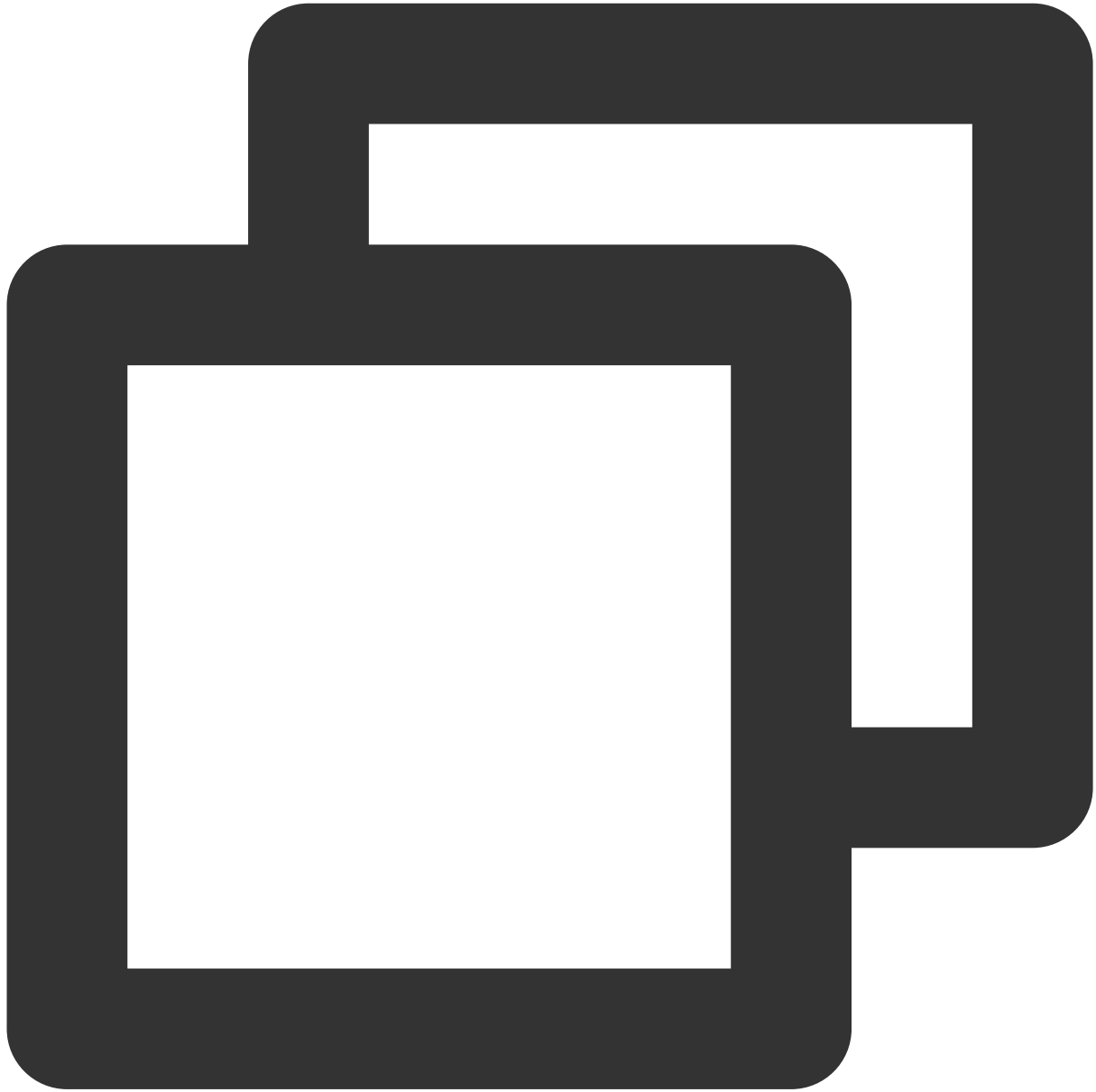


```
CREATE EXTENSION [ IF NOT EXISTS ] extension_name  
[ WITH ]  
[ SCHEMA schema_name ]  
[ VERSION version ]  
[ FROM old_version ]
```

Viewing Created Extension

If you have installed extensions, you can run the following command to view the list of extensions installed in the current database:

You can run the `\\dx` command if you use the psql client.



```
\\dx
```

```
List of installed extensions
```

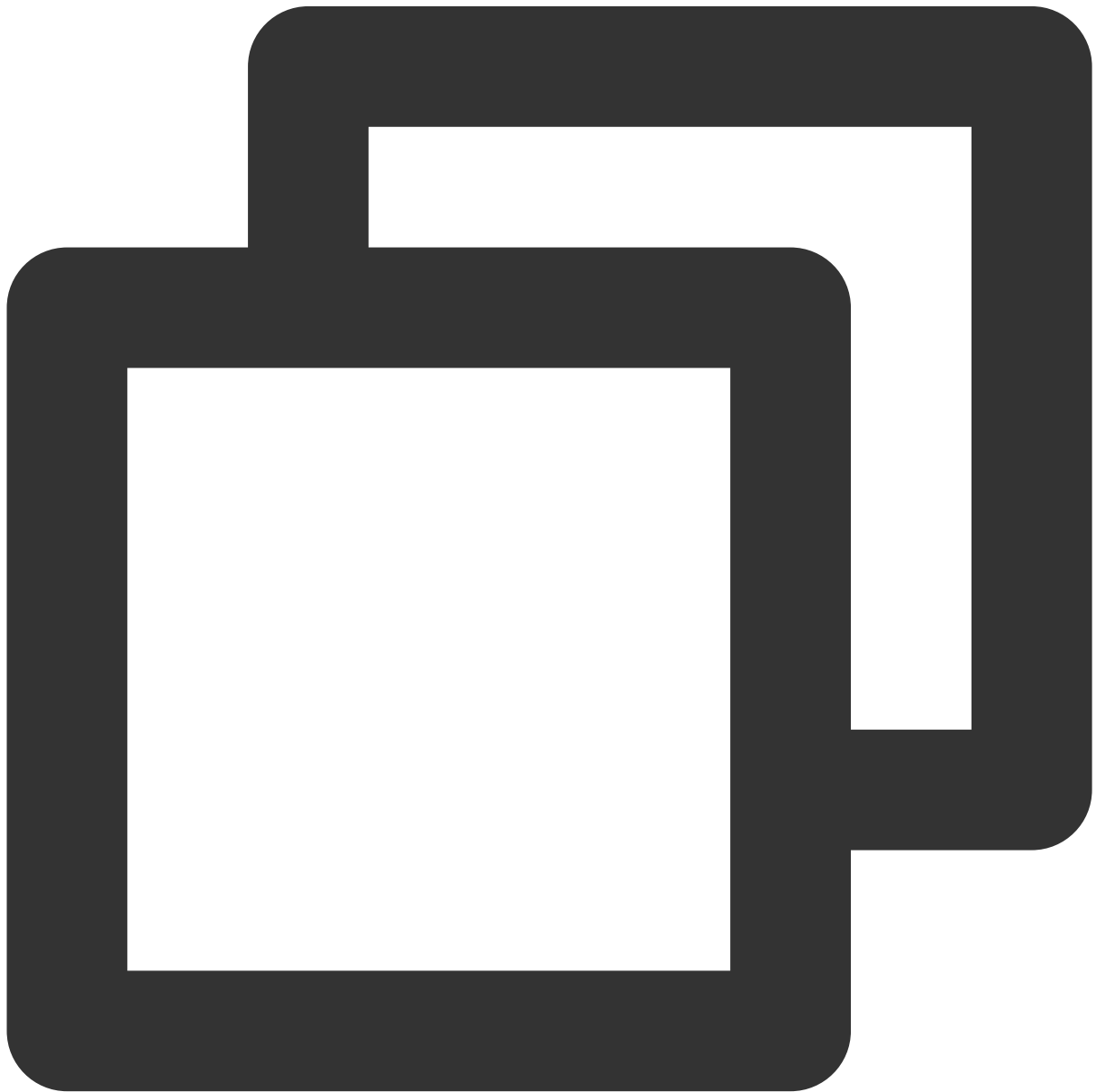
Name	Version	Schema	Description
amcheck	1.2	public	functions for verifying relation integrity
bloom	1.0	public	bloom access method - signature file based
hstore	1.6	public	data type for storing sets of (key, value)

hstore_plperl	1.0	public	transform between hstore and plperl
jsonb_plperl	1.0	public	transform between jsonb and plperl
plperl	1.0	pg_catalog	PL/Perl procedural language
plpgsql	1.0	pg_catalog	PL/pgSQL procedural language
postgis	3.0.2	public	PostGIS geometry, geography, and raster spa

(8 rows)

If you want to use SQL statements to view the extensions, run the `select * from`

`pg_available_extensions where installed_version is not null;` statement to view the list of installed extensions.



name	default_version	installed_version	
plperl	1.0	1.0	PL/Perl procedural language
amcheck	1.2	1.2	functions for verifying rela
hstore_plperl	1.0	1.0	transform between hstore and
plpgsql	1.0	1.0	PL/pgSQL procedural language
jsonb_plperl	1.0	1.0	transform between jsonb and
hstore	1.6	1.6	data type for storing sets o
bloom	1.0	1.0	bloom access method - signat
postgis	3.0.2	3.0.2	PostGIS geometry, geography,
(8 rows)			

List of Supported Extensions

TencentDB for PostgreSQL supports multiple powerful and high-performance extensions. For the list of extensions supported by each database version, see [Supported Extensions](#).

Supported Extensions

Overview

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL.

TencentDB for PostgreSQL supports creating custom extensions (CREATEEXTENSION < pluginName >). For extensions supported by TencentDB for PostgreSQL versions, see below:

[TencentDB for PostgreSQL 9.3](#)

[TencentDB for PostgreSQL 9.5](#)

[TencentDB for PostgreSQL 10](#)

[TencentDB for PostgreSQL 11](#)

[TencentDB for PostgreSQL 12](#)

[TencentDB for PostgreSQL 13](#)

[TencentDB for PostgreSQL 14](#)

Note:

The timescaledb, pipelinedb, wal2json, decoder_raw, and decoderbufs extensions cannot be directly created and used. If you want to use them or have any requirements or suggestions for extensions, [submit a ticket](#) for assistance.

TencentDB for PostgreSQL 9.3

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL 9.3.

Extension	v9.3.25_r1.3	v9.3.25_r1.2	v9.3.25_r1.1	v9.3.5_r1.0
pg_hint_plan	1.1.4	1.1.4	1.1.4	1.1.4
pg_prewarm	Unsupported	Unsupported	Unsupported	Unsupported
pg_stat_error	1	1	1	1
pg_stat_log	1	1	1	1
pg_stat_statements	1.1	1.1	1.1	1.1
pgrowlocks	1.1	1.1	1.1	1.1
sslnfo	1	1	1	1
tablefunc	1	1	1	1
tcn	1	1	1	1
unaccent	1	1	1	1
uuid-osp	1	1	1	1
pg_cron	Unsupported	Unsupported	Unsupported	Unsupported
pgagent	4	4	4	4
pg_partman	Unsupported	Unsupported	Unsupported	Unsupported
tsearch2	1	1	1	1
postgis	2.3.0	2.3.0	2.3.0	2.3.0
postgis_raster	Unsupported	Unsupported	Unsupported	Unsupported
postgis_sfcgal	2.3.0	2.3.0	2.3.0	Unsupported
postgis_tiger_geocoder	2.3.0	2.3.0	2.3.0	2.3.0
postgis_topology	2.3.0	2.3.0	2.3.0	2.3.0
pgrouting	2.4.1	2.4.1	2.4.1	2.4.1

address_standardizer	2.3.0	2.3.0	2.3.0	2.3.0
address_standardizer_data_us	2.3.0	2.3.0	2.3.0	2.3.0
earthdistance	1	1	1	1
plperl	1	1	1	1
plpgsql	1	1	1	1
pltcl	1	1	1	1
plv8	2.0.0	2.0.0	2.0.0	2.0.0
bool_plperl	Unsupported	Unsupported	Unsupported	Unsupported
jsonb_plperl	Unsupported	Unsupported	Unsupported	Unsupported
hstore	1.2	1.2	1.2	1.2
hstore_plperl	Unsupported	Unsupported	Unsupported	Unsupported
plcoffee	2.0.0	2.0.0	2.0.0	2.0.0
plls	2.0.0	2.0.0	2.0.0	2.0.0
timescaledb	Unsupported	Unsupported	Unsupported	Unsupported
pipelinedb	Unsupported	Unsupported	Unsupported	Unsupported
rdkit	Unsupported	Unsupported	Unsupported	Unsupported
imgsmr	1	1	1	1
zhparser	1	1	1	1
intagg	1	1	1	1
intarray	1	1	1	1
isn	1	1	1	1
xml2	1	1	1	1
jsonbx	Unsupported	Unsupported	Unsupported	Unsupported
dict_int	1	1	1	1
dict_xsyn	1	1	1	1

citext	1	1	1	1
ltree	1	1	1	1
postgres_fdw	1	1	1	1
orafce	3.3	3.3	3.3	3.3
chkpass	1	1	1	1
bloom	Unsupported	Unsupported	Unsupported	Unsupported
btree_gin	1	1	1	1
btree_gist	1	1	1	1
roaringbitmap	Unsupported	Unsupported	Unsupported	Unsupported
rum	Unsupported	Unsupported	Unsupported	Unsupported
cube	1	1	1	1
decoderbufs	Unsupported	Unsupported	Unsupported	Unsupported
pg_bigm	Unsupported	Unsupported	Unsupported	Unsupported
fuzzystrmatch	1	1	1	1
hll	Unsupported	Unsupported	Unsupported	Unsupported
pg_trgm	1.1	1.1	1.1	1.1
pg_hashids	1.2.1	1.2.1	1.2.1	1.2.1
pgcrypto	1	1	1	1
cos_fdw	Unsupported	Unsupported	Unsupported	Unsupported
topn	Unsupported	Unsupported	Unsupported	Unsupported

TencentDB for PostgreSQL 9.5

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL 9.5.

Extension	v9.5.25_r1.3	v9.5.25_r1.2	v9.5.25_r1.1	v9.5.4_r1.0
pg_hint_plan	1.1.5	1.1.5	1.1.5	1.1.5
pg_prewarm	1	1	1	Unsupported
pg_stat_error	1	1	1	1
pg_stat_log	1	1	1	1
pg_stat_statements	1.3	1.3	1.3	1.3
pgrowlocks	1.1	1.1	1.1	1.1
sslinfo	1	1	1	1
tablefunc	1	1	1	1
tcn	1	1	1	1
unaccent	1	1	1	1
uuid-osp	1	1	1	1
pg_cron	1.1	1.1	1.1	1.1
pgagent	1.2	1.2	1.2	4
pg_partman	2.6.4, 1.4	2.6.4, 1.4	2.6.4, 1.4	2.6.4, 1.4, 1.0
tsearch2	1	1	1	1
postgis	2.3.0	2.3.0	2.3.0	2.3.0
postgis_raster	Unsupported	Unsupported	Unsupported	Unsupported
postgis_sfcgal	2.3.0	2.3.0	2.3.0	Unsupported
postgis_tiger_geocoder	2.3.0	2.3.0	2.3.0	2.3.0
postgis_topology	2.3.0	2.3.0	2.3.0	2.3.0
pgrouting	2.4.1	2.4.1	2.4.1	2.4.1

address_standardizer	2.3.0	2.3.0	2.3.0	2.3.0
address_standardizer_data_us	2.3.0	2.3.0	2.3.0	2.3.0
earthdistance	1	1	1	1
plperl	1	1	1	1
plpgsql	1	1	1	1
pltcl	1	1	1	1
plv8	2.0.0	2.0.0	2.0.0	2.0.0
bool_plperl	Unsupported	Unsupported	Unsupported	Unsupported
jsonb_plperl	Unsupported	Unsupported	Unsupported	Unsupported
hstore	1.3	1.3	1.3	1.3
hstore_plperl	1	1	1	1
plcoffee	2.0.0	2.0.0	2.0.0	2.0.0
plls	2.0.0	2.0.0	2.0.0	2.0.0
timescaledb	Unsupported	Unsupported	Unsupported	Unsupported
pipelinedb	Unsupported	Unsupported	Unsupported	Unsupported
rdkit	Unsupported	Unsupported	Unsupported	Unsupported
imgsmr	1	1	1	1
zhparser	1	1	1	1
intagg	1	1	1	1
intarray	1	1	1	1
isn	1	1	1	1
xml2	1	1	1	1
jsonbx	1	1	1	1
dict_int	1	1	1	1
dict_xsyn	1	1	1	1

citext	1.1	1.1	1.1	1.1
ltree	1	1	1	1
postgres_fdw	1	1	1	1
orafce	3.3	3.3	3.3	3.3
chkpass	1	1	1	1
bloom	Unsupported	Unsupported	Unsupported	Unsupported
btree_gin	1	1	1	1
btree_gist	1.1	1.1	1.1	1.1
roaringbitmap	Unsupported	Unsupported	Unsupported	Unsupported
rum	Unsupported	Unsupported	Unsupported	Unsupported
cube	1	1	1	1
decoderbufs	Unsupported	Unsupported	Unsupported	Unsupported
pg_bigm	1.2	1.2	1.2	1.2
fuzzystrmatch	1	1	1	1
hll	2.14	2.14	2.14	2.14
pg_trgm	1.1	1.1	1.1	1.1
pg_hashids	1.2.1	1.2.1	1.2.1	1.2.1
pgcrypto	1.2	1.2	1.2	1.2
cos_fdw	Unsupported	Unsupported	Unsupported	Unsupported
topn	Unsupported	Unsupported	Unsupported	Unsupported

TencentDB for PostgreSQL 10

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL 10.

Extension	v10.17_r1.4	v10.17_r1.3	v10.17_r1.2	v10.17_r1.1	v10.4_r1
pg_hint_plan	1.3.6	1.3.6	1.3.6	1.3.6	1.3.6
pg_prewarm	1.1	1.1	1.1	1.1	1.1
pg_stat_error	1	1	1	1	1
pg_stat_log	1	1	1	1	1
pg_stat_statements	1.6	1.6	1.6	1.6	1.5
pgrowlocks	1.2	1.2	1.2	1.2	1.2
sslnfo	1.2	1.2	1.2	1.2	1.2
tablefunc	1	1	1	1	1
tcn	1	1	1	1	1
unaccent	1.1	1.1	1.1	1.1	1.1
uuid-oss	1.1	1.1	1.1	1.1	1.1
pg_cron	1.4	1.4	1.1	1.1	1.1
pgagent	4	4	4	4	4
pg_partman	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
tsearch2	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
postgis	3.0.1	3.0.1	3.0.1	3.0.1	2.3.7
postgis_raster	3.0.1	3.0.1	3.0.1	3.0.1	Unsuppo
postgis_sfcgal	3.0.1	3.0.1	3.0.1	3.0.1	Unsuppo
postgis_tiger_geocoder	3.0.1	3.0.1	3.0.1	3.0.1	2.4.1
postgis_topology	3.0.1	3.0.1	3.0.1	3.0.1	2.4.1
pgrouting	2.6.0	2.6.0	2.6.0	2.6.0	2.6.0

address_standardizer	3.0.1	3.0.1	3.0.1	3.0.1	2.4.1
address_standardizer_data_us	3.0.1	3.0.1	3.0.1	3.0.1	2.4.1
earthdistance	1.1	1.1	1.1	1.1	1.1
plperl	1	1	1	1	1
plpgsql	1	1	1	1	1
pltcl	1	1	1	1	1
plv8	2.3.4	2.3.4	2.3.4	2.3.4	2.3.4
bool_plperl	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
jsonb_plperl	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
hstore	1.4	1.4	1.4	1.4	1.4
hstore_plperl	1	1	1	1	1
plcoffee	2.3.4	2.3.4	2.3.4	2.3.4	2.3.4
plls	2.3.4	2.3.4	2.3.4	2.3.4	2.3.4
timescaledb	1.7.5	1.7.5	1.7.5	1.7.5	1.7.5
pipelinedb	1.0.0	1.0.0	1.0.0	1.0.0	1.0.0
rdkit	3.8	3.8	3.8	3.8	Unsuppo
imgsmplr	1	1	1	1	1
zhparser	1	1	1	1	1
intagg	1.1	1.1	1.1	1.1	1.1
intarray	1.2	1.2	1.2	1.2	1.2
isn	1.1	1.1	1.1	1.1	1.1
xml2	1.1	1.1	1.1	1.1	1.1
jsonbx	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
dict_int	1	1	1	1	1
dict_xsyn	1	1	1	1	1

citext	1.4	1.4	1.4	1.4	1.4
ltree	1.1	1.1	1.1	1.1	1.1
postgres_fdw	1	1	1	1	1
orafce	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
chkpass	1	1	1	1	1
bloom	1	1	1	1	1
btree_gin	1.2	1.2	1.2	1.2	1.2
btree_gist	1.5	1.5	1.5	1.5	1.5
roaringbitmap	0.5	0.5	0.5	0.5	Unsuppo
rum	1.3	1.3	1.3	1.3	1.3
cube	1.2	1.2	1.2	1.2	1.2
decoderbufs	0.1.0	0.1.0	0.1.0	0.1.0	0.1.0
pg_bigm	1.2	1.2	1.2	1.2	1.2
fuzzystrmatch	1.1	1.1	1.1	1.1	1.1
hll	2.14	2.14	2.14	2.14	2.14
pg_trgm	1.3	1.3	1.3	1.3	1.3
pg_hashids	1.2.1	1.2.1	1.2.1	1.2.1	1.2.1
pgcrypto	1.3	1.3	1.3	1.3	1.3
cos_fdw	1	Unsupported	Unsupported	Unsupported	Unsuppo
topn	2.4.0	2.4.0	Unsupported	Unsupported	Unsuppo

TencentDB for PostgreSQL 11

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL 11.

Extension	v11.12_r1.4	v11.12_r1.3	v11.12_r1.2	v11.12_r1.1	v11.8_r1
pg_hint_plan	1.3.6	1.3.6	1.3.6	1.3.6	1.3.6
pg_prewarm	1.2	1.2	1.2	1.2	1.2
pg_stat_error	1	1	1	1	1
pg_stat_log	1	1	1	1	1
pg_stat_statements	1.6	1.6	1.6	1.6	1.6
pgrowlocks	1.2	1.2	1.2	1.2	1.2
sslnfo	1.2	1.2	1.2	1.2	1.2
tablefunc	1	1	1	1	1
tcn	1	1	1	1	1
unaccent	1.1	1.1	1.1	1.1	1.1
uuid-oss	1.1	1.1	1.1	1.1	1.1
pg_cron	1.4	1.4	Unsupported	Unsupported	Unsuppo
pgagent	4	4	4	4	4
pg_partman	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
tsearch2	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
postgis	3.0.1	3.0.1	3.0.1	3.0.1	3.0.1
postgis_raster	3.0.1	3.0.1	3.0.1	3.0.1	3.0.1
postgis_sfcgal	3.0.1	3.0.1	3.0.1	3.0.1	Unsuppo
postgis_tiger_geocoder	3.0.1	3.0.1	3.0.1	3.0.1	3.0.1
postgis_topology	3.0.1	3.0.1	3.0.1	3.0.1	3.0.1
pgrouting	2.6.0	2.6.0	2.6.0	2.6.0	2.6.0

address_standardizer	3.0.1	3.0.1	3.0.1	3.0.1	3.0.1
address_standardizer_data_us	3.0.1	3.0.1	3.0.1	3.0.1	3.0.1
earthdistance	1.1	1.1	1.1	1.1	1.1
plperl	1	1	1	1	1
plpgsql	1	1	1	1	1
pltcl	1	1	1	1	1
plv8	2.3.15	2.3.15	2.3.15	2.3.15	2.3.15
bool_plperl	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
jsonb_plperl	1	1	1	1	1
hstore	1.5	1.5	1.5	1.5	1.5
hstore_plperl	1	1	1	1	1
plcoffee	2.3.15	2.3.15	2.3.15	2.3.15	2.3.15
plls	2.3.15	2.3.15	2.3.15	2.3.15	2.3.15
timescaledb	1.7.5	1.7.5	1.7.5	1.7.5	1.7.5
pipelinedb	1.0.0	1.0.0	1.0.0	1.0.0	1.0.0
rdkit	3.8	3.8	3.8	3.8	Unsuppo
imgsmplr	1	1	1	1	1
zhparser	1	1	1	1	1
intagg	1.1	1.1	1.1	1.1	1.1
intarray	1.2	1.2	1.2	1.2	1.2
isn	1.2	1.2	1.2	1.2	1.2
xml2	1.1	1.1	1.1	1.1	1.1
jsonbx	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
dict_int	1	1	1	1	1
dict_xsyn	1	1	1	1	1

citext	1.5	1.5	1.5	1.5	1.5
ltree	1.1	1.1	1.1	1.1	1.1
postgres_fdw	1	1	1	1	1
orafce	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
chkpass	1	1	1	1	1
bloom	1	1	1	1	1
btree_gin	1.3	1.3	1.3	1.3	1.3
btree_gist	1.5	1.5	1.5	1.5	1.5
roaringbitmap	0.5	0.5	0.5	0.5	Unsuppo
rum	1.3	1.3	1.3	1.3	1.3
cube	1.4	1.4	1.4	1.4	1.4
decoderbufs	0.1.0	0.1.0	0.1.0	0.1.0	0.1.0
pg_bigm	1.2	1.2	1.2	1.2	1.2
fuzzystrmatch	1.1	1.1	1.1	1.1	1.1
hll	2.14	2.14	2.14	2.14	2.14
pg_trgm	1.4	1.4	1.4	1.4	1.4
pg_hashids	1.2.1	1.2.1	1.2.1	1.2.1	1.2.1
pgcrypto	1.3	1.3	1.3	1.3	1.3
cos_fdw	1	Unsupported	Unsupported	Unsupported	Unsuppo
topn	2.4.0	2.4.0	Unsupported	Unsupported	Unsuppo

TencentDB for PostgreSQL 12

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL 12.

Extension	v12.7_r1.4	v12.7_r1.3	v12.7_r1.2	v12.7_r1.1	v12.4_r1
pg_hint_plan	1.3.6	1.3.6	1.3.6	1.3.6	1.3.6
pg_prewarm	1.2	1.2	1.2	1.2	1.2
pg_stat_error	1	1	1	1	1
pg_stat_log	1	1	1	1	1
pg_stat_statements	1.7	1.7	1.7	1.7	1.7
pgrowlocks	1.2	1.2	1.2	1.2	1.2
sslnfo	1.2	1.2	1.2	1.2	1.2
tablefunc	1	1	1	1	1
tcn	1	1	1	1	1
unaccent	1.1	1.1	1.1	1.1	1.1
uuid-oss	1.1	1.1	1.1	1.1	1.1
pg_cron	1.4	1.4	Unsupported	Unsupported	Unsuppo
pgagent	4	4	4	4	4
pg_partman	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
tsearch2	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
postgis	3.0.2	3.0.2	3.0.2	3.0.2	3.0.2
postgis_raster	3.0.2	3.0.2	3.0.2	3.0.2	3.0.2
postgis_sfcgal	3.0.2	3.0.2	3.0.2	3.0.2	3.0.2
postgis_tiger_geocoder	3.0.2	3.0.2	3.0.2	3.0.2	3.0.2
postgis_topology	3.0.2	3.0.2	3.0.2	3.0.2	3.0.2
pgrouting	3.1.0	3.1.0	3.1.0	3.1.0	3.1.0

address_standardizer	3.0.2	3.0.2	3.0.2	3.0.2	3.0.2
address_standardizer_data_us	3.0.2	3.0.2	3.0.2	3.0.2	3.0.2
earthdistance	1.1	1.1	1.1	1.1	1.1
plperl	1	1	1	1	1
plpgsql	1	1	1	1	1
pltcl	1	1	1	1	1
plv8	2.3.15	2.3.15	2.3.15	2.3.15	2.3.15
bool_plperl	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
jsonb_plperl	1	1	1	1	1
hstore	1.6	1.6	1.6	1.6	1.6
hstore_plperl	1	1	1	1	1
plcoffee	2.3.15	2.3.15	2.3.15	2.3.15	2.3.15
plls	2.3.15	2.3.15	2.3.15	2.3.15	2.3.15
timescaledb	1.7.4	1.7.4	1.7.4	1.7.4	1.7.4
pipelinedb	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
rdkit	3.8	3.8	3.8	3.8	3.8
imgsmplr	1	1	1	1	1
zhparser	1	1	1	1	1
intagg	1.1	1.1	1.1	1.1	1.1
intarray	1.2	1.2	1.2	1.2	1.2
isn	1.2	1.2	1.2	1.2	1.2
xml2	1.1	1.1	1.1	1.1	1.1
jsonbx	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
dict_int	1	1	1	1	1
dict_xsyn	1	1	1	1	1

citext	1.6	1.6	1.6	1.6	1.6
ltree	1.1	1.1	1.1	1.1	1.1
postgres_fdw	1	1	1	1	1
orafce	Unsupported	Unsupported	Unsupported	Unsupported	Unsuppo
chkpass	1	1	1	1	1
bloom	1	1	1	1	1
btree_gin	1.3	1.3	1.3	1.3	1.3
btree_gist	1.5	1.5	1.5	1.5	1.5
roaringbitmap	0.5	0.5	0.5	0.5	Unsuppo
rum	1.3	1.3	1.3	1.3	1.3
cube	1.4	1.4	1.4	1.4	1.4
decoderbufs	0.1.0	0.1.0	0.1.0	0.1.0	0.1.0
pg_bigm	1.2	1.2	1.2	1.2	1.2
fuzzystrmatch	1.1	1.1	1.1	1.1	1.1
hll	2.14	2.14	2.14	2.14	2.14
pg_trgm	1.4	1.4	1.4	1.4	1.4
pg_hashids	1.2.1	1.2.1	1.2.1	1.2.1	1.2.1
pgcrypto	1.3	1.3	1.3	1.3	1.3
cos_fdw	1	Unsupported	Unsupported	Unsupported	Unsuppo
topn	2.4.0	2.4.0	Unsupported	Unsupported	Unsuppo

TencentDB for PostgreSQL 13

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL 13.

