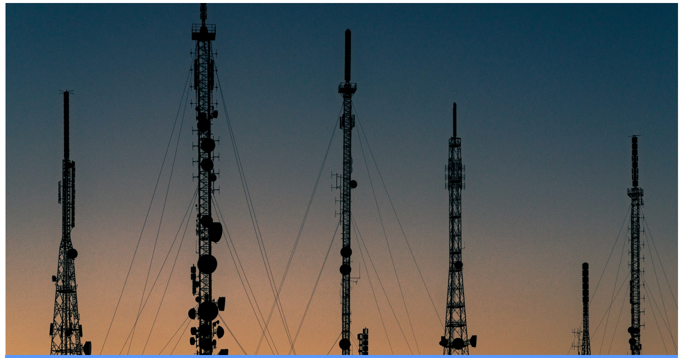
OBCA (OceanBase Database Certification Associate) Certification Training Course

Chapter 1: Difference between distributed database and centralized database

Databases are the core IT infrastructure

Finance

- The growth of Internet business drives the upgrading of core systems
- The core system introduces database distributed and cloud transformation to support horizontal smooth expansion



Operator

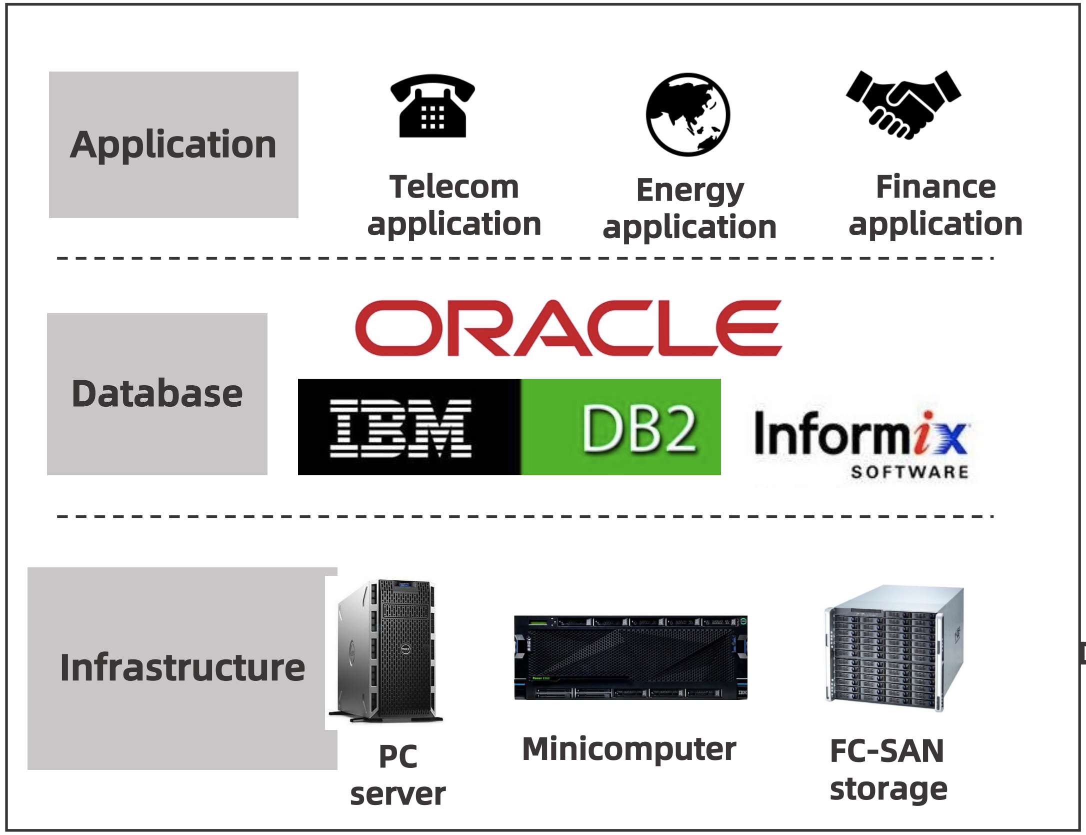
- The large-scale promotion of 5G drives the upgrading of IT systems
- With large bandwidth and ultra-low latency, 5G requires database systems to improve response speed and concurrency

Utility

- Building a smart government
- The "Internet +" business construction with the goal of smart government puts forward higher requirements for the performance and expansion of database

Challenges faced by traditional centralized databases

Traditional database architecture



Advantages

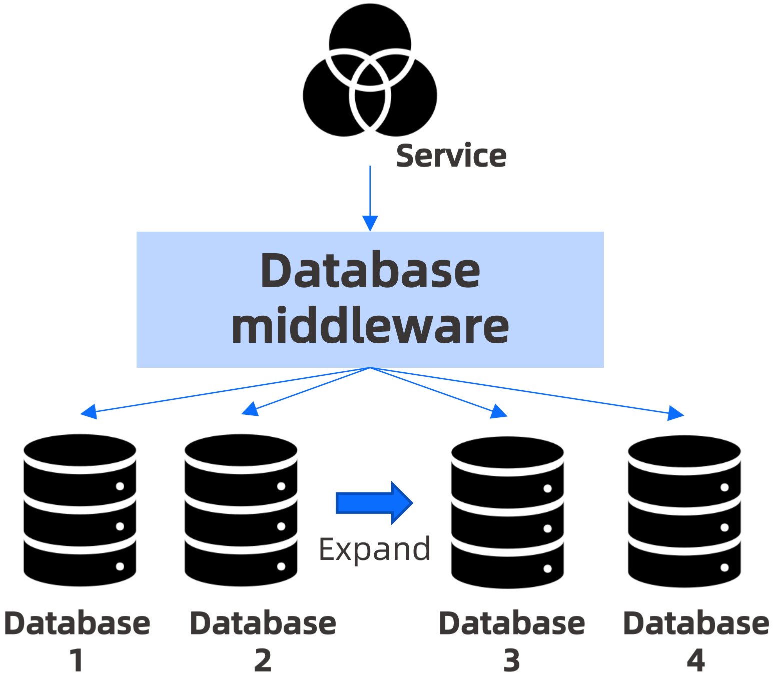
- **Mature and stable**: 40-year development, applications in various industries, mature and stable products & technologies
- **Industry adoption**: Adapted to various demands of different industries
- **Perfect eco-system**: Numerous ISV application developers and technology developers, perfect ecology of technology, industry and talent

Disadvantages

High cost: High software price, depending on high-end hardware, high CAPEX and OPEX cost

No horizontal expansion: Capacity expansion can be done only via increase of equipment performance (more CPU/memory/disk or upgrade from PC server to minicomputer), single-point limit exists

DB/Table sharding scheme by middleware - drawbacks



- The linear expansion can be achieved by generic database model.
- Database server is standalone, no connection in between, unaware of other database existence. Middleware is required to complete cross-database transactions.
- The database middleware connects each database, fulfilling DB shard or table shard.

Advantages

Linear expansion: DB or table sharding allow quick implementation of horizontal database expansion

low technology cost: Few or no re-construction of core database engine

Disadvantages

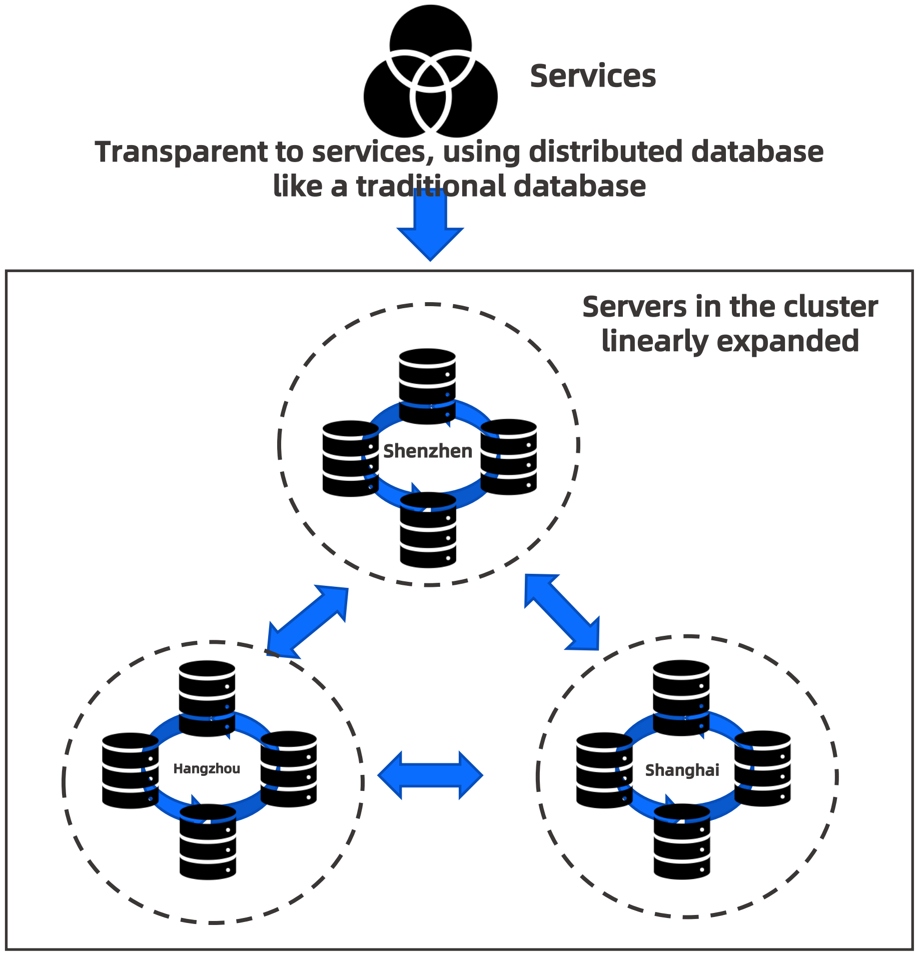
Cross-database distributed transactions: The database core engine has no distributed capability and can only complete distributed processing through middleware, but it is difficult for middleware to achieve RPO=0, so the ACID capability of distributed transactions cannot be guaranteed 100% when exceptions and failures occur

Global consistency: Because timestamps are inconsistent across multiple database servers, it is difficult to ensure global consistency of data version numbers across multiple databases

Load balancing: During capacity expansion or reduction, the underlying database engine cannot adjust data distribution rules online. Therefore, services need to be suspended and data needs to be converted again, posing serious challenges to services and O&M

Complicated database-across SQL: Cross-database complex SQL operations (such as multiple tables to do shard key-independent associative query) can only be completed in middleware, which does not have distributed parallel computing capability, and will eventually limit the use of SQL by applications, resulting in business invasion

Native distributed relational database architecture



Services

Transparent to services, using distributed database like a traditional database

Servers in the cluster linearly expanded

Advantages

High data reliability + High service availability: industrial level implementation of multi-replica consistency protocol Paxos ensures zero data loss (RPO=0) and fast service recovery (RTO<30 seconds) when individual nodes fail

Linear capacity expansion: Capacity expansion with the increase of business volume (such as during online promotion), capacity reduction with the decrease of business volume (such as after promotion)

Low cost: High availability based on common X86 servers without high-end minicomputers and storage

Global consistency: Supports distributed transactions to ensure global consistency and supports distributed complex queries

Flexible deployment mode: Supports multiple deployment modes, such as three-center, five-center, and active/standby mode

Transparent to services: Service systems can use a distributed database just like a single point database, and the cost of business migration and transformation is low

Comparison between OceanBase and Traditional Database

	Traditional centralized database	Distributed database represented by OceanBase
Product architecture	Classic "single point of centralization" architecture adopts "share-everything" architecture. Built on high-end hardware, such as IBM high-end servers and EMC high-end storage devices	Native "distributed" database, using the industry's most stringent Paxos distributed consistency protocol Design based on ordinary PC hardware, no need for high-end hardware
Data reliability and high availability of services	Use high-end hardware to ensure data reliability If the primary node is faulty, data is lost (RPO>0). The service cannot be automatically restored. The service recovery time (RTO) is usually measured in hours	Based on common PC hardware, Paxos distributed consistency protocol is used to ensure data reliability In the case of a primary node failure, Paxos can ensure that data is not damaged (RPO=0), and automatically vote and restore services. The service recovery time (RTO) is less than 30 seconds
Scalability	Data storage can only scale vertically within a single server, and eventually reaches the maximum capacity of the single architecture. Compute nodes usually cannot be extended. In a small number of scenarios (such as RAC and pureScale), compute nodes can be expanded. However, multiple compute nodes still need to access the shared storage, and the number of compute nodes that can be expanded is limited	Both data nodes and computing nodes can be scaled horizontally under the MPP architecture There is no limit on the number of data nodes and compute nodes. You can expand the number of data nodes and compute nodes to any number if the network bandwidth is sufficient
Applications	Core systems focused on corporate customers (finance, telecommunications, government and enterprise, etc.) Unable to cope with Internet business scenarios, few application cases	Core component of Alipay, core of MYBank, many businesses of Alibaba, and many external commercial banks. Gradually penetrating traditional business
Use cost	More expensive There are high-end infrastructure costs, high software licensing fees, and product and service costs	Relatively low PC based hardware design reduces hardware costs, software licensing costs and service costs also have advantages

Summary

After nearly 40 years of development, traditional centralized database has become mature. However, in the era of big data, traditional database still faces many challenges, distributed database can effectively solve these problems, and is the key direction of database development in the future.

1: Traditional databases often have high requirements on hardware infrastructure, and can only be extended vertically, but not horizontally, so it is easy to reach the upper limit of performance;

2: Although the subdatabase and subtable can be horizontally expanded, it also brings new problems such as ACID, which does not support complex SQL and is difficult to guarantee distributed transactions.

3: Distributed database can effectively solve these problems, applications can use distributed database as centralized database, distributed database has low hardware cost, high scalability, high availability and other characteristics.

Q&A

Simulation Questions

1. [True/false] Although the architecture of split database and table solves the scalability problem of centralized database, it also brings new problems (no support for complex SQL, ACID is difficult to guarantee distributed transactions, etc.) ()

2. [Multiple choice] What are the challenges of traditional centralized relational databases? ()

A. high cost: run on high-end servers, minicomputers, high-end storage and other proprietary hardware;

B. Lack of ecology: insufficient documentation, training, application, etc.

C. poor scalability: can not get rid of the stand-alone architecture, only vertical expansion, not horizontal expansion;

D. Poor performance: at any time, the performance of the traditional centralized database is worse than that of the distributed database;

Chapter 2: Introduction to OceanBase database products

1. **Product history and introduction**
2. TPC-C benchmark results
3. Internal and external application use cases

OceanBase： Enterprise-level distributed database

TPC-C world record

Native distributed relationships

Proven in large-scale financial scenario

Next generation database standard

Independent research and development

Service transparency

Strong consistency

Compatibility

Effectively support mixed load HTAP

High availability & expansibility

Multi-tenant

Low cost

Completely independent research and software development, run by individual company held by ANT Group, strategic long-term investment portfolio of the group

100% ownership of intellectual property, differentiated from products built based on open-source databases

Based on distributed architecture and generic hardware, financial level reliability and data consistency, independent of specific hardware architecture

Core technology advantages of high availability, scalability, low cost and high performance

OceanBase History – 10-year accumulation

2010

Product init.

the first user

淘宝 收藏夹

2013

Product

open-source

support SQL

淘宝/天猫直通车
阿里妈妈旗下产品

2014

Support

MYBank

finance-grade

core service

网商银行
MYbank

2016

Alipay core

transactions

100% traffic of

core trans.

支付宝
ALIPAY

2017

Financial

customers

Internet core

system

南京银行
BANK OF NANJING

2019

Break world

record

HTAP analysis

HTAP cloud database

TPC-C 60,880,000

西安银行
BANK OF XI'AN

天津銀行
BANK OF TIANJIN

广东农信
GDRC

2020

spin-off

commercialization

leading independent

research

OCEANBASE

TPC-C 7.07亿

2021

V3.0 launch

Official open

source

FORRESTER®

Listed in

Forrester

Distributed

Database Report

OCEANBASE

Certifications, Patents and Awards

Certifications

- Classified protection level 3
- CNAS（ISO 9001, ISO 27001)

Patents

- OceanBase has made technological breakthroughs in four aspects, including lossless DISASTER recovery and remote multi-live based on Paxos, distributed transaction processing supporting linear expansion, low-cost storage compression based on LSM tree, and HTAP mixed load supported by the same engine, forming a core patent group (49 issued patents, total 189 patent applications), a series of tool sets (3 software Copyrights) and industry technical standards (8 submitted manuscripts and proposals). Based on the project results, a Monograph in Chinese has been published.

Awards

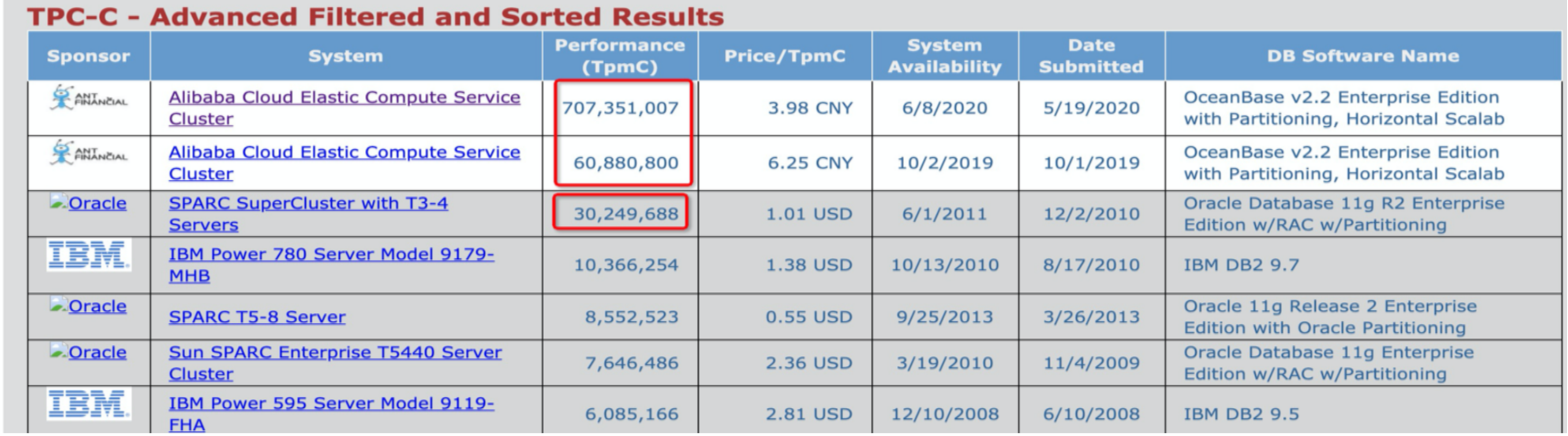
- 2011 China, Japan and South Korea Open Source Software Competition technical excellence Award
- 2016, the world Internet leading scientific and technological achievements
- 2016, The highest Award of Ant Financial Group – CEO Award
- 2019, the Best Innovative Product of the Year in China Database Technology Selection
- 2019, DOIT China Data and Storage Summit - Database Product of the Year Gold Award
- 2019, IT168 Database Selection - Annual Technical Excellence Award
- 2020, Ant Financial Superma -- Pursuit of Excellence Award

Chapter 2: Introduction to OceanBase database products

1. Product history and introduction
2. **TPC-C benchmark results**
3. Internal and external application use cases

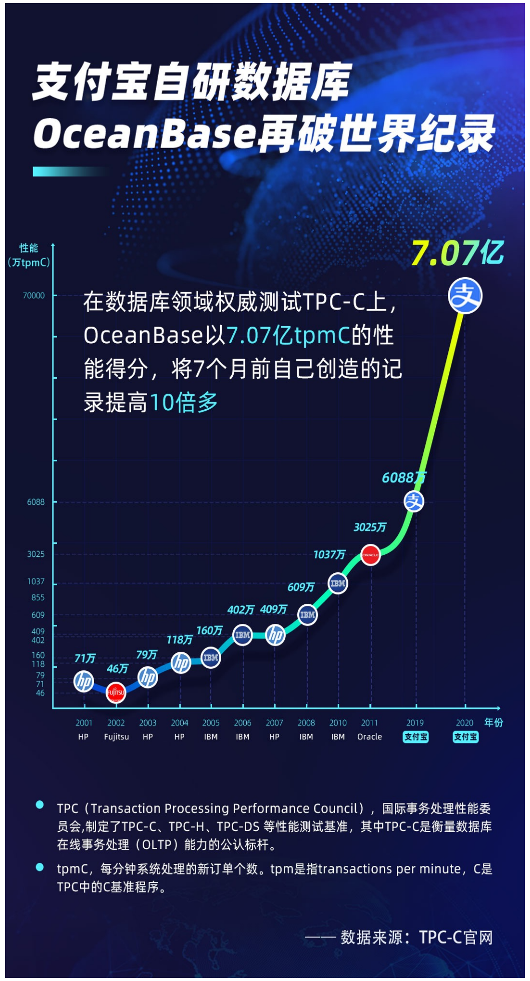
TPC-C Certification

milestones of distributed database in China



TPC-C - Advanced Filtered and Sorted Results

Sponsor	System	Performance (TpmC)	Price/TpmC	System Availability	Date Submitted	DB Software Name
ANT FINANCIAL	Alibaba Cloud Elastic Compute Service Cluster	707,351,007	3.98 CNY	6/8/2020	5/19/2020	OceanBase v2.2 Enterprise Edition with Partitioning, Horizontal Scalab
ANT FINANCIAL	Alibaba Cloud Elastic Compute Service Cluster	60,880,800	6.25 CNY	10/2/2019	10/1/2019	OceanBase v2.2 Enterprise Edition with Partitioning, Horizontal Scalab
Oracle	SPARC SuperCluster with T3-4 Servers	30,249,688	1.01 USD	6/1/2011	12/2/2010	Oracle Database 11g R2 Enterprise Edition w/RAC w/Partitioning
IBM	IBM Power 780 Server Model 9179-MHB	10,366,254	1.38 USD	10/13/2010	8/17/2010	IBM DB2 9.7
Oracle	SPARC T5-8 Server	8,552,523	0.55 USD	9/25/2013	3/26/2013	Oracle 11g Release 2 Enterprise Edition with Oracle Partitioning
Oracle	Sun SPARC Enterprise T5440 Server Cluster	7,646,486	2.36 USD	3/19/2010	11/4/2009	Oracle Database 11g Enterprise Edition w/RAC w/Partitioning
IBM	IBM Power 595 Server Model 9119-FHA	6,085,166	2.81 USD	12/10/2008	6/10/2008	IBM DB2 9.5



支付宝自研数据库
OceanBase再破世界纪录

在数据库领域权威测试TPC-C上，OceanBase以7.07亿tpmC的性能得分，将7个月前自己创造的记录提高10倍多

- TPC（Transaction Processing Performance Council），国际事务处理性能委员会,制定了TPC-C、TPC-H、TPC-DS 等性能测试基准，其中TPC-C是衡量数据库在线事务处理（OLTP）能力的公认标杆。
- tpmC，每分钟系统处理的新订单个数。tpm是指transactions per minute，C是TPC中的C基准程序。

—— 数据来源：TPC-C官网

The first certified database of China

- Chinese database participated for the first time, breaking the world record held by Oracle for nine years, showing the strength of Chinese database to the world

The first certified distributed database

- The world's first certified distributed database overcomes the challenges of tPC-C certification specification for distributed environment
- It is proved that distributed database can achieve horizontal scaling under ACID guarantee

Milestone of distributed database in China

- It proves that Chinese distributed database can also pass the most stringent OLTP evaluation and can be competent for the core transaction scenarios of key businesses

TPC-C Certification

Standard public cloud environment, cost-effective

- **OceanBase cluster scale**

  1,557 nodes, 65,394vCPU, ecs.i2d.21xlarge

- **Client scale**

  400 nodes, 20,800vCPU, ecs.ebmg6.26xlarge

- **Performance**

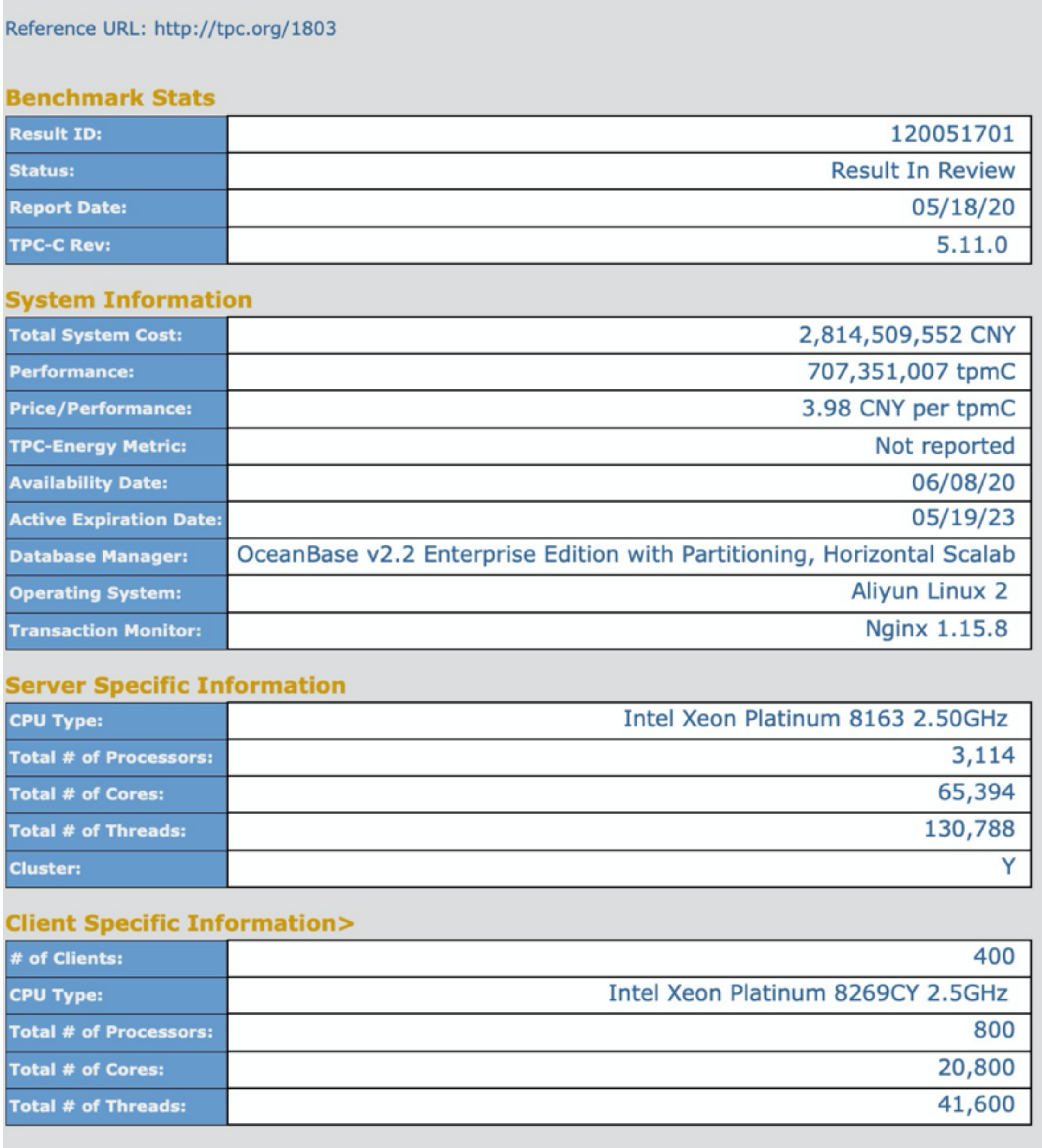
  707,350,000 tpmC

- **Stability**

  Continuous operation 8 hours with load, performance jitter is not more than 0.5%

- **Cost effectiveness**

  $0.6/tpmC



Reference URL: http://tpc.org/1803

Benchmark Stats

Result ID:	120051701
Status:	Result In Review
Report Date:	05/18/20
TPC-C Rev:	5.11.0

System Information

Total System Cost:	2,814,509,552 CNY
Performance:	707,351,007 tpmC
Price/Performance:	3.98 CNY per tpmC
TPC-Energy Metric:	Not reported
Availability Date:	06/08/20
Active Expiration Date:	05/19/23
Database Manager:	OceanBase v2.2 Enterprise Edition with Partitioning, Horizontal Scalab
Operating System:	Aliyun Linux 2
Transaction Monitor:	Nginx 1.15.8

Server Specific Information

CPU Type:	Intel Xeon Platinum 8163 2.50GHz
Total # of Processors:	3,114
Total # of Cores:	65,394
Total # of Threads:	130,788
Cluster:	Y

Client Specific Information>

# of Clients:	400
CPU Type:	Intel Xeon Platinum 8269CY 2.5GHz
Total # of Processors:	800
Total # of Cores:	20,800
Total # of Threads:	41,600

Public misunderstanding of TPC-C benchmark

Myth 1: "Just a run-scoring test, trying to improve your TPS by any means possible."

- TPC-C, with a history of nearly 30 years, constantly improves and optimizes the evaluation process, blocking out opportunistic means and striving to simulate the most realistic use scenarios. The hardest part of the TPC-C test is not to run a high score, but to run a high score when 100% of the strict specifications are met. After the test, auditors log on to the system, collect data, check data, and test results that do not meet the expected requirements are discarded.

Myth 2: "OceanBase compares the latest hardware to Oracle nine years ago, and it doesn't beat it."

- All the traditional database vendors have given up competing with Oracle, even as the hardware continues to evolve.

Myth 3: "Distributed databases have an advantage in TPC-C metrics, and separate databases and tables work well. "

- Distributed environment faces more challenges in TPC-C evaluation than traditional "centralized" database, such as performance challenges brought by distributed transactions and ACID challenges, and performance challenges brought by item tables. Therefore, prior to OceanBase, there had never been a distributed database participating in TPC-C metrics

Myth 4: "I can get a very high TPC-C score with very few machines, unlike OceanBase, which needs so many machines. "

- In order to make the evaluation as close as possible to the real business scenarios, TPC-C specifications stipulate that each warehouse can only produce 12.86 TPMCS at most, so to reach 700 million TPMCS, 54 million warehouses are needed, and 4320TB of stock data (single copy) is needed based on 80MB data of each warehouse. This does not include the historical order data to be saved during testing, which the TPC-C standard requires at least 60 days of historical order retention.
- As a result, few vendors comply with the specification during testing unless they participate in formal TPC-C assessments, so fewer machines are needed. However, such tests are not orders of magnitude more difficult than real TPC-C assessments and do not reflect the characteristics of real OLTP scenarios.

Chapter 2: Introduction to OceanBase database products

1. Product history and introduction

2. TPC-C benchmark results

3. **Internal and external application use cases**

Industrial applications and value point

Financial transactions

Easy to achieve multiple databases and live. Meets the requirements of high concurrency, low delay and transnationality of financial transaction system. In the minority failure case, RPO=0, RTO<30 seconds.

Telecom billing

The feature of quasi - memory database satisfies the real - time application scenario. At the same time, OceanBase's distributed concurrent SQL engine can well support OLAP applications with massive data.

Insurance

Based on common hardware and local storage, OceanBase naturally has good scale-out capabilities, enabling capacity expansion and reduction without interrupting services.

Internet start-ups

At the beginning of its establishment, OceanBase adopted the multi-tenant mode based on cloud database architecture. Resources among tenants are isolated from each other, which can provide DBaaS capability and reduce IT and operation and maintenance costs.

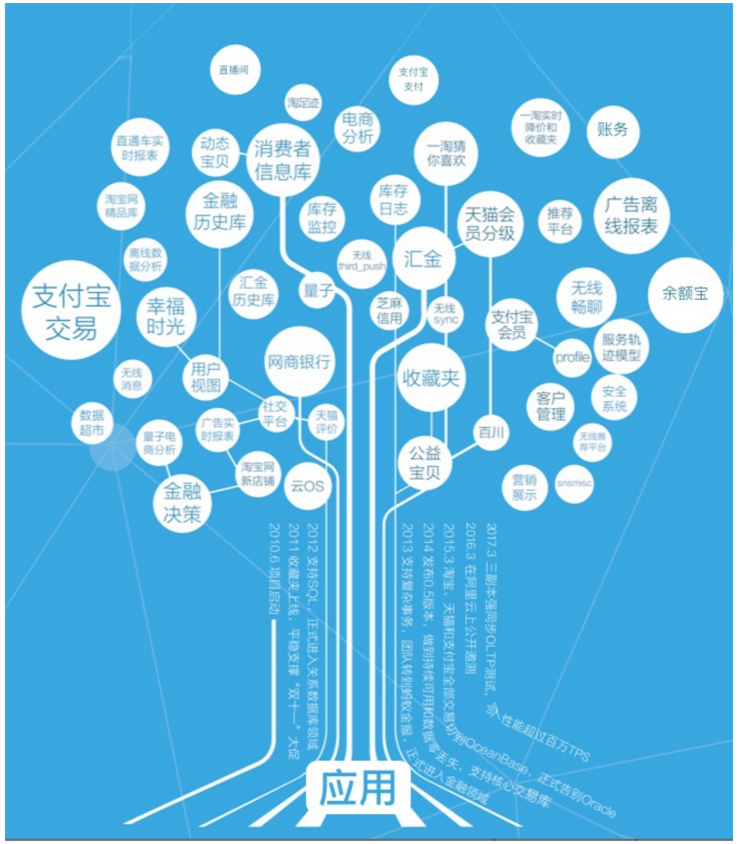
Internal application: All key core systems of Ant Group are running on OceanBase

Sustained the 61,000,000 database accesses/s (TPS+QPS) during the Double Eleven in 2019, proving the performance and stability of OceanBase database

Taobao has 300 million daily active users and 800 million monthly active users, each of whom uses "favorites" and "footprints" services. "Favorites" read requests run into millions of times per second, with the largest tables holding hundreds of billions of records

The OceanBase database was adopted for all transactions of MyBank at the beginning of its establishment, and the three-place and five-center architecture ensured data security in an all-round way

Developed international business, Paytm master station core database, adopted by two of the three e-payment providers



- **It has been applied in hundreds of key internal businesses of Alibaba/Ant, such as Alipay 100% core link, Yu 'ebao, Taobao favorites, Taobao Footprints, Ali Mom report analysis, etc.**

Join OceanBase Certification Program become a master of distributed database

Certification Benefits

01

Embrace the trend of distributed database development to update knowledge system

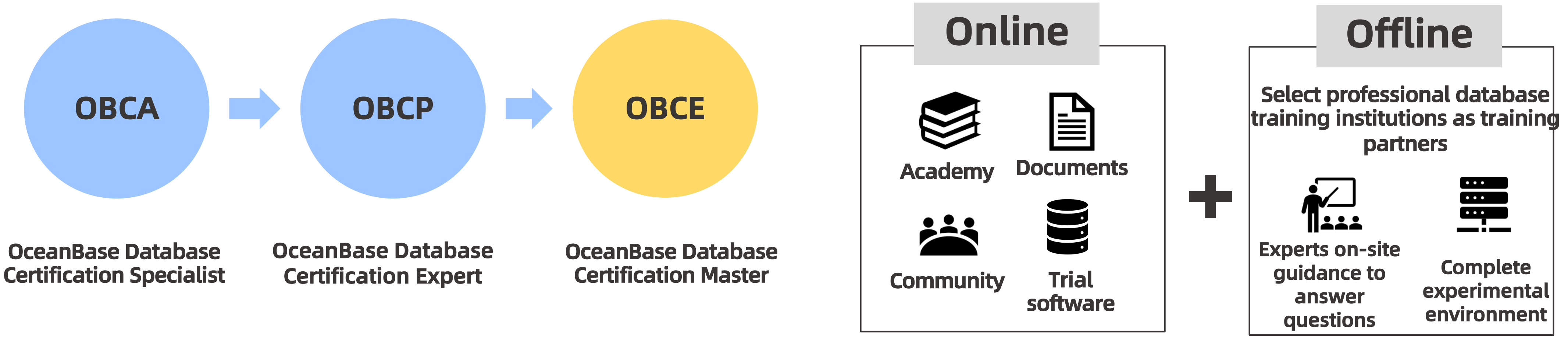
02

Explore the core technology of distributed database that broke TPC-C record twice

03

Join OceanBase's extensive industrial chain

Certification path



OBCA → OBCP → OBCE

OBCA: OceanBase Database Certification Specialist

OBCP: OceanBase Database Certification Expert

OBCE: OceanBase Database Certification Master

Online

- Academy
- Documents
- Community
- Trial software

+

Offline

Select professional database training institutions as training partners

- Experts on-site guidance to answer questions
- Complete experimental environment

Summary

This chapter introduces the basic information and development history of OceanBase products:

1. OceanBase is a database product with 100% intellectual property rights, which is different from the re-release product of open source database.

2. OceanBase realizes financial level reliability and data consistency based on distributed architecture and universal server, independent of specific hardware architecture;

3. TPC is currently the only public testing standard with credibility in the world for the combination of database functions and performance. TPC-C test has strict specifications, and only when it passes the formal audit of TPC organization is the real TPC-C score;

4. OceanBase broke the TPC-C test record twice and used ali Cloud public cloud general model. The test environment was consistent with the production system.

Q&A

OCEANBASE

Simulation Questions

1.[True/false] TPC-C is a running test, there are no rules, as long as you can run high. ()

2.[True/false] OceanBase database was incubated internally by Alibaba and Ant for 10 years before it was gradually rolled out to the outside market. ()

3.[True/false] OceanBase database is a re-release based on open source databases. ()

4. [Single choice] What type of database is OceanBase ()

A: Centralized database;

B: NoSQL database ;

C: Distributed relational database;

5. [Multiple choice] What are the core features of OceanBase? ()

A: High scalability, can use ordinary PC server for horizontal expansion;

B: High performance, peak peak 61 million times/SEC, single table maximum 320 billion rows;

C: High availability. Paxos ensures strong consistency. RPO=0 and RTO<30 seconds.