Extension	v13.3_r1.3	v13.3_r1.2	v13.3_r1.1	v13.3_r1.0
pg_hint_plan	1.3.7	1.3.7	1.3.7	1.3.7
pg_prewarm	1.2	1.2	1.2	1.2
pg_stat_error	1	1	1	1
pg_stat_log	1	1	1	1
pg_stat_statements	1.8	1.8	1.8	1.8
pgrowlocks	1.2	1.2	1.2	1.2
sslnfo	1.2	1.2	1.2	1.2
tablefunc	1	1	1	1
tcn	1	1	1	1
unaccent	1.1	1.1	1.1	1.1
uuid-osp	1.1	1.1	1.1	1.1
pg_cron	1.4	1.4	Unsupported	Unsupported
pgagent	4	4	4	4
pg_partman	Unsupported	Unsupported	Unsupported	Unsupported
tsearch2	Unsupported	Unsupported	Unsupported	Unsupported
postgis	3.0.2	3.0.2	3.0.2	3.0.2
postgis_raster	3.0.2	3.0.2	3.0.2	3.0.2
postgis_sfcgal	3.0.2	3.0.2	3.0.2	3.0.2
postgis_tiger_geocoder	3.0.2	3.0.2	3.0.2	3.0.2
postgis_topology	3.0.2	3.0.2	3.0.2	3.0.2
pgrouting	3.1.0	3.1.0	3.1.0	3.1.0

address_standardizer	3.0.2	3.0.2	3.0.2	3.0.2
address_standardizer_data_us	3.0.2	3.0.2	3.0.2	3.0.2
earthdistance	1.1	1.1	1.1	1.1
plperl	1	1	1	1
plpgsql	1	1	1	1
pltcl	1	1	1	1
plv8	2.3.15	2.3.15	2.3.15	2.3.15
bool_plperl	1	1	1	1
jsonb_plperl	1	1	1	1
hstore	1.7	1.7	1.7	1.7
hstore_plperl	1	1	1	1
plcoffee	2.3.15	2.3.15	2.3.15	2.3.15
plls	2.3.15	2.3.15	2.3.15	2.3.15
timescaledb	2.1.1	2.1.1	2.1.1	2.1.1
pipelinedb	Unsupported	Unsupported	Unsupported	Unsupported
rdkit	3.8	3.8	3.8	3.8
imgsmblr	1	1	1	1
zhparser	1	1	1	1
intagg	1.1	1.1	1.1	1.1
intarray	1.3	1.3	1.3	1.3
isn	1.2	1.2	1.2	1.2
xml2	1.1	1.1	1.1	1.1
jsonbx	Unsupported	Unsupported	Unsupported	Unsupported
dict_int	1	1	1	1
dict_xsyn	1	1	1	1

citext	1.6	1.6	1.6	1.6
ltree	1.2	1.2	1.2	1.2
postgres_fdw	1	1	1	1
orafce	Unsupported	Unsupported	Unsupported	Unsupported
chkpass	1	1	1	1
bloom	1	1	1	1
btree_gin	1.3	1.3	1.3	1.3
btree_gist	1.5	1.5	1.5	1.5
roaringbitmap	0.5	0.5	0.5	0.5
rum	1.3	1.3	1.3	1.3
cube	1.4	1.4	1.4	1.4
decoderbufs	0.1.0	0.1.0	0.1.0	0.1.0
pg_bigm	1.2	1.2	1.2	1.2
fuzzystrmatch	1.1	1.1	1.1	1.1
hll	2.15	2.15	2.15	2.15
pg_trgm	1.5	1.5	1.5	1.5
pg_hashids	1.2.1	1.2.1	1.2.1	1.2.1
pgcrypto	1.3	1.3	1.3	1.3
cos_fdw	1	Unsupported	Unsupported	Unsupported
topn	2.4.0	2.4.0	Unsupported	Unsupported

TencentDB for PostgreSQL 14

Last updated : 2024-01-24 11:16:51

This document lists the extensions supported by different kernel versions of TencentDB for PostgreSQL 14.

Extension	v14.2_r1.1	v14.2_r1.0
pg_hint_plan	1.4	1.4
pg_prewarm	1.2	1.2
pg_stat_error	Unsupported	Unsupported
pg_stat_log	1	1
pg_stat_statements	1.9	1.9
pgrowlocks	1.2	1.2
sslnfo	1.2	1.2
tablefunc	1	1
tcn	1	1
unaccent	1.1	1.1
uuid-osp	1.1	1.1
pg_cron	1.4	1.4
pgagent	4	4
pg_partman	Unsupported	Unsupported
tsearch2	Unsupported	Unsupported
postgis	3.2.1	3.2.1
postgis_raster	3.2.1	3.2.1
postgis_sfcgal	3.2.1	3.2.1
postgis_tiger_geocoder	3.2.1	3.2.1
postgis_topology	3.2.1	3.2.1
pgrouting	3.2.2	3.2.2

address_standardizer	3.2.1	3.2.1
address_standardizer_data_us	3.2.1	3.2.1
earthdistance	1.1	1.1
plperl	1	1
plpgsql	1	1
pltcl	1	1
plv8	2.3.15	2.3.15
bool_plperl	1	1
jsonb_plperl	1	1
hstore	1.8	1.8
hstore_plperl	1	1
plcoffee	2.3.15	2.3.15
plls	2.3.15	2.3.15
timescaledb	2.6.0	2.6.0
pipelinedb	Unsupported	Unsupported
rdkit	4.0.1	4.0.1
imgsmr	1	1
zhparser	2.2	2.2
intagg	1.1	1.1
intarray	1.5	1.5
isn	1.2	1.2
xml2	1.1	1.1
jsonbx	Unsupported	Unsupported
dict_int	1	1
dict_xsyn	1	1

citext	1.6	1.6
ltree	1.2	1.2
postgres_fdw	1.1	1.1
orafce	Unsupported	Unsupported
chkpass	1	1
bloom	1	1
btree_gin	1.3	1.3
btree_gist	1.6	1.6
roaringbitmap	0.5	0.5
rum	1.3	1.3
cube	1.5	1.5
decoderbufs	0.1.0	0.1.0
pg_bigm	1.2	1.2
fuzzystrmatch	1.1	1.1
hll	2.16	2.16
pg_trgm	1.6	1.6
pg_hashids	1.3	1.3
pgcrypto	1.3	1.3
cos_fdw	1	Unsupported
topn	2.4.0	2.4.0

TencentDB for PostgreSQL 15 supported extensions

Last updated : 2024-03-20 14:48:19

This article introduces the extensions supported in the latest version of TencentDB for PostgreSQL 15. Please [Upgrading Kernel Minor Version of Database Proxy](#) to use it.

Extension	Version of the extension
address_standardizer	3.3.2
address_standardizer_data_us	3.3.2
autoinc	1.0
bloom	1.0
bool_plperl	1.0
btree_gin	1.3
btree_gist	1.7
chkpass	1.0
citext	1.6
cos_fdw	1.0
cube	1.5
dblink	1.2
decoderbufs	0.1.0
dict_int	1.0
dict_xsyn	1.0
earthdistance	1.1
fuzzystrmatch	1.1
hll	2.16
hstore	1.8

hstore_plperl	1.0
imgsmr	1.0
insert_username	1.0
intagg	1.1
intarray	1.5
isn	1.2
jsonb_plperl	1.0
lo	1.1
ltree	1.2
moddatetime	1.0
mysql_fdw	1.1
old_snapshot	1.0
pg_bigm	1.2
pg_buffercache	1.3
pg_cron	1.4
pg_freespacemap	1.2
pg_hashids	1.3
pg_hint_plan	1.4
pg_prewarm	1.2
pg_squeeze	1.5.2
pg_stat_log	1.0
pg_stat_statements	1.10
pg_surgery	1.0
pg_trgm	1.6
pgagent	4.0

pgcrypto	1.3
pgrouting	3.2.2
pgrowlocks	1.2
pgstattuple	1.5
pgvector	0.4.2
plcoffee	2.3.15
plperl	1.0
plpgsql	1.0
pltcl	1.0
postgis	3.3.2
postgis_raster	3.3.2
postgis_sfcgal	3.3.2
postgis_tiger_geocoder	3.3.2
postgis_topology	3.3.2
postgres_fdw	1.1
rdkit	4.0.1
refint	1.0
roaringbitmap	0.5
rum	1.3
seg	1.4
sslinfo	1.2
tablefunc	1.0
tcn	1.0
tencentdb_failover_slot	1.0
tencentdb_system_stat	1.0

topn	2.4.0
tsm_system_rows	1.0
tsm_system_time	1.0
unaccent	1.1
uuid-osp	1.1
wal2json	2.5
xml2	1.1
zhparser	2.2

Plugins supported by PostgreSQL 16

Last updated : 2024-03-20 14:49:54

This document introduces the extensions supported by the latest version of TencentDB for PostgreSQL 16. If you need them, please [Upgrading Kernel Minor Version of Database Proxy](#) to use.

Extension	Version of the extension
address_standardizer	3.4.0
address_standardizer_data_us	3.4.0
autoinc	1.0
bloom	1.0
bool_plperl	1.0
btree_gin	1.3
btree_gist	1.7
chkpss	1.0
citext	1.6
cos_fdw	1.0
cube	1.5
dblink	1.2
decoderbufs	0.1.0
dict_int	1.0
dict_xsyn	1.0
earthdistance	1.1
fuzzystrmatch	1.1
hll	2.18
hstore	1.8
hstore_plperl	1.0

imgsmr	1.0
insert_username	1.0
intagg	1.1
intarray	1.5
isn	1.2
jsonb_plperl	1.0
lo	1.1
ltree	1.2
moddatetime	1.0
mysql_fdw	1.1
old_snapshot	1.0
pg_bigm	1.2
pg_buffercache	1.4
pg_cron	1.6
pg_freespacemap	1.2
pg_hashids	1.3
pg_hint_plan	1.6.0
pg_prewarm	1.2
pg_squeeze	1.6
pg_stat_log	1.0
pg_stat_statements	1.10
pg_surgery	1.0
pg_similarity	1.0
pg_trgm	1.6
pg_walinspect	1.0

pgagent	4.0
pgcrypto	1.3
pgrouting	3.2.2
pgrowlocks	1.2
pgstattuple	1.5
pgvector	0.5.0
plcoffee	2.3.15
plperl	1.0
plpgsql	1.0
pltcl	1.0
postgis	3.4.0
postgis_raster	3.4.0
postgis_sfcgal	3.4.0
postgis_tiger_geocoder	3.4.0
postgis_topology	3.4.0
postgres_fdw	1.1
rdkit	4.0.1
refint	1.0
roaringbitmap	0.5
rum	1.3
seg	1.4
ssinfo	1.2
tablefunc	1.0
tcn	1.0
tencentdb_failover_slot	1.0

tencentdb_system_stat	1.0
topn	2.4.0
tsm_system_rows	1.0
tsm_system_time	1.0
unaccent	1.1
uuid-oss	1.1
wal2json	2.5
xml2	1.1
zhparser	2.2

pgAgent Extension

Last updated : 2024-03-21 11:30:10

This document describes how to implement automatic job execution in TencentDB for PostgreSQL through the pgAgent feature. We recommend you use the pg_cron extension to schedule jobs.

Overview

If your business needs to perform specified actions in the database at scheduled times, such as clearing redundant data, updating materialized views, performing `VACUUM FULL`, and executing DML, PostgreSQL can help implement with the following features:

The crontab feature of Linux

The pgAgent feature of pgAdmin

pgAgent is an extension in the pgAdmin tool imported in pgAdmin III v1.4. It is mainly used as a PostgreSQL job scheduling agent and capable of running multi-step batch or shell scripts and SQL jobs on complex schedules.

It should be noted that pgAgent requires the support of certain databases, tables, and other objects, so you need to install it first.

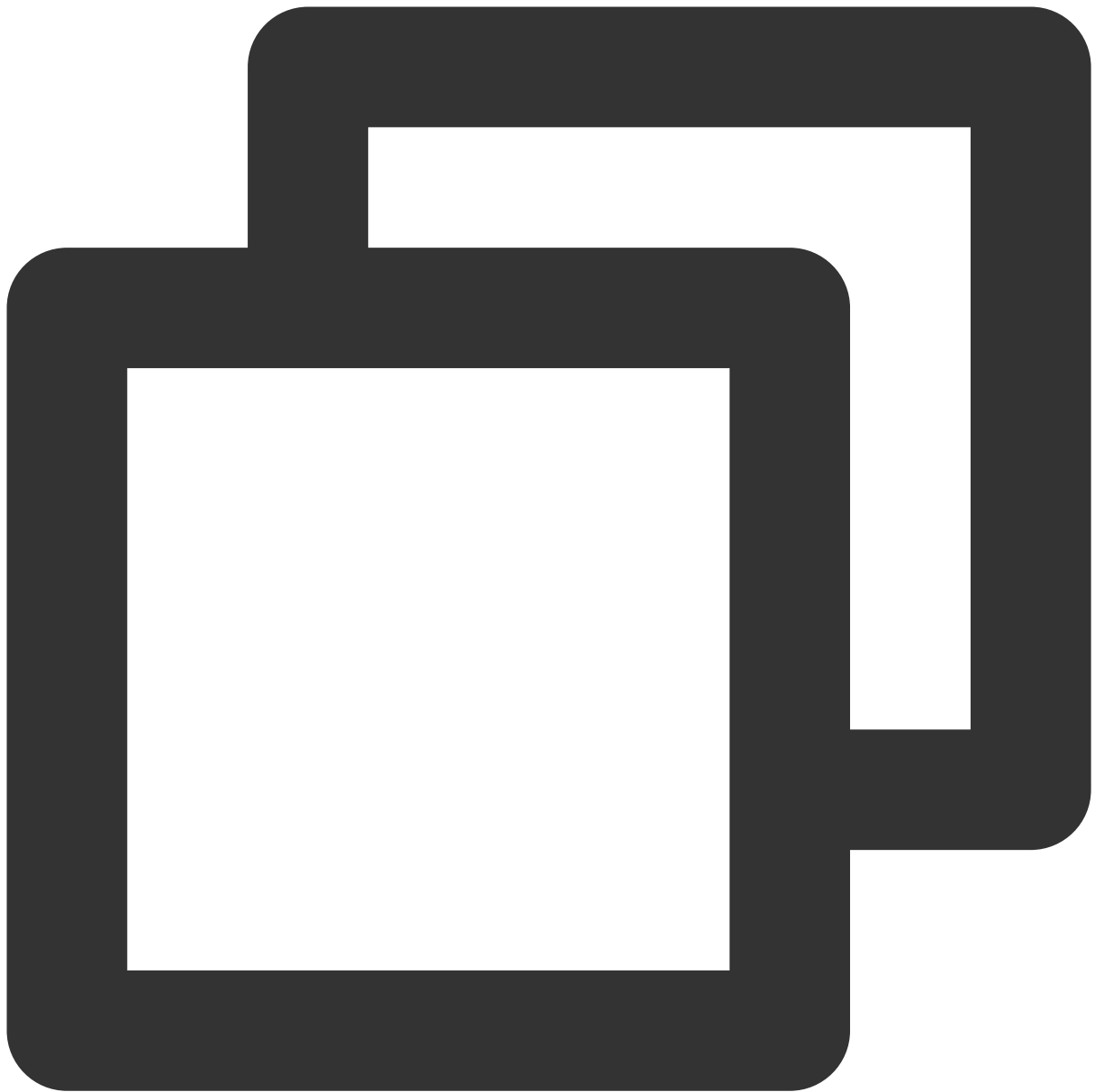
Directions

Configuring pgAgent

1. [Log in to the TencentDB for PostgreSQL instance](#) and create your business database.
2. Run the following statement in the database where you need to enable the pgAgent feature and the `postgres` database:

Note:

You must also create pgAgent in the `postgres` database.



```
psql > create extension pgagent;  
CREATE EXTENSION
```

3. After the configuration is completed, you need to start the job scheduler through the pgAgent tool.

[Log in to the CVM instance](#) (we recommend you put the CVM and TencentDB for PostgreSQL instances in the same VPC). Choose the pgAgent version according to the actual database version. This document uses v11.8 as an example to install pgagent available [here](#).

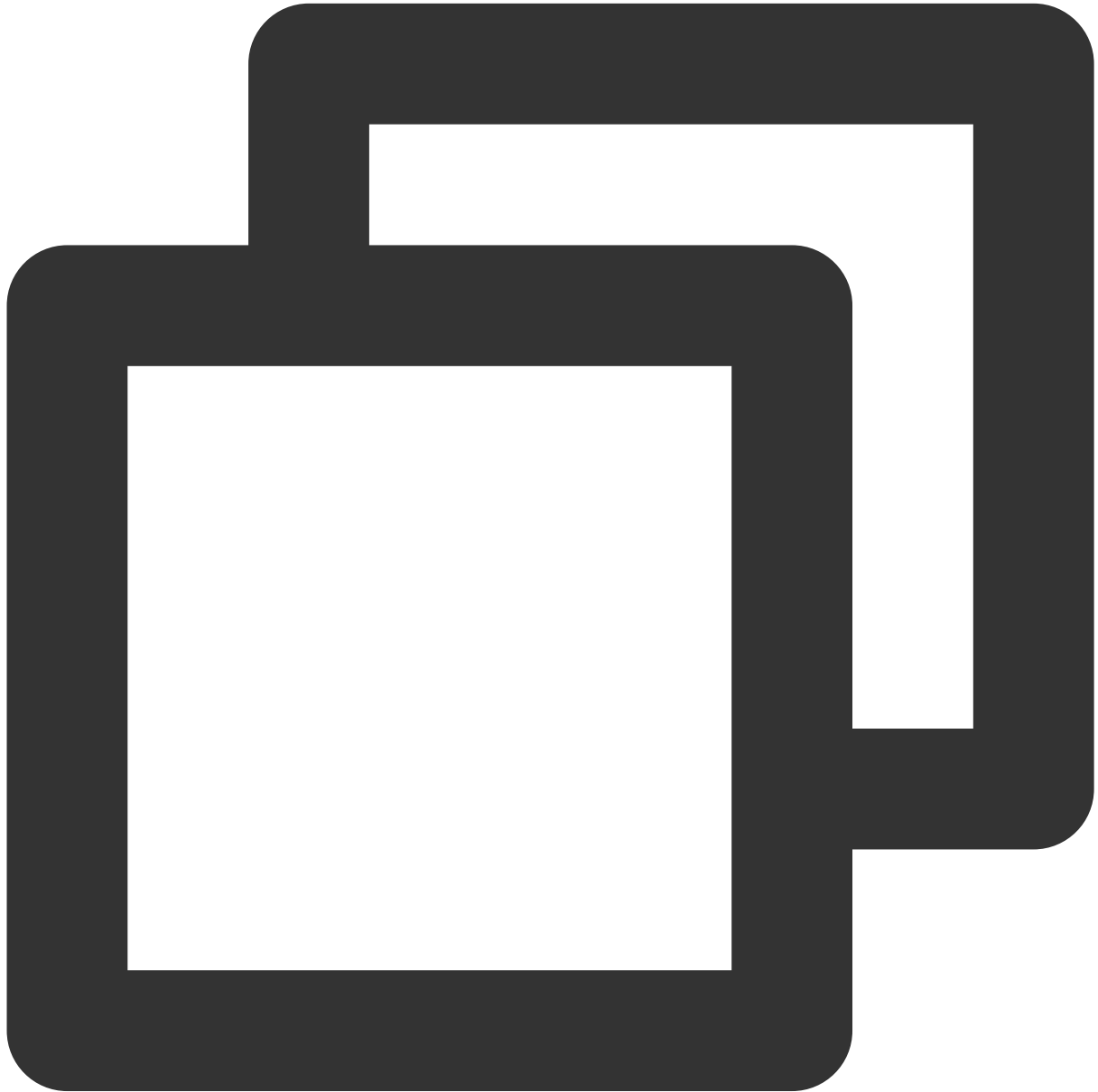
4. After pgAgent is installed, run the following statement to start the job scheduler:

Note:

Use the command based on the actually installed version of pgAgent. For example, if v10 is installed, the command should be `pgagent_10` .

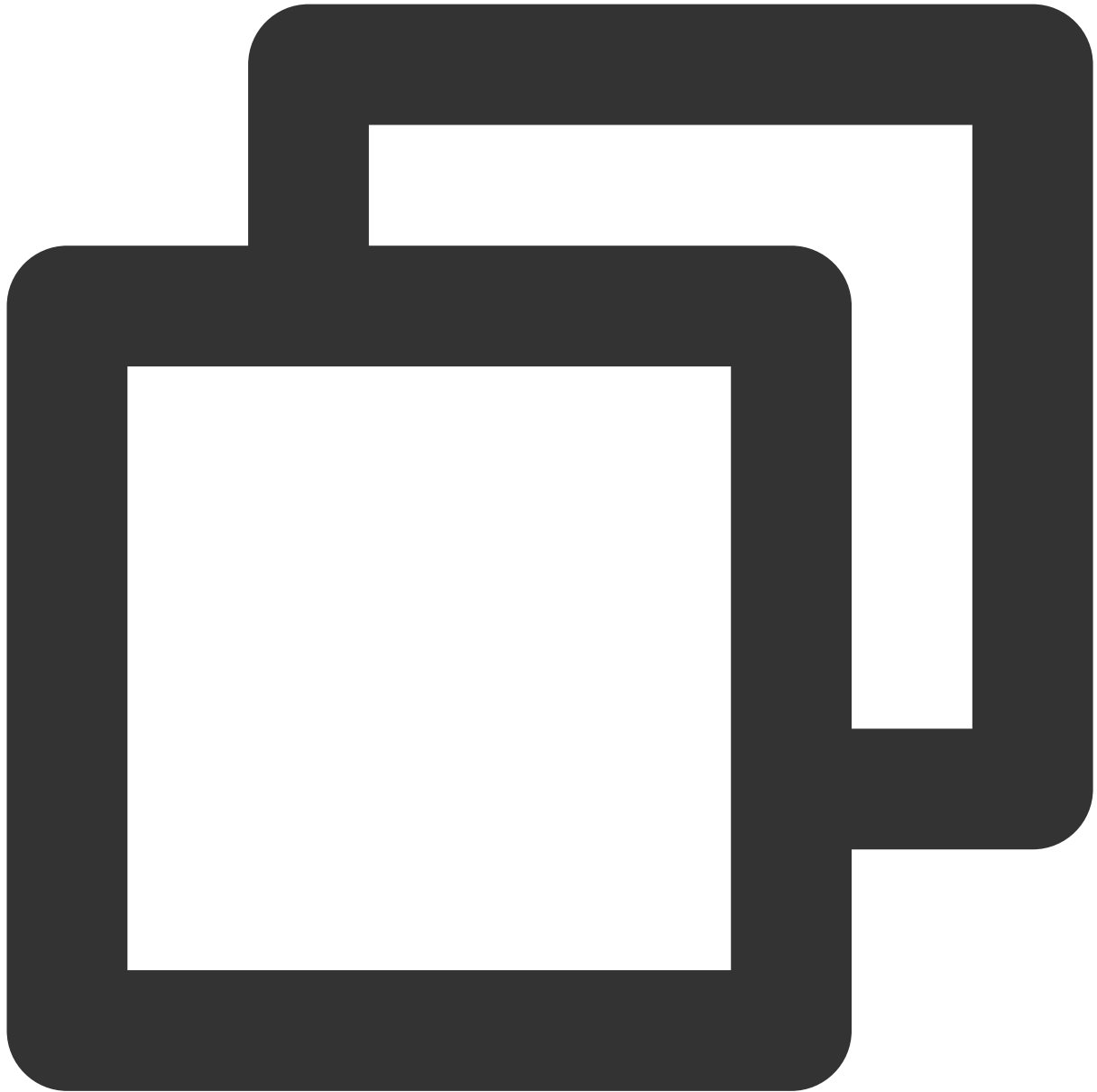
Note that `dbname` must be `postgres` rather than the name of the database that needs to run the scheduler; otherwise, the job configuration items will not be displayed on the pgAdmin page.

When the connection is executed, if the error "ERROR: Unsupported schema version" is reported, please [Submit a Ticket](#) for assistance.



```
pgagent_11 hostaddr=IP dbname=postgres user=username port=port password=password
```

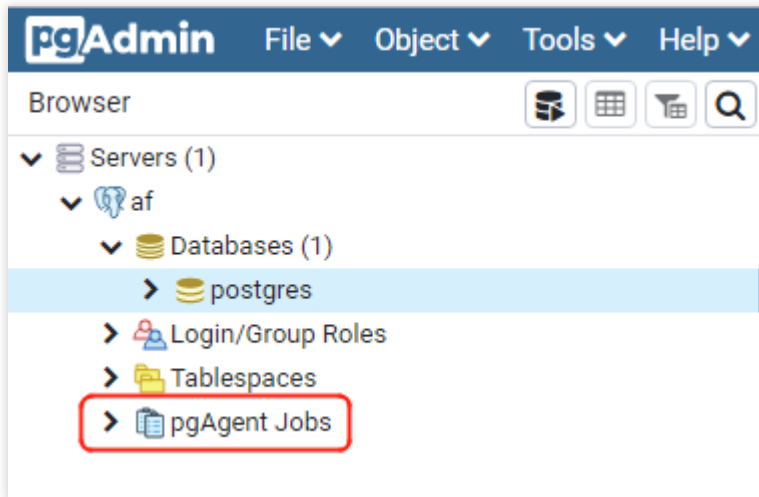
5. After successful execution, there is no echo, but you can use the following command to check whether the process is started successfully:



```
Run this statement, and if there is a `pgagent` process, it has been started successfully.
# ps -ef |grep pgagent
root      158553      1  0 Oct30 ?          00:00:15 pgagent_11 hostaddr=IP dbname=p
```

Configuring pgAgent Jobs through pgAdmin

1. Log in to the [TencentDB for PostgreSQL console](#), click an instance ID in the instance list to enter the instance details page, and enable the public network access.
2. Open pgAdmin 4 and access your TencentDB for PostgreSQL instance at the public network access address. At this time, you can see pgAgent Jobs on the page.



3. On the pgAdmin page, right-click and select **pgAgent Jobs** > **Create** > **Create Jobs** to create a scheduled job.
4. On the **General** page, configure the basic job information.

test

GeneralStepsSchedulesSQL

Name

test

Enabled?

Yes

Job class

Routine Maintenance

Please select a class to categorize the job. This option will not affect the

Host agent

Enter the hostname of a machine running pgAgent if you wish to ensure
Leave blank if any host may run the job.

Comment



i?

✕ C

5. Enter the **Steps** tab and configure the job that needs to be executed at the scheduled time. To do so, click **+** in the top-right corner to add a step, name it, and then configure the SQL statement to be executed on the **Code** tab.



test


General Steps Schedules SQL

	Name	Enabled?	Kind
 	test	<input checked="" type="checkbox"/> True	SQL

General Code


1 insert into t1 values(1);



 C

6. Enter the **Schedules** page and configure the scheduling information for job execution:

7. On the General tab below, configure the effective time of the job.



 **test**

General

Steps

Schedules

SQL

	Name	Enabled?	Start	E
 	test	<div>True</div>	2020-11-03 17:21:36 +08:00	

General

Repeat

Exceptions

Name

test

Enabled?

Yes


Start


2020-11-03 17:21:36 +08:00


End

2020-11-28 17:21:34 +08:00

Comment





 C

8. On the **Repeat** tab below, configure a crontab-style schedule.

The screenshot shows the 'test' configuration window with the 'Schedules' tab selected. The 'Repeat' sub-tab is active, displaying instructions on how to use cron-style format for scheduling. The 'Days' section includes 'Week Days', 'Month Days', and 'Months' with corresponding selection buttons. The 'Times' section includes 'Hours' and 'Minutes' with input fields; the 'Hours' field currently shows 'x 01'. The bottom of the window features an information icon, a help icon, a 'Cancel' button, and a save icon.

test

General Steps **Schedules** SQL

General **Repeat** Exceptions

Schedules are specified using a **cron-style** format.
For each selected time or date element, the schedule will execute.
e.g. To execute at 5 minutes past every hour, simply select '05' in the Minutes list box.
Values from more than one field may be specified in order to further control the schedule.
e.g. To execute at 12:05 and 14:05 every Monday and Thursday, you would click minute 05, hours 12 and 14, and week
For additional flexibility, the Month Days check list includes an extra Last Day option. This matches the last day of the

Days

Week Days

Month Days

Months

Times

Hours

Minutes

i **?**

9. After configuring the execution time, you can also configure the time when the job should not be executed on the **Exceptions** tab.

10. Click **Save** and this job will be automatically executed based on the configuration.

postgres_fdw Extension for Cross-database Access

Last updated : 2024-01-24 11:16:51

TencentDB for PostgreSQL provides extensions for accessing external data sources, including other databases in the same instance and other instances. The cross-database access extensions include homogeneous extensions (dblink and postgresql_fdw) and heterogeneous extensions (mysql_fdw and cos_fdw). You can enable cross-database access by following steps below:

1. Install the extensions by running `CREATE EXTENSION` .
2. Create a foreign server object and create link maps for each remote database that needs to be connected.
3. Use the corresponding command to access external tables to get the data.

As the cross-database access extensions can directly access across instances or perform cross-database access in the same instance, TencentDB for PostgreSQL optimizes access control over the creation of foreign server objects and implements categorized management based on the environment of the target instance. Auxiliary parameters are added to the open-source edition to verify user identity and adjust network policies. For more information, see [Auxiliary Parameters](#).

Auxiliary Parameters

host

This parameter is required for cross-instance access. IP address of the target instance

port

This parameter is required for cross-instance access. Port of the target instance.

instanceid

Instance ID

This parameter is required for access across TencentDB for PostgreSQL instances. It is in the format of `postgres-xxxxxx` or `pgro-xxxxxx` and can be viewed in the [console](#).

If the target instance is in a CVM instance, this parameter is the ID of the CVM instance in the format of `ins-xxxxxx` .

dbname

Name of the database in the remote PostgreSQL service to be accessed. For cross-database access in the same instance, you only need to configure this parameter and can leave other parameters empty.

access_type

This parameter is optional. Target instance types:

The target instance is a TencentDB for PostgreSQL or TencentDB for MySQL instance; if no other types are explicitly specified, this will be the default type.

The target instance is in a CVM instance.

The target instance is a self-built instance with public IP in Tencent Cloud.

The target instance is a Tencent Cloud VPN-based instance.

The target instance is a self-built VPN-based instance.

The target instance is a Direct Connect-based instance.

uin

This parameter is optional. ID of the account to which the instance belongs, which is used to verify user permissions and can be viewed in [Account Info](#).

own_uin

This parameter is optional. ID of the root account to which the instance belongs, which is also needed for verifying user permissions.

vpcid

This parameter is optional. VPC ID. It is required if the target instance is in a CVM instance in a VPC. It can be viewed in the [VPC console](#).

subnetid

This parameter is optional. VPC subnet ID. It is required if the target instance is in a CVM instance in a VPC. It can be viewed in the [VPC console](#).

dcgid

This parameter is optional. Direct Connect connection ID. It is required if the target instance is connected to the network over Direct Connect.

vpngwid

This parameter is optional. VPN gateway ID. It is required if the target instance is connected to the network over VPN.

region

This parameter is optional. It indicates the region where the target instance resides; for example, "ap-guangzhou" represents the Guangzhou region. It is required for cross-region access.

Sample for Using `postgres_fdw`

The `postgres_fdw` extension can be used to access data from other databases in the same instance or other instances.

Step 1. Prepare

1. Create test data in the instance.



```
postgres=>create role user1 with LOGIN  CREATEDB PASSWORD 'password1';  
postgres=>create database testdb1;  
CREATE DATABASE
```

Note:

If an error occurs during creation, [submit a ticket](#) for assistance.