D: High compatibility, supporting MySQL and Oracle modes, reducing the cost of business migration and transformation;

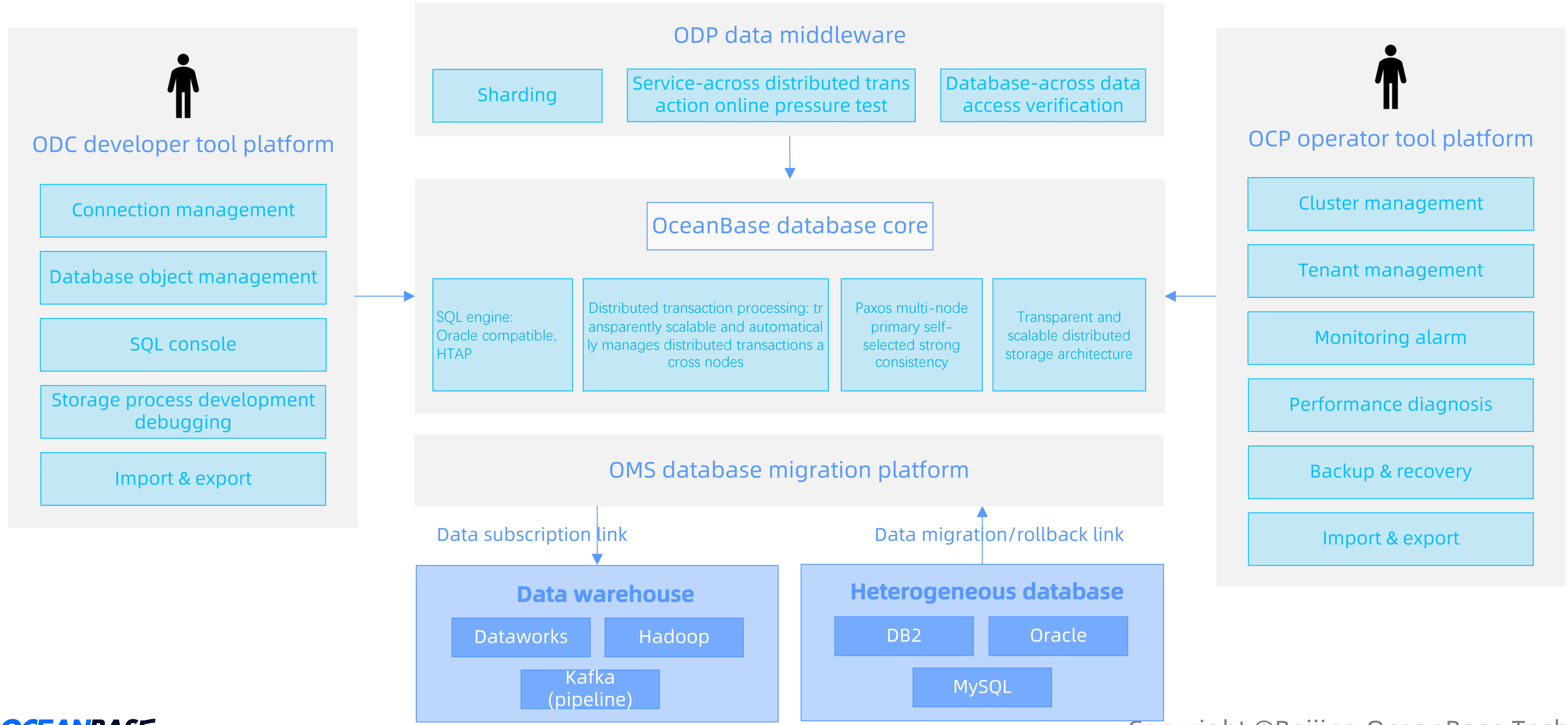
E: High cost, using minicomputers, high-end storage and other proprietary hardware;

Chapter 3: OceanBase product family and basic concepts

1. **Product family**
2. Installation & deployment
3. Data Import
4. Basic concepts

OceanBase database product family

1 + 4 perfect overall product platform scheme



ODP data middleware

- Sharding
- Service-across distributed transaction online pressure test
- Database-across data access verification

OceanBase database core

- SQL engine: Oracle compatible, HTAP
- Distributed transaction processing: transparently scalable and automatically manages distributed transactions across nodes
- Paxos multi-node primary self-selected strong consistency
- Transparent and scalable distributed storage architecture

ODC developer tool platform

- Connection management
- Database object management
- SQL console
- Storage process development debugging
- Import & export

OCP operator tool platform

- Cluster management
- Tenant management
- Monitoring alarm
- Performance diagnosis
- Backup & recovery
- Import & export

OMS database migration platform

Data subscription link

Data migration/rollback link

Data warehouse

- Dataworks
- Hadoop
- Kafka (pipeline)

Heterogeneous database

- DB2
- Oracle
- MySQL

OCEANBASE

OceanBase Database Core

Scalability

- Horizontal scale
- On-demand online capacity scale-out, scale-in, un-interrupted service
- Over 100 servers per cluster

High performance

- Peak 61,000,000/s (real business system)
- Up to 320,0 billion rows per table (real business system)
- semi-memory processing performance

High availability

- Based on Paxos protocol, strong consistency
- No data loss or service interruption caused by failed data replicas
- RPO=0; RTO<30s

High compatibility

- Oracle/MySQL compatibility mode
- Lower cost of business migration/revamp

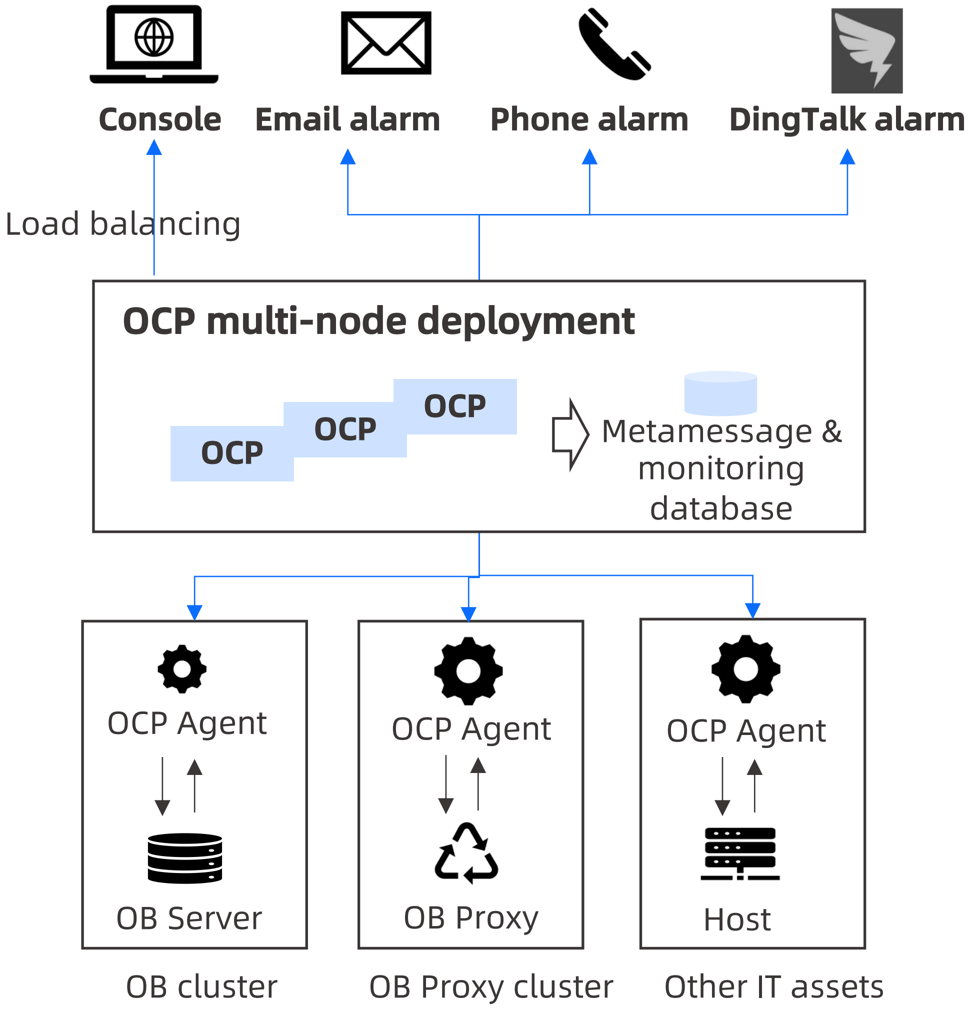
Multi-tenant

- DBaaS framework
- Resource isolation
- Automatic load balancing

High transparency

- Global consistency snapshot
- Global index
- Automatic two-phase commit

OCP product architecture and function: One-stop management O&M tool



Product architecture

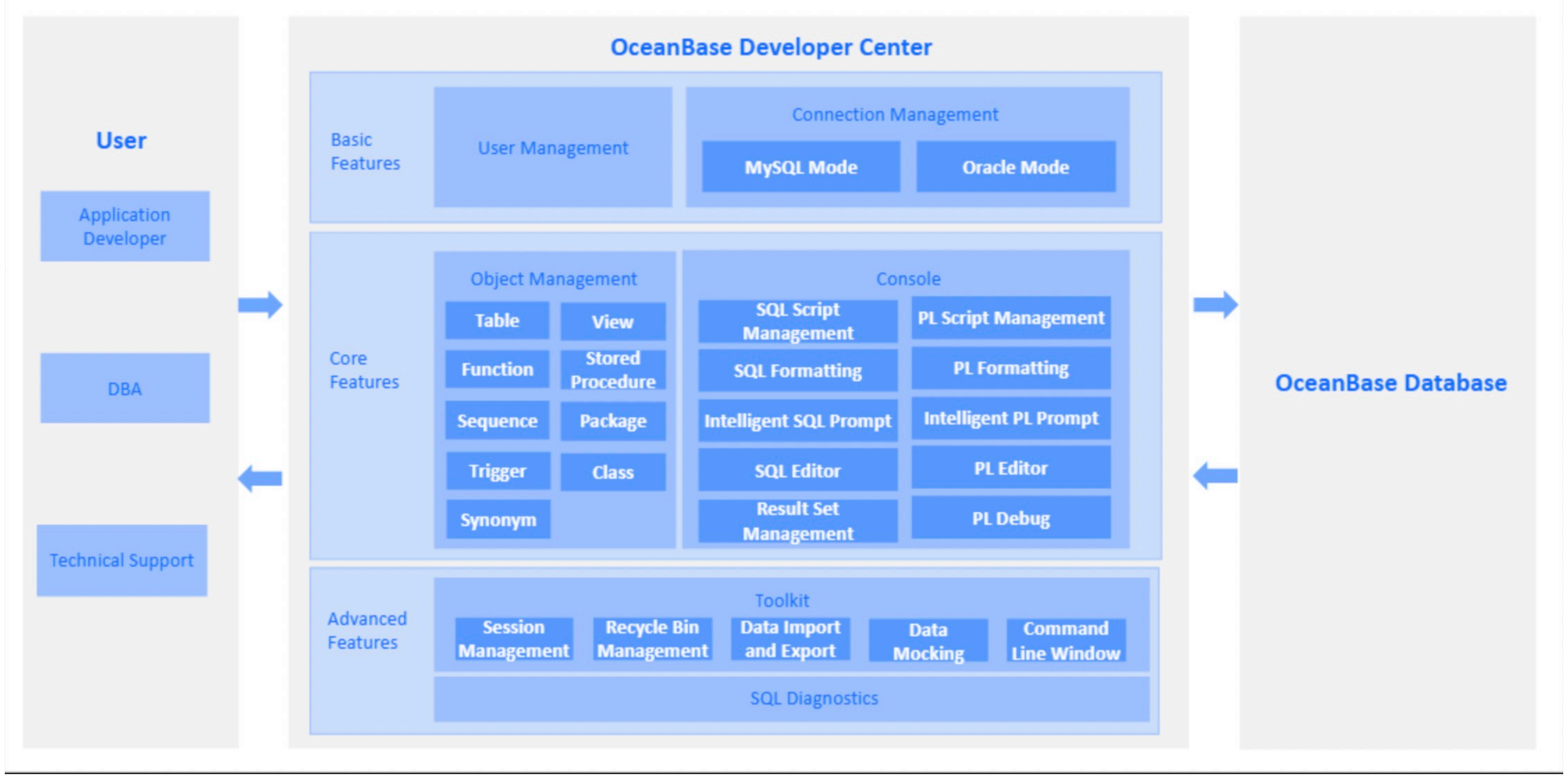
- Each managed administrator installs the OCP Agent. The OCP manages and monitors the managed administrator through the Agent
- OCP provides management, monitoring, and alarm functions for administrators
- Each OCP node has complete and complete functions. A single node can provide all OCP capabilities. When an OCP node is unavailable, it automatically dispatches to available OCP nodes
- OCP Server supports multi-node deployment and implements load balancing using DNS, HAProxy, Nginx, or F5 to ensure high system availability

Dependent software/hardware resources

- OCP Server can be installed on a physical machine or installed and run in a Docker container
- The X86 OCP Server supports oss such as RHEL, CentOS, AliOS, and OpenSUSE. It also supports AliOS, Bid-Winning Kirin, and Huawei EulerOS operating systems based on the ARM architecture
- Ocp-agent occupies less resources and has no special requirements on hardware resources
- Clients use Web browsers to access the OCP service, such as Chrome, Firefox, Safari, and Edge

OceanBase Developer Center (ODC) Architecture

- OceanBase Developer Center (ODC) is an enterprise-level database development platform tailored for OceanBase database.
- ODC supports the connection between MySQL and Oracle databases in OceanBase, and provides database developers with daily database development operations, WebSQL, SQL diagnosis, session management, data import and export and other functions.
- You can download a special client or log in directly using a browser.



User

Application Developer

DBA

Technical Support

OceanBase Developer Center

Basic Features

User Management

Connection Management

MySQL Mode

Oracle Mode

Core Features

Object Management

Table

View

Function

Stored Procedure

Sequence

Package

Trigger

Class

Synonym

Console

SQL Script Management

PL Script Management

SQL Formatting

PL Formatting

Intelligent SQL Prompt

Intelligent PL Prompt

SQL Editor

PL Editor

Result Set Management

PL Debug

Advanced Features

Toolkit

Session Management

Recycle Bin Management

Data Import and Export

Data Mocking

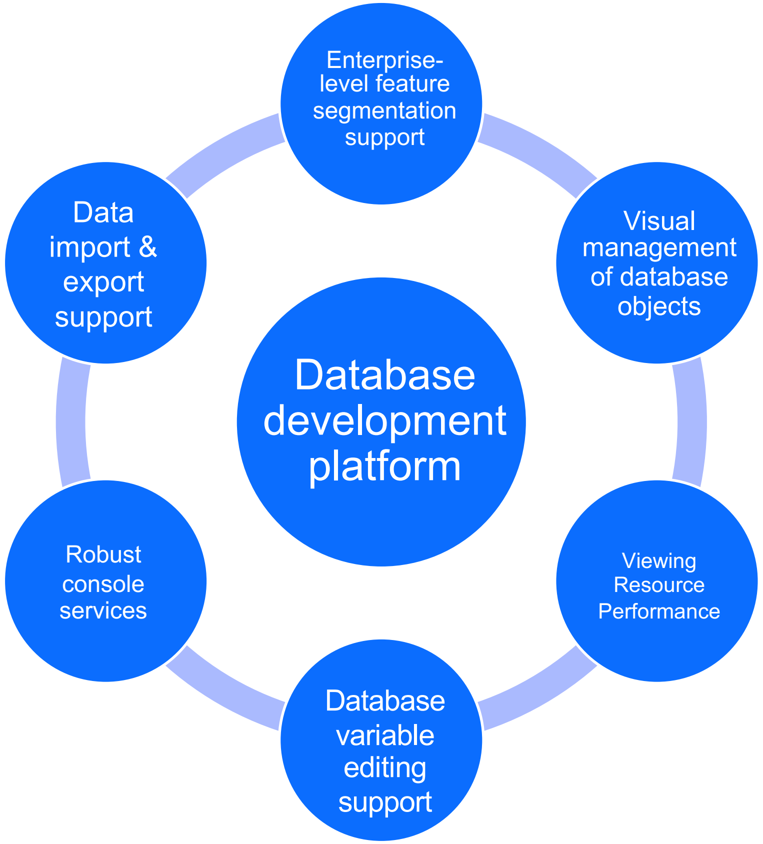
Command Line Window

SQL Diagnostics

OceanBase Database

6 Core functions of ODC help developers get started with OceanBase quickly

Supports full partition types in OceanBase MySQL and Oracle modes


Enterprise-level feature segmentation support
Data import & export support
Visual management of database objects
Database development platform
Robust console services
Viewing Resource Performance
Database variable editing support

Supports import and export of various file formats

Provides wizard-style creation and visual modification of database objects

Through WebSQL technology for developers to provide SQL syntax highlighting, formatting, intelligent tips and other intimate features, support PL object and anonymous block compilation, running, debugging

Monitors database session access in real time, supports viewing and terminating sessions, and provides SQL execution plan analysis and SQL tuning guidance services

Supports visual modification of session variables and system global variables, reducing the difficulty of user memory variables

OMS Core Function Overview

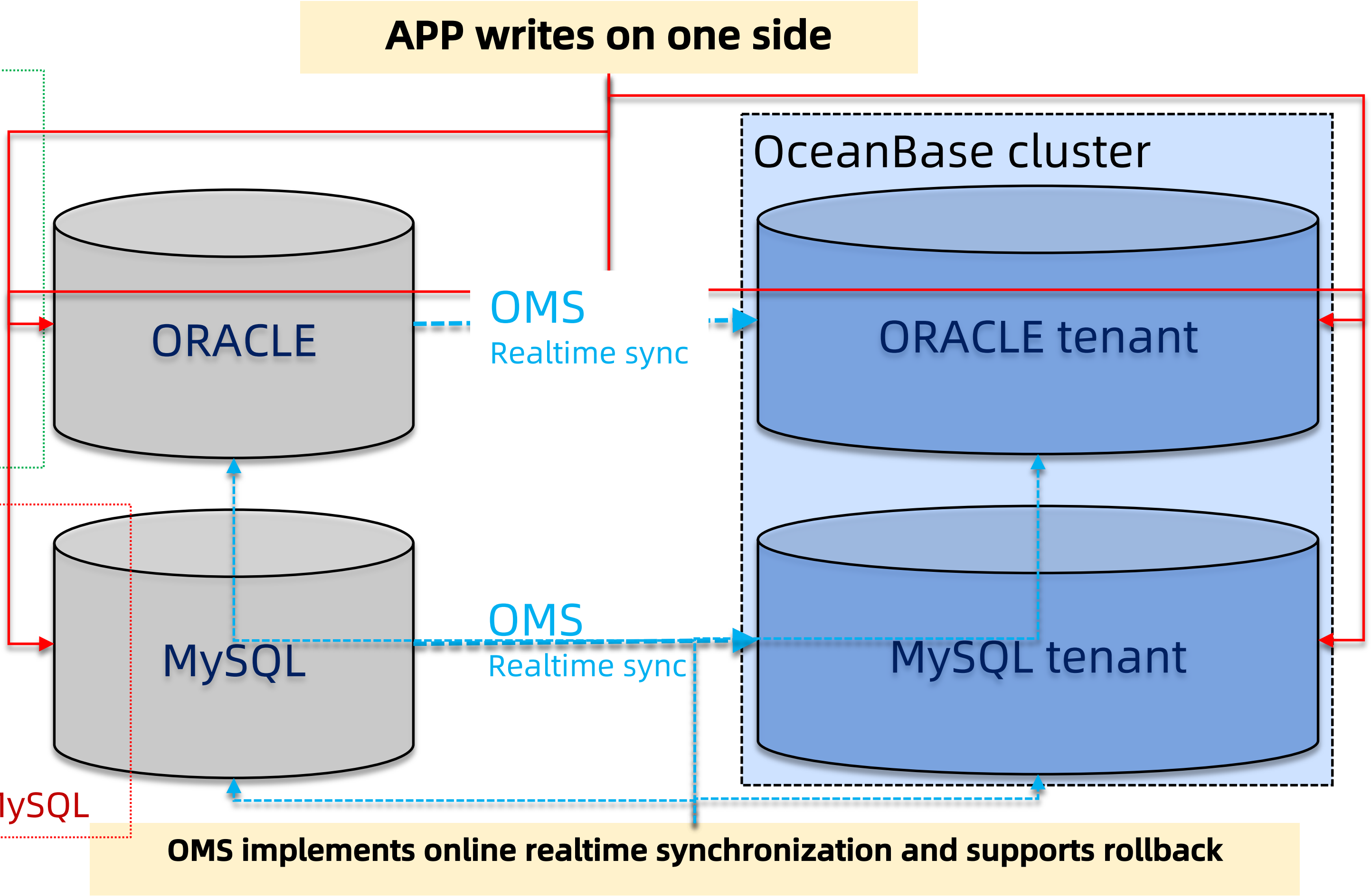
- **Supports multiple data sources**: Supports full migration and incremental real-time data synchronization from databases such as Oracle, MySQL, DB2, and OceanBase to OceanBase
- **Compatibility evaluation and modification**: Heterogeneous data migration OceanBase provides object compatibility evaluation and modification suggestions, greatly reducing the threshold of service migration and difficulty of service transformation
- **One-stop interaction**: Data migration life cycle management. The creation, configuration, and monitoring of data migration are performed coherently on the management interface, facilitating interaction
- **Multi-data verification**: provides protection of multi-mode verification, ensuring data quality more comprehensively, time-saving and efficiently; At the same time, the difference data is displayed to provide a fast correction way

OMS for smooth migration:
Real-time data synchronization + Fast Switchover + fallback plan

- ➢ Application reads/writes ORACLE/MySQL
- ➢ OMS realtime sync: ORACLE/MySQL -> OB
- ➢ Application stops write ORACLE/MySQL
- ➢ OMS full data check: OB vs ORACLE/MySQL
- ➢ OMS realtime sync: OB -> ORACLE/MySQL
- ➢ Application switches read/write OB

<rollback plan>

- ➢ Application stops write OB
- ➢ OMS full data check: ORACLE/MySQL vs OB
- ➢ OMS realtime sync: ORACLE/MySQL -> OB
- ➢ Application switches read/write ORACLE/MySQL



OMS implements online realtime synchronization and supports rollback

Chapter 3: OceanBase product family and basic concepts

1. Product family
2. **Installation & deployment**
3. Data import
4. Basic concepts

OceanBase supports multiple deployment modes

Cloud deployment

- **Public cloud**: Ali cloud has been released, single user single cluster, enterprises can buy on demand
- **Private cloud**: Ali Cloud proprietary cloud 3.5 or above, but also to support the enterprise's already built proprietary cloud infrastructure

Standalone deployment

The recommended configurations for commercial use are as follows. You can download the trial version of OceanBase from the OceanBase official website, which supports 2C8G servers

Server type	Qty	Min. function configuration	Min. performance configuration
OCP control server	1	32C, 128G, 1.5TB storage	32C, 128G, 1.5TB SSD storage, 10 gigabit network card
OceanBase server	3	32C, 128G, 1.2TB storage	32C, 256G, 2TB SSDstorage, 10 gigabit network card

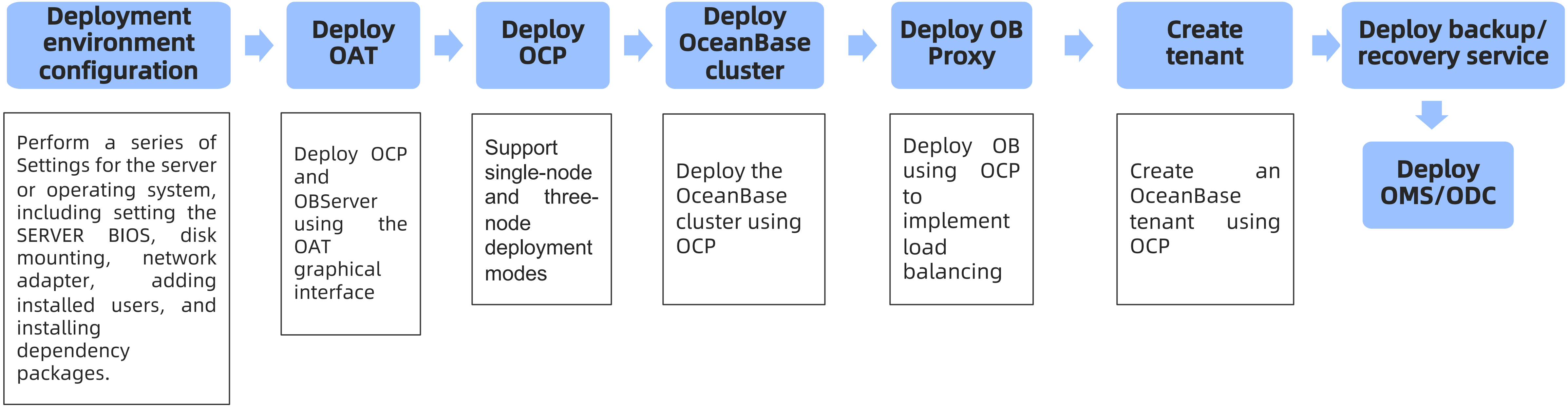
Supports the following CPU & OS

- X86 series
- Hygon 7185
- Hisilicon (Kunpeng 920)
- Phytium (FT1500a, FT2000)

- CentOS
- Red Hat
- SUSE、
- Debian/ Ubuntu、

- AliOS
- NeoKylin V7u5。
- GalaxyKylin 4.0.2

OceanBase Deployment Process



Deployment environment configuration

Perform a series of Settings for the server or operating system, including setting the SERVER BIOS, disk mounting, network adapter, adding installed users, and installing dependency packages.

Deploy OAT

Deploy OCP and OBServer using the OAT graphical interface

Deploy OCP

Support single-node and three-node deployment modes

Deploy OceanBase cluster

Deploy the OceanBase cluster using OCP

Deploy OB Proxy

Deploy OB using OCP to implement load balancing

Create tenant

Create an OceanBase tenant using OCP

Deploy backup/ recovery service

Deploy OMS/ODC

Deploy OceanBase Cluster

Deploy OceanBase Cloud Platform (OCP)

OCP can perform multiple tasks, such as installing the OceanBase cluster, creating the OceanBase cluster tenant, and installing the OB Proxy service, as well as the cluster O&M and cluster monitoring tasks.

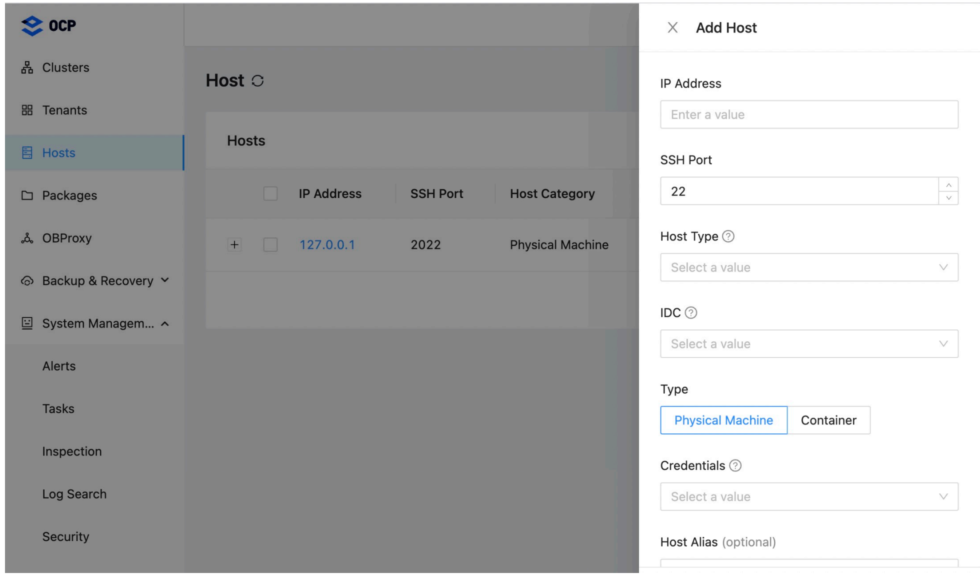
- The cluster can be deployed on three machines with high availability. The OceanBase metadata cluster provides persistent storage of metadata and monitoring data and ensures data reliability in OCP cluster mode
- Each OCP machine has two Dockers running OCP (Web service) and OB Server (OceanBase metadata cluster) respectively.
- The OCP cluster has its own HIGH availability DNS server to provide domain name resolution services
- Provide config URL service to store key information about OceanBase cluster, such as RootService List (rslist)

- Log in to the OCP server as user root and run the bash init_obcluster_conf.sh command
- In the displayed mode selection, select 1-point deployment and 3-3 node deployment.
- The system generates a configuration mode
- Based on the comments, modify the configuration template to start deployment

Deploy OceanBase cluster

Deploy OceanBase cluster

OCP supports adding host and creating cluster operations on a graphical interface, facilitating installation and deployment.



Host

Hosts

	IP Address	SSH Port	Host Category
+ □	127.0.0.1	2022	Physical Machine

Add Host

IP Address: Enter a value

SSH Port: 22

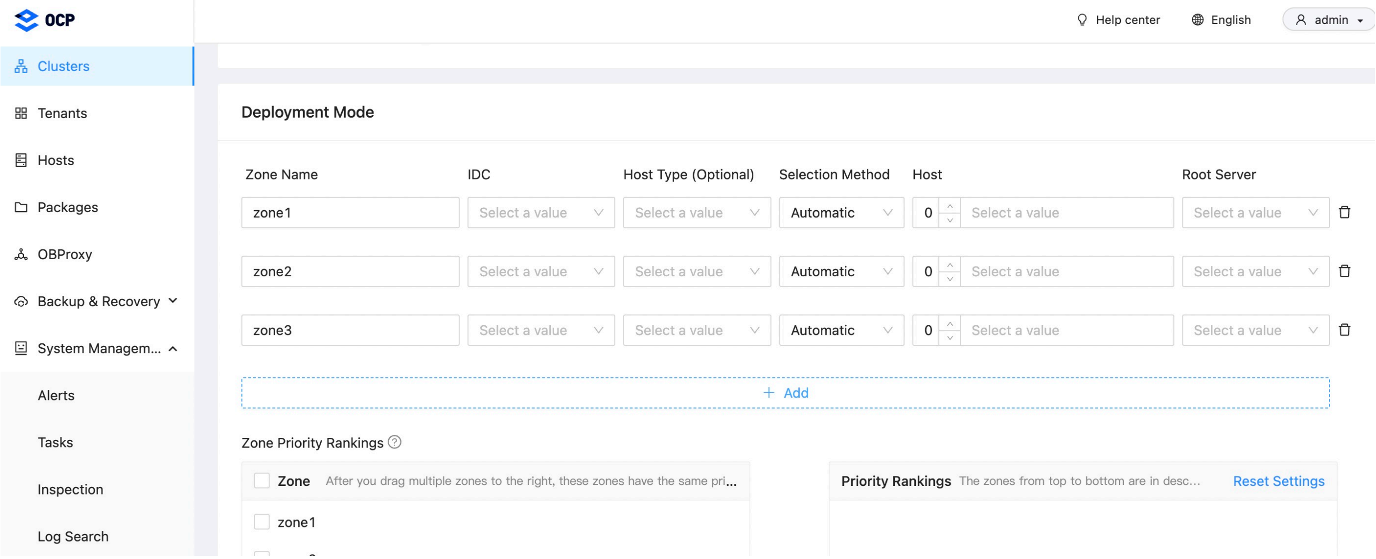
Host Type: Select a value

IDC: Select a value

Type: Physical Machine | Container

Credentials: Select a value

Host Alias (optional)



Deployment Mode

Zone Name	IDC	Host Type (Optional)	Selection Method	Host		Root Server
zone1	Select a value	Select a value	Automatic	0	Select a value	Select a value
zone2	Select a value	Select a value	Automatic	0	Select a value	Select a value
zone3	Select a value	Select a value	Automatic	0	Select a value	Select a value

+ Add

Zone Priority Rankings

□ Zone After you drag multiple zones to the right, these zones have the same pri...

□ zone1

Priority Rankings The zones from top to bottom are in desc... Reset Settings

Deploy OceanBase Cluster

In addition to the OCP installation mode, you can also deploy the OceanBase cluster using the CLI

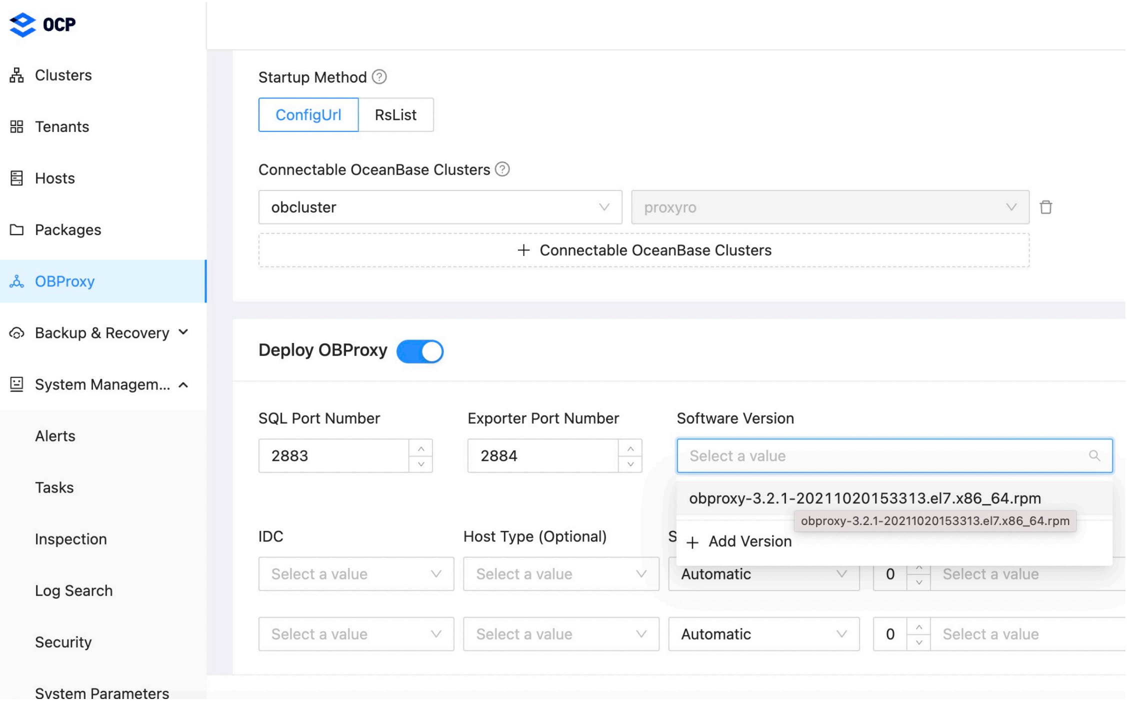
- After a cluster is created using OCP, an OB Server process is started on each Server, specifying the correct parameters: network adapter name, internal service port, external service port, cluster name, cluster ID, zone to which the OB Server belongs, disk directory, and Rslist information
- **Common causes of OceanBase cluster initialization failure (Bootstrap)**
  - ➢ If the clock error between the two machines is too large (more than 100 milliseconds), you can use commands such as NTPQ and clockdiff to check the clock difference between the two machines
  - ➢ The information is specified incorrectly, such as the wrong zone name (OCP cluster creation does not have this problem)
  - ➢ Other issues, such as hardware exceptions

Deploy OceanBase Cluster

Deploy OB Proxy

- Suggest using OCP to deploy OB Proxy. Choose OCP > O&M >OB Proxy > Install OB Proxy
- It can also be deployed in command line mode

- **Host:** Generally choose at least one OBServer machine to deploy OBProxy on each Zone, so three hosts are chosen here
- **OB Proxy version:** Upload OB Proxy RPM version package



Startup Method: ConfigUrl | RsList

Connectable OceanBase Clusters: obcluster | proxyro

+ Connectable OceanBase Clusters

Deploy OBProxy

SQL Port Number: 2883

Exporter Port Number: 2884

Software Version: Select a value

obproxy-3.2.1-20211020153313.el7.x86_64.rpm

obproxy-3.2.1-20211020153313.el7.x86_64.rpm

+ Add Version

IDC	Host Type (Optional)	S		
Select a value	Select a value	Automatic	0	Select a value
Select a value	Select a value	Automatic	0	Select a value

Deploy OceanBase Cluster

Other notes

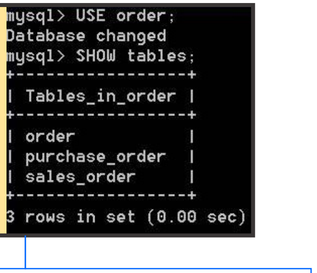
- When creating an OceanBase cluster, you only need to specify three or more machines where the RootService resides. You do not need to specify all machines when creating the cluster. After the cluster is created, you can add new machines
- Obproxy is not automatically installed when an OceanBase cluster is created. You need to install obProxy separately
- After the OceanBase cluster is created, each observer can accept connections. The default connection port is 2881
- After a cluster is created, a tenant named sys is created by default. The sys tenant has a default root user
- If the tenant is directly connected to the Observer, specify the username in the following format: <user name>@< Tenant Name >, for example, "root@sys"

Chapter 3: OceanBase product family and basic concepts

1. Product family
2. Installation & deployment
3. **Data Import**
4. Basic concepts

OceanBase： Various client tools

CLI tools



```
mysql> USE order;
Database changed
mysql> SHOW tables;
+-----------------+
| Tables_in_order |
+-----------------+
| order           |
| purchase_order  |
| sales_order     |
+-----------------+
3 rows in set (0.00 sec)
```

OceanBase client

OceanBase client (OBClient) is compatible with both MySQL and Oracle tenants that access the OceanBase, so it is the recommended blank screen client tool.

MySQL client

OceanBase is fully compatible with MySQL. Standard MySQL client can be used to connect to OceanBase MySQL tenants. Suggesting using a MySQL client version 5.6 or 5.7

GUI tools

OceanBase cloud platform

It not only provides life-cycle management services for OceanBase cluster and tenant components, but also provides management services for OceanBase-related resources (hosts, networks, software packages, etc.).

OceanBase developer center

Enterprise database development platform. ODC supports the connection between MySQL and Oracle databases in OceanBase, and provides database developers with daily database development operations, WebSQL, SQL diagnosis, session management, data import and export and other functions

Connect to MySQL tenants through MySQL clients

To use the MySQL tenant of OceanBase, you can use the MySQL client to connect to the tenant

Operation steps

- Open a command line terminal and make sure that the environment variable PATH contains the MySQL client command directory
- Provide the MySQL running parameters in the following format

```
$mysql -h192.168.1.101 -uroot@obmysql#obdemo -P2883 -pabcABC123 -c -A oceanbase
```

Parameter descriptions

- -h: provides the OceanBase database connection IP address, usually an OBProxy address
- -u: provides the connection account of the tenant in either of the following formats: Username @ Tenant name # Cluster name or cluster name: Tenant name: user name. The default administrator user name of the MySQL tenant is root
- -p: provides the OceanBase database connection port and OBProxy listening port. The default port is 2883 and can be customized
- -p: Provides the account password. For security purposes, you can enter the password at the following prompt instead. The password text is invisible
- -c: do not ignore comments in the MySQL operating environment
- -a: indicates that the MySQL database does not automatically obtain statistics when it connects to the database
- OceanBase: specifies the name of the accessed database. You can change it to a service database

After the connection is successful, the default command line prompt is MySQL [OceanBase]>. To exit the OceanBase command line, type Exit and press Enter, or press the shortcut key CTRL + D

Connect to OceanBase tenants through obclient

Obclient is a command line client tool dedicated to OceanBase. Through obClient, you can connect MySQL and ORACLE tenants in OceanBase to provide operating parameters of OBClient in the following format:

```
$ obclient -h192.168.1.101 -usys@t_oracle0_91#obdoc -P2883 -pabcABC123 -c -A sys
```

Note:

- -h: Provides the OceanBase database connection IP, which is generally OBProxy address.
- -u: Provides the tenant connection account.

  Format: "username@tenant name#cluster name" or "cluster name:tenant name:username".

  Oracle tenant administrator name is sys by default.
- -P: Provides OceanBase database connection port, which is also the OBProxylisting port, default 2883, customizable.
- -p: Provide account password. It can be not supplied for sake of safety and to be entered under the prompt later. The password text is invisible.
- -c: Indicates sending the annotation in SQL statement to the database.
- -A: Indicates not getting all table information in connecting database, which speeds up the logon to the database.
- sys: Accessed database name, which can be changed to the service database.

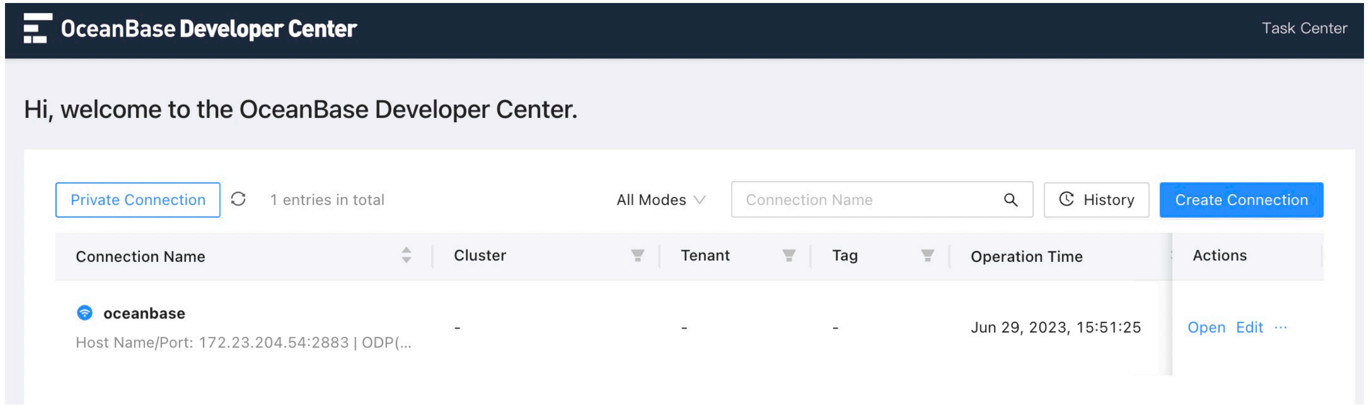
- After the connection is successful, the command line prompt obclient >,
- To exit the OceanBase command line, type Exit and press Enter, or press the shortcut key CTRL + D.