2. Create test data in the target instance.

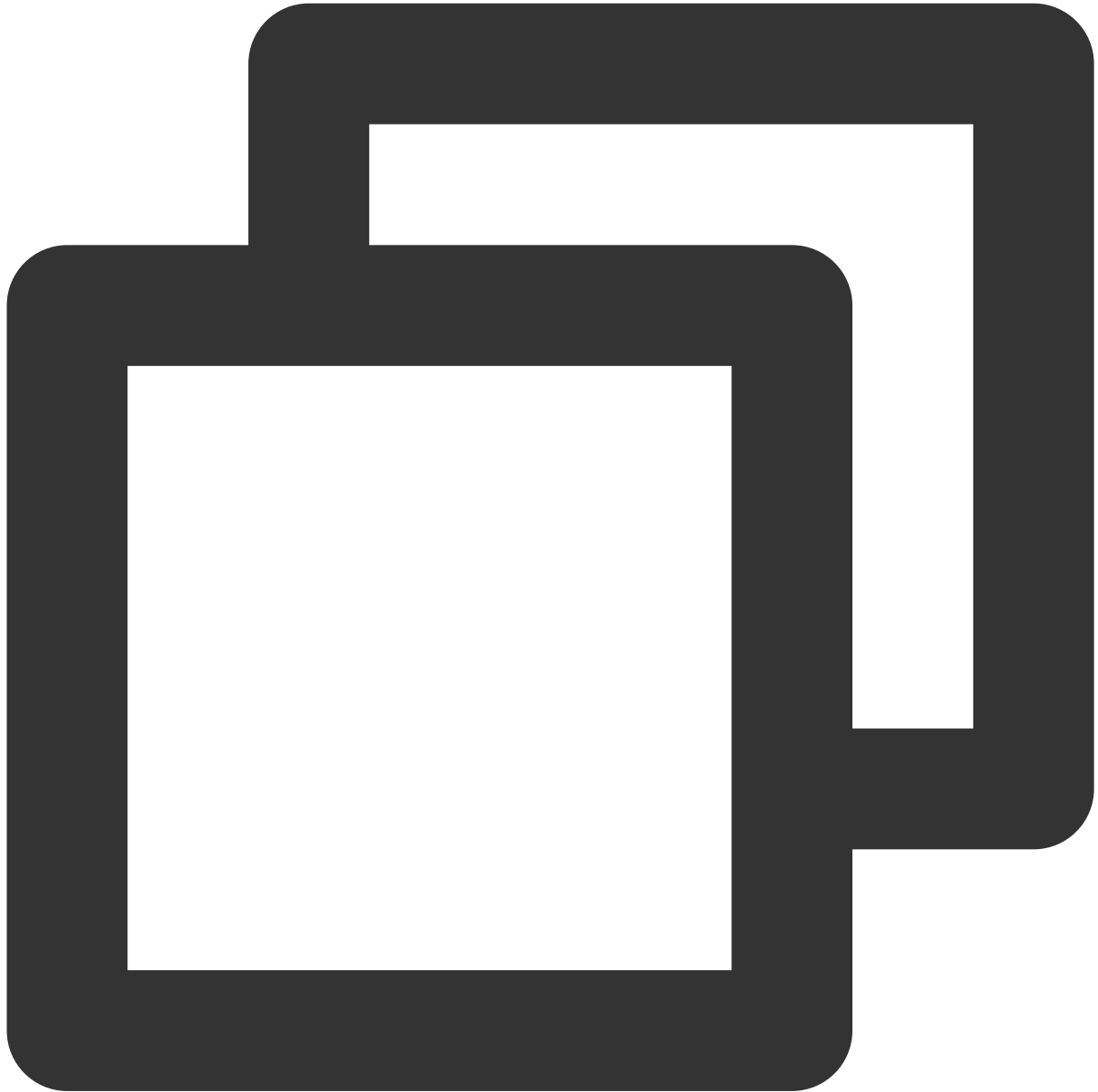


```
postgres=>create role user2 with LOGIN CREATEDB PASSWORD 'password2';
postgres=> create database testdb2;
CREATE DATABASE
postgres=> \c testdb2 user2
You are now connected to database "testdb2" as user "user2".
testdb2=> create table test_table2(id integer);
CREATE TABLE
testdb2=> insert into test_table2 values (1);
INSERT 0 1
```

Step 2. Create the postgres_fdw extension

Note:

If you are prompted that the extension does not exist or you have insufficient permissions during creation, [submit a ticket](#) for assistance.



```
# Create
postgres=> \c testdb1
You are now connected to database "testdb1" as user "user1".
testdb1=> create extension postgres_fdw;
CREATE EXTENSION
```

```
# View
testdb1=> \dx

               List of installed extensions
  Name          | Version | Schema  | Description
-----+-----+-----+-----
 plpgsql        | 1.0     | pg_catalog | PL/pgSQL procedural language
 postgres_fdw   | 1.0     | public   | foreign-data wrapper for remote PostgreSQL
(2 rows)
```

Step 3. Create a server

Note:

Cross-instance access is supported only for kernel v10.17_r1.2, v11.12_r1.2, v12.7_r1.2, v13.3_r1.2, v14.2_r1.0, and later.

Cross-instance access



```
# Access the data of the target instance's `testdb2` from the current instance's `t
testdb1=>create server srv_test1 foreign data wrapper postgres_fdw options (host 'x
CREATE SERVER
```

For cross-database access in the same instance, you only need to enter the `dbname` parameter.



```
# Access the data of `testdb2` from `testdb1` in the current instance
create server srv_test1 foreign data wrapper postgres_fdw options (dbname 'testdb2'
```

The target instance is in a CVM instance in the classic network.



```
testdb1=>create server srv_test foreign data wrapper postgres_fdw options (host '  
CREATE SERVER
```

The target instance is in a CVM instance in a VPC.



```
testdb1=>create server srv_test1 foreign data wrapper postgres_fdw options (host  
CREATE SERVER
```

The target instance is a self-built instance with public IP in Tencent Cloud.



```
testdb1=>create server srv_test1 foreign data wrapper postgres_fdw options (host  
CREATE SERVER
```

The target instance is a Tencent Cloud VPN-based instance.



```
testdb1=>create server srv_test1 foreign data wrapper postgres_fdw options (host
```

The target instance is a self-built VPN-based instance.



```
testdb1=>create server srv_test1 foreign data wrapper postgres_fdw options (host
```

The target instance is a Direct Connect-based instance.

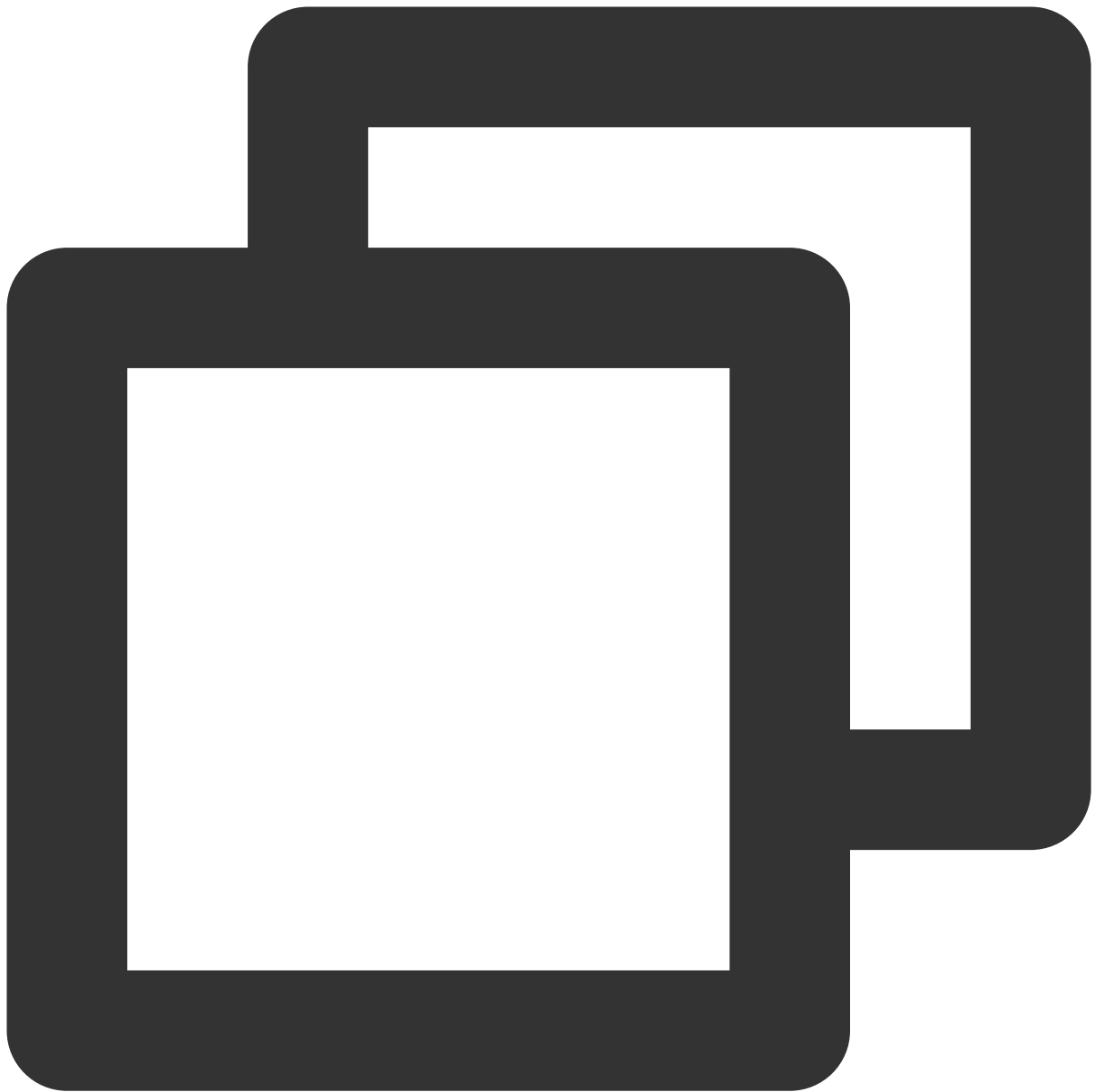


```
testdb1=>create server srv_test1 foreign data wrapper postgres_fdw options (host  
CREATE SERVER
```

Step 4. Create a user mapping

Note:

You can skip this step for cross-database access in the same instance.



```
testdb1=> create user mapping for user1 server srv_test1 options (user 'user2',pass  
CREATE USER MAPPING
```

Step 5. Create a foreign table



```
testdb1=> create foreign table foreign_table1(id integer) server srv_test1 options(  
CREATE FOREIGN TABLE
```

Step 6. Access data from foreign table



```
testdb1=> select * from foreign_table1;
 id
----
  1
(1 row)
```

References

[postgres_fdw Overview](#)

[Create a server on v9.3](#)

[Create a server on v9.5](#)[Create a server on v10](#)[Create a server on v11](#)[Create a server on v12](#)[Create a server on v13](#)[Create a server on v14](#)

Sample for Using `dblink`

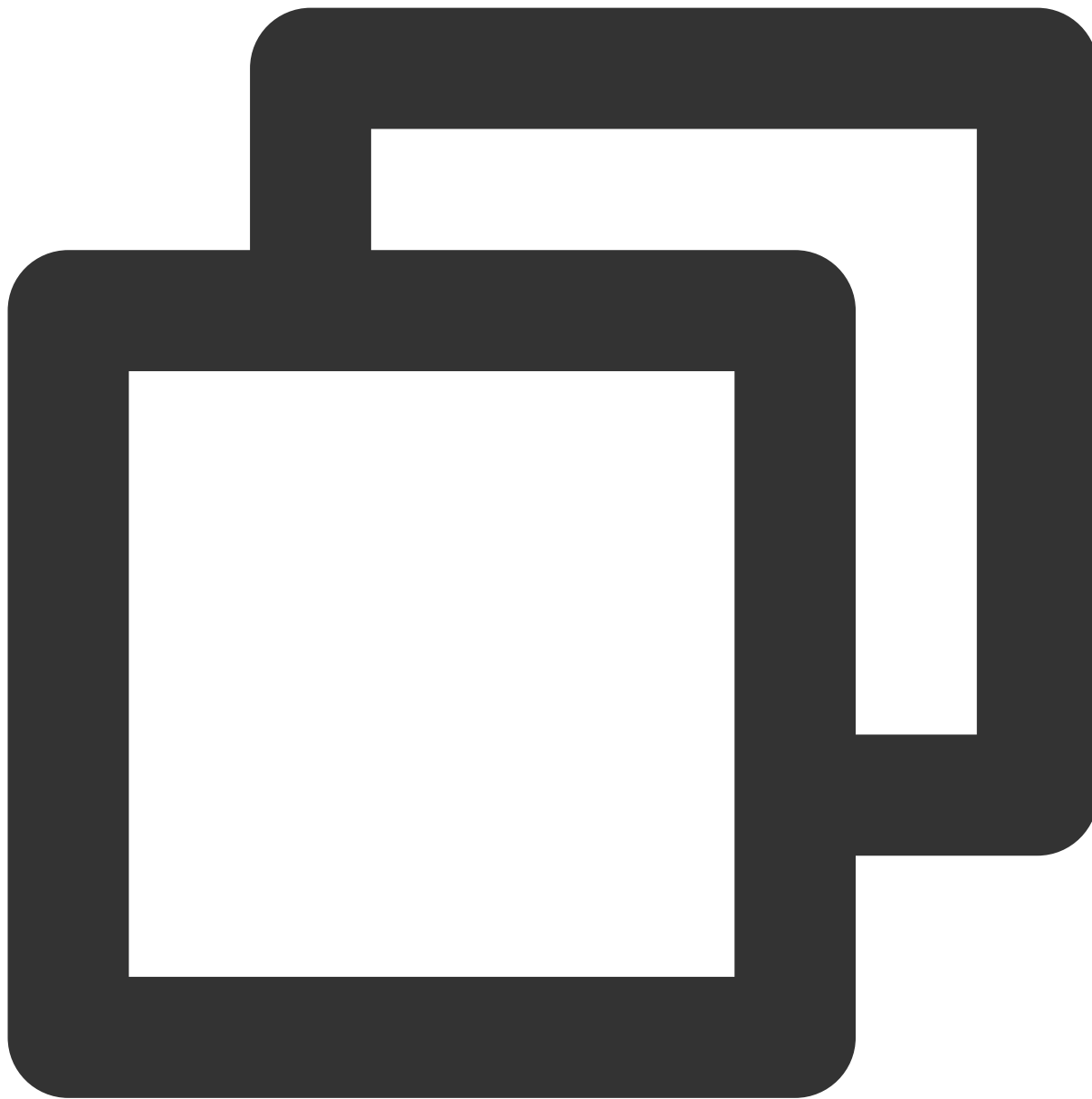
Step 1. Create the `dblink` extention



```
postgres=> create extension dblink;
postgres=> \dx
```

```
                                List of installed extensions
   Name | Version | Schema | Description
-----+-----+-----+-----
dblink  | 1.2     | public | connect to other PostgreSQL databases
pg_stat_log | 1.0    | public | track runtime execution statistics of
pg_stat_statements | 1.6 | public | track execution statistics of all SQL statemen
plpgsql | 1.0     | pg_catalog | PL/pgSQL procedural language
(4 rows)
```

Step 2. Create the `dblink` link



```
select dblink_connect('yunpg1','host=10.10.10.11 port=5432 instanceid=postgres-2123')
 dblink_connect
-----
OK
(1 row)
```

Step 3. Access external data



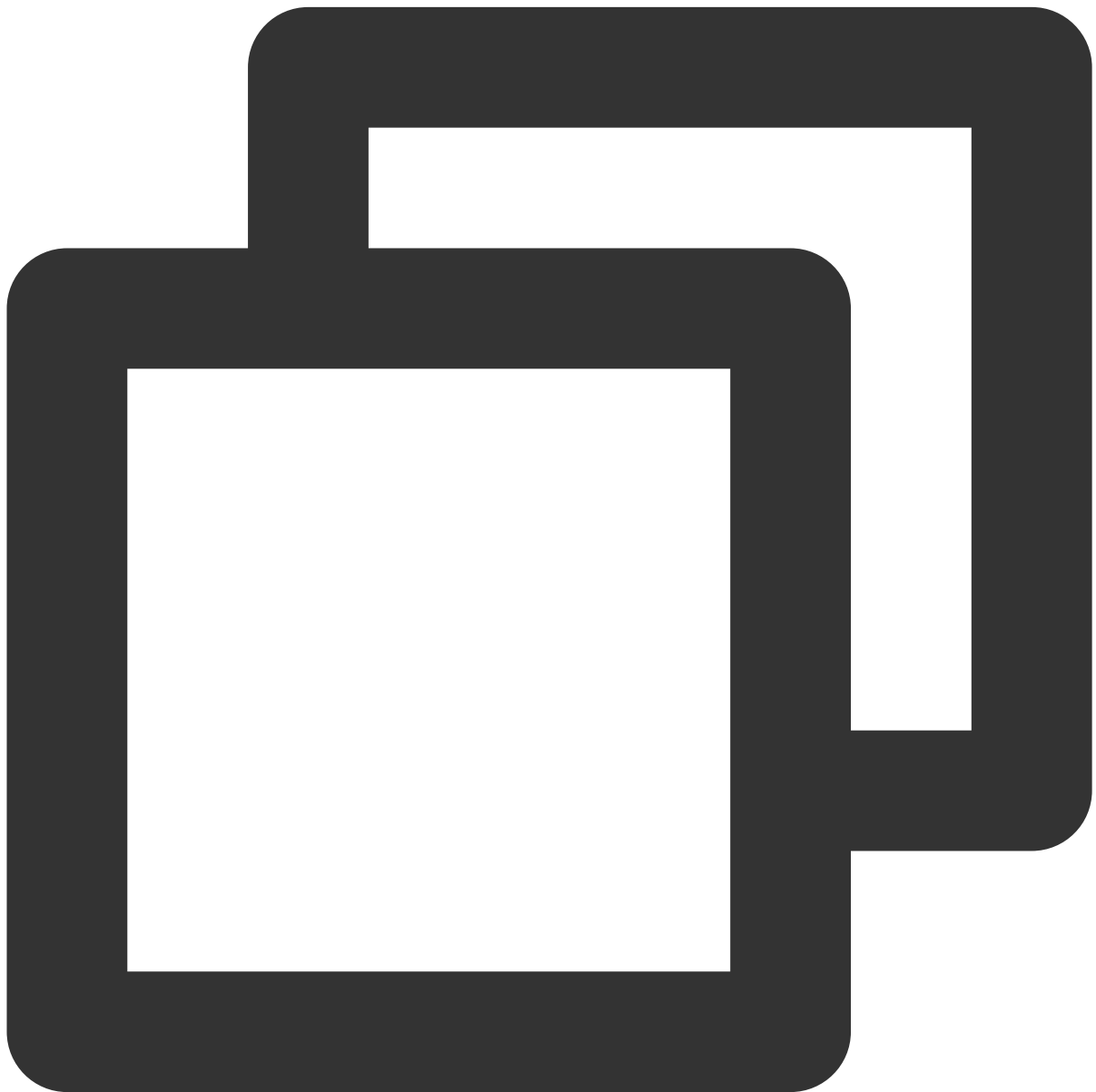
```
postgres=> select * from dblink('yunpg1','select catalog_name,schema_name,schema_ow
  a      |      b      |      c
-----+-----+-----
postgres | pg_toast      | user_00
postgres | pg_temp_1      | user_00
postgres | pg_toast_temp_1 | user_00
postgres | pg_catalog      | user_00
postgres | public          | user_00
postgres | information_schema | user_00
(6 rows)
```

References

[dblink Overview](#)

Sample for Using `mysql_fdw`

Step 1. Create the `mysql_fdw` extension



```
postgres=> create extension mysql_fdw;  
CREATE EXTENSION
```

```
postgres=> \dx;
```

List of installed extensions

Name	Version	Schema	Description
dblink	1.2	public	connect to other PostgreSQL databases
mysql_fdw	1.1	public	Foreign data wrapper for querying a MySQL se
pg_stat_log	1.0	public	track runtime execution statistics of
pg_stat_statements	1.9	public	track planning and execution statistic
plpgsql	1.0	pg_catalog	PL/pgSQL procedural language

(5 rows)

Step 2. Create a server



```
postgres=> CREATE SERVER mysql_svr  FOREIGN DATA WRAPPER mysql_fdw OPTIONS (host '1  
CREATE SERVER
```

Step 3. Create a user mapping



```
postgres=> CREATE USER MAPPING FOR PUBLIC SERVER mysql_svr OPTIONS (username 'fdw_u  
CREATE USER MAPPING
```

Step 4. Access external data



```
postgres=> IMPORT FOREIGN SCHEMA hrdb FROM SERVER mysql_svr INTO public;
```

References

[mysql_fdw Overview](#)

Sample for Using `cos_fdw`

For samples of using `cos_fdw` , see [Supporting Tiered Storage Based on cos_fdw Extension](#).

Note

Pay attention to the following for the target instance:

1. The hba in PostgreSQL needs to be modified to allow the created mapped user (e.g., user2) to access via MD5. For more information on how to modify hba, see [PostgreSQL's official documentation](#).
2. If the target instance is not a TencentDB instance and has a hot backup mode configured, after a primary-standby switch, you need to update the server connection address or create a server again.

pg_roaringbitmap Extension for Bitwise Operation

Last updated : 2024-01-24 11:16:51

TencentDB for PostgreSQL provides the `pg_roaringbitmap` extension to use the bitwise operation feature to improve the query performance.

Prerequisites

Your TencentDB for PostgreSQL instance is on v10, 11, 12, 13, 14 or 15.

Background

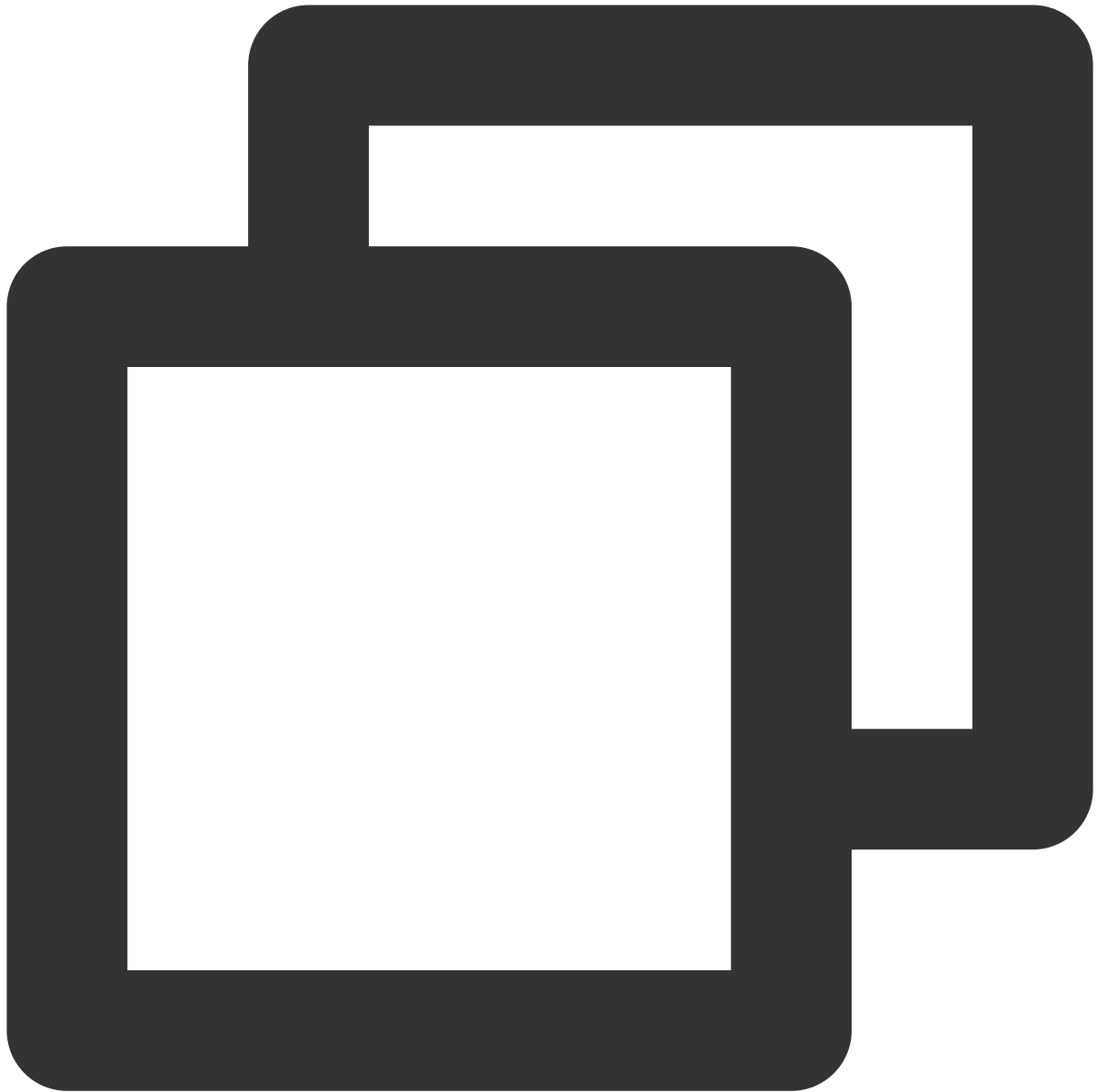
The roaring bitmap algorithm divides a 32-bit INT value into 216 data chunks, each of which corresponds to the higher 16 bits of an integer and uses a container to store the lower 16 bits.

Roaring bitmap stores the containers in a dynamic array as a level-1 index. Containers are in two different structures: array container for sparse chunks and bitmap container for dense chunks. If a container has less than 4,096 integers, the values are stored in an array container; otherwise, the values are stored in a bitmap container.

By using this storage structure, roaring bitmap can quickly search for a specific value. During bitwise operations (AND, OR, and XOR), roaring bitmap provides the corresponding algorithms to efficiently implement operations between two containers, making it powerful in both storage and computing performance.

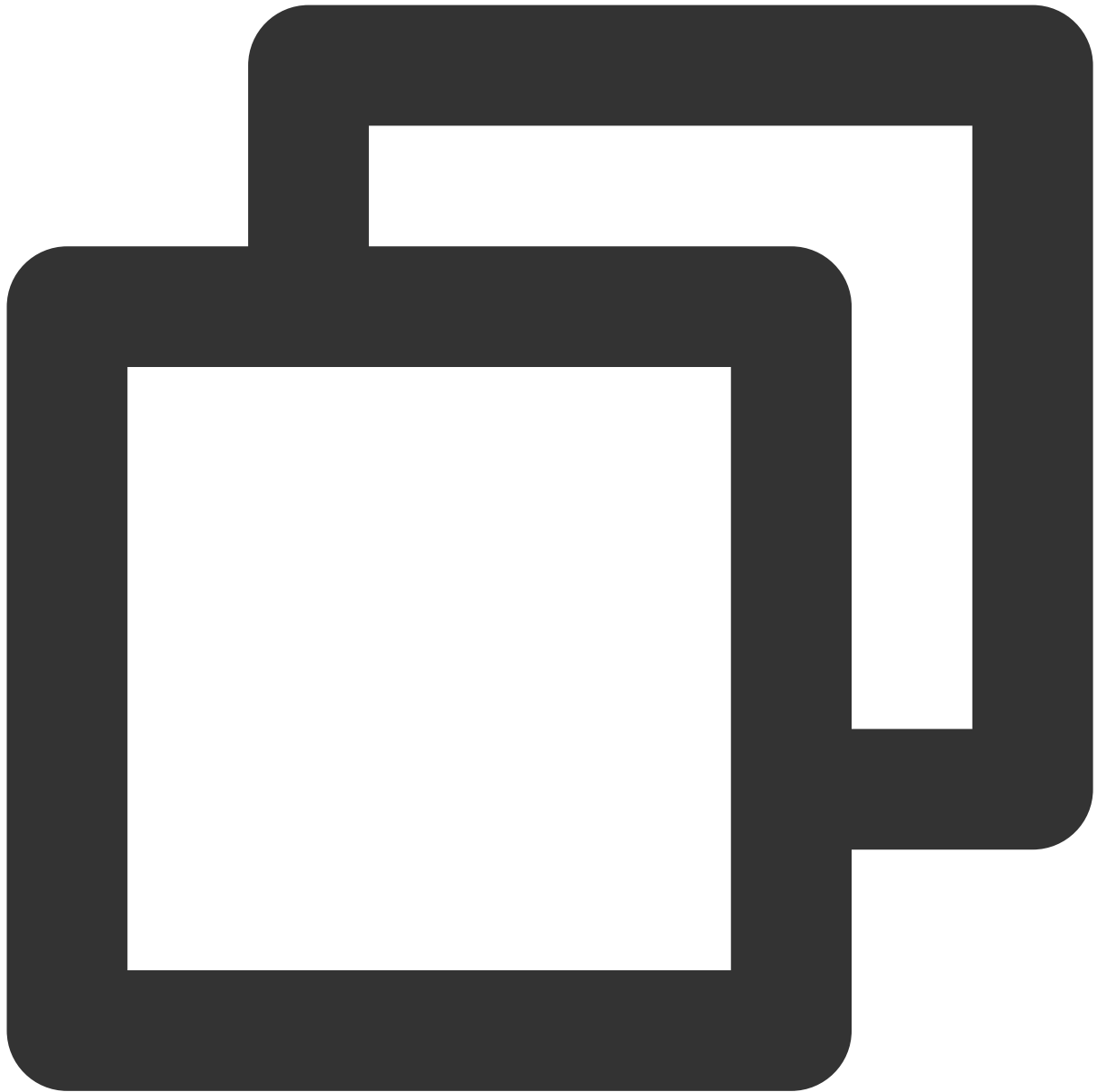
Directions

1. Run the following command to create an extension:



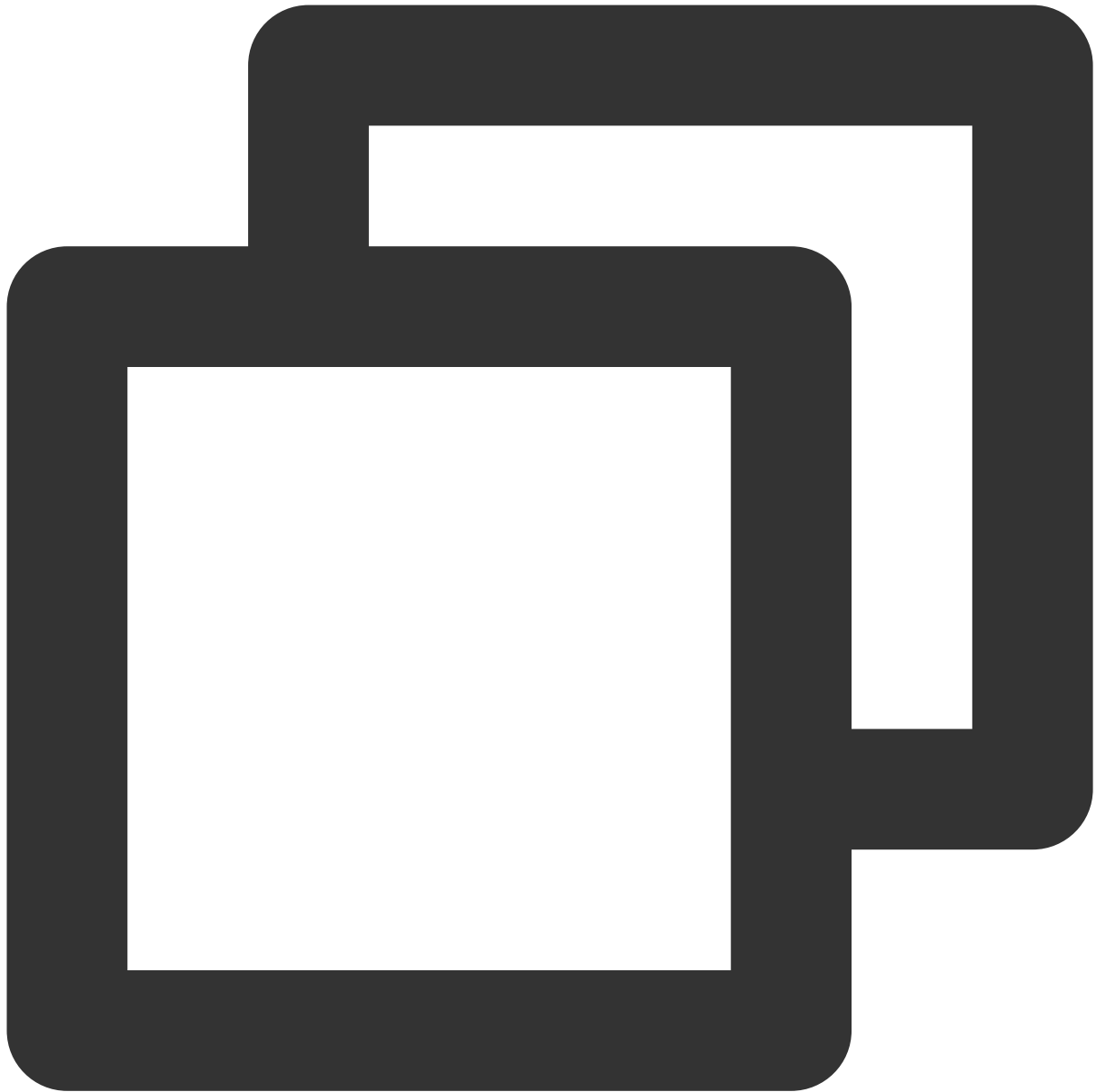
```
CREATE EXTENSION roaringbitmap;
```

2. Run the following command to create a table with data of `roaringbitmap` type:



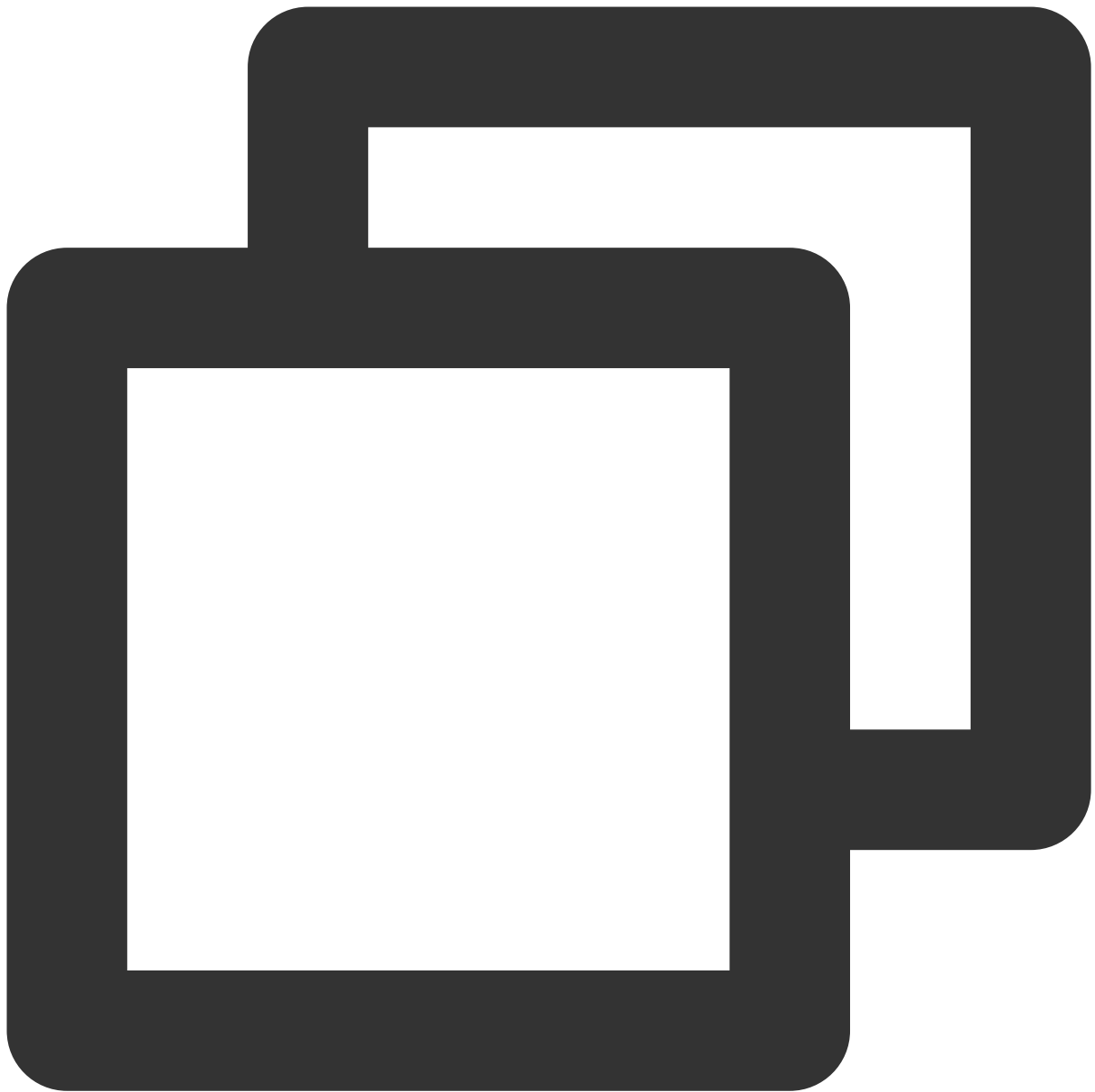
```
CREATE TABLE t1 (id integer, bitmap roaringbitmap);
```