Java database connection driver

- OceanBase implements its own JDBC driver, which enables Java to send SQL statements to OceanBase's MySQL and Oracle tenants. OceanBase JDBC supports exposing SQL data types, PL/SQL objects, and fa st access to SQL data for Java
- The name of the OceanBase JDBC driver file is OceanBase-client -[version number].jar
- OceanBase database driver file 1.0 related version class name: com.alipay.oceanbase.obproxy.mysql.jdbc.Driver
- Since 1.1.0, the OceanBase database driver file changes class name to: com.alipay.oceanbase.jdbc.Driver; the original class name is kept but not recommended

The OceanBase MySQL tenant is compatible with the MySQL connection protocol. The standard MySQL JDBC can be used to connect to the OceanBase MySQL tenant. By default, JDBC does not support the connection protocol for Oracle tenants.

Connect to OceanBase database through ODC



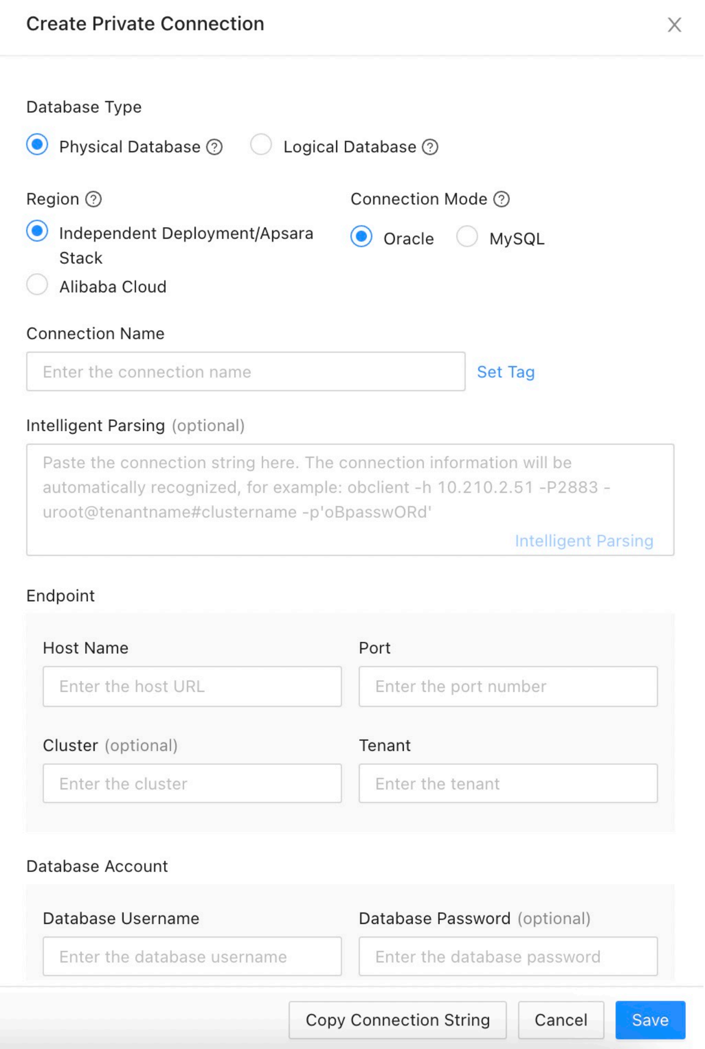
OceanBase Developer Center	Task Center

Hi, welcome to the OceanBase Developer Center.

Private Connection ⟳ 1 entries in total	All Modes	Connection Name	History	Create Connection

Connection Name	Cluster	Tenant	Tag	Operation Time	Actions
oceanbase Host Name/Port: 172.23.204.54:2883 | ODP(...	-	-	-	Jun 29, 2023, 15:51:25	Open Edit ···

On the Create Connection page, set the connection mode to MySQL/Oracle, enter the connection name, host name, port, cluster, tenant, database user name, and database password, and click Save. If the connection is saved successfully, the database is successfully connected.



Create Private Connection

Database Type

● Physical Database ○ Logical Database

Region

● Independent Deployment/Apsara Stack
○ Alibaba Cloud

Connection Mode

● Oracle ○ MySQL

Connection Name

Enter the connection name	Set Tag

Intelligent Parsing (optional)

Paste the connection string here. The connection information will be automatically recognized, for example: obclient -h 10.210.2.51 -P2883 -uroot@tenantname#clustername -p'oBpasswORd'

Intelligent Parsing

Endpoint

Host Name: Enter the host URL
Port: Enter the port number
Cluster (optional): Enter the cluster
Tenant: Enter the tenant

Database Account

Database Username: Enter the database username
Database Password (optional): Enter the database password

Copy Connection String	Cancel	Save

About data migration and synchronization

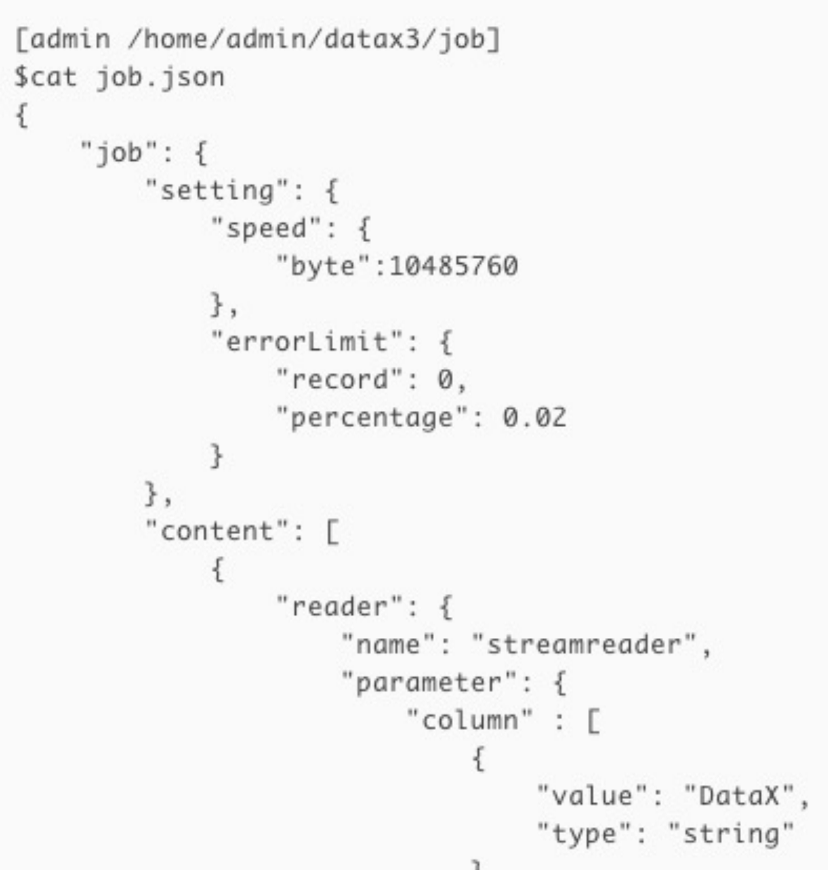
- After data is migrated from a traditional database to the OceanBase database, you can export the data as CSV files or SQL files and then import the data to the OceanBase database
- You can also use The OceanBase product DataX or OMS to migrate data offline or online between a traditional database and OceanBase
- **Note: When the amount of data being migrated is very large, if the migration is fast, OceanBase's incremental memory consumption may be faster than the dump and merge free memory. In this case, modify data migration parameters, limit memory write speed, or expand the memory capacity of instances**

Introduction to general data synchronization framework DataX

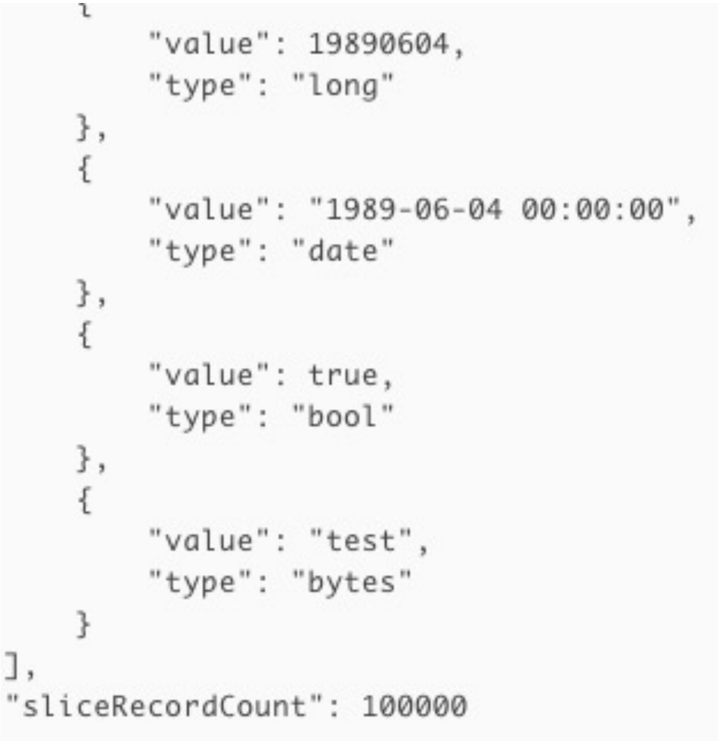
- DataX is a widely used offline data synchronization tool/platform within Alibaba Group. The implementation includes MySQL, Oracle, SqlServer, Postgre, HDFS, Hive, ADS, HBase, TableStore (OTS), MaxCompute (ODPS), DRDS, and OceanBase Efficient data synchronization between heterogeneous data sources
- As a data synchronization framework, DataX abstracts the synchronization of different data sources into Reader plug-in that reads data from the source data source and Writer plug-in that writes data to the target. Theoretically, DataX framework can support data synchronization of any data source type
- After DataX is installed, the default directory is /home/admin/datax3. The job folder contains the configuration file of data migration tasks by default. You can also customize the directory
- The parameter file for each task is a JSON format, consisting of a reader and a writer. There is a default example task configuration file job.json in the job folder
- The DataX website supports read and write plugins for most major data sources and has detailed usage documentation. For Oracle database, use Oracle Reader and Oracle Writer plug-in to read and write

Example of general data synchronization framework DataX

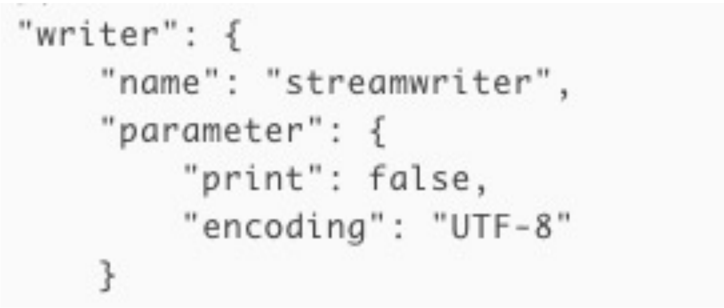
- After DataX is installed, the default directory is /home/admin/datax3. The directory contains the job folder, which stores the configuration files of data migration tasks by default. You can also customize the directory
- The parameter file for each task is a JSON format, consisting of a reader and a writer. There is a default example task configuration file job.json in the job folder



```
[admin /home/admin/datax3/job]
$cat job.json
{
    "job": {
        "setting": {
            "speed": {
                "byte":10485760
            },
            "errorLimit": {
                "record": 0,
                "percentage": 0.02
            }
        },
        "content": [
            {
                "reader": {
                    "name": "streamreader",
                    "parameter": {
                        "column" : [
                            {
                                "value": "DataX",
                                "type": "string"
                            },
```



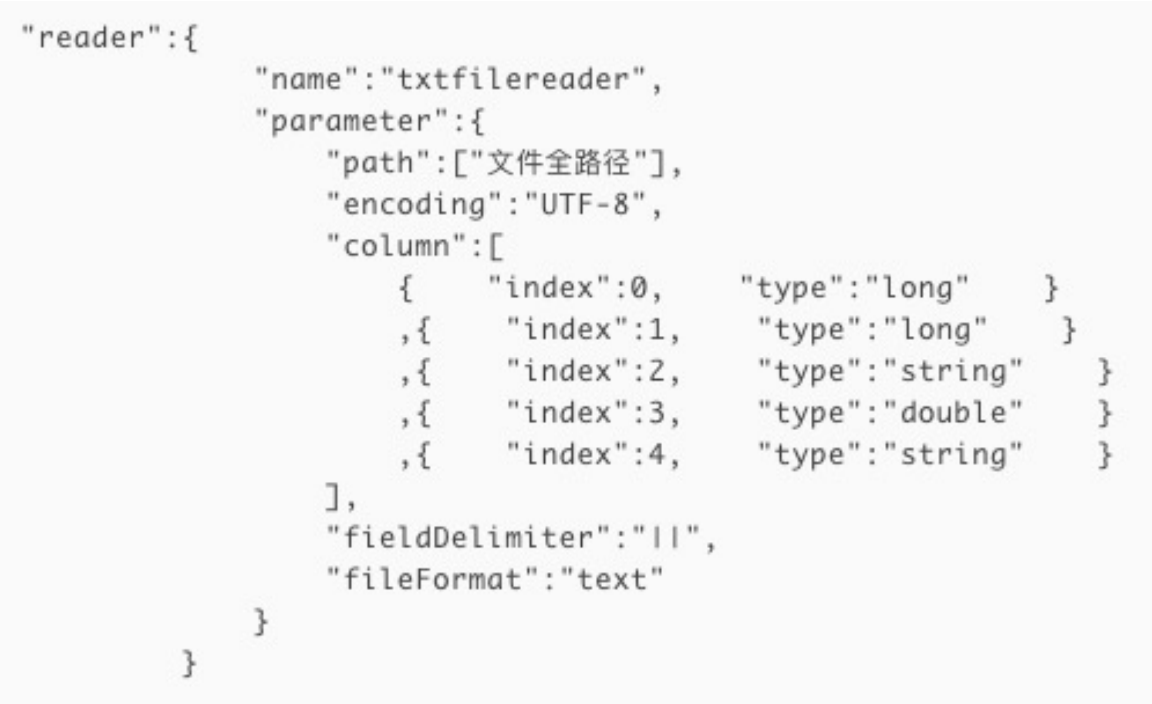
```
    {
        "value": 19890604,
        "type": "long"
    },
    {
        "value": "1989-06-04 00:00:00",
        "type": "date"
    },
    {
        "value": true,
        "type": "bool"
    },
    {
        "value": "test",
        "type": "bytes"
    }
],
"sliceRecordCount": 100000
```



```
"writer": {
     "name": "streamwriter",
     "parameter": {
          "print": false,
          "encoding": "UTF-8"
     }
```

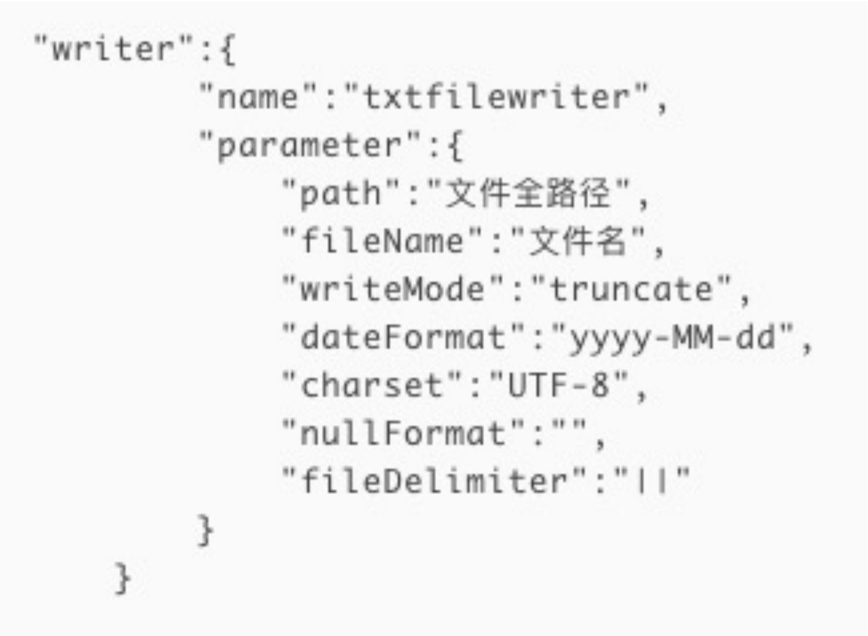
DataX - CSV file read/write plugin

Txtreader configuration example:



```
"reader":{
            "name":"txtfilereader",
            "parameter":{
                "path":["文件全路径"],
                "encoding":"UTF-8",
                "column":[
                    {    "index":0,     "type":"long"    }
                   ,{     "index":1,     "type":"long"     }
                   ,{     "index":2,     "type":"string"     }
                   ,{     "index":3,     "type":"double"     }
                   ,{     "index":4,     "type":"string"     }
                ],
                "fieldDelimiter":"||",
                "fileFormat":"text"
            }
        }
```

txtwriter configuration example :

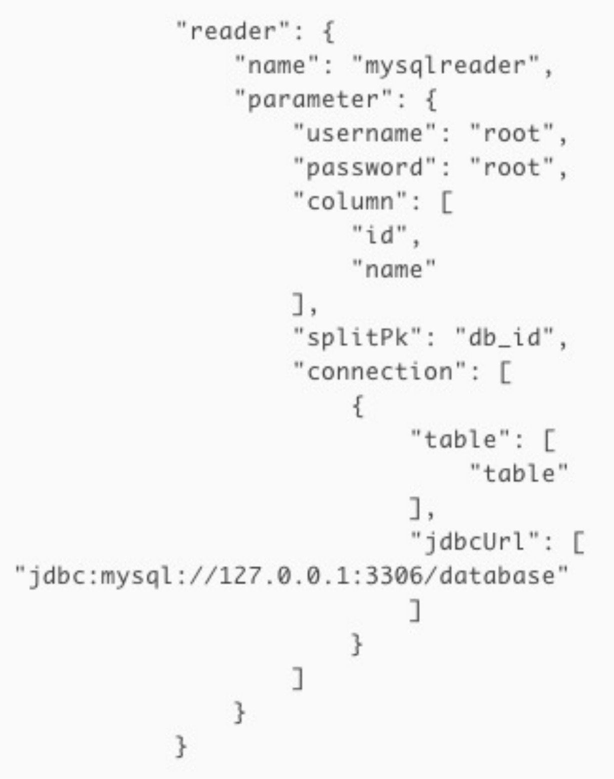


```
"writer":{
        "name":"txtfilewriter",
        "parameter":{
            "path":"文件全路径",
            "fileName":"文件名",
            "writeMode":"truncate",
            "dateFormat":"yyyy-MM-dd",
            "charset":"UTF-8",
            "nullFormat":"",
            "fileDelimiter":"||"
        }
    }
```

DataX - MySQL database read/write plugin

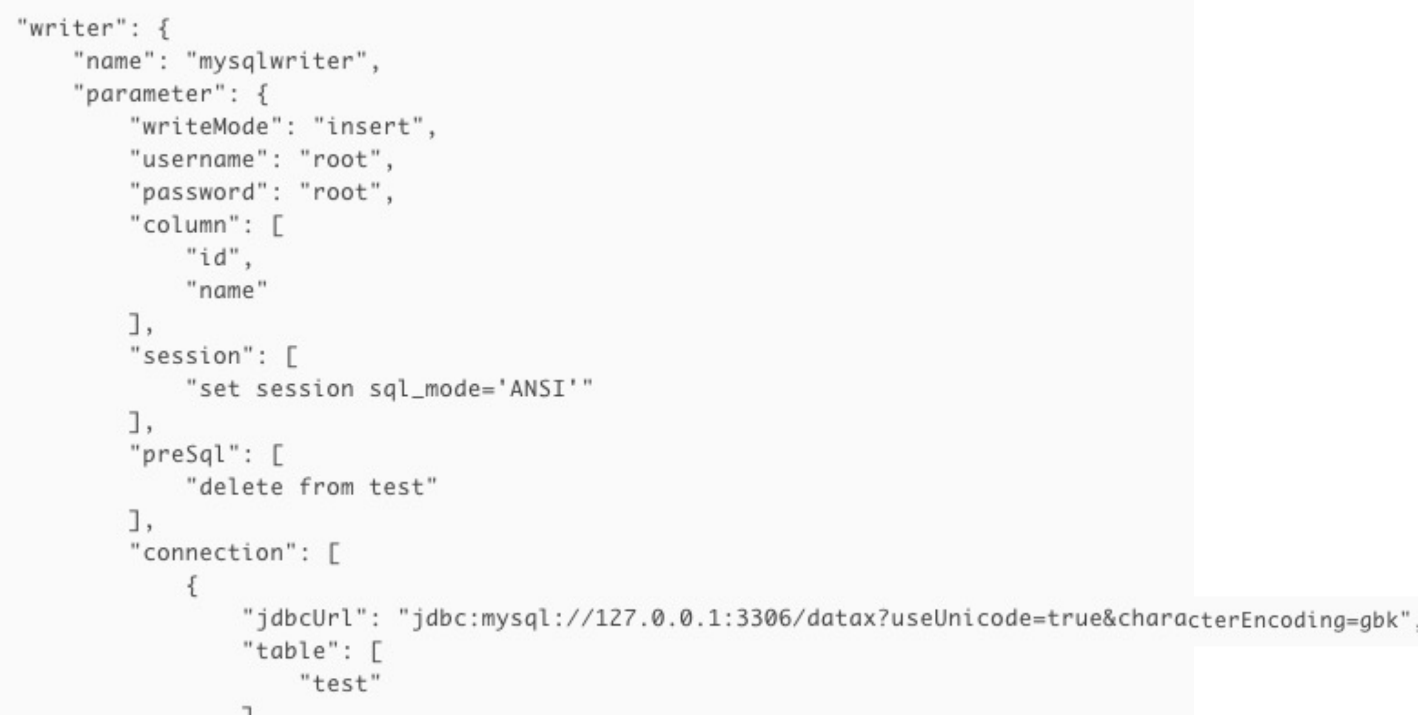
For MySQL database, the mysqlreader and mysqlwriter read/write plugins can be used.

mysqlreader configuration example:



```
            "reader": {
                "name": "mysqlreader",
                "parameter": {
                    "username": "root",
                    "password": "root",
                    "column": [
                        "id",
                        "name"
                    ],
                    "splitPk": "db_id",
                    "connection": [
                        {
                            "table": [
                                "table"
                            ],
                            "jdbcUrl": [
"jdbc:mysql://127.0.0.1:3306/database"
                            ]
                        }
                    ]
                }
            }
```

mysqlwriter configuration example:

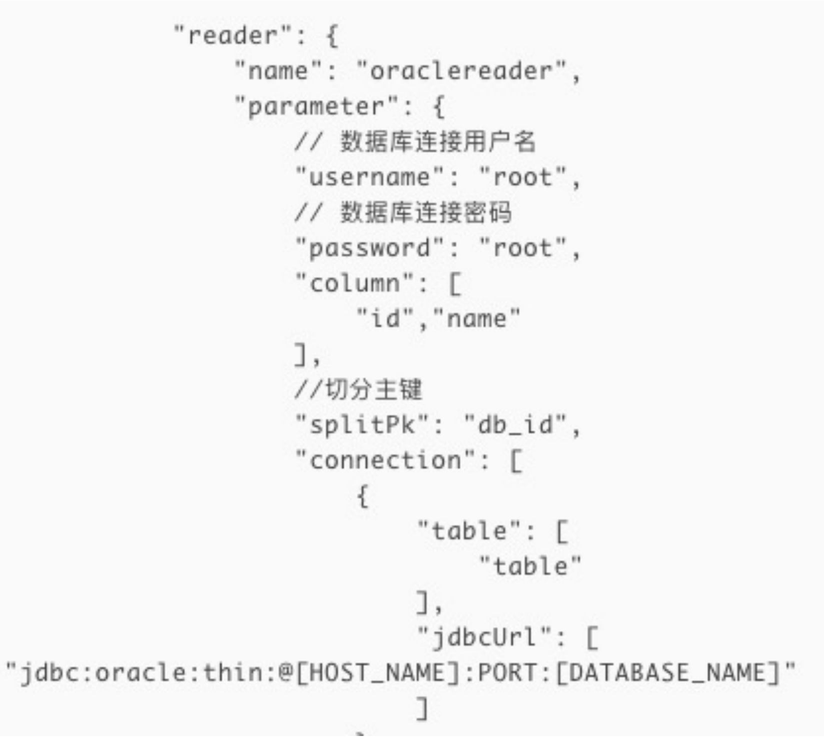


```
"writer": {
    "name": "mysqlwriter",
    "parameter": {
        "writeMode": "insert",
        "username": "root",
        "password": "root",
        "column": [
            "id",
            "name"
        ],
        "session": [
            "set session sql_mode='ANSI'"
        ],
        "preSql": [
            "delete from test"
        ],
        "connection": [
            {
                "jdbcUrl": "jdbc:mysql://127.0.0.1:3306/datax?useUnicode=true&characterEncoding=gbk",
                "table": [
                    "test"
                ]
```

DataX - Oracle database read/write plugin

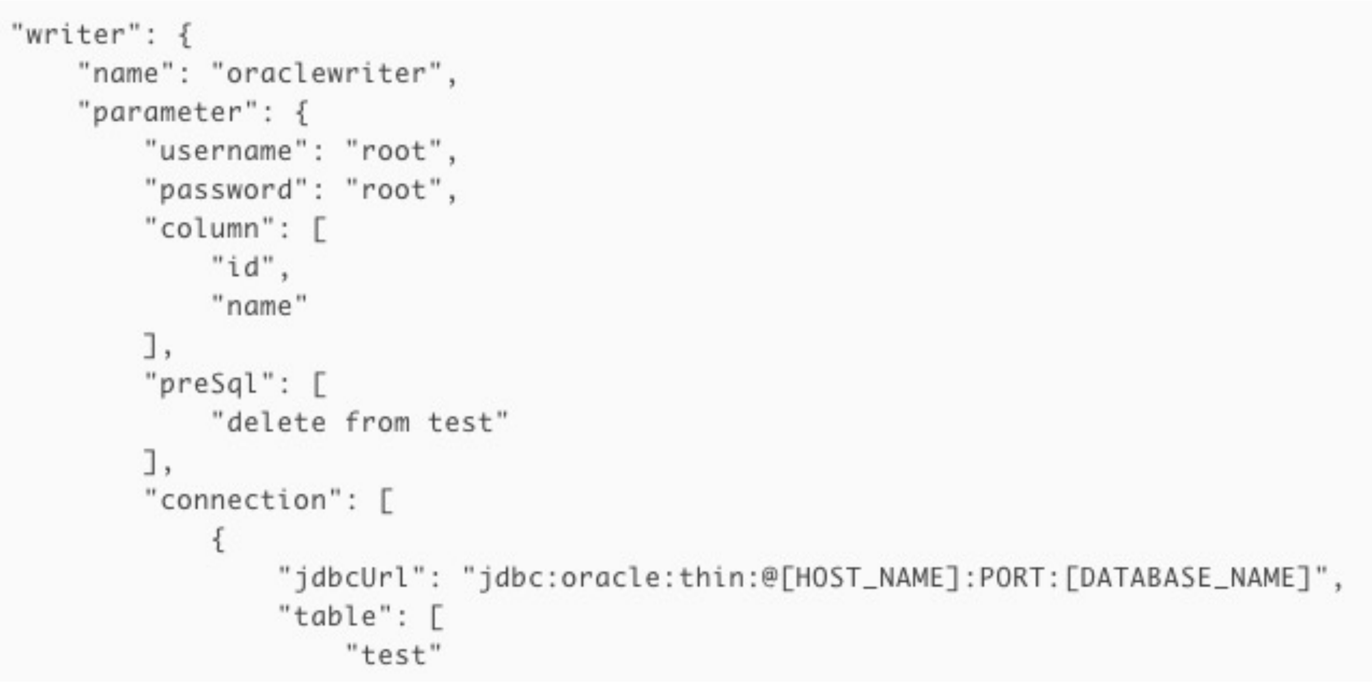
For Oracle database, the oraclereader and oraclewriter read/write plugins can be used.

oraclereader configuration example:



```
           "reader": {
               "name": "oraclereader",
               "parameter": {
                   // 数据库连接用户名
                   "username": "root",
                   // 数据库连接密码
                   "password": "root",
                   "column": [
                       "id","name"
                   ],
                   //切分主键
                   "splitPk": "db_id",
                   "connection": [
                       {
                           "table": [
                               "table"
                           ],
                           "jdbcUrl": [
"jdbc:oracle:thin:@[HOST_NAME]:PORT:[DATABASE_NAME]"
                           ]
```

oraclewriter configuration example:

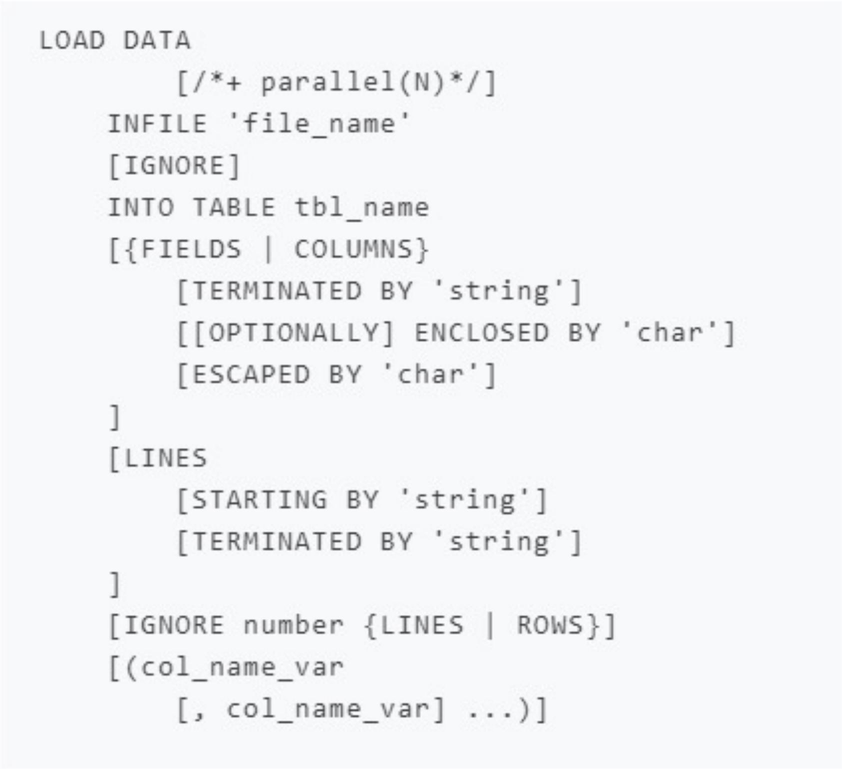


```
"writer": {
    "name": "oraclewriter",
    "parameter": {
        "username": "root",
        "password": "root",
        "column": [
            "id",
            "name"
        ],
        "preSql": [
            "delete from test"
        ],
        "connection": [
            {
                "jdbcUrl": "jdbc:oracle:thin:@[HOST_NAME]:PORT:[DATABASE_NAME]",
                "table": [
                    "test"
```

OCEANBASE

Import data via LOAD DATA command

OceanBase supports the LOAD DATA command to load the contents of external text files into database tables



```
LOAD DATA
        [/*+ parallel(N)*/]
    INFILE 'file_name'
    [IGNORE]
    INTO TABLE tbl_name
    [{FIELDS | COLUMNS}
        [TERMINATED BY 'string']
        [[OPTIONALLY] ENCLOSED BY 'char']
        [ESCAPED BY 'char']
    ]
    [LINES
        [STARTING BY 'string']
        [TERMINATED BY 'string']
    ]
    [IGNORE number {LINES | ROWS}]
    [(col_name_var
        [, col_name_var] ...)]
```

Load Data can import text files in CSV format. The import process is as follows :

1. Parse file: OceanBase reads the data in the file according to the file name entered by the user, and parses the data in the input file in parallel or serial according to the degree of parallelism entered by the user
2. Distribute Data: Because OceanBase is a distributed database system, Data of each partition may be distributed to different observers. Load Data calculates the parsed Data and determines which OBServer the Data needs to be sent to
3. Insert data: When the target OBServer receives the sent data, it performs an INSERT operation locally to INSERT the data into the corresponding partition

Load Data options (1)

- **Parallel:** /*+ parallel (N) */ option specifies the degree of parallelism to load data. The recommended value range is [0 - tenant's maximum NUMBER of cpus]
- **Input file**: the INFILE 'file_name' keyword specifies the path and file name of the input file. Currently, Load Data supports only local input files in the OBServer. Therefore, before importing, the user needs to copy the file to an OBServer and connect to the OBServer where the file resides to run a Load Data statement
- **Execute permission**: Users need to grant permission to access files on the machine. There are two steps: First, change the path of the security file to empty (that is, no check is required). Next, the user is granted read permission to the directory
- **Duplicate data Processing**: This section specifies how to handle duplicate data. Replace Replaces the original data in the table with the data in the input file. Ignore: Duplicate data is ignored. Load Data determines whether Data is duplicate by the primary key of the table. If the table does not have a primary key, the Load Data statement cannot determine whether Data is duplicate. Replace and Ignore are the same. If the user does not specify this option, the Load Data statement will record the incorrect Data to a log file when duplicate Data is encountered

Load Data options (2)

- **Destination table option:** INTO TABLE tbl_name keyword is used to specify the name of the destination table. Load Data supports partitioned table and non-partitioned table.
- **Field format:** this part of the specified input file each field delimiter options, through the Fields | clause to specify the Columns, including: Terminated By keyword is used to specify the field separator; The Enclosed By keyword specifies whether the beginning and end of each field contain a specific character; The Escaped By keyword specifies the wildcard in a field
- **Line format**: This section specifies the start and end characters for each line in the input file, set through the Lines clause. Where Starting By is used to specify the beginning character of each line; Terminated By the user specifies the end character of each line. IGNORE the number {LINES | ROWS} clause specifies IGNORE number before the line of the input file data
- **Column mapping option**: This part is used to specify the relationship between the columns of the target table and the fields of the input file by (col_name_var [, col_name_var...). Keyword specified. If not specified by the user, fields in the input file are mapped to columns in the table by default. If a user uses the col_name_OR_user_var keyword to specify the mapping between the fields in the input file and the columns in the table, the Load Data will correspond to the columns in the table based on the specified column name, and the columns that are not specified will be empty

OBLOADER import tool

- **The OBLOADER tool provides very flexible command-line options for importing structures and data into OceanBase databases in a variety of complex scenarios**
- **OBLOADER is mainly used with OBDUMPER. OBLOADER also supports SQL or CSV files exported by tools such as Navicat, Mydumper and SQLDeveloper**

OBLOADER has the following key features:

- Support the importing Schema definition statements
- Support the CSV and SQL format data files
- Support importing complex data formats such as byte offsets, string separations, and mixed data files
- Support the configuration of simple data cleaning rules
- Support field mapping between configuration files and tables
- Support traffic limiting, explosion prevention, breakpoint recovery, and automatic retry
- Support custom log directories and saves bad data and conflicting data
- Support importing data in restricted mode, which does not depend on any account information of system tenants
- Support OBLOADER to import data from OSS to OceanBase database

OBLOADER supports the following file formats:

- DDL file format: A file contains only DDL statements
- SQL file format: The file contains only INSERT statements, and records do not cross rows
- MIX mixed file format: A file contains mixed statements such as DDL and DML
- POS file format: Data in the file is generated based on byte offset position, not character offset position
- CUT file format: Data in the file is separated by character strings, which is different from single character delimiters in CSV format

OBLOADER import tool instructions

Running environment

Environment	Requirements
System version	It runs on Linux, MacOSX, and Windows 7+. PC is only used for testing. High-specification machine is recommended for large-scale data import.
Java environment	installed JDK 1.8+ and configured JAVA_HOME in system environment variable
Character set	UTF-8
JVM parameter	Adjust JVM memory parameter in script as required data volume

permission

The following permissions are required when the OBLOADER is used to connect to the OceanBase database and import data:

- The user account used to connect to the OceanBase database must have the permission to run the SELECT, INSERT, MERGE, and UPDATE commands
- The OBLOADER user root@sys or proxyro@sys is used to query system tables. Therefore, you must specify the values of --sys-user and --sys-password in the data import command (except in restricted mode).

OBLOADER import tool syntax

```
[admin@localhost]>./obloader -h <host IP> -P <port> -u <user> -p <password> --sys-password <account password in system tenant> -c <cluster> -t <tenant> -D <Schema library name> [--ddl] [--csv|--sql] [--all|--table ‘table name'] -f<data file or directory>
```

Parameter	Description
--ddl	Import DDL file, default file suffix -schema.sql
--sql	Import SQL file, default file suffix .sql
--all	Import the structure definition of all database objects
--sys-password	Specify password for a user in sys tenant, default empty

OBLOADER import tool example

Cluster information

Database info	Example value
Cluster	Cluster A
OBProxy address	192.168.0.0
OBProxy port number	2883
Tenant name	TenantA
Password for a user under sys tenant (at least read-only permission)	Passroot123 (for example password for the root@sys user)
Account for a user under service tenant (requiring read/write permission)	userA
Password for a user under service tenant	Pass123
Schema name	USERA

Example

1. Import all structure information in /home/admin/LOAD-1/ directory into Schema USERA

```
[admin@localhost]> ./obloader -h 192.168.0.0 -P 2883 -u userA -p
Pass123 --sys-password Passroot123 -c ClusterA -t tenantA -D USERA --
ddl --all -f /Users/admin/LOAD-1/
```

2. Import the CSV data file in /home/admin/LOAD-1/ directory into Schema USERA

```
[admin@localhost]> ./obloader -h 192.168.0.0 -P 2883 -u userA -p
Pass123 --sys-password Passroot123 -c ClusterA -t tenantA -D USERA --
csv --all -f /Users/admin/LOAD-1/
```

3. Import the SQL data file in /home/admin/LOAD-1/ directory into Schema USERA

```
[admin@localhost]> ./obloader -h 192.168.0.0 -P 2883 -u userA -p
Pass123 --sys-password Passroot123 -c ClusterA -t tenantA -D USERA --
sql --all -f /Users/admin/LOAD-1/
```

OBDUMPER introduction

OBDUMPER can export OceanBase data to files in SQL or CSV format. You can also use the tool to export objects defined in the database to a file. OBDUMPER has the following obvious advantages in terms of functionality and performance over export tools such as MyDumper and SQLDeveloper:

- High performance: Provides performance optimization in scenarios without primary key tables and partitioned tables
- Multiple functions: Provides limited data export, diverse data formats, and globally consistent and unlocked data export

Core features

•Support exporting structure definition statements (DDL) for objects in a database

•Support exporting table data to a file in CSV or SQL format

•For partitioned tables, specify a partition name to export some of the data within the partition

•Specify global filtering criteria and export only data that meets the criteria

•Support configuration of simple data processing rules

•Support globally consistent non-locking reads to ensure global consistency of exported data

•Support SCN/TIMESTAMP-based flashback query to ensure global consistency of exported data

•Support exporting data in restricted mode, which does not depend on any account information of system tenants

•Support OBDUMPER to export and upload data to Aliyun OSS

OBDUMPER introduction

Running environment

Environment	Requirements
System version	Support the operation on Linux and MacOSX system. Trial version available for PC only, high-specifications machines recommended for large-scale data export
Java environment	Installed JDK 1.8+ and configured JAVA_HOME in system environment variable
Character set	UTF-8
JVM parameter	Adjust JVM memory parameter in script as required data volume

permission

The following permissions are required when the OBDUMPER is used to connect to the OceanBase database and export data:

- The user account used to connect to the OceanBase database must have the permission to run the SELECT command
- The OBDUMPER user root@sys or proxyro@sys is used to query system tables. Therefore, you must specify the values of --sys-user and --sys-password in the data export command (except in restricted mode).

others

The foreign key definition contains multiple columns, and the order of the columns is not guaranteed when exported. Example: FOREIGN KEY (C1, C2) REFERENCE (C1, C2)

When importing or exporting large-scale data, modify vm memory parameters (default: -xMS4g -XMx4g) in the running script.