3. Run the following command to use the `rb_build` function to insert the `roaringbitmap` data:



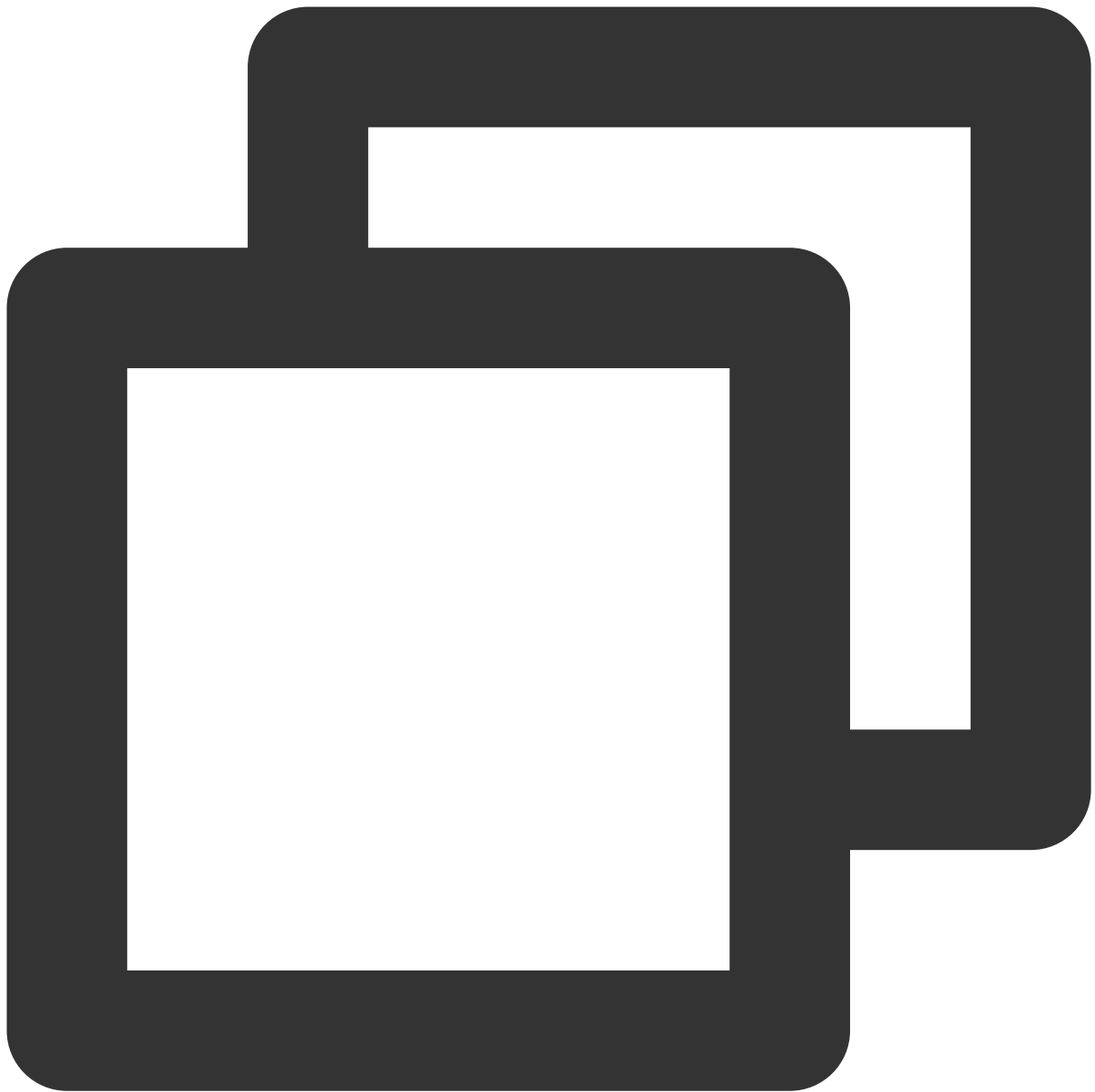
```
-- Set the bit value of the array to 1.  
INSERT INTO t1 SELECT 1,RB_BUILD (ARRAY[1,2,3,4,5,6,7,8,9,200]);  
-- Set the bit values of multiple records to 1 and aggregate the bit values into a  
INSERT INTO t1 SELECT 2,RB_BUILD_AGG(e) FROM GENERATE_SERIES(1,100) e;
```

4. Run the following command to perform bitwise operations (OR, AND, XOR, and ANDNOT):



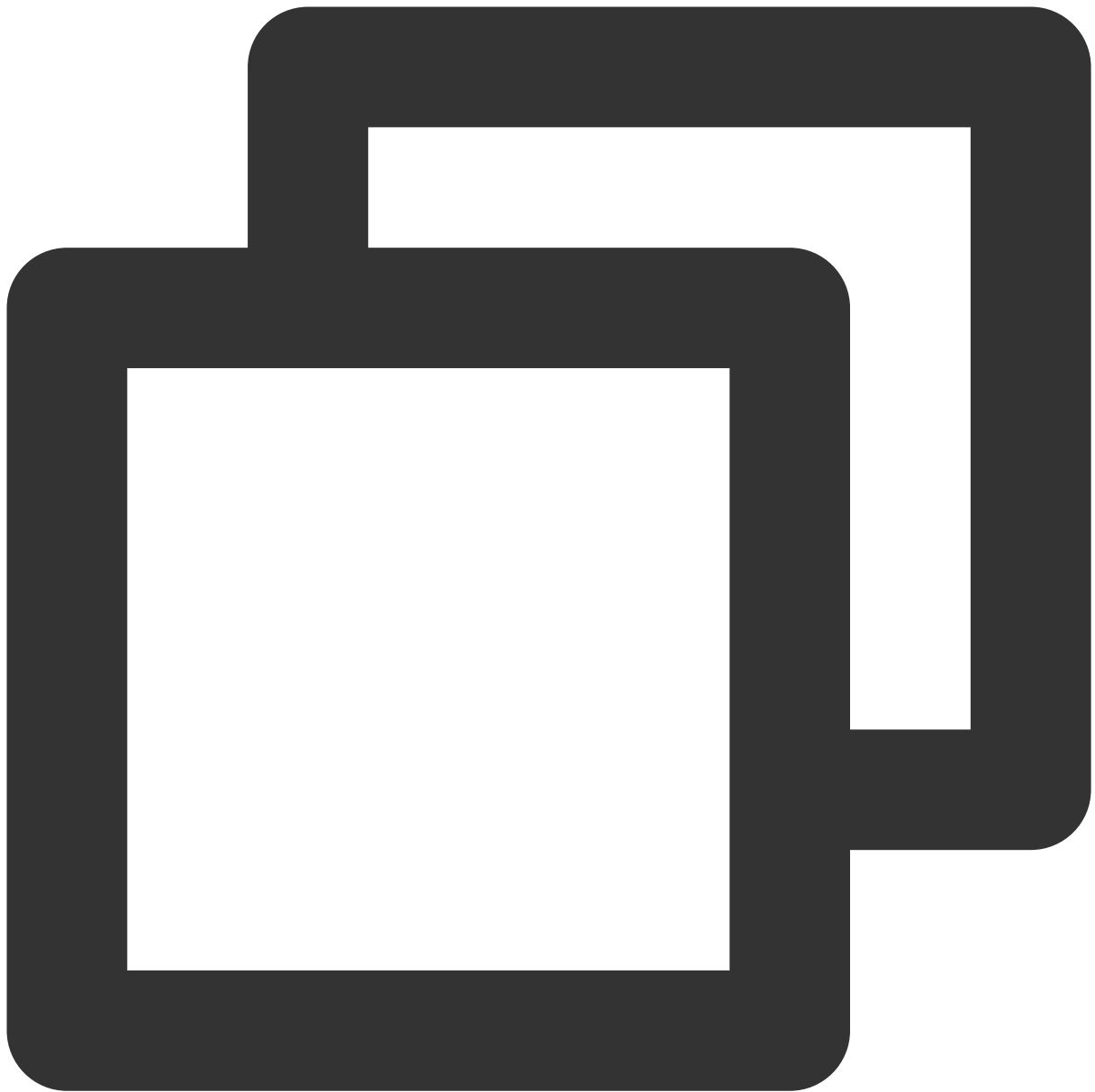
```
-- Set the bit value of the array to 1.  
SELECT RB_OR(a.bitmap,b.bitmap) FROM (SELECT bitmap FROM t1 WHERE id = 1) AS a, (SEL
```

5. Run the following command to perform aggregated bitwise operations (OR, AND, XOR, and BUILD) to generate a new Roaring bitmap:



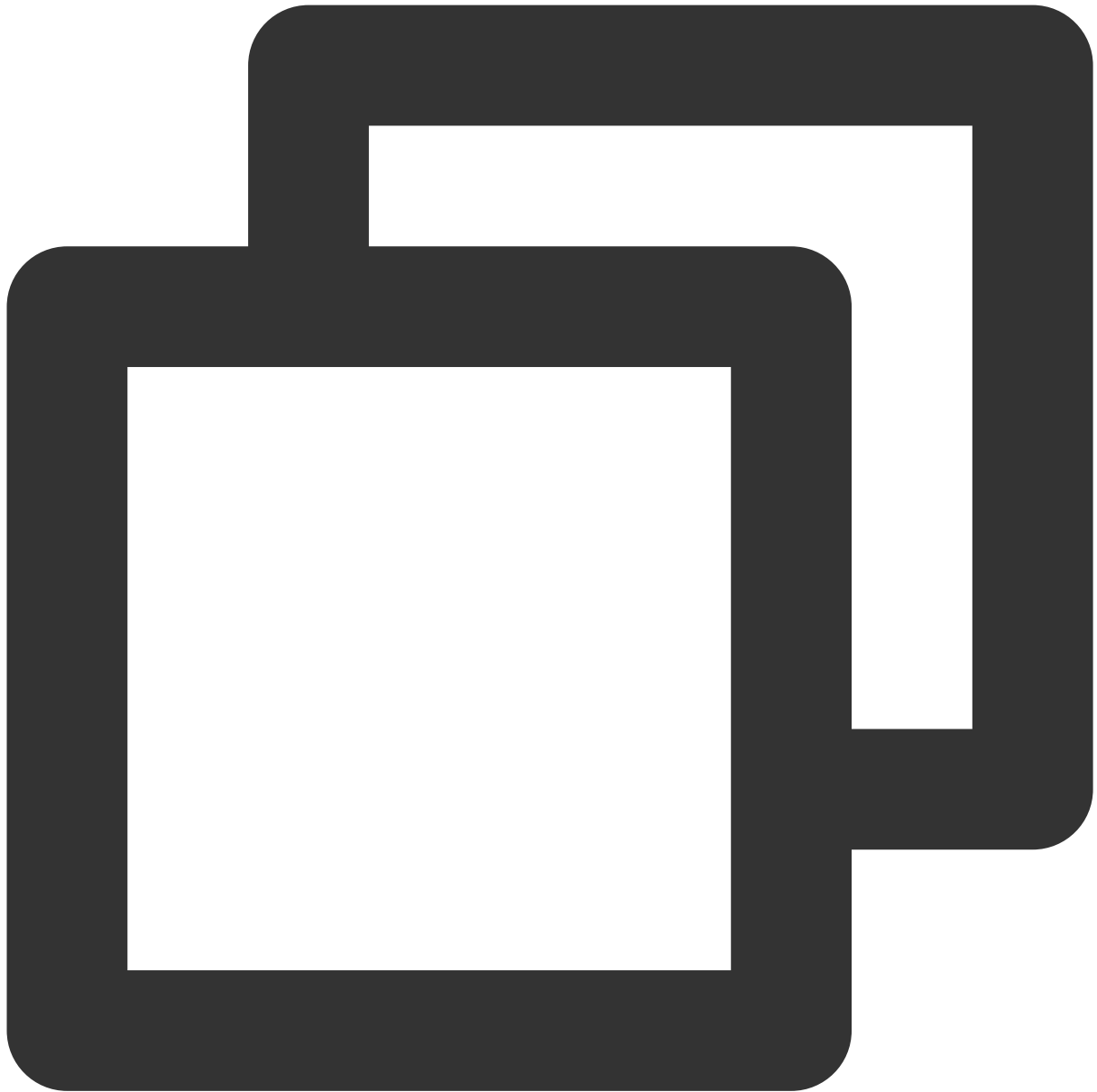
```
SELECT RB_OR_AGG(bitmap) FROM t1;  
SELECT RB_AND_AGG(bitmap) FROM t1;  
SELECT RB_XOR_AGG(bitmap) FROM t1;  
SELECT RB_BUILD_AGG(e) FROM GENERATE_SERIES(1,100) e;
```

6. Run the following command to calculate the cardinality, i.e., number of bits set to 1 in the Roaring bitmap:



```
SELECT RB_CARDINALITY(bitmap) FROM t1;
```

7. Run the following command to return the subscripts of the bits set to 1 in the Roaring bitmap:



```
SELECT RB_ITERATE(bitmap) FROM t1 WHERE id = 1;
```

Feature Function List

Function	Input	Output	Description
rb_build	integer[]	roaringbitmap	Create roaringbitmap from integer array

rb_index	roaringbitmap, integer	bigint	Return the 0-based index of element i in this roaringbitmap, or -1 if it does not exist
rb_cardinality	roaringbitmap	bigint	Return cardinality of the roaringbitmap
rb_and_cardinality	roaringbitmap, roaringbitmap	bigint	Return cardinality of the AND of two roaringbitmaps
rb_xor_cardinality	roaringbitmap, roaringbitmap	bigint	Return cardinality of the XOR of two roaringbitmaps
rb_andnot_cardinality	roaringbitmap, roaringbitmap	bigint	Return cardinality of the ANDNOT of two roaringbitmaps
rb_is_empty	roaringbitmap	boolean	Check if roaringbitmap is empty.
rb_fill	roaringbitmap, range_start bigint, range_end bigint	roaringbitmap	Fill the specified range (not include the range_end)
rb_clear	roaringbitmap, range_start bigint, range_end bigint	roaringbitmap	Clear the specified range (not include the range_end)
rb_flip	roaringbitmap, range_start bigint, range_end bigint	roaringbitmap	Negative the specified range (not include the range_end)
rb_range	roaringbitmap, range_start bigint, range_end bigint	roaringbitmap	Return new set with specified range (not include the range_end)
rb_range_cardinality	roaringbitmap, range_start bigint, range_end bigint	bigint	Return the cardinality of specified range (not include the range_end)
rb_min	roaringbitmap	integer	Return the smallest offset in roaringbitmap. Return NULL if the bitmap is empty
rb_max	roaringbitmap	integer	Return the greatest offset in roaringbitmap. Return NULL if the bitmap is empty
rb_rank	roaringbitmap, integer	bigint	Return the number of elements that are smaller or equal to the specified offset
rb_jaccard_dist	roaringbitmap, roaringbitmap	double precision	Return the jaccard distance (or the Jaccard similarity coefficient) of two bitmaps

rb_select	roaringbitmap,bitset_limit bigint,bitset_offset bigint=0,reverse boolean=false,range_start bigint=0,range_end bigint=4294967296	roaringbitmap	Return subset [bitset_offset,bitset_offset+bitset_limit of bitmap between range [range_start,range_end)
rb_to_array	roaringbitmap	integer[]	Convert roaringbitmap to integer array
rb_iterate	roaringbitmap	SET of integer	Return set of integer from a roaringbitmap data.

Aggregate Function List

Aggregate Function	Input	Output	Description	Example
rb_build_agg	integer	roaringbitmap	Build a roaringbitmap from a integer set	<pre>select rb_build_agg(id (values (1), (2), (3)) t (</pre>
rb_or_agg	roaringbitmap	roaringbitmap	AND Aggregate calculations from a roaringbitmap set	<pre>select rb_or_agg(bitmap from (values (roaringbitmap('{1,2,3} (roaringbitmap('{2,3,4} t(bitmap)</pre>
rb_and_agg	roaringbitmap	roaringbitmap	AND Aggregate calculations from a roaringbitmap set	<pre>select rb_and_agg(bitmap from (values (roaringbitmap('{1,2,3} (roaringbitmap('{2,3,4} t(bitmap)</pre>
rb_xor_agg	roaringbitmap	roaringbitmap	XOR Aggregate calculations from a roaringbitmap set	<pre>select rb_xor_agg(bitmap from (values (roaringbitmap('{1,2,3} (roaringbitmap('{2,3,4} t(bitmap)</pre>
rb_or_cardinality_agg	roaringbitmap	bigint	OR	<pre>select</pre>

			Aggregate calculations from a roaringbitmap set, return cardinality.	<code>rb_or_cardinality_agg(b from (values (roaringbitmap('{1,2,3} (roaringbitmap('{2,3,4} t(bitmap)</code>
<code>rb_and_cardinality_agg</code>	roaringbitmap	bigint	AND Aggregate calculations from a roaringbitmap set, return cardinality	<code>select rb_and_cardinality_agg(from (values (roaringbitmap('{1,2,3} (roaringbitmap('{2,3,4} t(bitmap)</code>
<code>rb_xor_cardinality_agg</code>	roaringbitmap	bigint	XOR Aggregate calculations from a roaringbitmap set, return cardinality	<code>select rb_xor_cardinality_agg(from (values (roaringbitmap('{1,2,3} (roaringbitmap('{2,3,4} t(bitmap)</code>

pg_cron Extension for Job Scheduling

Last updated : 2024-01-24 11:16:51

pg_cron is a simple cron-based job scheduler for PostgreSQL 10 and later. It runs in the database as an extension and uses common cron syntax to schedule and execute database commands directly in the database.

This document describes how to use the pg_cron extension of PostgreSQL.

Enabling pg_cron Extension

1. To use pg_cron, [submit a ticket](#) to add it to the `shared_preload_libraries` parameter of your database. Modifying this parameter requires an instance restart; therefore, do so during off-peak hours.
2. After the parameter is modified, enter the `postgres` database and run the following command with the admin account:



```
CREATE EXTENSION pg_cron;
```

3. Currently, `pg_cron` can execute scheduled jobs only in the `postgres` database. You can run scheduled jobs in other databases as instructed in [Setting Scheduled Job for Other Databases](#).

4. By default, after `pg_cron` is created, its configuration data and job execution can be configured only by the admin. If you want to use another user account to configure or run `pg_cron`, grant the account the cron metadatabase permission by running the following command:



```
postgres=> GRANT USAGE ON SCHEMA cron TO other-user;
```

This permission grants another user the permission to access cron metadata to schedule and cancel cron jobs. To successfully execute a cron job, the user needs the permission to access objects in the job. If the user doesn't have such permission, the job will fail, and an error will be displayed in `postgresql.log`.

In the following sample code, the user doesn't have the permission to access the `pgbench_accounts` table:



```
2020-12-08 16:41:00 UTC::@[30647]:ERROR: permission denied for table pgbench_accou
2020-12-08 16:41:00 UTC::@[30647]:STATEMENT: update pgbench_accounts set abalance
2020-12-08 16:41:00 UTC::@[27071]:LOG: background worker "pg_cron" (PID 30647) exi
```

Below are other messages in the `cron.job_run_details` table:



```
postgres=> select jobid, username, status, return_message, start_time from cron.job
jobid | username | status | return_message |
-----+-----+-----+-----+-----
143 | unprivuser | failed | ERROR: permission denied for table pgbench_accounts | 2
143 | unprivuser | failed | ERROR: permission denied for table pgbench_accounts | 2
143 | unprivuser | failed | ERROR: permission denied for table pgbench_accounts | 2
143 | unprivuser | failed | ERROR: permission denied for table pgbench_accounts | 2
143 | unprivuser | failed | ERROR: permission denied for table pgbench_accounts | 2
(5 rows)
```

pg_cron Scheduled Job Configuration

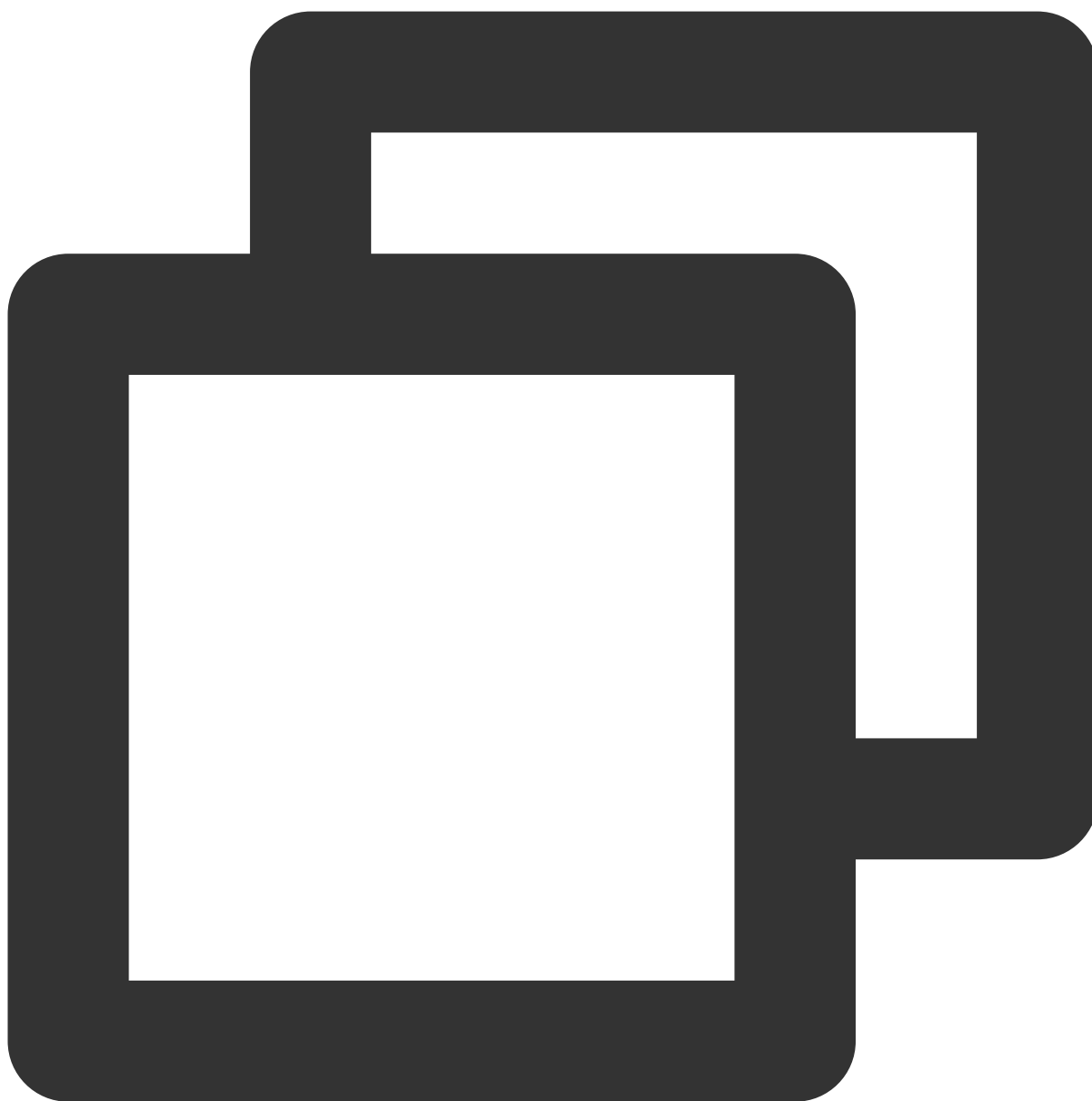
pg_cron provides three main operations: adding and deleting jobs and viewing job information.

cron.schedule() function

This function is used to schedule a cron job. Jobs are scheduled in the `postgres` database initially by default. This function returns a bigint value indicating the job identifier. To schedule a job in other databases in a TencentDB for PostgreSQL instance, refer to the example in [Setting Scheduled Job for Other Databases](#).

This function has two syntax formats:

Syntax



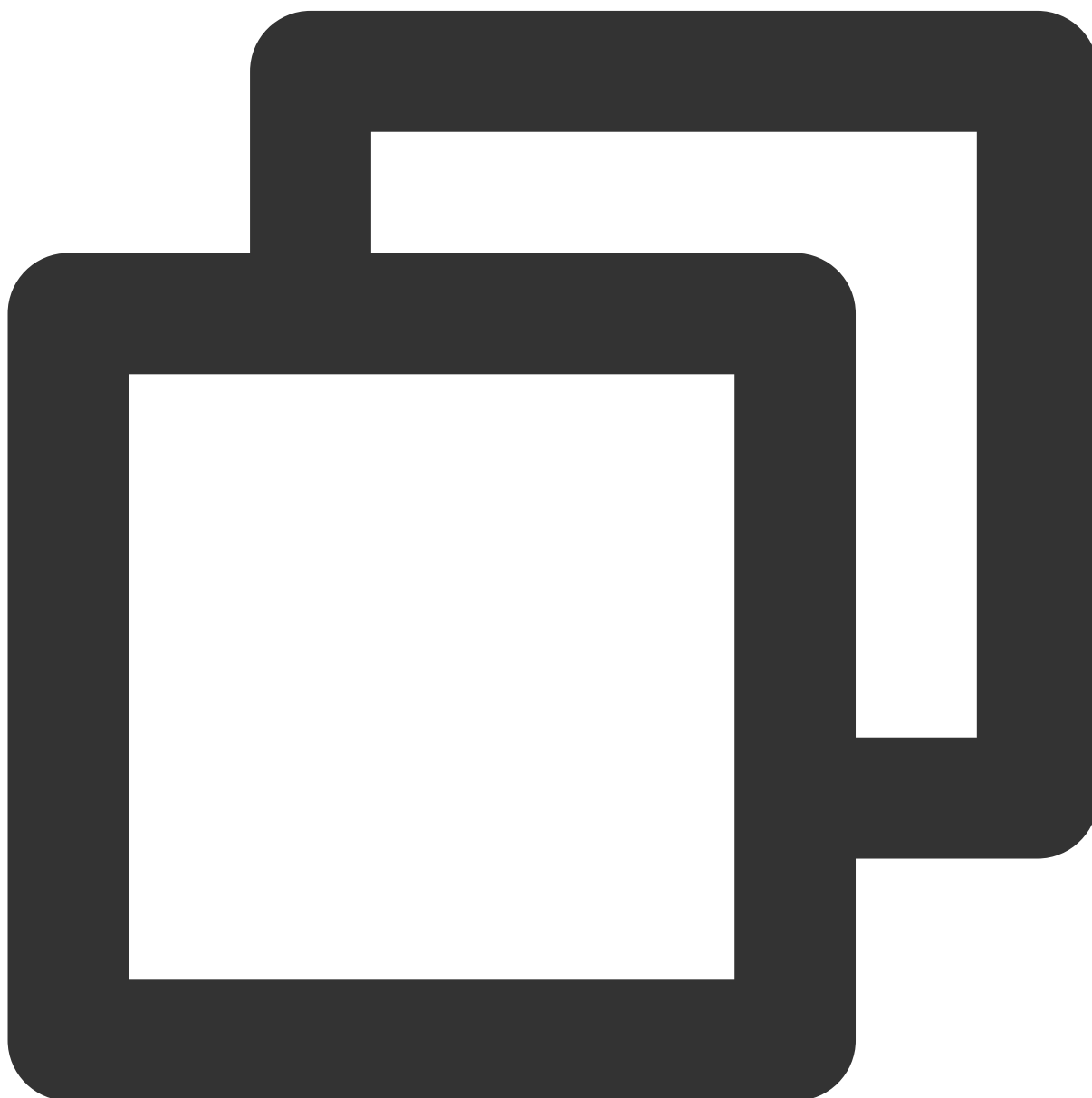
```
cron.schedule (job_name,  
               schedule,  
               command  
);  
  
cron.schedule (schedule,  
               command  
);
```

Parameters

--	--

Parameter	Description
job_name	cron job name, which can be left empty.
schedule	cron job schedule text, which is in the standard cron format.
command	Text of the command to be executed.

Sample



```
postgres=> SELECT cron.schedule ('test','0 10 * * *', 'VACUUM pgbench_history');
```

```
schedule
-----
      145
(1 row)

postgres=> SELECT cron.schedule ('0 15 * * *', 'VACUUM pgbench_accounts');
schedule
-----
      146
(1 row)
```

`schedule` uses the standard cron syntax. Here, `*` indicates to run the job at the specified time, and specific numbers indicate to run the job at the time specified by the numbers.



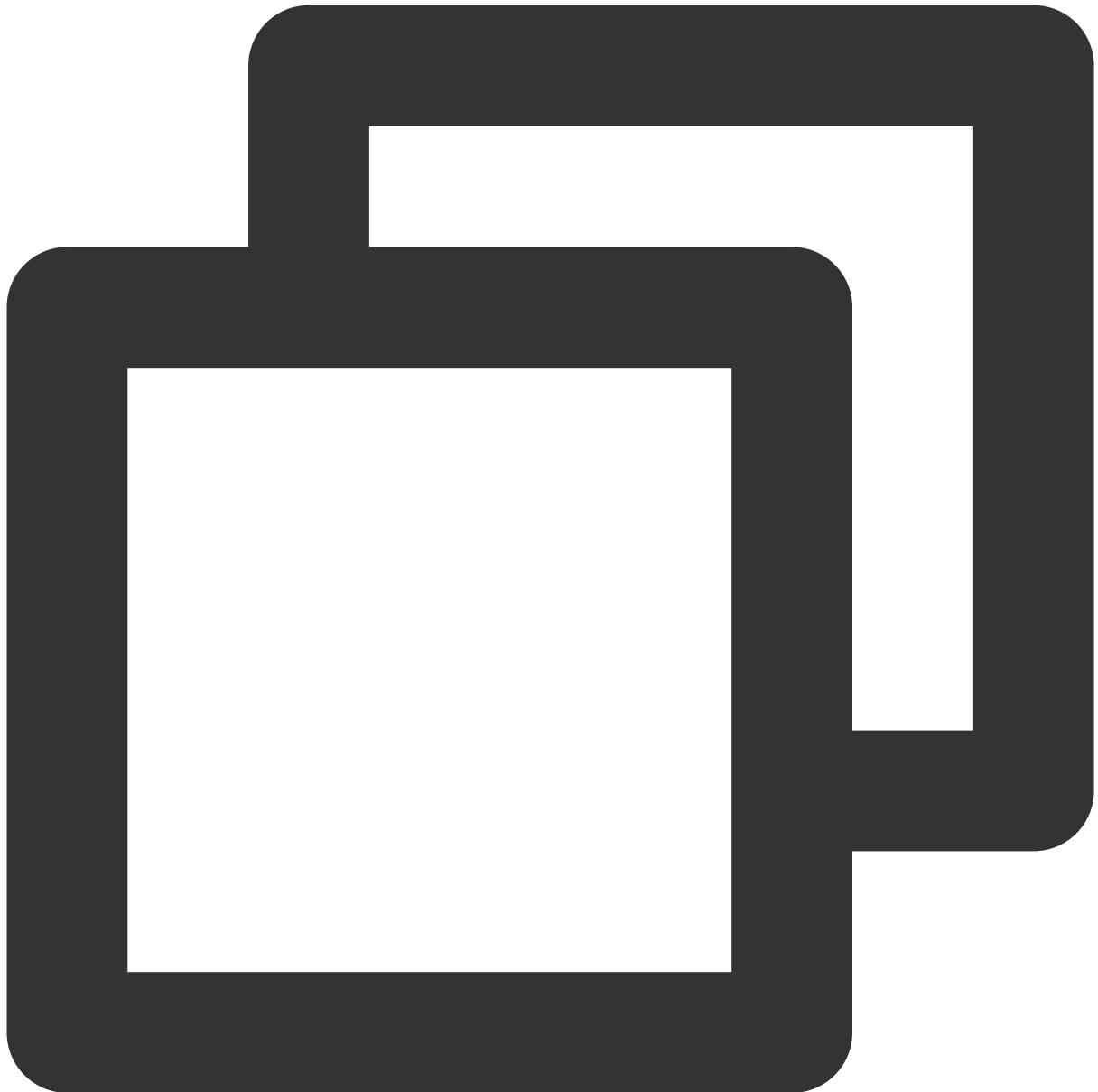
```
# Format: minute hour day of month month day of week
# week (0 - 6) = sun,mon,tue,wed,thu,fri,sat
# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,...,sat
# | | | | |
# * * * * *
```

cron.unschedule() function

This function is used to delete a cron job. You can pass in a `job_name` or `job_id` . Make sure that you own the policy corresponding to the `job_id` passed in. This function returns a boolean value indicating success or failure.

This function uses the following syntax format:

Syntax

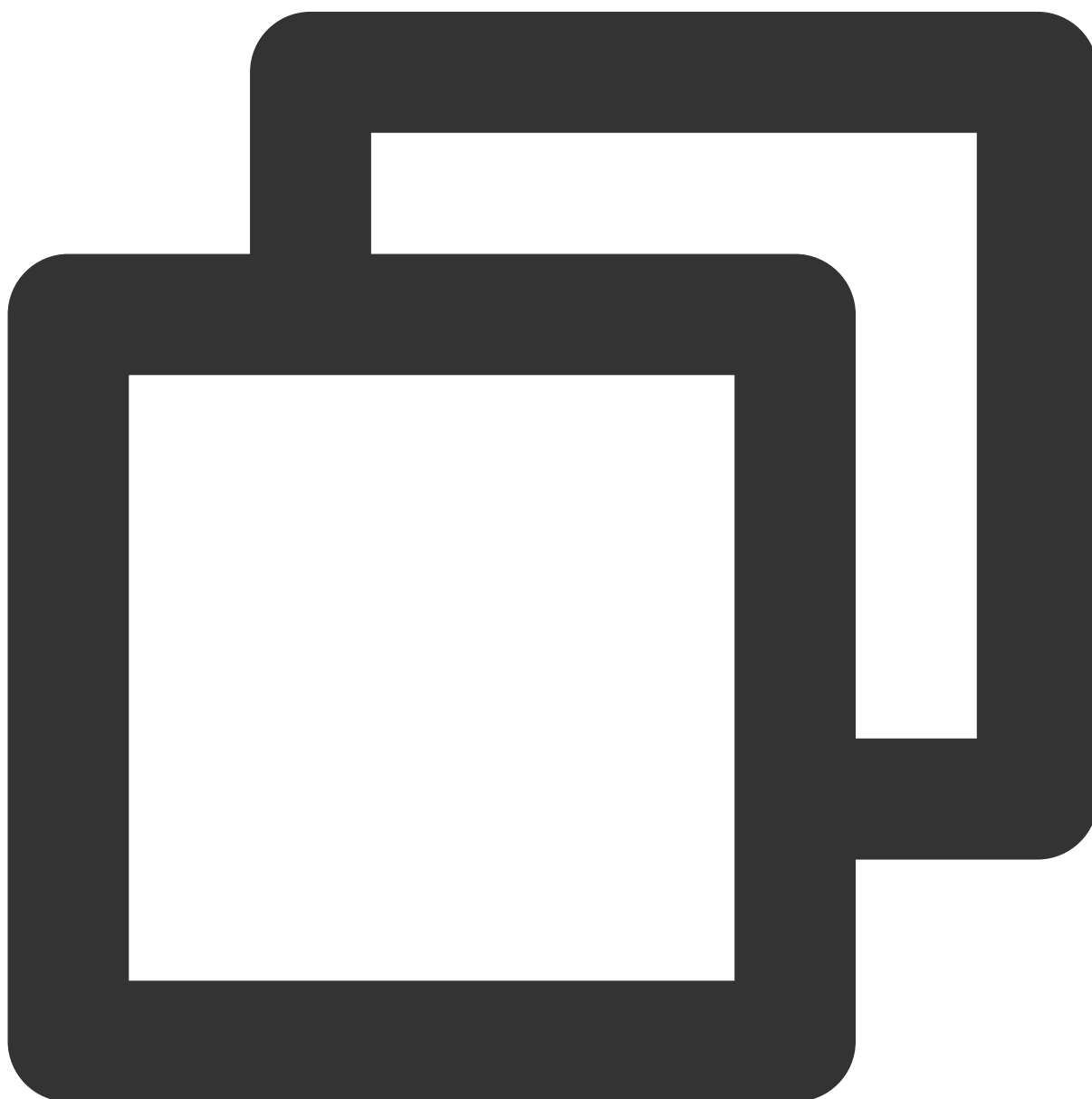


```
cron.unschedule (job_id);  
  
cron.unschedule (job_name);
```

Parameters

Parameter	Description
job_id	Job ID returned by the <code>cron.schedule</code> function during cron job scheduling.
job_name	Name of the cron job scheduled by the <code>cron.schedule</code> function.

Sample



```
postgres=> select cron.unschedule(108);
```

```
unschedule
-----
t
(1 row)

postgres=> select cron.unschedule('test');
unschedule
-----
t
(1 row)
```

pg_cron tables

The following tables are used to schedule jobs and record job execution methods.

Table	Description
cron.job	It contains the metadata of each scheduled job. Most interactions with this table are implemented by using the <code>cron.schedule</code> and <code>cron.unschedule</code> functions. Note that we recommend you not directly grant the permission to update or insert data into this table.
cron.job_run_details	It contains the historical information of previously scheduled jobs. It is very useful for checking the statuses, returned messages, and start/end times of executed jobs. To prevent this table from growing continuously, clear it regularly.

Setting pg_cron Scheduled Job

1. If you want to perform the `VACUUM` operation on a specified table at the selected time, use the `cron.schedule` function to schedule a job. For example, you can run `VACUUM FREEZE` on the specified table at 22:00 (GMT) every day. The number returned by the scheduling statement indicates the current job ID.

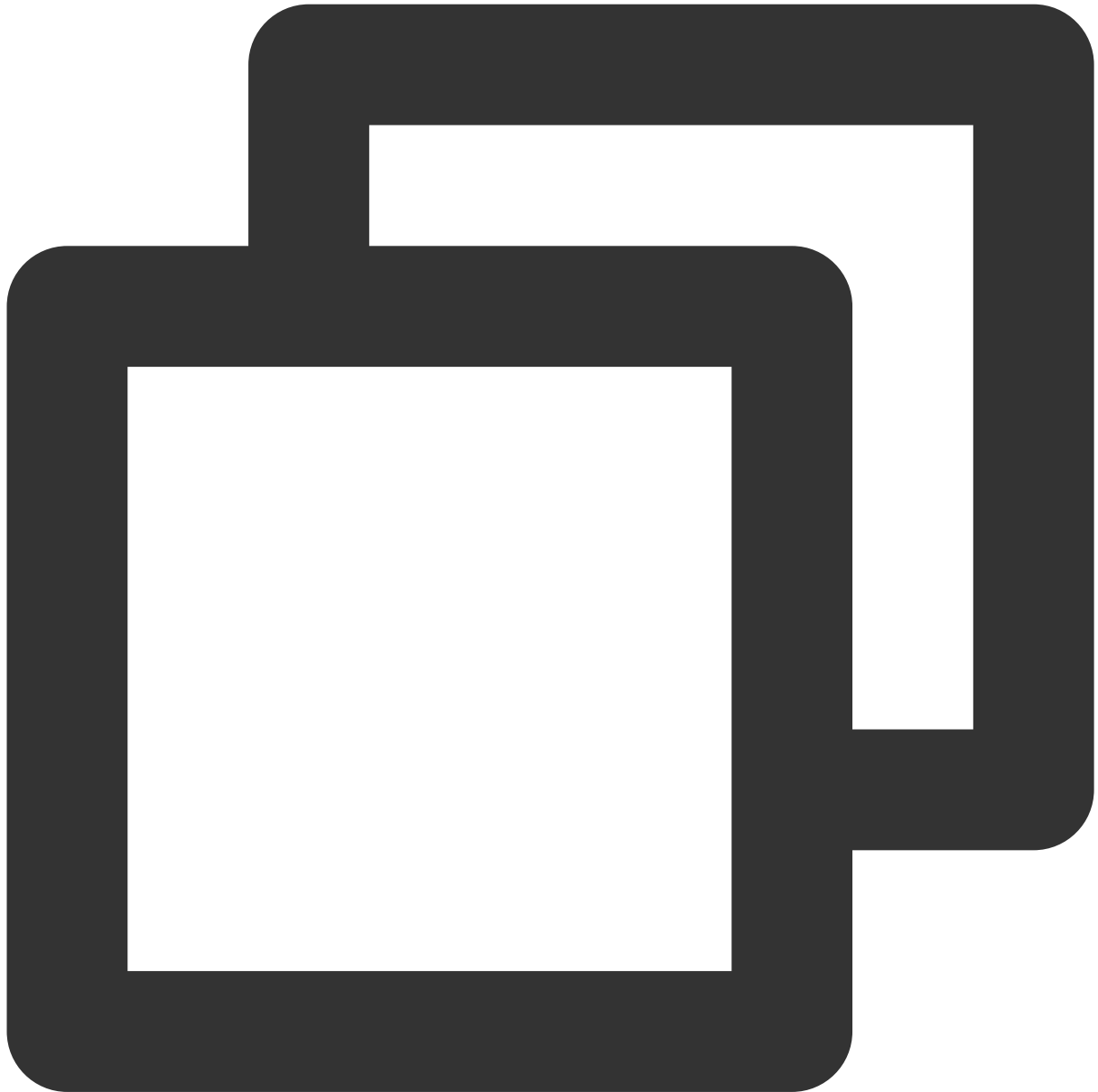


```
SELECT cron.schedule('manual vacuum', '0 22 * * *', 'VACUUM FREEZE pgbench_accounts
  schedule
-----
1
(1 row)
```

2. This function has three input parameters: the job name (string), the cron scheduling syntax, and the specific SQL statement to be executed.

Viewing pg_cron Scheduled Job

After scheduling a job, you can view it in the `cron.job` table by running the following statement:

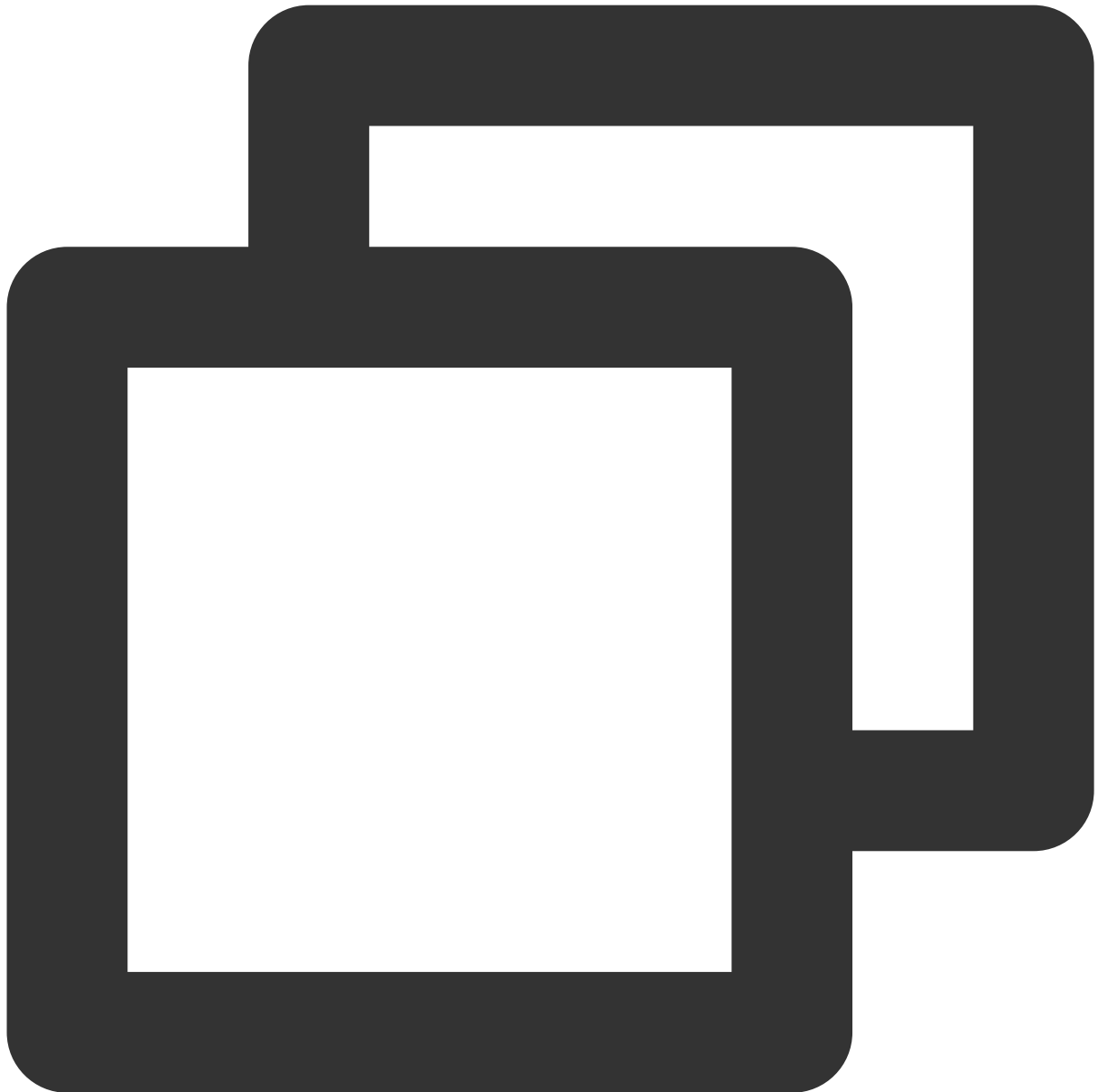


```
SELECT * FROM cron.job;
```

jobid	schedule	command	nodename	nodeport	database	username	activ
1	0 22 * * *	VACUUM ...	localhost	5432	postgres	test	t

Deleting pg_cron Scheduled Job

If a scheduled job is no longer needed, you can run the following statement to delete it:



```
SELECT cron.unschedule(1);
```

```
unschedule
```

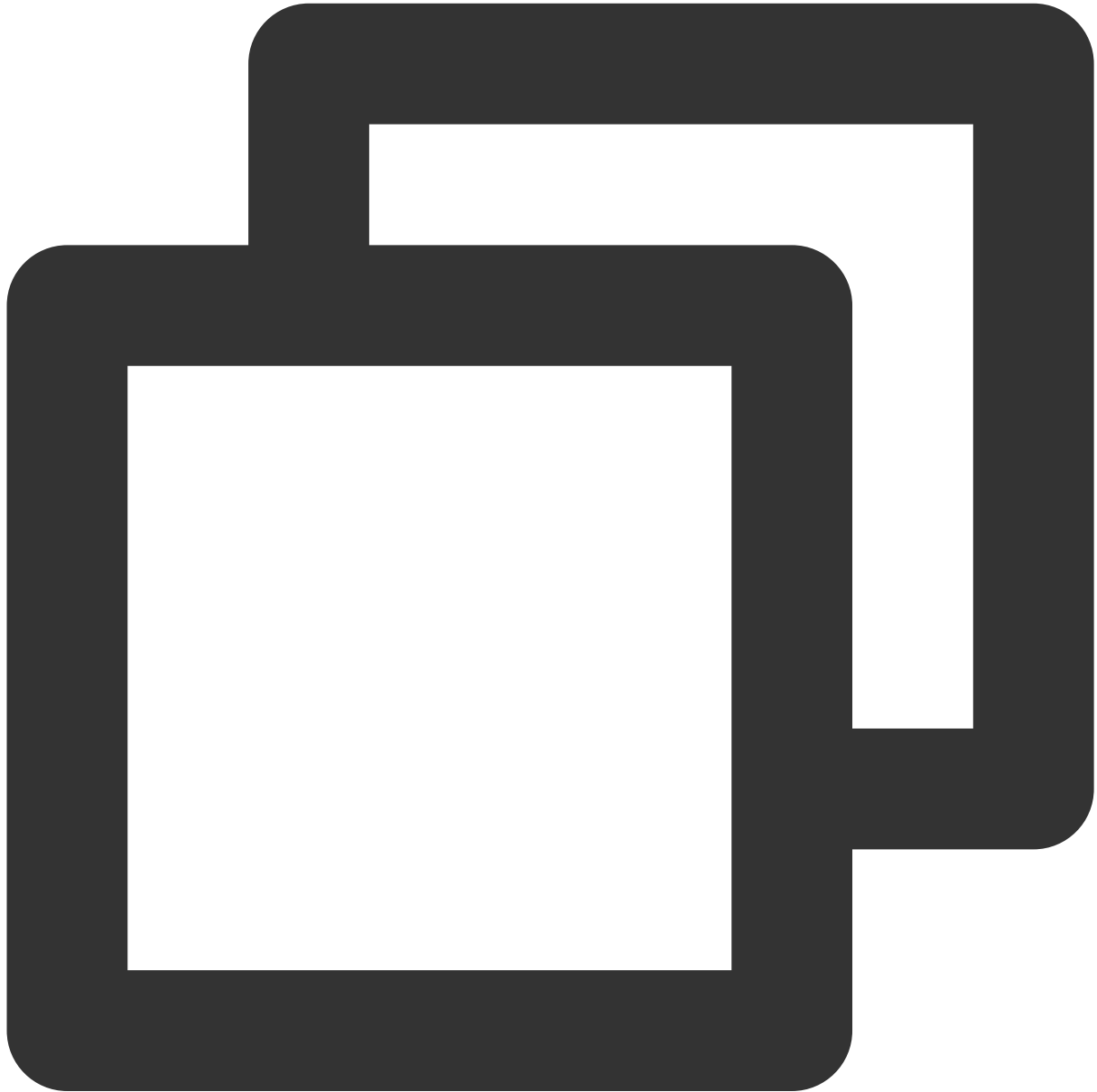
```
-----
```

```
t
```

Viewing the Execution History of Scheduled Job

After running the above sample code, you can check the job status and execution result in the

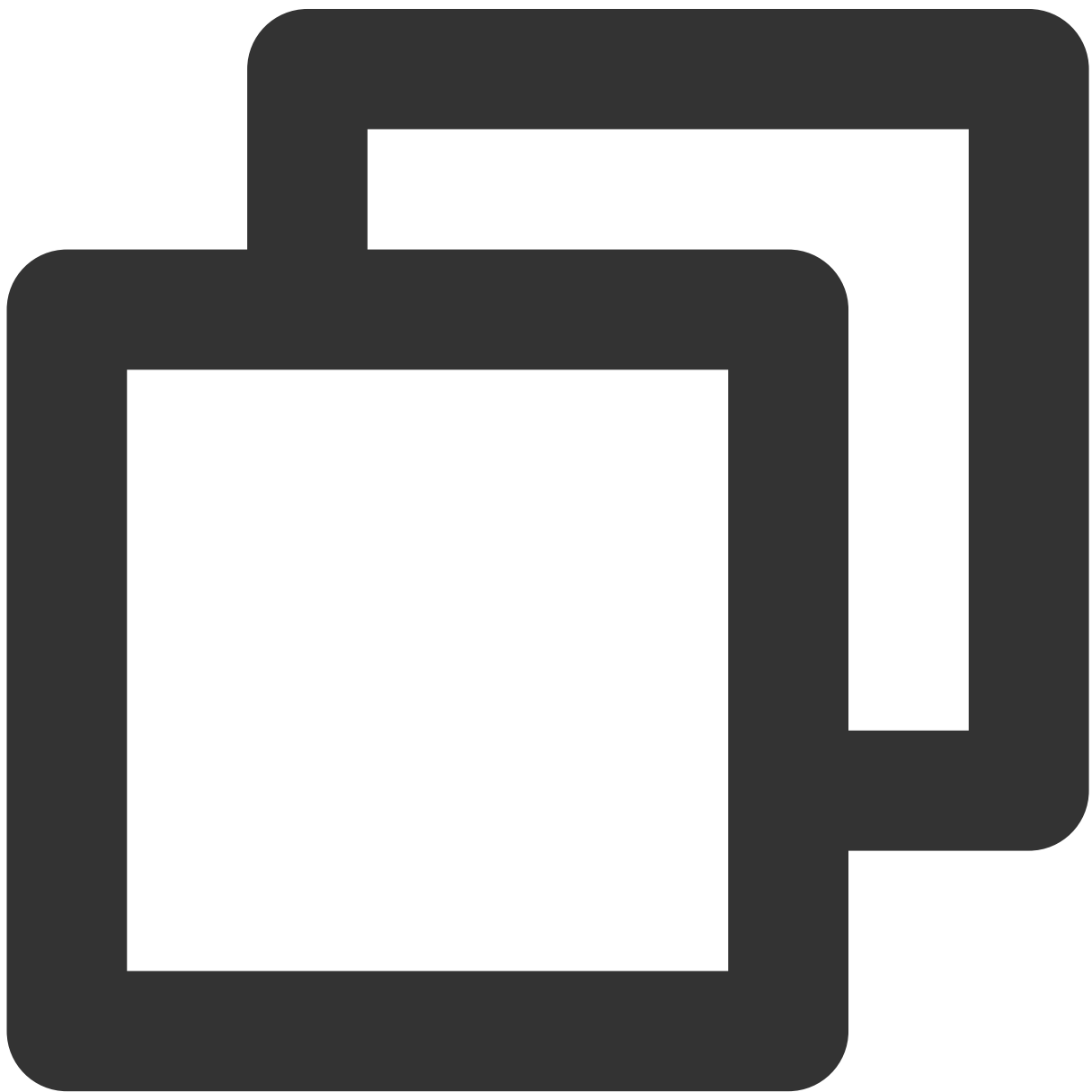
`cron.job_run_details` table as follows:



```
postgres=> select * from cron.job_run_details;
 jobid | runid | job_pid | database | username | command | status | return_message
-----+-----+-----+-----+-----+-----+-----+-----
  1    | 1    | 3395    | postgres | adminuser| vacuum freeze pgbench_accounts | succeeded |
(1 row)
```


Clearing pg_cron Record Table

1. The `cron.job_run_details` table contains the records of historical cron jobs, which may get very large over time. We recommend you clear it regularly. For example, it may be sufficient to retain the records of jobs in the past week for troubleshooting.
2. In the following sample code, the `cron.schedule` function is used to schedule the job of clearing records in the `cron.job_run_details` table at 00:00 every day and retaining only records of jobs in the past seven days.



```
SELECT cron.schedule('0 0 * * *', $$DELETE
FROM cron.job_run_details
WHERE end_time < now() - interval '7 days'$$);
```

Disabling pg_cron Records

To completely disable writing any content into the `cron.job_run_details` table, set the `cron.log_run` parameter to `off` in the console.

If you do so, the `pg_cron` extension will no longer write data to this table and will only generate errors in the `postgresql.log` file. You can view all error messages in the error logs in the console.

Run the following command to check the value of the `cron.log_run` parameter.

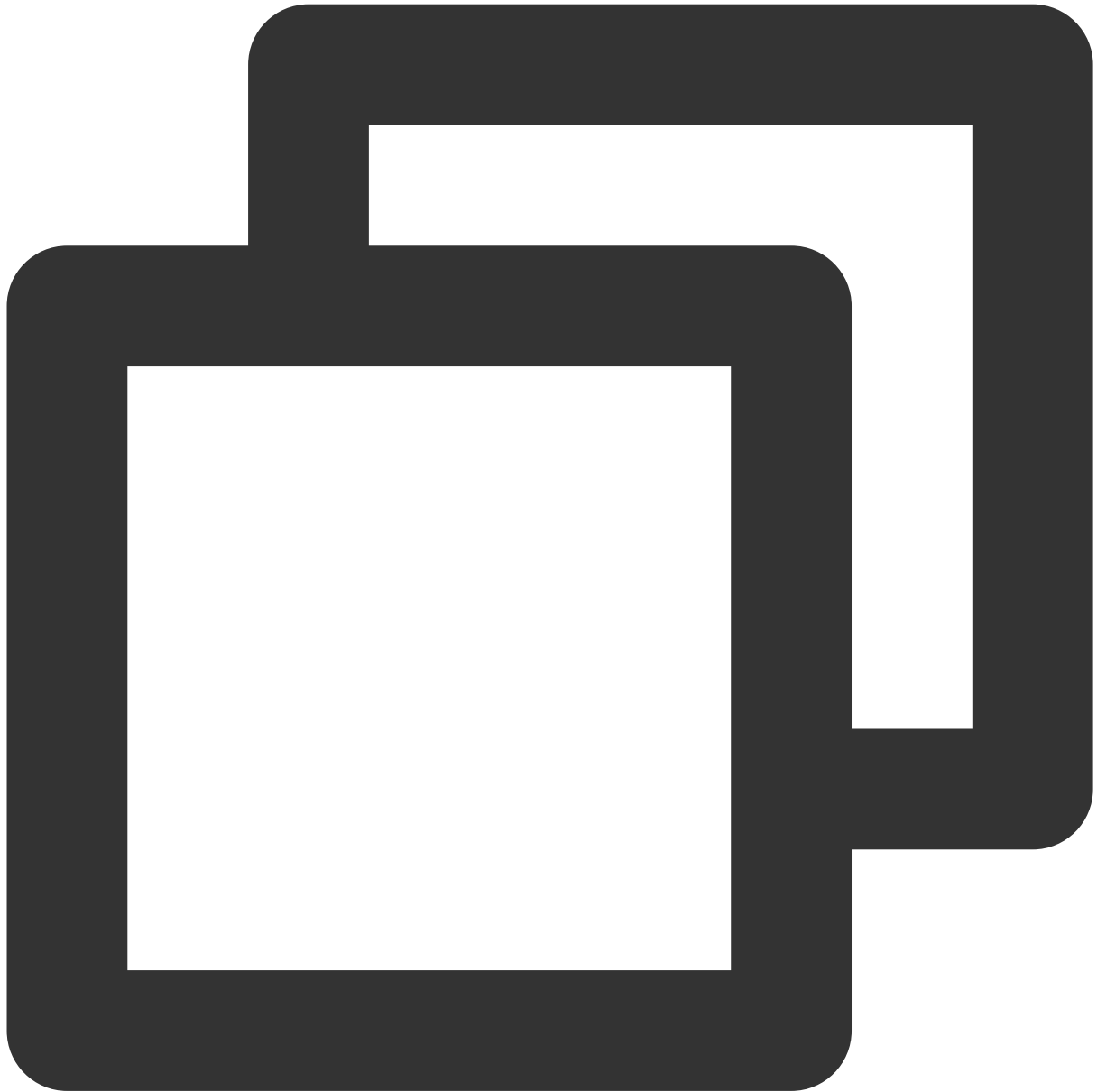


```
postgres=> SHOW cron.log_run;
```

Setting Scheduled Job for Other Databases

By default, all metadata of `pg_cron` is stored in the `postgres` database. To run a scheduled job for objects in another database, perform the following operations:

1. To perform the `VACUUM` operation on a table in the `test` database, you first need to use the admin account of `pg_cron` to run the `cron.schedule` function in the `postgres` database to schedule a job.



```
postgres=> SELECT cron.schedule('test manual vacuum', '29 03 * * *', 'vacuum freeze
```

2. Run the following command with the admin account to set the database of the schedule job to the target database.

Note that `jobid` must be the `jobid` returned in step 1.



```
postgres=> UPDATE cron.job SET database = 'test' WHERE jobid = 106;
```

3. Query the `cron.job` table to verify the operation result.



```
postgres=> select * from cron.job;
 jobid | schedule | command | nodename | nodeport | database | username | active |
-----+-----+-----+-----+-----+-----+-----+-----
  2 | 29 03 * * * | vacuum freeze test_table | localhost | 8192 | test | adminuser |
  1 | 59 23 * * * | vacuum freeze pgbench_accounts | localhost | 8192 | postgres | a
(2 rows)
```

pg_cron Parameters

Parameter used to control the behaviors of the pg_cron extension are as listed below:

Parameter	Description
cron.database_name	pg_cron metadatabase.
cron.host	Name of the host to connect to PostgreSQL, which cannot be modified.
cron.log_run	Specifies whether to record all executed jobs into the <code>job_run_details</code> table. Valid values: <code>on</code> , <code>off</code> .
cron.log_statement	Specifies whether to record all cron statements into logs before running them. Valid values: <code>on</code> , <code>off</code> .
cron.max_running_jobs	Maximum number of concurrent jobs. To run more jobs, submit a ticket for assistance.
cron.use_background_workers	Specifies to use backend workers instead of client sessions. You cannot modify the value.