When exporting data, suggest triggering a data merge to improve the performance of the export

OBDUMPER syntax

```
[admin@localhost]>./obdumper -h <host IP> -P <port> -u <user> -p <password> --sys-password
<password of a user in sys tenant> -c <cluster> -t <tenant> -D <database name> [--ddl] [--csv|--sql--
cut]] [--all|--table ‘table name'] -f<data directory>
```

Parameter	Description
--ddl	Export DDL file, default file suffix-schema.sql
--sql	Export SQL file, default file suffix.sql
--all	Export the structure definition of all database objects, default: table, view, trigger, function, storage process, series, synonym, etc.
--sys-password	Specify password for a user in sys tenant, default empty

OBDUMPER application example

Cluster information

Database info	Example value
Cluster	Cluster A
OBProxy address	192.168.0.0
OBProxy port	2883
Tenant name	TenantA
Password for a user under sys tenant (at least read-only permission)	Passroot123 (for example password for the root@sys user)
Account for a user under service tenant (requiring read/write permission)	userA
Password for a user under service tenant	Pass123
Schema name	USERA

Example

1. Export structure statement of all objects in Schema USERA to /home/admin/DUMP-1/ directory

```
[admin@localhost]> ./obdumper -h 192.168.0.0 -P 2883 -u userA -p Pass123 -
-sys-password Passroot123 -c ClusterA -t tenantA -D USERA --ddl --all -f
/Users/admin/DUMP-1/
```

2. Export data of all objects in Schema USERA to /home/admin/DUMP-1/ directory, with data in SQL format

```
[admin@localhost]> ./obdumper -h 192.168.0.0 -P 2883 -u userA -p Pass123 -
-sys-password Passroot123 -c ClusterA -t tenantA -D USERA --sql --all -f
/Users/admin/DUMP-1/
```

3. Export structure and data of all objects in Schema USERA to /home/admin/DUMP-1/ directory, with data in CSV format

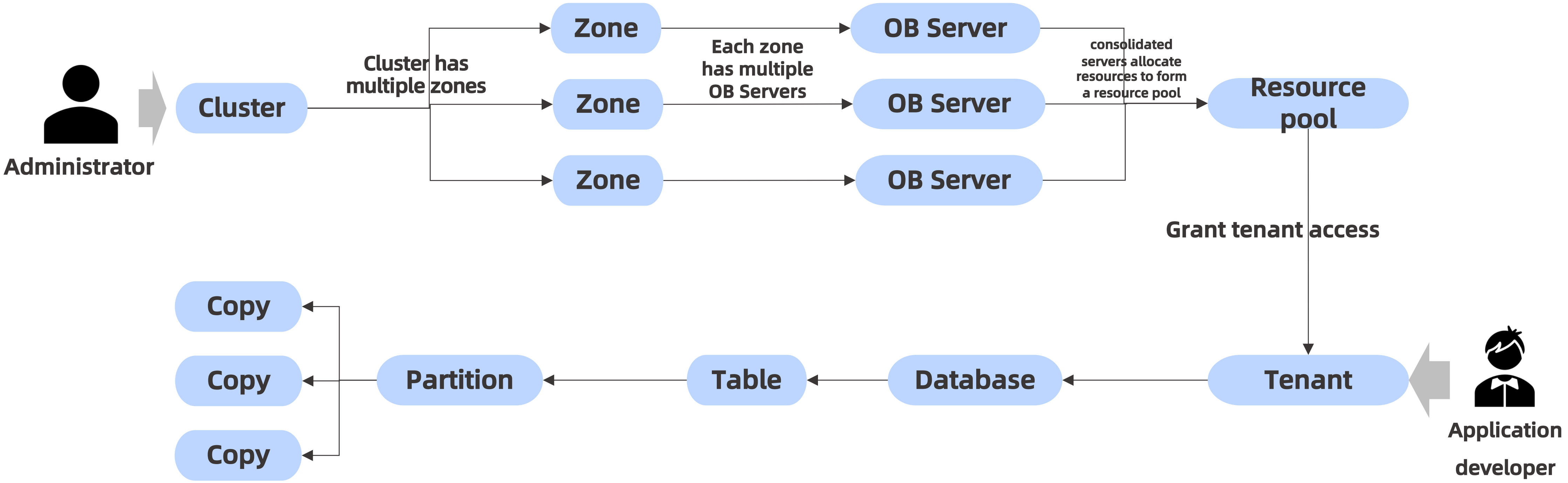
```
[admin@localhost]> ./obdumper -h 192.168.0.0 -P 2883 -u userA -p Pass123 -
-sys-password Passroot123 -c ClusterA -t tenantA -D USERA --csv --all -f
/Users/admin/DUMP-1/
```

Chapter 3: OceanBase product family and basic concepts

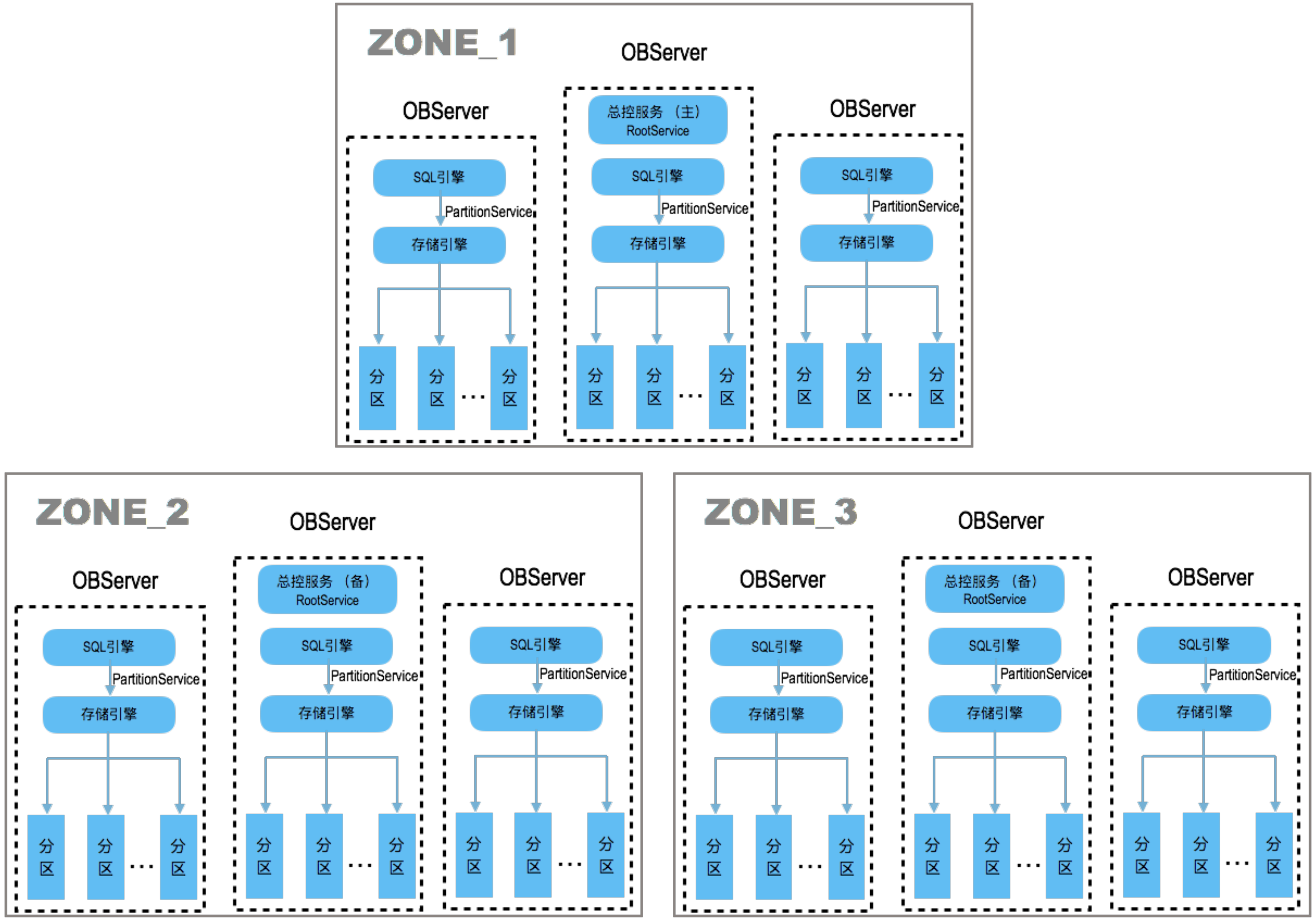
1. Product family
2. Installation & deployment
3. Data Import
4. **Basic concepts**

OceanBase basic concept introduction

Understand fundamentals of OceanBase: Cluster, Zone, OB Server, resource pool, tenant, and zone; from both system mgt. and application dev. perspective



Cluster, Zone and OB Server



- A cluster is made of multi-zones, a bunch of machines with same tag in the cluster belong to one zone

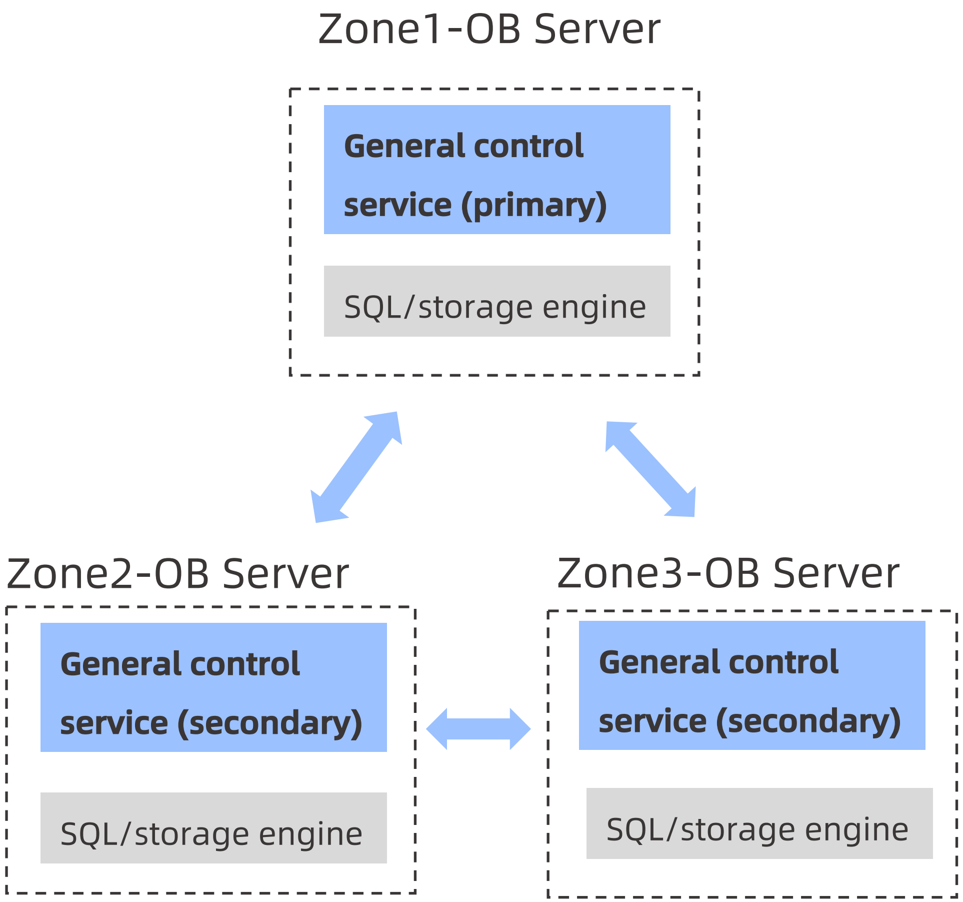
- Different zone can map to different city, or different IDCs of the same city, or different racks in the same IDC

- No. of zone should be >=3, suggest odd number

- Each zone has a full data replica; single zone failure not affect business

- Each OB Server is independent, which has individual compute and storage engine

RootService: General Control Service (RS)


Zone1-OB Server
General control service (primary)
SQL/storage engine
Zone2-OB Server
General control service (secondary)
SQL/storage engine
Zone3-OB Server
General control service (secondary)
SQL/storage engine

General control service of OceanBase

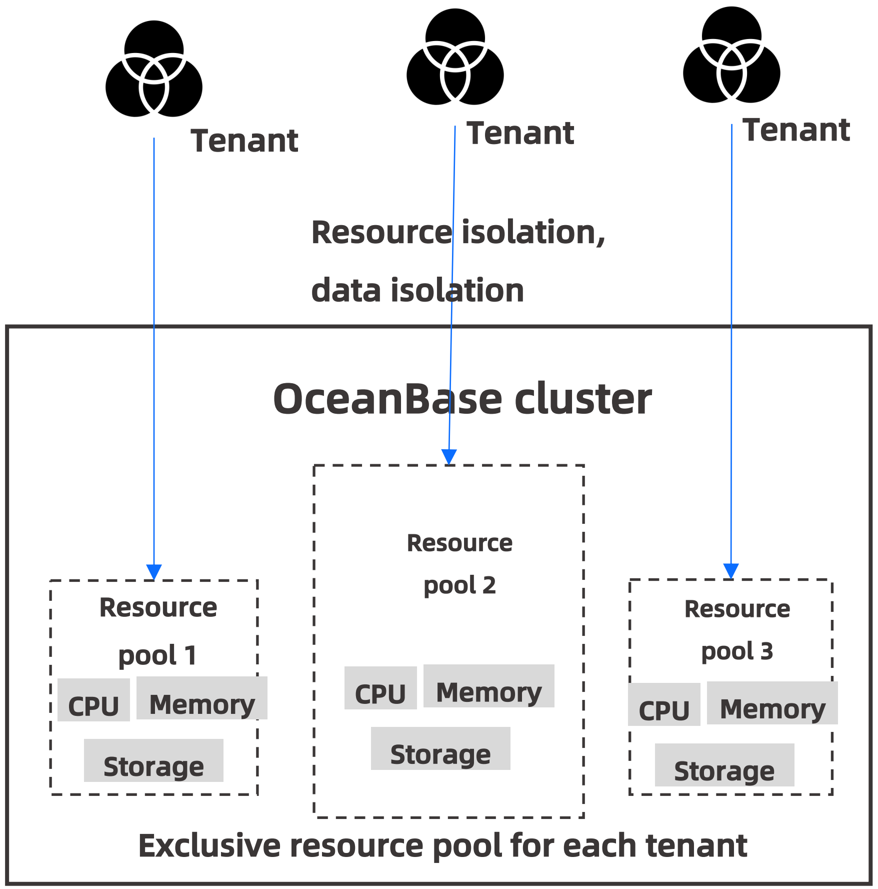
- OceanBase core module, managing the whole cluster
- Built-in service of the cluster, no additional software/hardware deployment
- Built-in high availability capability, no risk of single-point fault

Core Functionality

- System initialization (BootStrap); system metadata management
- Resource allocation and dispatch: zone and data copy management, dynamic load balancing, scale in/out, etc.
- Global DDL; cluster data merge

OCEANBASE

Multi-tenant mechanism, resource isolation, data isolation



Tenant introduction

- A database cluster is divided into multiple resource pools based on specifications (CPU, memory, storage, TPS, and QPS) and allocated to different tenants. Tenants are isolated from each other
- Generally, one application occupies one tenant

Tenant features

- A tenant is an instance of a traditional database that is created by a system tenant on demand (for example, for a business). When creating a tenant, in addition to specifying the tenant name, the most important thing is to specify the resources it occupies. The tenant has the following features:
  - ➢ Allow creating own user (different username and password)
  - ➢ Allow creating all object objects such as databases and tables
  - ➢ Independent system databases such as information_schema
  - ➢ Independent system variables
  - ➢ Additional features that database instances have

OCEANBASE

Each tenant has several resource pools

Unit

- ➢ Each UNIT describes a group of computing and storage resources on a Server. Each UNIT can belong to only one tenant
- ➢ Each Unit can be regarded as a lightweight VIRTUAL machine, including several CPU resources, memory resources, and disk resources

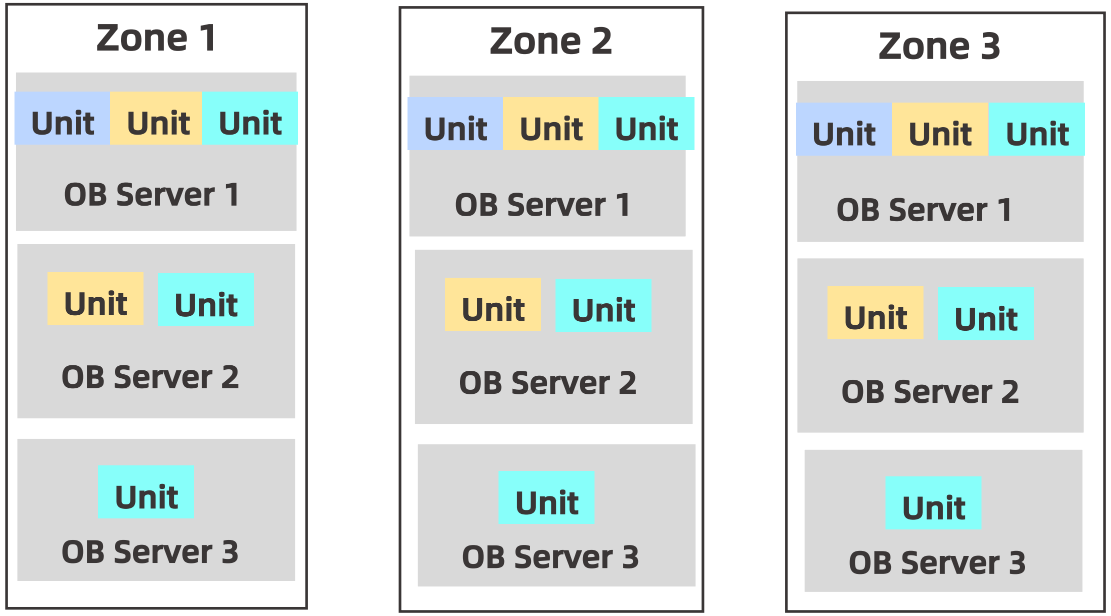
Tenant resource pool

- ➢ A tenant has multiple resource pools. The collection of these resource pools describes all the resources available to this tenant
- ➢ A tenant can have a maximum of one UNIT on a Server. In fact, conceptually, a copy is stored in a UNIT, which is a container for the copy

Firstly, define the specification of each unit (CPU and memory). For example U1=2C8G;

- Tenant 1: Assign a resource pool, Unit=U1, Unit Num=1
- Tenant 2: Assign a resource pool, Unit=U1, Unit Num=2
- Tenant 3: Assign a resource pool, Unit=U1, Unit Num=3

Tenant 1Unit 🟦
Tenant 2Unit 🟨
Tenant 3Unit 🟩



OceanBase can allocate different types and quantities of units to different types of applications to meet different service requirements.

Resources are not static and can be adjusted (up or down) as business grows

Create tenant

Basic process in creating tenant (or use OCP for the operation)

OCP also facilitates creating tenant. For better explanation for the creating process, the command line is used below:

- Step 1: Create Resource Unit Specifications. Run the create resource unit command to specify resource unit specifications.
- Step 2: Create a resource pool. Run the create resource pool command to create a resource unit based on the specifications of the resource unit and assign it to a new resource pool.
- Step 3: Create a tenant. Run the create Tenant command to assign a resource pool to a new tenant.