You can run the following SQL command to display these parameters and their values:



```
postgres=> SELECT name, setting, short_desc FROM pg_settings WHERE name LIKE 'cron.'
```


Network Management

Overview

Last updated : 2024-01-24 11:16:51

This document describes the network of TencentDB for PostgreSQL. TencentDB for PostgreSQL offers the network management feature to protect your instance security and provide service to internal and external businesses efficiently and freely.

Network Types

There are two types of TencentDB network environments: VPC and classic network.

VPC: it is a logically isolated network space that can be customized in Tencent Cloud. Even in the same region, different VPCs cannot communicate with each other by default. Similar to the traditional network in an IDC, a VPC is where your Tencent Cloud service resources are managed.

Classic network: it is the public network resource pool for all Tencent Cloud users. All your Tencent Cloud resources will be centrally managed by Tencent Cloud.

Note:

Currently, resources cannot be created in the classic network.

Feature comparison

Feature	Classic Network	VPC
Custom network	Unsupported	Supported
Custom routing	Unsupported	Supported
Custom IP	Unsupported	Supported
Interconnection rule	Interconnection in the same region	Interconnection between subnets in the same VPC in the same region
Security control	Security group	Security group

Network Access

Tencent Cloud services can be accessed over both the public and private networks.

Public network access: it is a service provided by Tencent Cloud to implement public data transfer for an instance. You can enable the public IP of the instance for it to communicate with other computers and allow access over the public network.

Private network access: it is used to provide Local Area Network (LAN) service. Tencent Cloud assigns resources with private IP addresses to allow a free private network communication in the same region or instance access over the private network.

Note:

Security groups that currently support public network access are available only in the Guangzhou, Shanghai, Beijing, and Chengdu regions. Instances in other regions may be attacked if the public network access is enabled. We do not recommend that you enable public network access for instances in production environment. If you need to enable public network access, security group rules must be configured.

Network Configuration

You can configure one or two networks for each TencentDB for PostgreSQL instance.

In scenarios where the instance supports two networks:

An instance can be accessed through different VIPs that belong to different VPCs and subnets.

You can use this feature to change the instance network, for example, from the classic network to VPC or from VPC A to VPC B.

You can use this feature to implement the multi-plane network feature in scenarios where businesses in two different VPCs need to access the same database instance.

Managing Instance Network

You can add, delete, and change networks in the TencentDB for PostgreSQL console. For more information, see [Modifying Network](#).

Modifying Network

Last updated : 2024-01-24 11:16:51

This document describes how to configure and manage instance networks in the TencentDB for PostgreSQL console. You can add, delete, and modify instance networks based on your business needs.

Overview

Tencent Cloud supports **classic network** and **VPC**, which are capable of offering a diversity of smooth services. On this basis, we provide more flexible services as shown below to help you configure and manage network connectivity with ease.

Changing network

Switch from classic network to VPC: a single TencentDB source instance can be switched from classic network to VPC.

Switch from VPC A to VPC B: a single primary or read-only TencentDB instance can be switched from VPC A to VPC B.

Customizing access IP address

Custom primary instance IP: you can specify the IP address when adding a network on the instance details page of the primary instance.

Custom read-only instance IP: you can specify the IP address when adding a network on the instance details page of a read-only instance.

Notes

The change from classic network to VPC is irreversible. After the switch to a VPC, the TencentDB instance cannot communicate with Tencent Cloud services in another VPC or classic network.

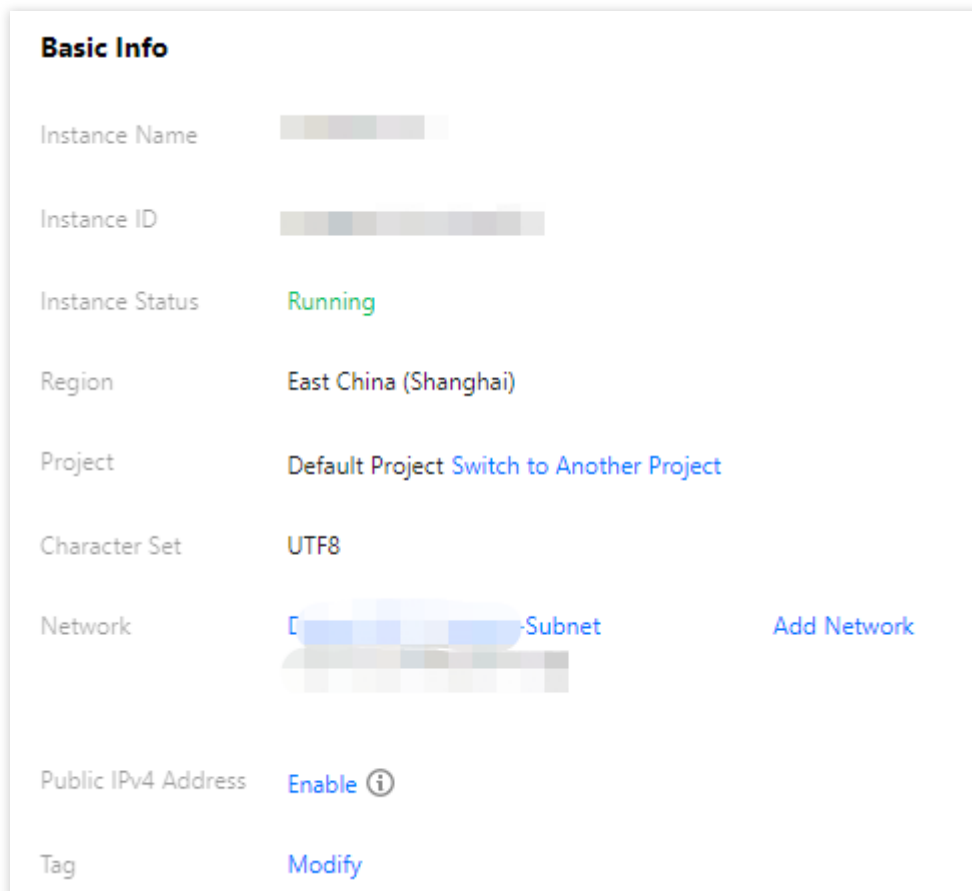
After you change a primary instance's network, the networks of read-only instances associated with the primary instance won't be automatically switched; that is, you need to separately switch them.

A new network added to an instance does not affect the IP address in the original network configuration.

Directions

Adding network

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. On the instance details page, click **Add Network** in **Basic Info** > **Network**.



3. In the pop-up window, select a network. You can let the system automatically set an IP or manually specify an IP. After confirming that everything is correct, click **OK**.

Note:

You can configure one or two networks for each instance.

If an instance has two networks, both are controlled by the security group associated with the instance.

You can only select a new VPC and subnet in the same region where the instance resides.

Add Network

1. You can configure one or two networks for each instance.

2. If an instance has two networks, both are controlled by the security group associated with the instance.

Select Network

CIDR

subnet IP/available

In the current network environment, only CVM in the "Default-VPC" can access this database.[Create VPC](#) [Create Subnet](#)

☒ Auto-Assign IP

☐ Specify IP

OK

Cancel

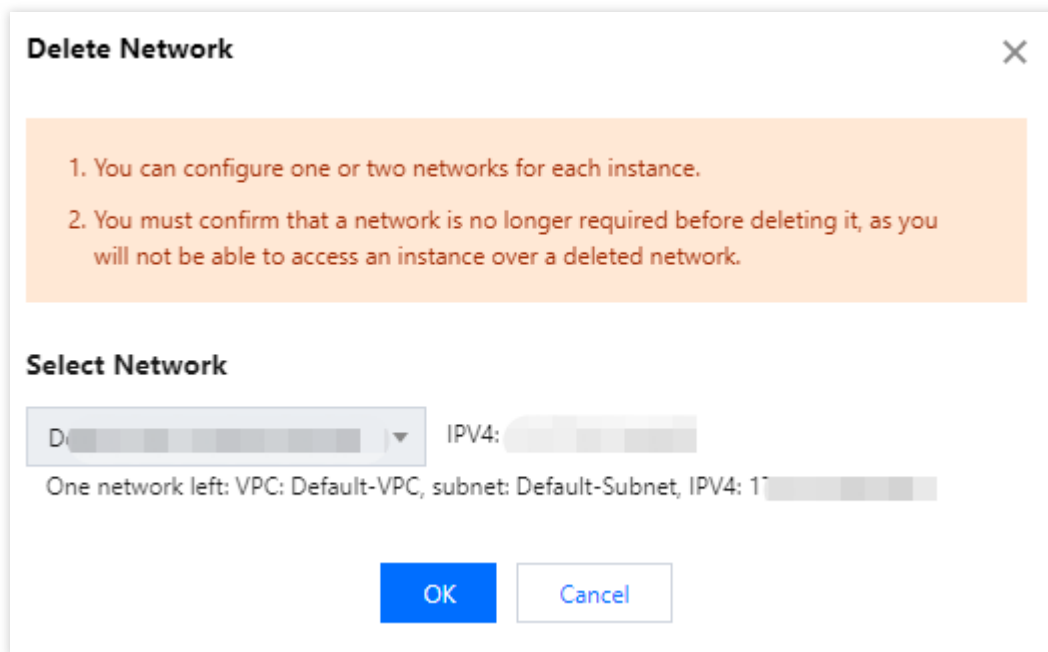
4. After the instance status changes from **Changing network** to **Running**, you can query the changed instance network on the instance details page.

Deleting network

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. On the instance details page, click **Delete Network** in **Basic Info > Network**.

Modify

You must confirm that a network is no longer required before deleting it, as you will not be able to access an instance over a deleted network.



4. After the instance status changes from **Changing network** to **Running**, you can query the changed instance network on the instance details page.

Modifying network

If you want to change the current network of the instance, for example, from the classic network to a VPC or from VPC A to VPC B, you can add and delete a network as detailed above for this need.

Example 1: changing from classic network to VPC

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. On the instance details page, click **Add Network** in **Basic Info > Network**, select the target VPC, and click **OK**.
3. After the instance status becomes **Running**, click **Delete Network** after the instance network, select the classic network, and click **Delete**. At this point, the instance network has changed from the original classic network to the new VPC.

Example 2: changing from VPC A to VPC B

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click an instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. On the instance details page, click **Add Network** in **Basic Info > Network**, select VPC B, and click **OK**.
3. After the instance status becomes **Running**, click **Delete Network** after the instance network, select VPC A, and click **Delete**. At this point, the instance network has changed from VPC A to VPC B.

Note:

After a network is deleted, you cannot access the instance over it. Make sure that a network is no longer needed before deleting it.

Enabling Public Network Address

Last updated : 2024-01-24 11:16:51

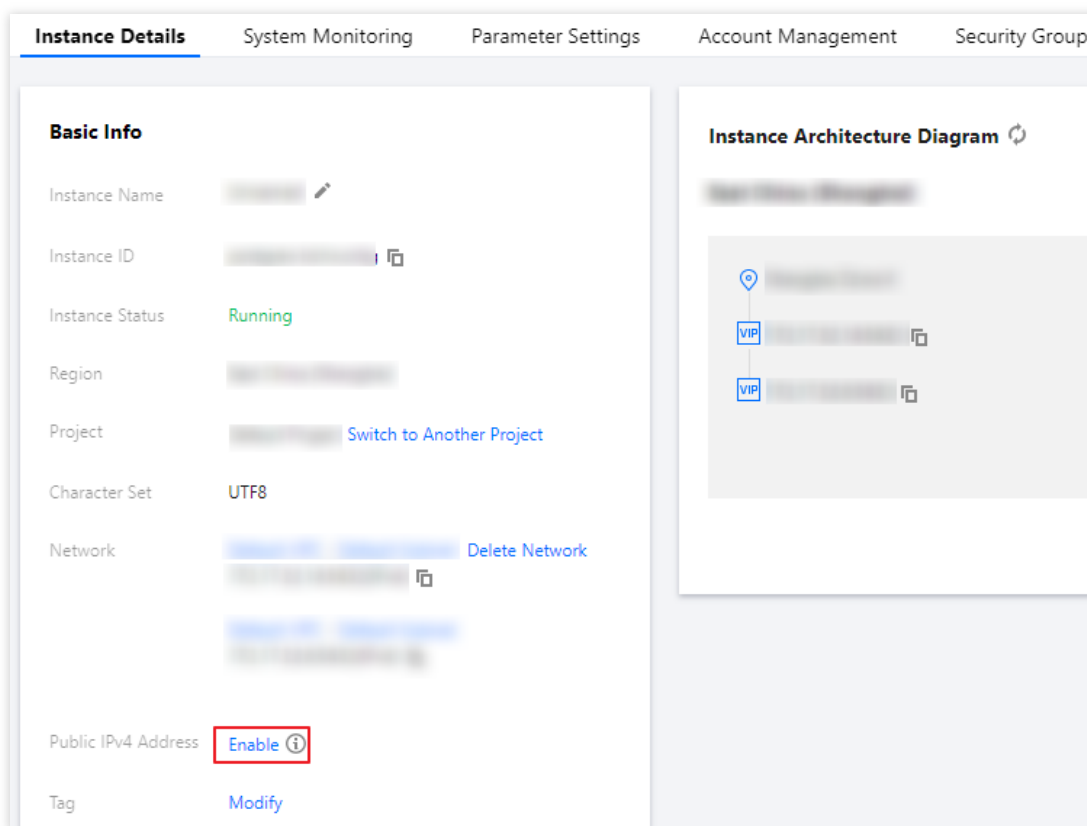
TencentDB for PostgreSQL supports private and public network addresses. By default, a private network address is provided for you to access your instance over the private network. If you want to enable public network access, you can enable it in the console to build a public network address.

Note:

Currently, the public network security group feature is available in the Beijing, Shanghai, Guangzhou, and Chengdu regions, so you can enable the public network address in the console.

Enabling Public Network Address in Console

1. Log in to the [TencentDB for PostgreSQL console](#) and select the region. In the instance list, click the target instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. On the **Instance Details** page, click **Enable** in **Basic Info** > **Public IPv4 Address**.



3. In the pop-up window, read the notes and click **OK**.
4. After the instance status is updated to **Running**, you can view the public network address on the instance details page.

Configure a TencentDB for PostgreSQL security group

1. Log in to the [TencentDB for PostgreSQL console](#) and select the region. In the instance list, click the target instance ID or **Manage** in the **Operation** column to enter the instance management page.
2. On the instance management page, select the **Security Group** tab, click **Configure Security Group**, configure the security group rule to open all ports, and confirm that the security group allows access from public IPs. For more information on configuration, see [Managing Security Groups](#).

Added to security group		
<div>EditConfigure Security Group</div>		
Priority	Security Group ID	Security Group
1		Open all ports

Check public network connectivity

You can use a client tool to access TencentDB for PostgreSQL. For detailed directions, see [Connecting to PostgreSQL Instances](#).

Access Management Overview

Last updated : 2024-01-24 11:16:51

Known Issues

If you have multiple users managing different Tencent Cloud services such as CVM, VPC, TencentDB for PostgreSQL, and other TencentDB products, and they all share your Tencent Cloud account access key, you may face the following problems:

The risk of your key being compromised is high since multiple users are sharing it.

Your users might introduce security risks from misoperations due to the lack of user access control.

Solutions

You can avoid the problems above by allowing different users to manage different services through sub-accounts. By default, a sub-account does not have permissions to use TencentDB for PostgreSQL or its resources. Therefore, you need to create a policy to grant different permissions to the sub-accounts.

[Cloud Access Management \(CAM\)](#) is a Tencent Cloud service that helps you securely manage and control access to your Tencent Cloud resources. Using CAM, you can create, manage, and terminate users and user groups. You can manage identities and policies to allow specific users to access your Tencent Cloud resources.

When using CAM, you can associate a policy with a user or user group to allow or forbid them to use specified resources to complete specified tasks. For more information on CAM policies, please see [Element Reference](#).

You can skip this section if you do not need to manage permissions to PostgreSQL resources for sub-accounts. This will not affect your understanding and use of the other sections of the document.

Getting started

A CAM policy is used to allow or deny one or more PostgreSQL instance operations. When configuring a policy, you must specify the target resources of the operations, which can be all resources or specified resources. A policy can also include conditions where the resources can be used.

Some PostgreSQL APIs do not support resource-level permissions, which means that you cannot specify resources when using those APIs.

Task	Link
Understand the basic structure of policies	Access Policy Syntax > Policy Syntax

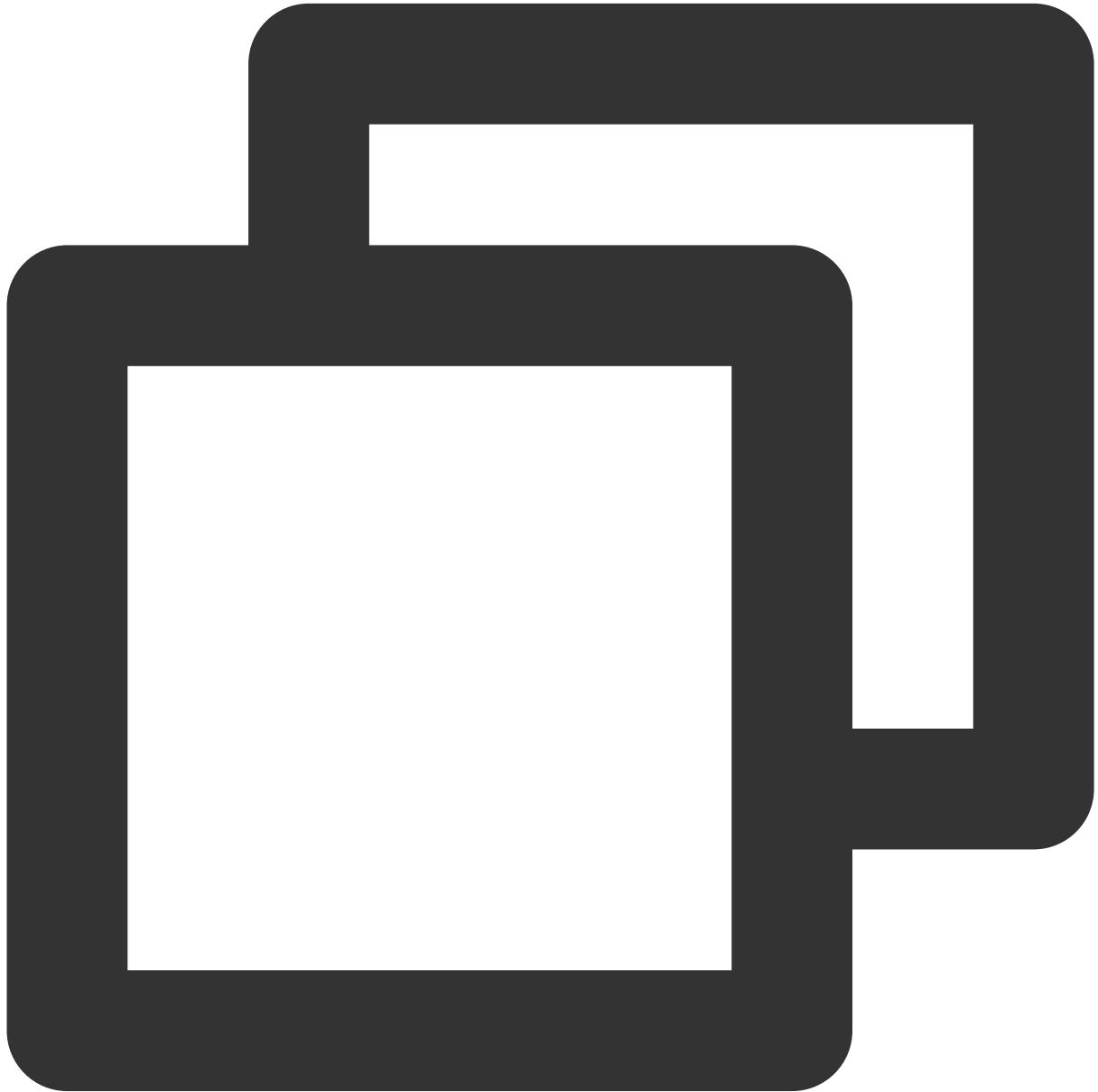
Define operations in a policy	Access Policy Syntax > PostgreSQL Operations
Define resources in a policy	Access Policy Syntax > PostgreSQL Resource Paths
View supported resource-level permissions	Authorizable Resource Types
View console examples	Console Examples

Access Policy Syntax

Last updated : 2024-01-24 11:16:51

Policy Syntax

CAM policy:



```
{  
    "version": "2.0",  
    "statement":
```

```
[
  {
    "effect": "effect",
    "action": ["action"],
    "resource": ["resource"],
    "condition": { "key": { "value" } }
  }
]
```

version is required. Currently, only the value "2.0" is allowed.

statement describes the details of one or more permissions, and therefore contains the permission(s) other elements such as `effect` , `action` , `resource` , and `condition` . One policy has only one `statement` .

effect is required. It describes the result of a statement. The result can be "allow" or an explicit "deny".

action is required. It describes the allowed or denied operation. An operation can be an API or a feature set (a set of specific APIs prefixed with "permid").

resource is required. It describes the details of authorization. A resource is described in a six-segment format.

Detailed resource definitions vary by product.

condition describes the condition for the policy to take effect. Conditions consist of operators, operation keys, and operation values. PostgreSQL currently does not support special conditions, so this element can be left empty.

PostgreSQL Operations

You can use CAM policy statements to authorize any API operations for any services that support CAM. To authorize PostgreSQL operations, please specify the APIs prefixed with "postgres:", such as "postgres:DescribeDBInstances" and "postgres:DescribeDBInstanceAttribute".

To specify multiple operations in a single statement, separate them with commas as shown below:



```
"action":["postgres:action1","postgres:action2"]
```

You can also specify multiple operations using a wildcard. For example, you can specify all operations whose names begin with "Describe" as shown below:



```
"action":["postgres:Describe*"]
```

To specify all PostgreSQL operations, use the wildcard (*) as shown below:



```
"action":["postgres:*"]
```

PostgreSQL Resource Paths

Each CAM policy statement for PostgreSQL is resource-specific.

The general form of a resource path is as follows:



```
qcs:project_id:service_type:region:account:resource
```

project_id describes the project information, which is only used to enable compatibility with legacy CAM logic and can be left empty.

service_type describes the abbreviated service name, such as "postgres".

region describes the [region information](#), such as "ap-shanghai".

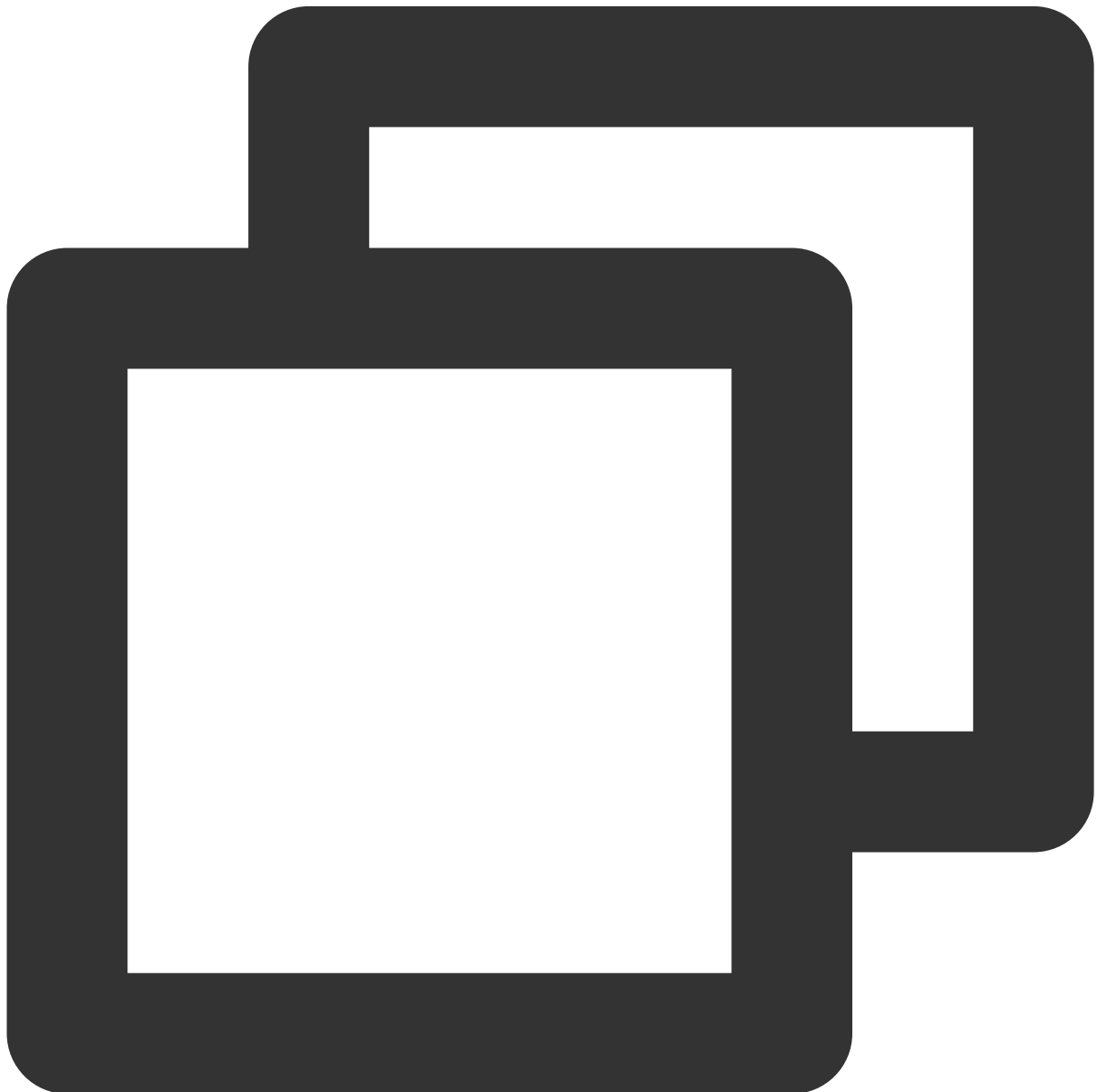
account: the root account information of the resource owner (which is the "Account ID" on the [Account Information](#) page), such as "164xxx472".

resource describes detailed resource information of each product, such as "DBInstanceId/postgres-0xssvm8e" or

"DBInstanceid/*". The table below describes the resources that can be used by PostgreSQL and the corresponding resource description methods.

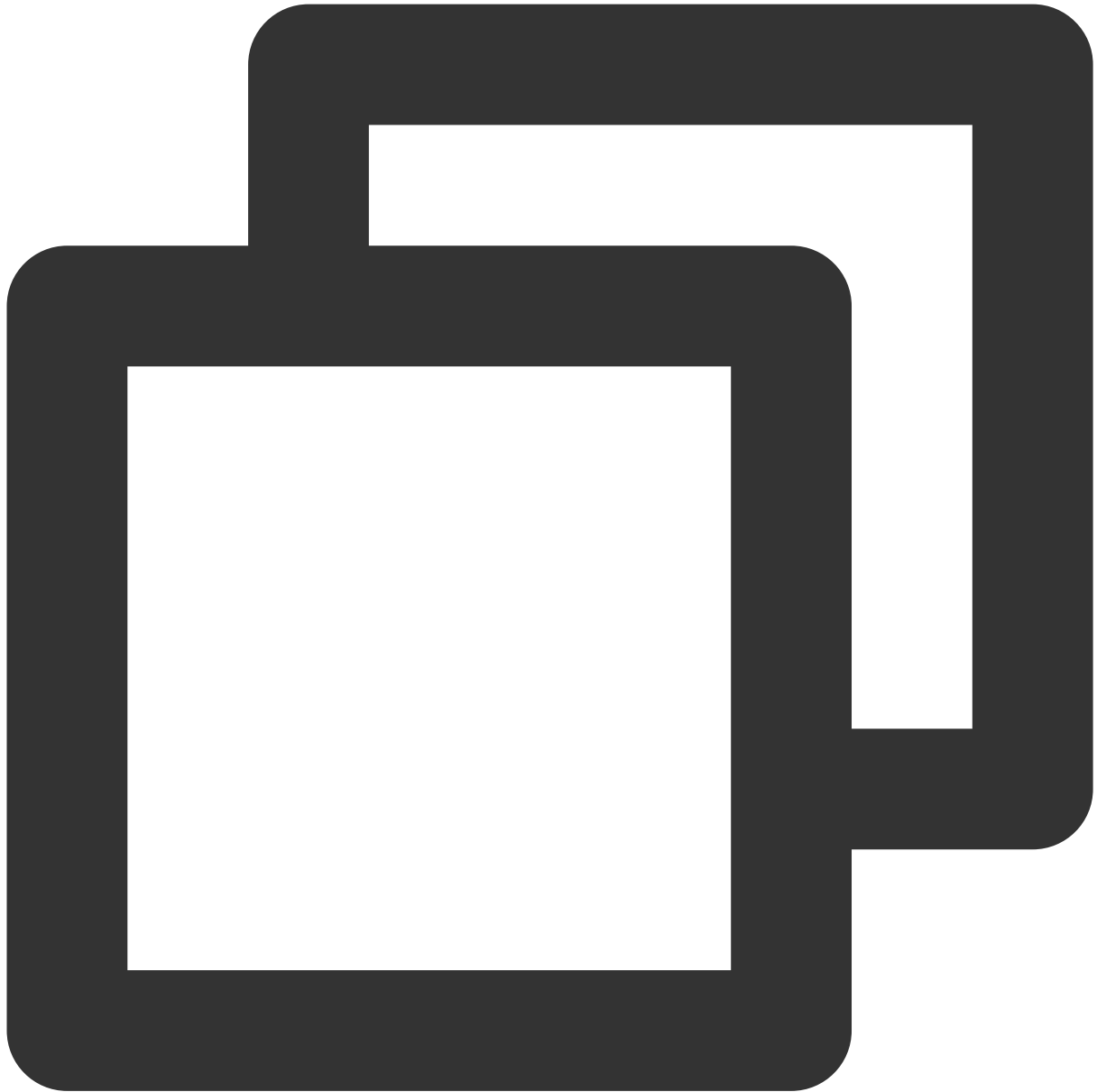
Resource	Resource Description Method in Access Policies
Instance	qcs::postgres:\$region:\$account:DBInstanceid/\$DBInstanceid

For example, you can specify an instance (instance ID: postgres-0xssvm8e) in the statement as shown below:



```
"resource": [ "qcs::postgres:ap-shanghai:164xxx472:DBInstanceId/postgres-0xssvm8e"]
```

You can also use the wildcard (*) to specify all instances in the Shanghai region that belong to a specific account as shown below:



```
"resource": [ "qcs::postgres:ap-shanghai:164xxx472:DBInstanceId/*"]
```

If you want to specify all resources or if a specific API operation does not support resource-level permission control, you can use the wildcard (*) in the `resource` element as shown below:



```
"resource": ["*"]
```

To specify multiple resources in a single statement, separate them with commas. In the following example, we specified two instances:



```
"resource":["qcs::postgres::164xxx472:DBInstanceId/postgres-0xf1f41e","qcs::postgre
```

Authorizable Resource Types

Last updated : 2024-01-24 11:16:51

Resource-Level Permission Overview

Resource-level permissions specify resources a user can operate. TencentDB for PostgreSQL supports specific resource-level permissions, i.e., allowing the user to perform operations or use specific resources. In Cloud Access Management (CAM), the types of PostgreSQL resources that can be authorized are as follows:

Resource Type	Resource Description Method in Access Policies
Instance	<div>qcs::postgres:\$region:\$account:DBInstanceId/\$DBInstanceId</div> <div>qcs::postgres:\$region:\$account:DBInstanceId/*</div>

The [PostgreSQL instance APIs](#) section in this document describes PostgreSQL API operations that currently support resource-level permissions as well as resources and condition keys supported by each operation. When configuring the resource path, you need to replace values of the parameters such as `$region` and `$account` with your actual values. You can also use the wildcard (*) in the path. For more information, please see [Console Examples](#).

Note: For a PostgreSQL API operation that does not support authorization at the resource level, you can still authorize a user to perform the operation. In this case, you must specify `*` as the resource element in the policy statement.

List of APIs Not Supporting Resource-Level Permissions

API Operation	API Description
CreateDBInstances	Creates an instance
CreateServerlessDBInstance	Creates a PostgreSQL for Serverless instance
DescribeOrders	Obtains order information
DescribeRegions	Queries available regions
DescribeZones	Queries available availability zones
DescribeProductConfig	Queries product specifications
InquiryPriceCreateDBInstances	Queries prices

DescribeServerlessDBInstances	Queries the list of PostgreSQL for Serverless instances
-------------------------------	---

List of APIs Supporting Resource-Level Permissions

[PostgreSQL instance APIs]

PostgreSQL for Serverless instance APIs

API Name	API Description
CloseServerlessDBExtranetAccess	Disables the public network access for a PostgreSQL for Serverless instance
DeleteServerlessDBInstance	Deletes a PostgreSQL for Serverless instance
OpenServerlessDBExtranetAccess	Enables the public network access for a PostgreSQL for Serverless instance

Backup and restoration APIs

API Name	API Description
DescribeDBBackups	Queries the list of instance backups
DescribeDBErrlogs	Obtains error logs
DescribeDBSlowlogs	Obtains slow query logs
DescribeDBXlogs	Obtains the Xlog list

Instance APIs

API Name	API Description
CloseDBExtranetAccess	Disables the public network address for an instance
DescribeDBInstanceAttribute	Queries instance details
DescribeDatabases	Pulls the instance list
DestroyDBInstance	Terminates an instance
InitDBInstances	Initializes an instance
InquiryPriceRenewDBInstance	Queries the instance renewal price

InquiryPriceUpgradeDBInstance	Queries the instance upgrade price
ModifyDBInstanceName	Modifies the instance name
ModifyDBInstancesProject	Transfers an instance to another project
OpenDBExtranetAccess	Enables public network access
RenewInstance	Renews an instance
RestartDBInstance	Restarts an instance
SetAutoRenewFlag	Sets auto-renewal
UpgradeDBInstance	Upgrades an instance
DescribeDBInstances	Queries the instance list

Account APIs

API Name	API Description
DescribeAccounts	Obtains the list of instance users
ModifyAccountRemark	Modifies the account password
ResetAccountPassword	Resets the account password

Console Examples

Last updated : 2024-01-24 11:16:51

Overview

You can grant a user permissions to view and use specific resources in the [TencentDB for PostgreSQL console](#) by using Cloud Access Management (CAM) policies. This document provides examples to describe how to create and use such policies to grant these permissions.

Directions

Note:

To grant a user only the permissions of specific APIs, at least the permissions of the following APIs must be granted, or else the console fails to display correctly.

The sample code of `action` is as follows:



```
"action": [
  "postgres:DescribeProductConfig",
  "postgres:InquiryPriceCreateDBInstances",
  "postgres:DescribeRegions",
  "postgres:DescribeZones"
]
```

Note:

To grant a user the permissions to monitor and view instances, the API permissions related to monitoring needs to be granted. The sample code of `action` is as follows:



```
{ "effect": "allow",  
  "action": [  
    "monitor:Get*",  
    "monitor:Describe*"  
  ],  
  "resource": "*" }  
}
```

Full read/write permission policy for PostgreSQL

To grant a user permissions to create and manage PostgreSQL instances, you can associate the

`QcloudPostgreSQLFullAccess` policy with the user.

This policy grants the user permissions to operate all PostgreSQL resources. You can find more details below:

Associate the default policy `QcloudPostgreSQLFullAccess` with the user as instructed in [Authorization Management](#).

Read-only permission policy for PostgreSQL

To grant a user permissions to only view PostgreSQL instances, you can associate the

`QcloudPostgreSQLReadOnlyAccess` policy with the user. Users assigned will not have the access to create, delete, or modify PostgreSQL instances.

This policy grants the user permissions of all PostgreSQL operations that begin with the word "Describe" or "Inquiry".

The detailed steps are as follows:

Associate the default policy `PostgreSQL` with the user as instructed in [Authorization Management](#).

Policy for granting a user permissions to operate specific PostgreSQL instances

To grant a user permissions to operate specific PostgreSQL instances, you can associate the following policy with the user. The detailed steps are as follows:

1. Create a custom policy as instructed in [Policy](#).

The example policy syntax is as follows. This example policy grants a user permissions of all operations on the PostgreSQL instance whose ID is "postgres-0xxx8e".



```
{
  "version": "2.0",
  "statement": [
    {
      "action": "postgres:*",
      "resource": "qcs::postgres:ap-shanghai:103xxx1481:DBInstanceId/postgres-0x",
      "effect": "allow"
    }
  ]
}
```

2. Locate the created policy and click **Bind User/Group** in the "Operation" column.
3. In the pop-up window, select the user/group you want to authorize and click **OK**.

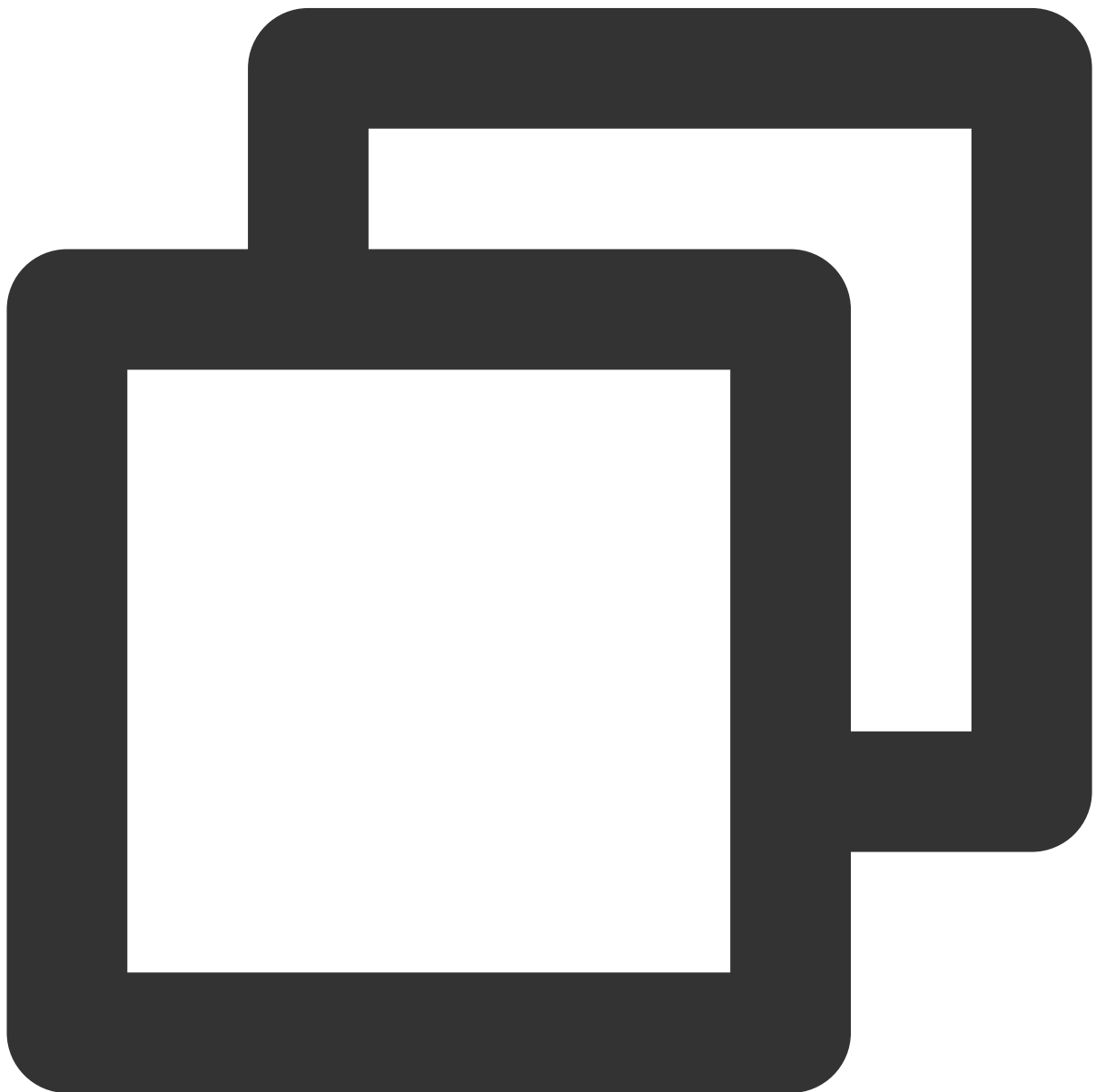
Policy for granting a user permissions to use all PostgreSQL resources

To grant a user permissions to use all PostgreSQL resources, you can associate the following policy with the user.

The detailed steps are as follows:

1. Create a custom policy as instructed in [Policy](#).

The example policy syntax is as follows. This example policy grants a user permissions to operate all PostgreSQL resources.



```
{
  "version": "2.0",
  "statement": [
    {
      "action": "postgres:*",
      "resource": "qcs::postgres:::*",
      "effect": "allow"
    }
  ]
}
```

2. Locate the created policy and click **Bind User/Group** in the "Operation" column.

3. In the pop-up window, select the user/group you want to authorize and click **OK**.

Policy for denying a user permissions to operate specific PostgreSQL instances

To deny a user permissions to operate specific PostgreSQL instances, you can associate the following policy with the user. The detailed steps are as follows:

1. Create a custom policy as instructed in [Policy](#).

The example policy syntax is as follows. This example policy denies a user permissions to operate the PostgreSQL instances whose IDs are "postgres-c8xxxa4" and "postgres-d8xxxb4" respectively.



```
{
  "version": "2.0",
  "statement": [
    {
      "action": "postgres:*",
      "resource": [
        "qcs::postgres::16xxx472:DBInstanceId/postgres-c8xxxa4",
        "qcs::postgres::16xxx472:DBInstanceId/postgres-d8xxxb4",
      ],
      "effect": "deny"
    }
  ]
}
```



```
]
}
```

2. Locate the created policy and click **Bind User/Group** in the "Operation" column.
3. In the pop-up window, select the user/group you want to authorize and click **OK**.

Custom policies

If preset policies do not meet your requirements, you can create custom policies as needed.

For detailed instructions, see [Policy](#).

For more PostgreSQL-related policy syntax, see [Access Policy Syntax](#).

Data Encryption

TDE Overview

Last updated : 2023-02-14 18:26:43

Overview

Transparent Data Encryption (TDE) provides file-level encryption for data stored in the disk. It is imperceptible to applications at the upper layer of the database and doesn't require you to modify the business code. It encrypts data before it is written to disk and decrypts data when it is read from the disk in a transparent manner.

TDE is usually used to address security and compliance issues in various scenarios where the static data needs to be protected, such as PCI DSS and CCP compliance.

Encryption

In cryptography, encryption refers to converting information in plaintext into unreadable content in ciphertext.

Modern cryptography is a study based on number and probability theories. Its ultimate goal is the perfect security (also called **information-theoretic security**) defined by Claude Shannon.

Let $E = (E, D)$ be a Shannon cipher defined over (K, M, C) . Consider a probabilistic experiment in which the random variable k is uniformly distributed over K . If for all $m_0, m_1 \in M$, and all $c \in C$, we have

$$\Pr[E(k, m_0) = c] = \Pr[E(k, m_1) = c],$$

then we say that E is a perfectly secure Shannon cipher.

Simply put, ciphertext c can be encrypted from any plaintext m , and its relevance to the plaintext cannot be found in itself.

Encryption Algorithm Types

There are two types of encryption algorithms: symmetric and asymmetric.

- Symmetric encryption: The same key is used for both encryption and decryption. It is much faster than asymmetric encryption and is required in many scenarios.
- Asymmetric encryption: It is also called public-key cryptography. It uses different keys for encryption and decryption and is mainly used to transfer user information.

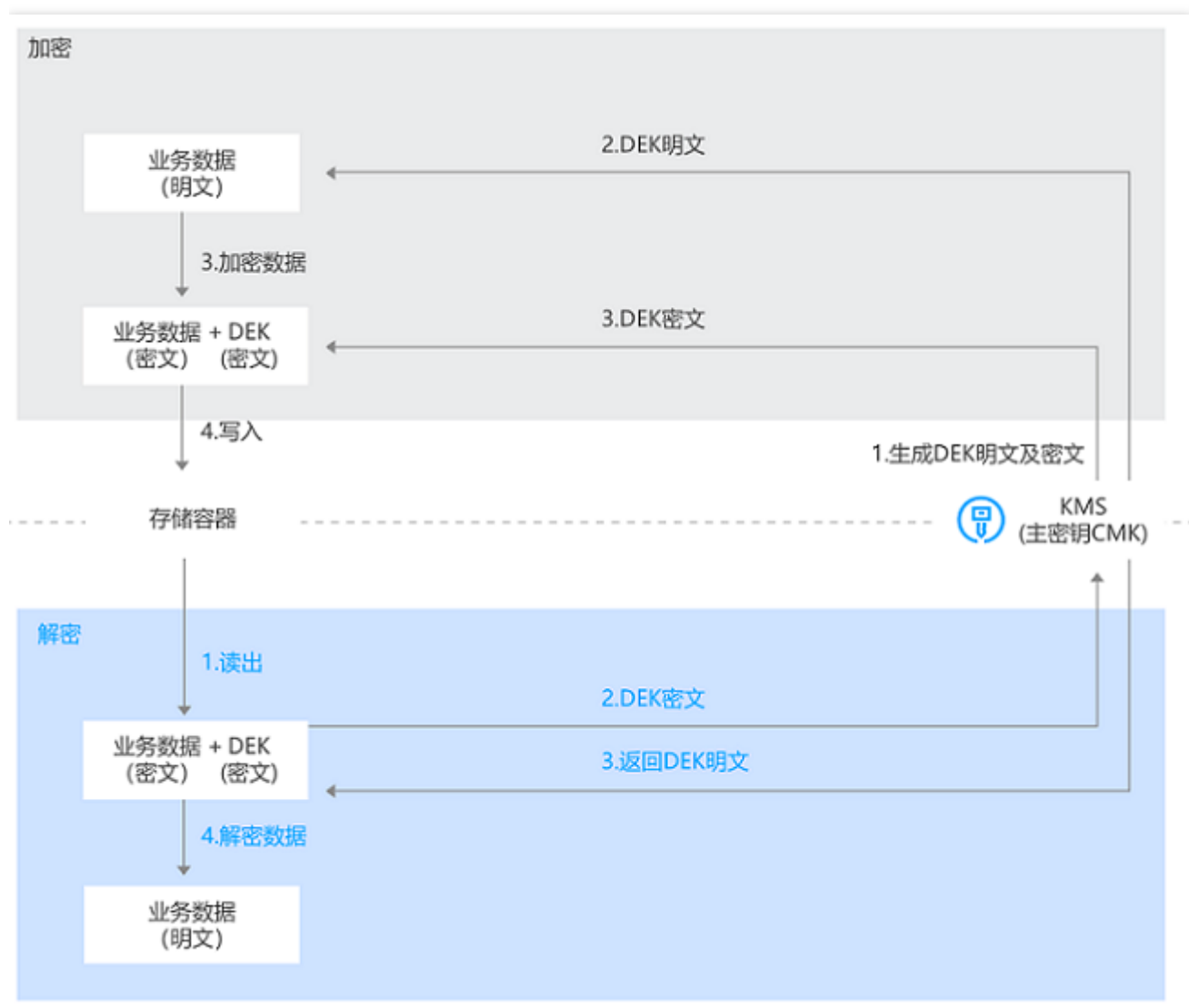
Common symmetric encryption algorithms include AES and 3DES. The most popular asymmetric encryption algorithm is RSA, whose reliability lies in the difficulty in factorizing extremely large integers.

TDE Threat Model

TDE is mainly used to protect static data (data at rest) to prevent data leakage caused by disk theft.

TencentDB for PostgreSQL Data Encryption Implementation

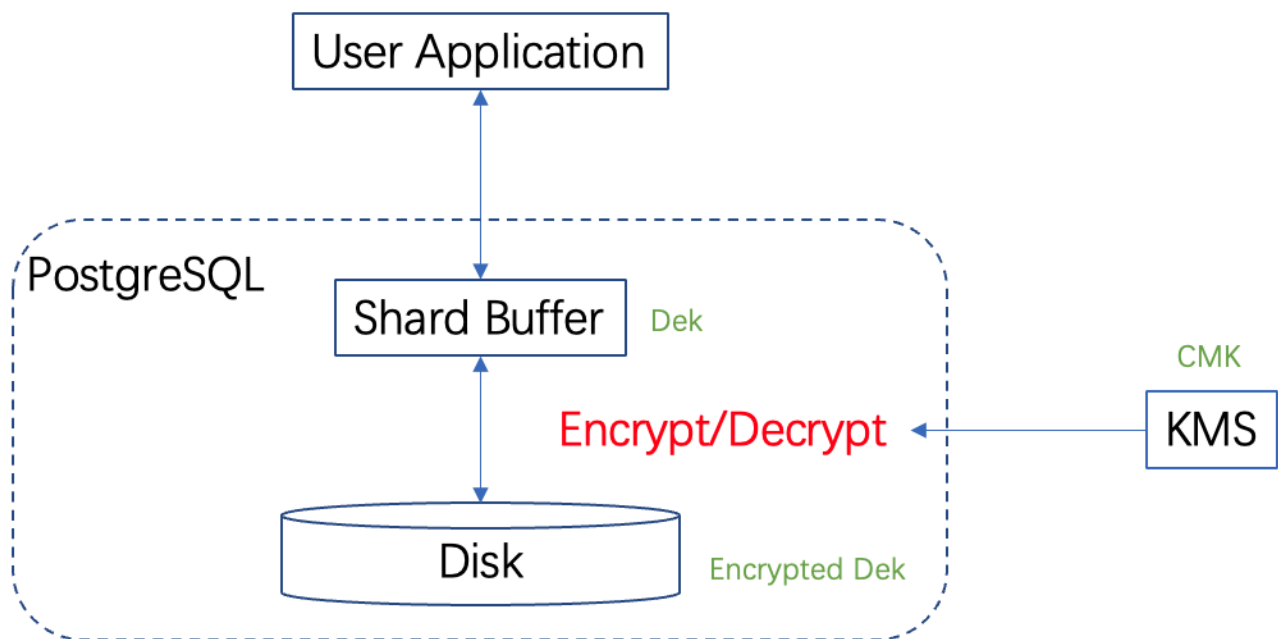
TencentDB for PostgreSQL applies to use the CMK stored in KMS to generate the DEK ciphertext and plaintext. Then, it uses them to encrypt and decrypt the keys used to encrypt Tencent Cloud product data.



This encryption scheme is called envelope encryption, where a key is used to encrypt another key. It has a high

performance in encrypting and decrypting massive amounts of data. Specifically, it can generate DEKs to encrypt and decrypt local data, which guarantees the randomness and security of data keys based on KMS while meeting the requirements for robust business encryption.

As encryptions and decryptions are in-memory operations, the database will get key materials from KMS every time it is restarted or its memory is closed. No key materials used for decryption are stored in the local storage.



Enabling TDE

Last updated : 2022-07-31 16:54:56

Overview

TencentDB for PostgreSQL comes with the transparent data encryption (TDE) feature. Transparent encryption means that the data encryption and decryption are transparent to users. TDE supports real-time I/O encryption and decryption of data files. It encrypts data before it is written to disk, and decrypts data when it is read into memory from disk, which meets the compliance requirements of static data encryption.

Prerequisites

- TDE can be enabled only during instance creation and cannot be disabled once enabled.
- Only the kernel version PostgreSQL v10.17_r1.2、v11.12_r1.2、v12.7_r1.2、v13.3_r1.2、v14.2_r1.0 supports TDE.
- KMS must be activated. If it is not activated, you can purchase and activate it [here](#).
- To use TDE with a sub-account, you must create a service role for authorizing TencentDB for PostgreSQL to manipulate KMS. You can create a role [here](#) with your root account.
- The sub-account must have the `cam:PassRole` , `kms:GetServiceStatus` , and `kms:GetRegions` permissions. If a permission is not granted, use the root account to grant it to the sub-account.

Note :

- The keys used for encryption are generated and managed by [KMS](#). TencentDB for PostgreSQL does not provide keys or certificates required for encryption.
- TDE does not incur fees, but KMS may. For more information, see [Billing Overview](#).

- If your account has overdue payment, you can't get keys from KMS, which may cause migration, upgrade, and other tasks to fail. For more information, see [Notes on Arrears](#).

Notes

- Once enabled, TDE cannot be disabled. If you revoke the key authorization, TDE will become unavailable after the database is restarted.

- After TDE is enabled, data backups will also be encrypted, so even if a backup file is leaked, you don't need to worry about data leakage. To restore data from a backup, use the [instance cloning](#) feature of TencentDB for PostgreSQL.
- TDE enhances the security of static data while compromising the read-write performance of encrypted databases. Therefore, use the feature based on your actual needs. Tests show that the average performance loss is around 2%–3%.
- If the primary instance is associated with a read-only instance, TDE will be automatically enabled for and cannot be disabled by the read-only instance.
- After TDE is enabled, your account balance must be greater than or equal to zero. Otherwise, instance migration may fail as KMS is inaccessible.
- To avoid accidental instance deletion, Tencent Cloud supports key protection. If an instance is configured with [data encryption](#), the key won't be immediately unbound after the instance is isolated and eliminated. The key cannot be [deleted](#) from KMS until three days after the instance is eliminated from the recycle bin.

Directions

1. Log in to the [TencentDB for PostgreSQL purchase page](#) and enable the database encryption feature in **Enable Encryption**.
2. In the pop-up window, select a key and click **OK**.

Note :

- An instance with TDE enabled cannot be restored from a physical backup to a self-created database on another server.
- Once you enable TDE, you cannot disable it.

- **KMS Service:** If KMS is not activated, you need to [purchase it](#) first.
- **KMS Key Authorization:** If a message indicating that you are not authorized is displayed, you can click the **authorization link** to enter the role authorization page and authorize TencentDB for PostgreSQL to manipulate KMS with a service role.
- **Select Key:**
 - Select the KMS region based on your instance region. If **No KMS service in the region** is displayed, KMS is unavailable in the selected region, and you cannot enable encryption.
 - If you select **Use key auto-generated by Tencent Cloud**, the key will be auto-generated by Tencent Cloud.
 - If you select **Use existing custom key (BYOK)**, you can select a key created by yourself.

Note :

If there are no custom keys, click **Go to create** to create keys in the KMS console. For more information, see [Creating a Key](#).

Tenant and Resource Isolation

Database Resource Isolation

Last updated : 2024-08-09 15:51:32

Application Scenario

In some service scenarios, the database object database corresponds to related service logic. A typical example is the SaaS scenario where the database corresponds to the tenant. Therefore, it is necessary to address the issue of database resource isolation. This document mainly describes how to achieve CPU isolation for databases in TencentDB for PostgreSQL instances.

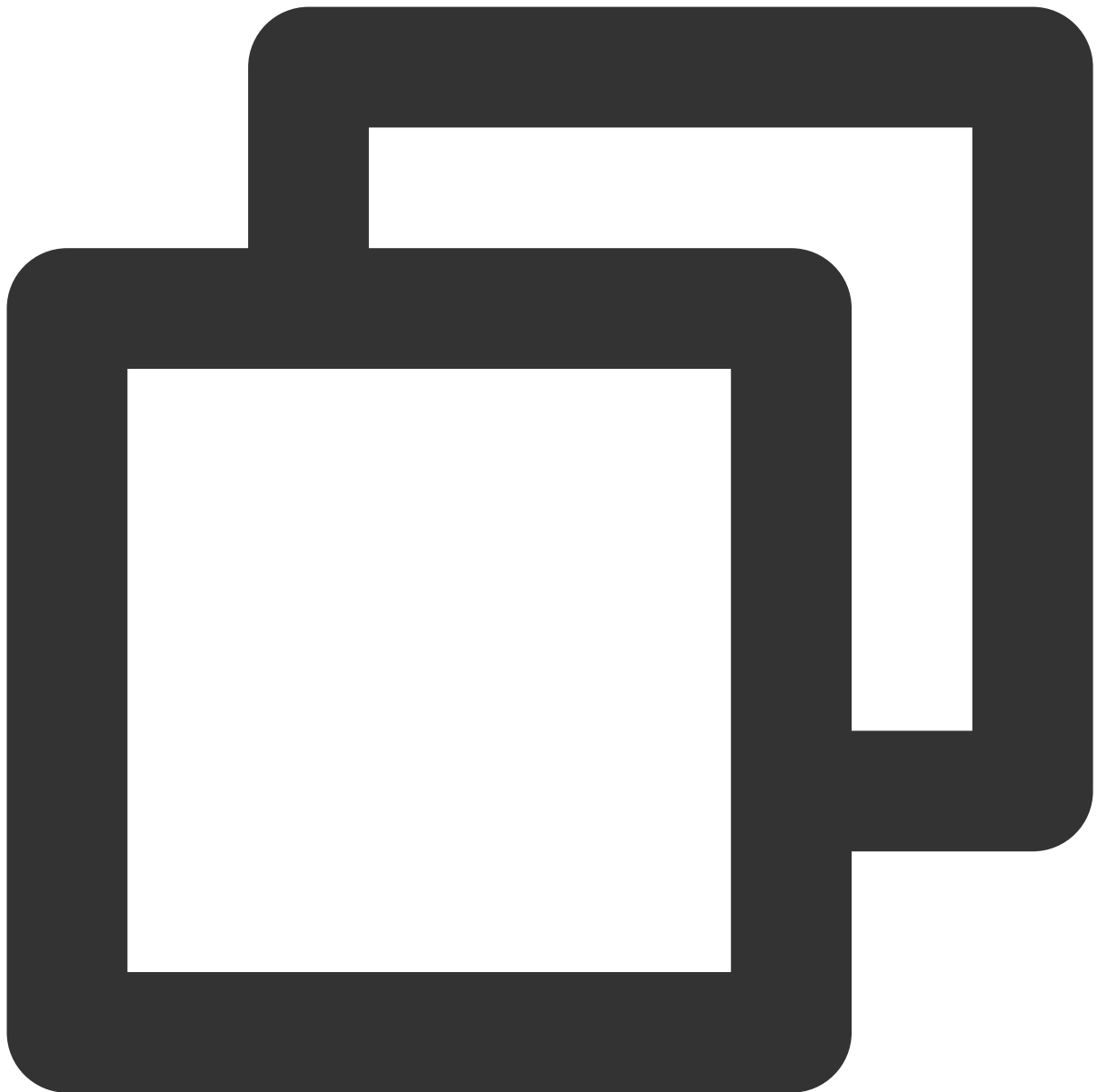
Setting Database Resource Isolation

Note:

Currently, only PostgreSQL 14 kernel versions v14.11_r1.21 and above support resource isolation capabilities. To enable the database resource isolation mode, [submit a ticket](#) to contact us to enable the `tencentdb_serverless` plugin and set relevant parameters.

Among others, `tencentdb_serverless.min_cpu_cores` is the minimum number of CPU cores that can be set for the instance, and `tencentdb_serverless.max_cpu_cores` is the maximum number of CPU cores that can be set for the instance. These two parameters are mainly used for the back-end management system to control the resource isolation of the database within the instance, and users do not need to modify them.

Once you have enabled the `tencentdb_serverless` plugin and set the relevant plugin parameters `tencentdb_serverless.min_cpu_cores` and `tencentdb_serverless.max_cpu_cores`, you can start the configuration. You can use the following command to check that the plugin has been installed successfully:



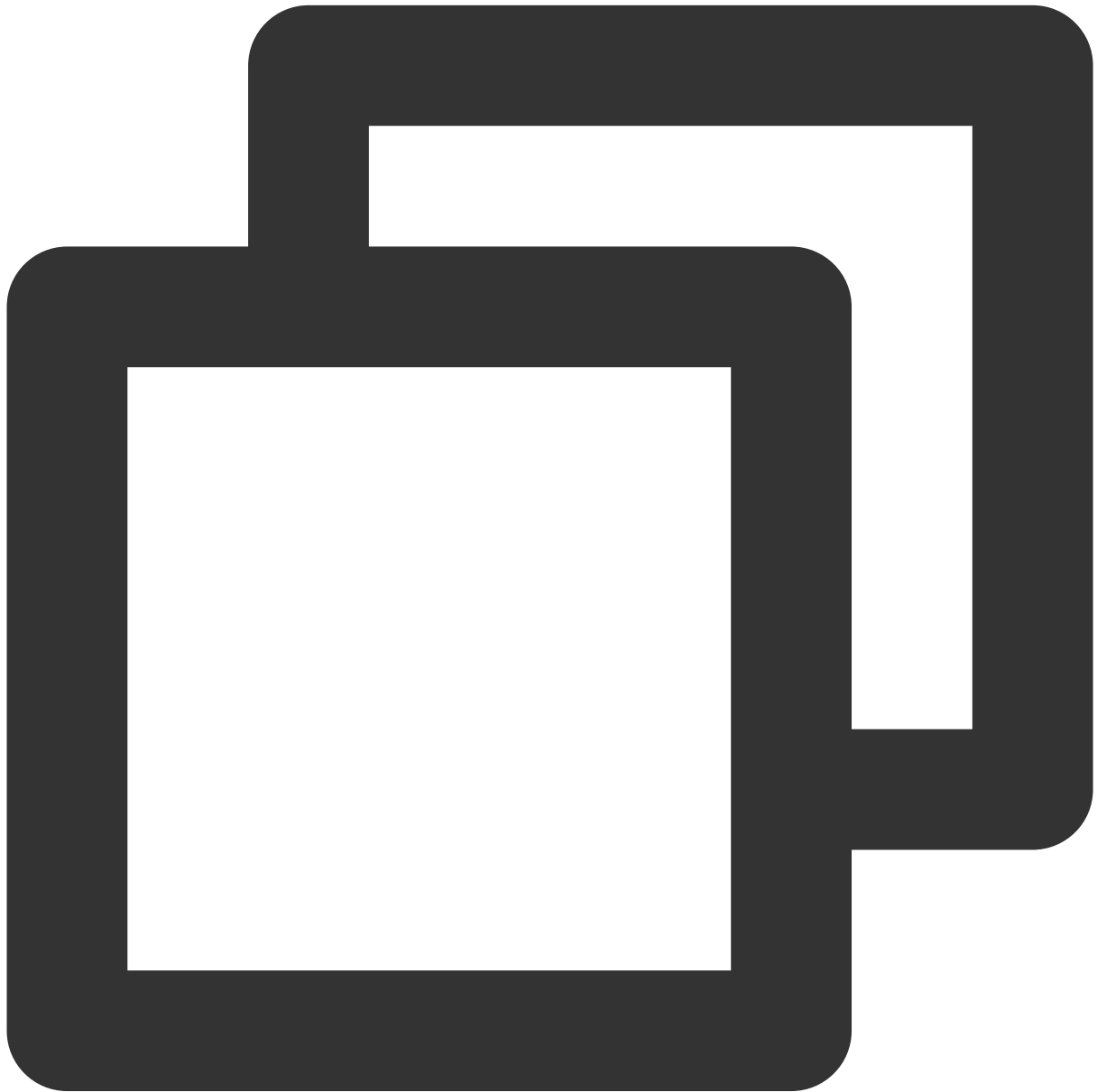
```
postgres=> \dx;
```

List of installed extensions

Name	Version	Schema	Description
pg_stat_log	1.0	public	track runtime execution statistics
pg_stat_statements	1.9	public	track planning and execution statistics
plpgsql	1.0	pg_catalog	PL/pgSQL procedural language
tencentdb_serverless	1.0	public	extension for serverless mode
tencentdb_system_stat	1.0	public	track execution statistics of queries

(5 rows)

The initial values of the `tencentdb_serverless.min_cpu_cores` and `tencentdb_serverless.max_cpu_cores` parameters are the current number of cores of the instance. These two parameters are mainly used for the back-end management system to control the resource isolation of the database within the instance, and users do not need to modify them. If the instance configuration changes later, the two parameters will change accordingly. You can check the current value of the parameters using the following command.



```
postgres=> show tencentdb_serverless.min_cpu_cores;
tencentdb_serverless.min_cpu_cores
```

```
-----
8
```

```
(1 row)

postgres=> show tencentdb_serverless.max_cpu_cores;
 tencentdb_serverless.max_cpu_cores
-----
      8
(1 row)
```

Note:

When there are multiple database objects in the instance, a CPU resource limit needs to be set for each database for the configuration to take effect.

After checking the plugin and parameters, you can start setting the upper and lower CPU resource limits for the database. We provide corresponding functions or views for you to call. Details are as follows:

Setting the CPU Resource Limits of the Specified Database

The function definition is as follows:



```
tencentdb_serverless.set_database_cpu_limit(database_name text [, min_cpu_cores num
```

Call example:



```
postgres=> select tencentdb_serverless.set_database_cpu_limit('tenant_001',2,2.5);
set_database_cpu_limit
-----
(1 row)
```

Clearing the CPU Resource Limits of the Specified Database

The function definition is as follows:



```
tencentdb_serverless.reset_database_limit(database_name text)
```

Call example:



```
postgres=> select tencentdb_serverless.reset_database_limit('tenant_001');
reset_database_limit
-----
(1 row)
```

Clearing the CPU Resource Limits of All Databases in the Instance

The function definition is as follows:



```
tencentdb_serverless.reset_all_database_limit()
```

Call example:



```
postgres=> select tencentdb_serverless.reset_all_database_limit();
reset_all_database_limit
-----

(1 row)
```

Viewing the Details of All Configured CPU Resource Limits in the Current Instance.

We offer the view `tencentdb_serverless.resource_limit_view` for you to view the details of all configured CPU resource limits in the current instance. The field definitions are as follows:

Column name	Meaning
-------------	---------

database_name	The name of the database
min_cpu_cores	The minimum number of CPU cores that the current database can use
max_cpu_cores	The maximum number of CPU cores that the current database can use
min_mem_kilobytes	The maximum memory size that the current database can use, measured in kB. Reserved field, currently not in use.
max_mem_kilobytes	The maximum memory size that the current database can use, measured in kB. Reserved field, currently not in use.

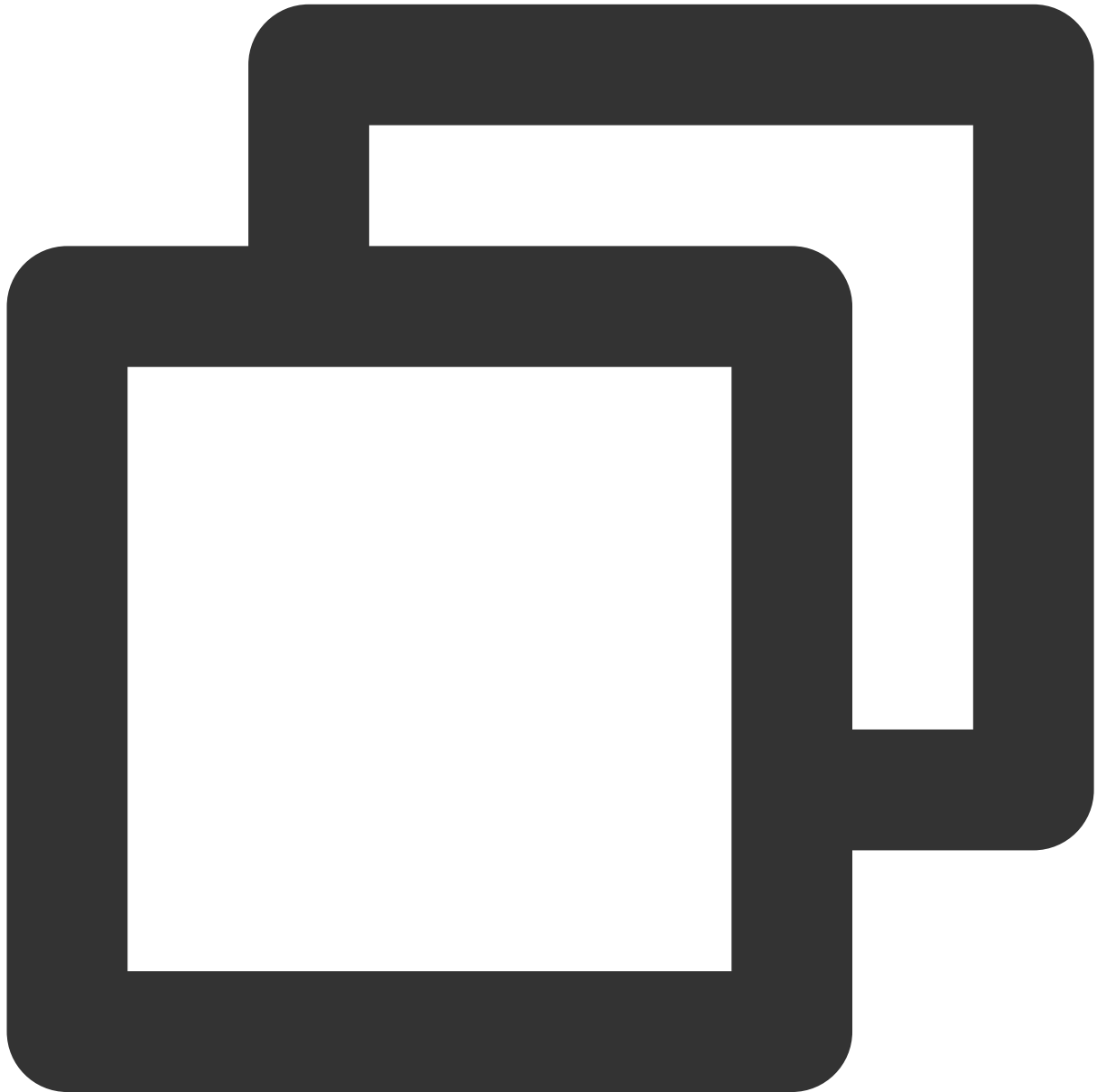
Call example:



```
postgres=> select * from tencentdb_serverless.resource_limit_view;
 database_name | min_cpu_cores | max_cpu_cores | min_mem_kilobytes | max_mem_kilobytes
-----+-----+-----+-----+-----
tenant_001    |          2.0 |          2.5 |                | 
tenant_002    |          2.0 |          2.5 |                | 
(2 rows)
```

Process Monitoring

When all databases in an instance are configured with CPU resource isolation, if the overall resource utilization of the instance is high, each database can ensure the use of the minimum configured CPU cores. Additionally, if you need to check which databases are using more resources in the current system, you can use [Process Monitoring](#) capabilities. To view the CPU resource usage details of all databases, you can use the following statement:



```
postgres=> select datname,sum(cpu_usage) as cpu_usage from tencentdb_process_system
 datname    | cpu_usage
-----+-----
 postgres   |          3
 tenant_001  |         1.99
 tenant_002  |          1
```

(3 rows)

When process monitoring finds that certain database resources have high utilization, you can adjust the CPU configuration of the database in real time. The configuration can **take effect in real time**.

Resource Migration

When a database corresponds to a tenant, and we discover through process monitoring that when the database's resource usage is consistently high and requires resource reintegration, TencentDB for PostgreSQL offers data migration capabilities. You can configure TencentDB for PostgreSQL's [Logical Migration](#). The figure below shows how to configure a migration task:

← **Configure a migration task**

1 ✓ Set source and target databases > 2 Set migration options and select migration objects > 3 Verify task

Migration Method ⓘ • Physical migration Logical migration

Migration Type ⓘ • Structural migration Full migration Full + incremental migration

Migration Object ⓘ • Entire instance Specify object

ⓘ For migration notes, see [Migration FAQs](#) ↗

Previous Save Next

Disabling Database Resource Isolation

If you need to disable the database's CPU resource isolation mode, [submit a ticket](#) to contact us for data cleanup. After receiving the ticket, the back-end engineer will reset all of the CPU resource parameters `tencentdb_serverless.min_cpu_cores` and `tencentdb_serverless.max_cpu_cores`, remove all resource configurations, and finally delete the plugin `tencentdb_serverless`.

Security Groups

Managing Security Groups

Last updated : 2022-03-30 15:39:08

Overview

A [security group](#) is a stateful virtual firewall capable of filtering. As an important means for network security isolation provided by Tencent Cloud, it can be used to set network access controls for one or more TencentDB instances. Instances with the same network security isolation demands in one region can be put into the same security group, which is a logical group. TencentDB and CVM share the security group list and are matched with each other within the security group based on rules. For specific rules and limitations, please see [Security Group Overview](#).

Note :

- TencentDB for PostgreSQL security groups currently only support network access control for VPCs and public networks but not the classic network.
- Security groups that currently support public network access are available only in the Beijing, Shanghai, Guangzhou, and Chengdu regions.
- As TencentDB does not have active outbound traffic, outbound rules are not applicable to TencentDB.
- TencentDB for PostgreSQL primary instances, read-only instances, and read-only instance groups (RO groups) support security groups.

Configuring Security Groups

Step 1. Create a security group

1. Log in to the [CVM console](#).
 2. Select **Security Group** on the left sidebar, select a region, and click **New**.
 3. In the pop-up dialog box, configure the following items and click **OK**.
- **Template:** select a template based on the service to be deployed on the TencentDB instance in the security group, which simplifies the security group rule configuration, as shown below:

Template	Description	Remarks
Open all ports	All ports are open. May present security issues.	-

Open ports 22, 80, 443, and 3389 and the ICMP protocol	Ports 22, 80, 443, and 3389 and the ICMP protocol are opened to the internet. All ports are opened to the private network.	This template does not take effect for TencentDB.
Custom	You can create a security group and then add custom rules. For detailed directions, please see "Step 2. Add a security group rule" below.	The custom template is recommended.

- **Name:** name of the security group.
- **Project:** by default, **DEFAULT PROJECT** is selected. Select a project for easier management.
- **Notes:** a short description of the security group for easier management.

Step 2. Add a security group rule

1. On the [Security Group](#) page, click **Modify Rule** in the **Operation** column on the row of the security group for which to configure a rule.
2. On the security group rule page, click **Inbound rule > Add Rule**.
3. In the pop-up dialog box, set the rule.

- **Type:** **Custom** by default.
- **Source** or **Target:** traffic source (inbound rules) or target (outbound rules). You need to specify one of the following options:

Source or Target	Description
A single IPv4 address or an IPv4 range	In CIDR notation, such as <code>203.0.113.0</code> , <code>203.0.113.0/24</code> or <code>0.0.0.0/0</code> , where <code>0.0.0.0/0</code> indicates all IPv4 addresses will be matched.
A single IPv6 address or an IPv6 range	In CIDR notation, such as <code>FF05::B5</code> , <code>FF05:B5::/60</code> , <code>::/0</code> or <code>0::0/0</code> , where <code>::/0</code> or <code>0::0/0</code> indicates all IPv6 addresses will be matched.
ID of referenced security group. You can reference the ID of: <ul style="list-style-type: none"> ◦ Current security group ◦ Other security group 	<ul style="list-style-type: none"> ◦ To reference the current security group, please enter the ID of security group associated with the CVM. ◦ You can also reference another security group in the same region and belongs to the same project by entering the security group ID.
Reference an IP address object or IP address group object in a parameter template .	-

- **Protocol Port:** enter the protocol type and port range or reference a protocol/port or protocol/port group in a [parameter template](#).

Note :

To connect to TencentDB for PostgreSQL, port 5432 must be opened.

- **Policy:** **Allow** or **Reject**. **Allow** is selected by default.
 - **Allow:** traffic to this port is allowed.
 - **Reject:** data packets will be discarded without any response.
- **Notes:** a short description of the rule for easier management.

4. Click **Complete**.

Use cases

Scenario: you have created a TencentDB for PostgreSQL instance and want to access it from a CVM instance.

Solution: add an inbound security group rule where TCP:5432 is opened.

You can also set **Source** to all or specific IPs (IP ranges) as needed to allow them to access TencentDB for PostgreSQL from a CVM instance.

Inbound or Outbound	Type	Source	Protocol and Port	Policy
Inbound	Custom	All IPs: 0.0.0.0/0 Specific IPs: specify IPs or IP ranges	TCP:5432	Allow

Importing Security Group Rules

1. On the [Security Group](#) page, click the ID/name of the desired security group.
2. On the inbound rule or outbound rule tab, click **Import Rule**.
3. In the pop-up dialog box, select an edited inbound/outbound rule template file and click **Import**.

Note :

As existing rules will be overwritten after importing, we recommend that you export the existing rules before importing new ones.

Cloning Security Groups

1. On the [Security Group](#) page, locate the desired security group and click **More > Clone** in the **Operation** column.
2. In the pop-up dialog box, select the target region and target project, enter the new security group name, and click **OK**. If the new security group needs to be associated with a CVM instance, do so by managing the CVM instances in the security group.

Deleting Security Groups

1. On the [Security Group](#) page, locate the security group to be deleted and click **More > Delete** in the **Operation** column.
2. Click **OK** in the pop-up dialog box. If the current security group is associated with a CVM instance, it must be disassociated before it can be deleted.

Associating Instances with Security Groups

Last updated : 2021-07-27 15:31:25

A security group is an instance-level firewall provided by Tencent Cloud for controlling inbound traffic of TencentDB. You can associate a security group with an instance when purchasing it or later in the console. In TencentDB for PostgreSQL, primary instances, read-only instances, and read-only instance groups (RO groups) can use security groups which are independent from each other.

Note :

- A TencentDB for PostgreSQL instance can associate with up to five security groups.
- The security group of an RO group controls the access address of the RO group itself rather than the access addresses of the read-only instances in this RO group. For example, if the access from an IP is allowed by the security group of an RO group but denied by that of a read-only instance in the RO group, this IP can still access to the read-only instance using the access address of the RO group instead of the access address of the read-only instance.

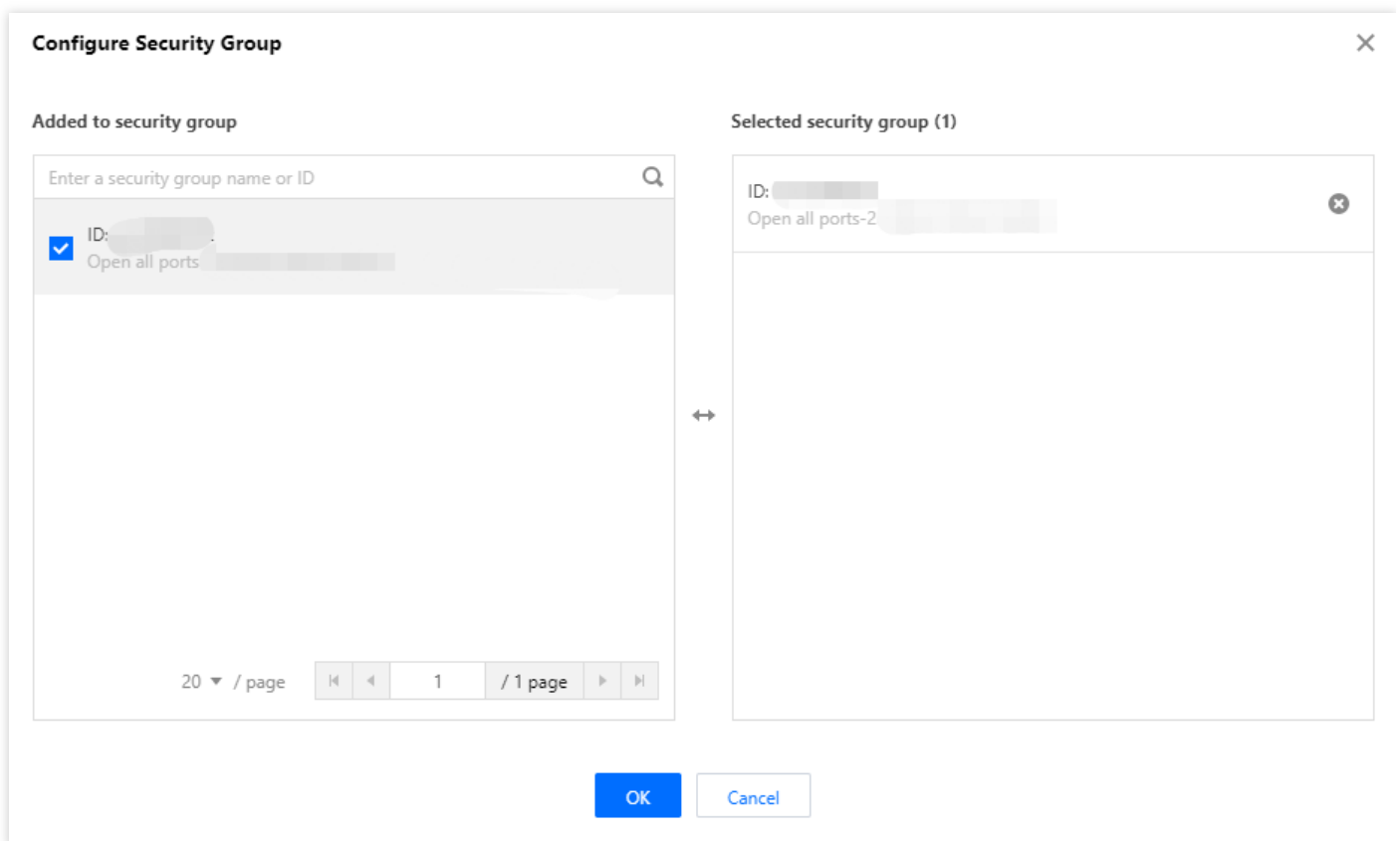
Prerequisites

You have created security groups for TencentDB instances in the security group console. For more information, please see [Managing Security Groups](#).

Associating Security Groups with Primary/Read-Only Instances

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click the ID of the instance to be associated and enter the instance management page.
2. On the **Security Group** page, click **Configure Security Group**.

3. In the pop-up dialog box, select the security group to be associated and click **OK**.



Associating Security Groups with RO Groups

1. Log in to the [TencentDB for PostgreSQL console](#). In the instance list, click the ID of a read-only instance in the desired RO group and enter the instance management page.
2. On the **Security Group** page, select **RO Group** for **Object Type** in the **Security Group Object** section and click **Configure Security Group** in the **Associated Security Group** section.
3. In the pop-up dialog box, select the security group to be associated and click **OK**.

Adjusting the Priorities of Security Groups

You can associate up to five security groups with a TencentDB instance. If you have associated multiple security groups, these security groups are executed based on their priorities. You can adjust the priorities as follows.

1. Log in to the [TencentDB for PostgreSQL console](#), click an instance ID in the instance list, and enter the instance management page.
2. Select the **Security Group** page.

3. In the **Associated Security Group** section, click **Edit**, and click the **Move up** or **Move down** icon in the **Operation** column to adjust the priorities. A security group ranking higher in the list has a higher priority.
Configurations of all the security groups are connected by OR. If the configurations of two security groups conflict, whichever has a higher priority will prevail.
4. After adjusting the priority, click **Save**.

Monitoring and Alarms

Monitoring Feature

Last updated : 2024-01-24 11:20:59

Monitoring in Console

To make it easier for you to view and stay up to date with how instances work, TencentDB for PostgreSQL provides a wide variety of performance monitoring metrics. You can log in to the [TencentDB for PostgreSQL console](#) and view them on the **System Monitoring** tab on the corresponding instance management page.

Monitoring metrics

Metric Name	Metric	Unit	Description
CPU Utilization	cpu	%	Actual CPU utilization
Used Storage Space	storage	GB	Used instance space
Data File Size	data_file_size	GB	Size of data files
WAL File Size	log_file_size	GB	Size of WAL log files
Temp File Size	temp_file_size	MB	Size of temporary files
Storage Space Utilization	storage_rate	%	Total storage space utilization, which includes temporary, data, and log files, as well as other types of database files
Queries per Second	qps	Counts/sec	Average number of executed SQL statements per second
Connection Count	connections	-	Total number of current connections when metric collection is initiated on the database
Connections Created in the Last 5 Sec	new_conn_in5s	-	Total number of connections established in the past five seconds when metric collection is initiated on the database
Active Connections	active_conns	-	Number of currently active (non-idle) connections when metric collection is initiated on the database

Idle Connections	idle_conns	-	Number of idle connections when metric collection is initiated on the database
Waiting Sessions	waiting	-	Number of sessions in waiting status when metric collection is initiated on the database
Sessions Waiting for More Than 5 Sec	long_waiting	-	Number of sessions that stay in waiting status for over five seconds in a collection period
Idle Transactions	idle_in_xact	-	Number of idle transactions when metric collection is initiated on the database
Transactions Executed for More Than 1 Sec	long_xact	-	Number of transactions with an execution duration longer than one second in a collection period
Transactions Idle for More Than 5 Sec	long_idle_in_xact	-	Number of transactions that remain idle for over five seconds during a collection period
Transactions per Second	tps	Counts/sec	Average number of successfully executed transactions (including rollbacks and commits) per second
Transactions Committed /sec	xact_commit	Counts/sec	Average number of committed transactions per second
Transactions Rolled Back /sec	xact_rollback	Counts/sec	Average number of rolled back transactions per second
Request Count	read_write_calls	-	Total number of requests in a statistical period
Read Request Count	read_calls	-	Number of read requests in a statistical period
Write Request Count	write_calls	-	Number of write requests in a statistical period
Other Request Count	other_calls	-	Number of other requests (BEGIN, CREATE, Non-DML, DDL, and DQL operations) in a statistical period
Buffer Cache Hit Rate	hit_percent	%	Hit rate of execution of all SQL statements in a request period

Average Execution Latency	sql_runtime_avg	ms	Average execution latency of all SQL statements in a statistical period
Average of Top 10 Longest SQL Execution Time	sql_runtime_max	ms	Average execution latency of the top ten SQL statements with the longest latency in a statistical period
Average of Top 10 Shortest SQL Execution Time	sql_runtime_min	ms	Average execution latency of the top ten SQL statements with the shortest latency in a statistical period
Remaining XID Count	remain_xid	-	Number of remaining XIDs of the database with the fewest remaining XIDs when metric collection is initiated on the database. This metric is unavailable for read-only instances
Differences Between sent_lsn and replay_lsn	xlog_diff	Byte	Difference between the size of the log sent from the primary node to the standby node and the log replayed on the standby node, which mainly reflects the log application speed of the standby node as well as its performance and network transfer speed. This metric is unavailable for read-only instances
WAL Flush Lag	xlog_diff_time	s	Time difference between the time point when the log is sent from the primary node to the standby node and the time point when the standby node receives the log and flushes it. This metric is unavailable for read-only instances and instances earlier than v10.x
Primary-Standby Sync Delay	slave_apply_delay	s	Primary-Standby sync delay. For primary instances, this metric can reflect the RTO of failover. For read-only instances, it indicates the time after which the data written to primary instances can be queried on the read-only instances. The metric for read-only instances has the same name
Slow Query Count	slow_query_cnt	-	Number of slow queries in a collection period
SQLs Executed for	long_query	-	Number of SQL statements with execution time over one second when metric collection is initiated on the database

More Than 1 Sec			
2PC Transactions	2pc	-	Number of current 2PC transactions when metric collection is initiated on the database
2PC Transactions Uncommitted for More Than 5 Sec	long_2pc	-	Number of current transactions with execution time over five seconds when metric collection is initiated on the database
Rows Deleted /sec	tup_deleted	-	Average number of deleted tuples per second. This metric is unavailable for read-only instances
Rows Inserted /sec	tup_inserted	-	Average number of inserted tuples per second. This metric is unavailable for read-only instances
Rows Updated /sec	tup_updated	-	Average number of updated tuples per second. This metric is unavailable for read-only instances
Rows Scanned in Index Scans /sec	tup_fetched	-	Average number of tuples scanned by the index per second
Rows Scanned in Sequential Scans /sec	tup_returned	-	Average number of tuples scanned in the full table per second
Deadlocks	deadlocks	-	Total number of deadlocks in a collection period

Alarming Feature

Last updated : 2024-01-24 11:20:59

Overview

You can create alarm policies to trigger alarms and send alarm notifications when the TencentDB instance status changes. The created alarm policies can determine whether an alarm needs to be triggered according to the difference between the monitoring metric value and the given threshold at intervals.

You can take appropriate precautionary or remedial measures in a timely manner when the alarm is triggered by changed product status. Therefore, properly created alarm policies can help you improve the robustness and reliability of your applications. For more information on alarms, see [Creating Alarm Policy](#) in Cloud Monitor.

To send an alarm for a specific status of a product, you need to create an alarm policy at first. An alarm policy is composed of three compulsory components, that is, the name, type and alarm triggering conditions. Each alarm policy is a set of alarm triggering conditions with the logical relationship "or", that is, as long as one of the conditions is met, an alarm will be triggered. The alarm will be sent to all users associated with the alarm policy. Upon receiving the alarm, the user can view the alarm and take appropriate actions in time.

Note:

Make sure that you have set the default alarm recipient; otherwise, the default alarm policy of TencentDB won't be able to send notifications.

Directions

Creating an alarm policy

1. Log in to the [Cloud Monitor](#) console and select **Alarm Configuration > Alarm Policy** on the left sidebar.

2. In the alarm policy list, click **Create**.

3. On the **Create Alarm Policy** page, set the policy name, policy type, alarm object, and trigger condition.

An alarm trigger is a semantic condition composed of metric, comparison, threshold, statistical period, and duration.

For example, if the metric is disk utilization, the comparison is $>$, the threshold is 80%, the statistical period is 5 minutes, and the duration is two statistical periods, then the data on disk utilization of a database will be collected once every five minutes, and an alarm will be triggered if the disk utilization exceeds 80% for two consecutive times.

The object instance to be associated with can be found by selecting the region where the object is located or searching for the instance ID of the object.


4. After confirming everything is correct, click **Complete**.

Associating alarm objects

After the alarm policy is created, you can associate alarm objects with it. When an alarm object satisfies an alarm trigger condition, an alarm notification will be sent.

1. In the [alarm policy](#) list, click the name of an alarm policy to enter the alarm policy management page.
2. Click **Add Object** in the **Alarm Object** section.

Alarm Object [Edit](#)

 Regions that have no instances bound to alarm policy are not displayed

Add Object

Unassociate

Unassociate All

Guangzhou(1)

<input type="checkbox"/> ID/Host Name	Network Type	IPv4 Addresses
<input type="checkbox"/> ins [redacted] Unnamedaa	VPC Network	[redacted]

Total items: 1

3. In the pop-up dialog box, select the alarm objects to be associated with, and click **OK**.

Tag

Overview

Last updated : 2024-01-24 11:20:59

Introduction

Tags are key-value pairs provided by Tencent Cloud to easily identify resource. For more information, please see [Product Overview](#).

You can use tags to categorize and manage TencentDB for PostgreSQL resources by various metrics such as business, purpose, and owner. You can also quickly locate a resource by its tag. In Tencent Cloud, the tag key-value pairs have no semantic meaning and are strictly parsed and matched as strings. To use tags, please pay attention to [use limits](#) first.

Here we describe a use case to show how a tag is used.

Use Case Background

A company has three PostgreSQL instances in Tencent Cloud. Those instances are distributed in three gaming businesses whose OPS owners are John, Jane, and Harry.

Configuring Tags

To manage the resources better, the company categorizes its TencentDB for PostgreSQL resources with tags and defines the following tag key-value pairs.

Tag Key	Tag Value
Business	Game 1, game 2, and game 3
OPS owner	John, Jane, and Harry

These tag key-value pairs are bound to TencentDB for PostgreSQL instances in the following way:

instance-id	Business	OPS Owner
postgres-abcdef1	Game 1	Harry
postgres-abcdef2	Game 2	Jane
postgres-abcdef3	Game 3	John

Using Tags

For more information on how to create and delete a tag, please see [Querying Resources by Tag](#).

For more information on how to edit a tag in TencentDB for PostgreSQL, please see [Editing a Tag](#).

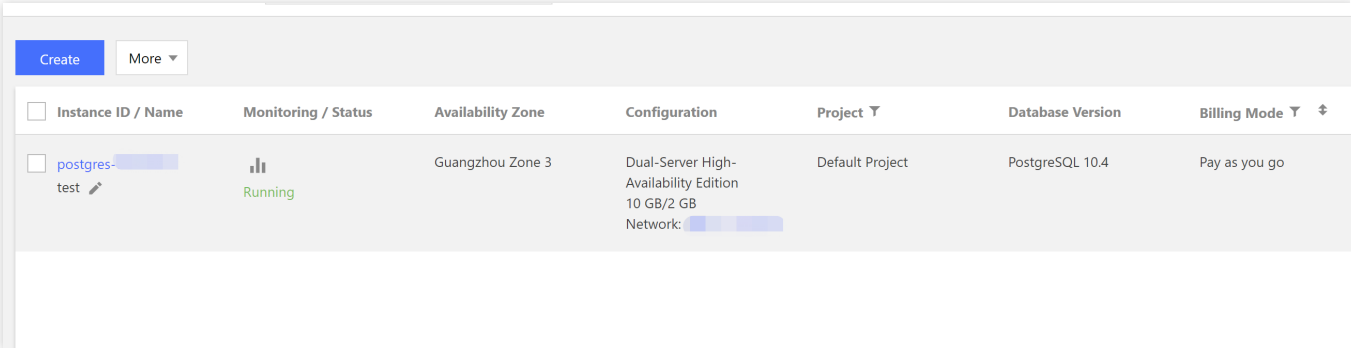
Editing Tag

Last updated : 2024-01-24 11:20:59

You can edit resource tags by the following steps.

Editing the Tag of a Single Instance

1. Log in to the [TencentDB for PostgreSQL Console](#), locate the desired instance in the instance list, and click **More > Edit Tag** in the "Operation" column.



2. In the pop-up dialog box, add, modify, or delete a tag, and click **OK**.

Edit Tags ✕

The tag is used to manage resources by category from different dimensions. If the existing tag does not meet your requirements, please go to [Manage Tags](#) 🔗

1 resource selected

test ▼

test ▼ ✕

Tag key ▼

Tag value ▼ ✕

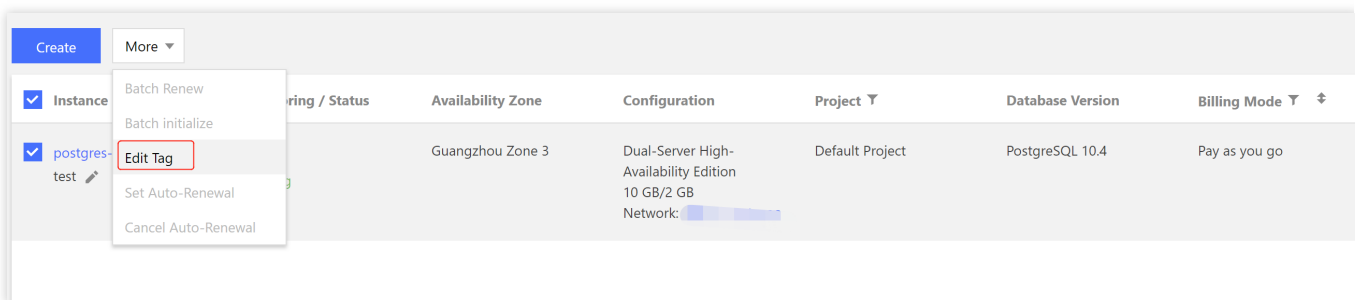
[+ Add](#)

OK

Cancel

Editing the Tags of Multiple Instances

1. Log in to the [TencentDB for PostgreSQL Console](#), select desired instances in the instance list, and click **More > Edit Tag** at the top.



2. In the pop-up dialog box, add, modify, or delete tags, and click **OK**.