Create a tenant

Create resource unit (just defining the specifications without actual resource allocation)

```
CREATE RESOURCE UNIT unit1
  max_cpu = 4,
  max_memory = 10737418240, -- 10GB
  min_memory = 10737418240, -- 10GB
  max_iops = 1000,
  min_iops = 128,
  max_session_num = 300,
  max_disk_size = 21474836480 -- 20GB
;
```

Create a tenant

Create resource pool (create a unit actually and allocate resources by the specification definition)

```
CREATE RESOURCE POOL pool1
  UNIT = 'unit1',
  UNIT_NUM = 1,
  ZONE_LIST =  ('zone1', 'zone2', 'zone3')
;
```

- Each resource pool can have only one resource unit on each OB Server. If unit_num is greater than 1, each zone must have the same number of machines as unit_num
- Generally, the zone List is the same as the number of zones
- If no machine in a zone has sufficient resources to create a resource unit, the resource pool creation fails

Create a tenant

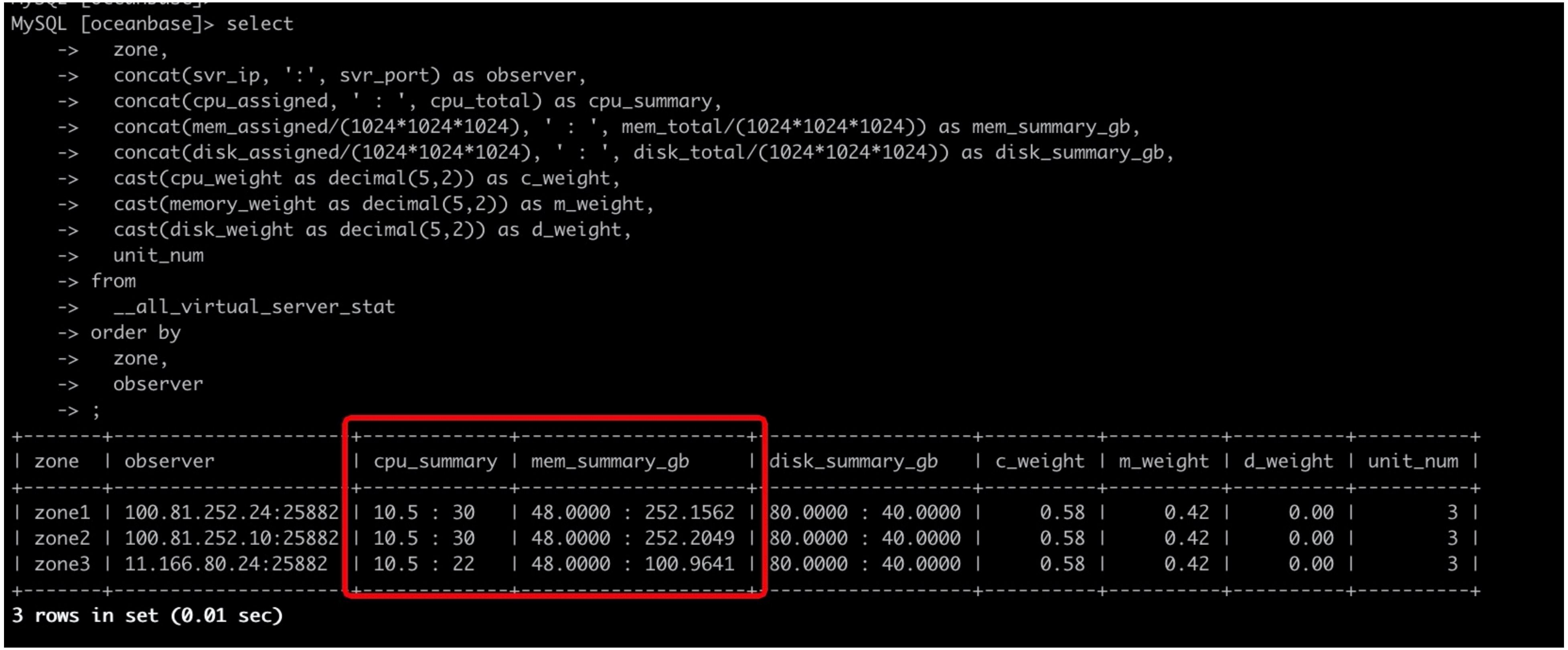
Create a tenant (allocate resource pool to the tenant)

```
CREATE TENANT mysql_tenant

  RESOURCE_POOL_LIST =  ('pool1') ,

  primary_zone = 'zone1,zone2,zone3'

  set ob_tcp_invited_nodes = '%', ob_compatibility_mode = 'mysql', recyclebin =
off, ob_timestamp_service = 'GTS’

;
```

- Primary Zone: Specifies the priority of the Primary copy assigned to the Zone. The priorities of both sides of a comma (,) are the same, and the priorities of the left side of a semicolon (,) are higher than those of the right side. For example, zone1, zone2; Zone3 (more on this later)
- Specify whether the tenant type is MySQL or Oracle

Check cluster status

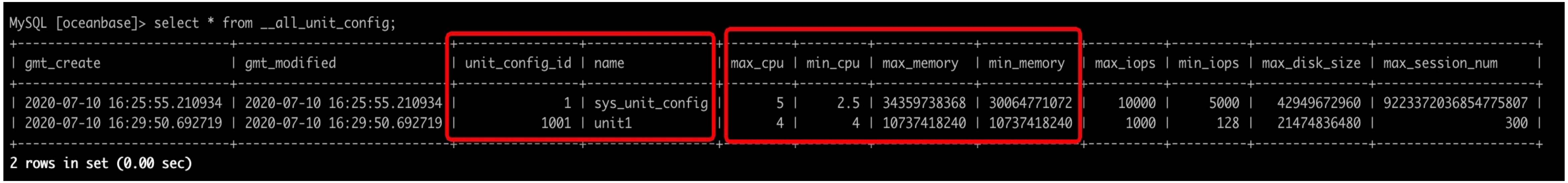
- Check the whole resource allocations in the cluster: __all_virtual_server_stat;



```
MySQL [oceanbase]> select
    ->   zone,
    ->   concat(svr_ip, ':', svr_port) as observer,
    ->   concat(cpu_assigned, ' : ', cpu_total) as cpu_summary,
    ->   concat(mem_assigned/(1024*1024*1024), ' : ', mem_total/(1024*1024*1024)) as mem_summary_gb,
    ->   concat(disk_assigned/(1024*1024*1024), ' : ', disk_total/(1024*1024*1024)) as disk_summary_gb,
    ->   cast(cpu_weight as decimal(5,2)) as c_weight,
    ->   cast(memory_weight as decimal(5,2)) as m_weight,
    ->   cast(disk_weight as decimal(5,2)) as d_weight,
    ->   unit_num
    -> from
    ->   __all_virtual_server_stat
    -> order by
    ->   zone,
    ->   observer
    -> ;
+-------+---------------------+-------------+-------------------+-------------------+----------+----------+----------+----------+
| zone  | observer            | cpu_summary | mem_summary_gb    | disk_summary_gb   | c_weight | m_weight | d_weight | unit_num |
+-------+---------------------+-------------+-------------------+-------------------+----------+----------+----------+----------+
| zone1 | 100.81.252.24:25882 | 10.5 : 30   | 48.0000 : 252.1562 | 80.0000 : 40.0000 |     0.58 |     0.42 |     0.00 |        3 |
| zone2 | 100.81.252.10:25882 | 10.5 : 30   | 48.0000 : 252.2049 | 80.0000 : 40.0000 |     0.58 |     0.42 |     0.00 |        3 |
| zone3 | 11.166.80.24:25882  | 10.5 : 22   | 48.0000 : 100.9641 | 80.0000 : 40.0000 |     0.58 |     0.42 |     0.00 |        3 |
+-------+---------------------+-------------+-------------------+-------------------+----------+----------+----------+----------+
3 rows in set (0.01 sec)
```

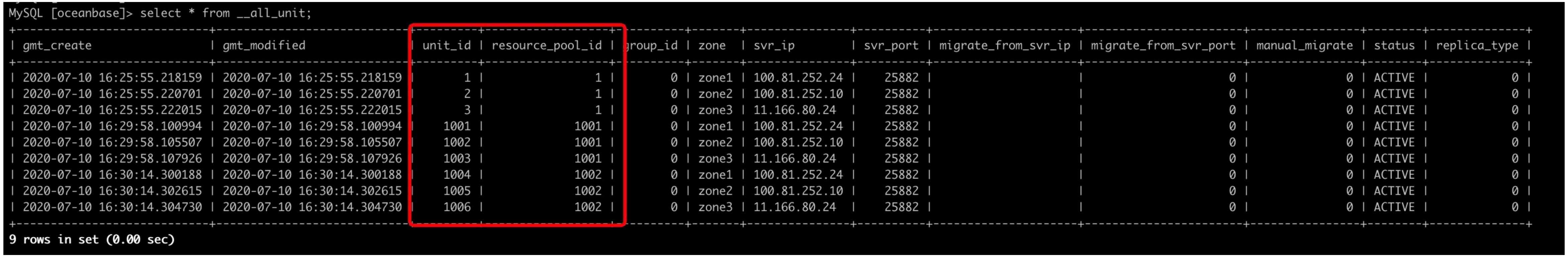
Check cluster status

- Check the resource unit specifications defined in the system: select * from __all_unit_config;



```
MySQL [oceanbase]> select * from __all_unit_config;
+----------------------------+----------------------------+----------------+-----------------+---------+---------+-------------+-------------+----------+----------+---------------+---------------------+
| gmt_create                 | gmt_modified               | unit_config_id | name            | max_cpu | min_cpu | max_memory  | min_memory  | max_iops | min_iops | max_disk_size | max_session_num     |
+----------------------------+----------------------------+----------------+-----------------+---------+---------+-------------+-------------+----------+----------+---------------+---------------------+
| 2020-07-10 16:25:55.210934 | 2020-07-10 16:25:55.210934 |              1 | sys_unit_config |       5 |     2.5 | 34359738368 | 30064771072 |    10000 |     5000 |   42949672960 | 9223372036854775807 |
| 2020-07-10 16:29:50.692719 | 2020-07-10 16:29:50.692719 |           1001 | unit1           |       4 |       4 | 10737418240 | 10737418240 |     1000 |      128 |   21474836480 |                 300 |
+----------------------------+----------------------------+----------------+-----------------+---------+---------+-------------+-------------+----------+----------+---------------+---------------------+
2 rows in set (0.00 sec)
```

- Check the resource unit allocated in the system: select * from __all_unit;



```
MySQL [oceanbase]> select * from __all_unit;
+----------------------------+----------------------------+---------+------------------+----------+-------+---------------+----------+---------------------+-----------------------+----------------+--------+--------------+
| gmt_create                 | gmt_modified               | unit_id | resource_pool_id | group_id | zone  | svr_ip        | svr_port | migrate_from_svr_ip | migrate_from_svr_port | manual_migrate | status | replica_type |
+----------------------------+----------------------------+---------+------------------+----------+-------+---------------+----------+---------------------+-----------------------+----------------+--------+--------------+
| 2020-07-10 16:25:55.218159 | 2020-07-10 16:25:55.218159 |       1 |                1 |        0 | zone1 | 100.81.252.24 |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:25:55.220701 | 2020-07-10 16:25:55.220701 |       2 |                1 |        0 | zone2 | 100.81.252.10 |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:25:55.222015 | 2020-07-10 16:25:55.222015 |       3 |                1 |        0 | zone3 | 11.166.80.24  |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:29:58.100994 | 2020-07-10 16:29:58.100994 |    1001 |             1001 |        0 | zone1 | 100.81.252.24 |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:29:58.105507 | 2020-07-10 16:29:58.105507 |    1002 |             1001 |        0 | zone2 | 100.81.252.10 |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:29:58.107926 | 2020-07-10 16:29:58.107926 |    1003 |             1001 |        0 | zone3 | 11.166.80.24  |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:30:14.300188 | 2020-07-10 16:30:14.300188 |    1004 |             1002 |        0 | zone1 | 100.81.252.24 |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:30:14.302615 | 2020-07-10 16:30:14.302615 |    1005 |             1002 |        0 | zone2 | 100.81.252.10 |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
| 2020-07-10 16:30:14.304730 | 2020-07-10 16:30:14.304730 |    1006 |             1002 |        0 | zone3 | 11.166.80.24  |    25882 |                     |                     0 |              0 | ACTIVE |            0 |
+----------------------------+----------------------------+---------+------------------+----------+-------+---------------+----------+---------------------+-----------------------+----------------+--------+--------------+
9 rows in set (0.00 sec)
```

Check system log

OceanBase log file

OB Server log (/home/admin/oceanbase/log directory)

- observer.log: Log file during the observer operation
- rootservice.log: Log file of RootServer on observer
- election.log: Log file of election module on observer

Controls the number of OB Server log files

- Enable log recycle to prevent the disk from being filled by logs
- enable_syslog_recycle = True; max_syslog_file_count = <count>

Log level

- syslog_level = [DEBUG,TRACE,**INFO**,WARN,USER_ERR,ERROR]

Summary

This chapter describes the OceanBase product family and some basic concepts (cluster, Zone, OB Server, resource pool, tenant, etc.)

- OceanBase product family consists of database kernel, OCP cloud management platform, ODC developer center and OMS migration service tool
- The OceanBase cluster consists of multiple zones, and each Zone consists of multiple PC servers. Zones have multiple copies. Each Zone stores one and only one copy
- Tenants are similar to traditional database instances. Each tenant has independent resources and can create its own users, databases, tables and other objects
- When creating a tenant, you need to specify the resource specifications of the tenant (how many resource units are allocated to the tenant by the server), the number of resource units (how many servers in the Zone allocate resource units to the tenant), and the compatibility mode (MySQL or Oracle).
- Tenants' resources are not constant, but can be adjusted dynamically (for example, by increasing resource specifications or adjusting the number of resource units).

Q&A

OCEANBASE

Simulation Questions (1)

1.[True/false] OceanBase has been published in Aliyun public cloud and private cloud ()

2.[True/false] OceanBase only supports CPUS of X86 architecture, not other CPUs (such as Kunpeng, Hygon, Phytium, etc.) ()

3.[True/false] Zone is a logical concept that assigns the same tag to a group of servers in a cluster. Servers with the same tag belong to the same Zone ()

4.[True/false] Zones can correspond to different cities, equipment rooms in a city, or racks in an equipment room ()

5.[True/false] Once a tenant's resource pool is created completely, it cannot be changed ()

6. [Single choice] OceanBase is a cluster. Which component manages the entire cluster and supports functions such as global DDL and cluster data consolidation? ()

A: OB Proxy B: RootService general control service C: OCP management platform D: ODC developer center

7. [Single choice] OceanBase cluster can support both MySQL and Oracle tenants, which blank screen tool can connect to Oracle tenants? ()

A: OceanBase client; B: Standard MySQL client

Simulation Questions (2)

8. [Single choice] Which operating system does OceanBase not support? ()

A: CentOS； B: Windows C: NeoKylin D: Galaxy Kylin

9. [Single choice] If an OceanBase cluster has three zones, each Zone has five OB serers. How many universal copies can a partition have at most? ()

A: 10 B: 3 C: 6 D: 5

10. [Single choice] If a cluster has three zones, each Zone has five OB servers. Unit Num of the resource pool corresponding to a tenant is 3. How many resource units does the tenant have? ( )

A: 15 B: 9 C: 45 D: 30

11. [Multiple choice] Which products is OceanBase composed of? ()

A: Database kernel: SQL engine and storage engine, compatible with MySQL and Oracle modes; Use the Paxos protocol to ensure high availability;

B: OCP cloud management platform: provides management tools for administrators, such as cluster management, Zone management, and tenant management.

C: OMS data migration tool: synchronizes baseline data and incremental data, subscribes to data links from data warehouses, and migrates data from heterogeneous databases.

D: ODC Developer Center: provides daily database development, SQL diagnosis, session management and data import and export functions.

Chapter 4: OceanBase Cluster Architecture

1. **Paxos protocol and load balance**
2. Dynamic scale-out and scale-in
3. Data reliability and availability
4. Distributed transaction, MVCC, transaction isolation level

Data partition and partition copy

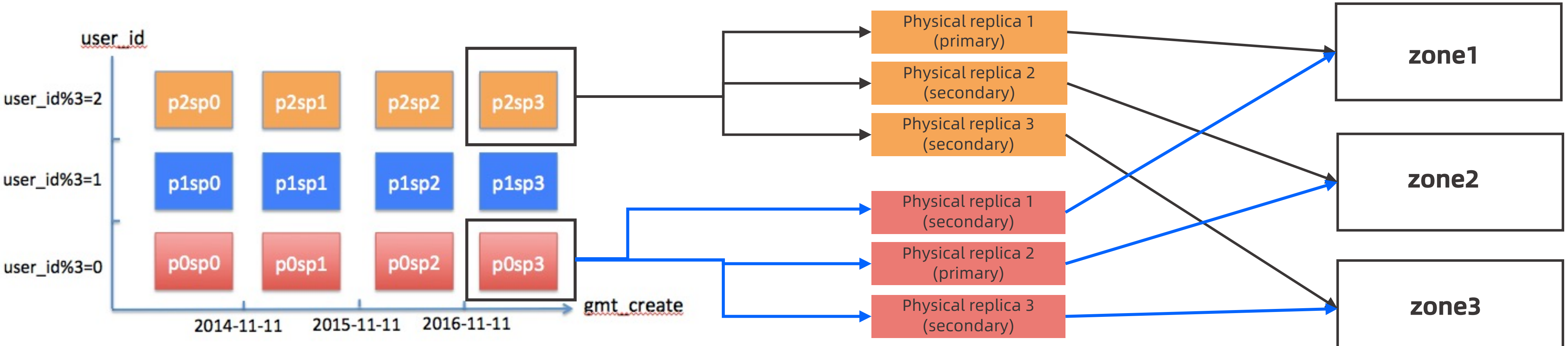
Partition

- When a table is large, it can be split horizontally into partitions, each containing rows of the table. Partitions can be categorized into Hash partitions, List partitions, and Range partitions based on the mapping relationship between row data and partitions
- Each partition can also be further divided into several partitions with different partition type, which is called subpartitions
- Partitioning is the basic unit of OceanBase data architecture. It is the implementation of partitioned table of traditional database in a distributed system

Copy

- For data security and high availability, the data of each partition is physically stored in multiple copies, each is called a partition replica
- replicas are automatically scheduled by the system to be distributed across multiple servers based on the load and specific policies. Replicas support management operations such as data migration, replication, addition, deletion, and type conversion

For example, the transaction record table is divided into three hash partitions (red, blue, and yellow) based on the user ID. Each level-1 hash partition is further divided into four range partitions based on the transaction time.



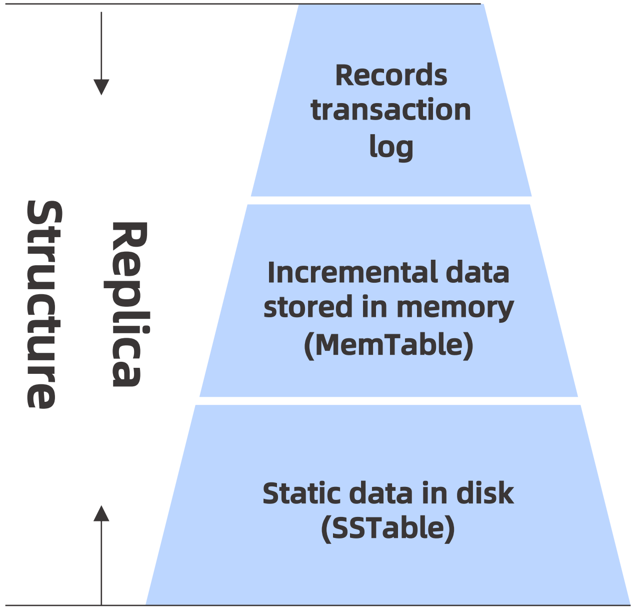
user_id
user_id%3=2
user_id%3=1
user_id%3=0
p2sp0
p2sp1
p2sp2
p2sp3
p1sp0
p1sp1
p1sp2
p1sp3
p0sp0
p0sp1
p0sp2
p0sp3
2014-11-11
2015-11-11
2016-11-11
gmt_create
Physical replica 1 (primary)
Physical replica 2 (secondary)
Physical replica 3 (secondary)
Physical replica 1 (secondary)
Physical replica 2 (primary)
Physical replica 3 (secondary)
zone1
zone2
zone3

Each table is divided into multiple partitions

Each partition has multiple replicas

Individual replica of partition is stored in individual zone

Three types of replica to meet different service demands


Records transaction log
Incremental data stored in memory (MemTable)
Static data in disk (SSTable)
Replica
Structure

- **A partition can have ONLY one full-featured or log type replica in one partition**
- **Multiple read-only replicas can exist in one partition**

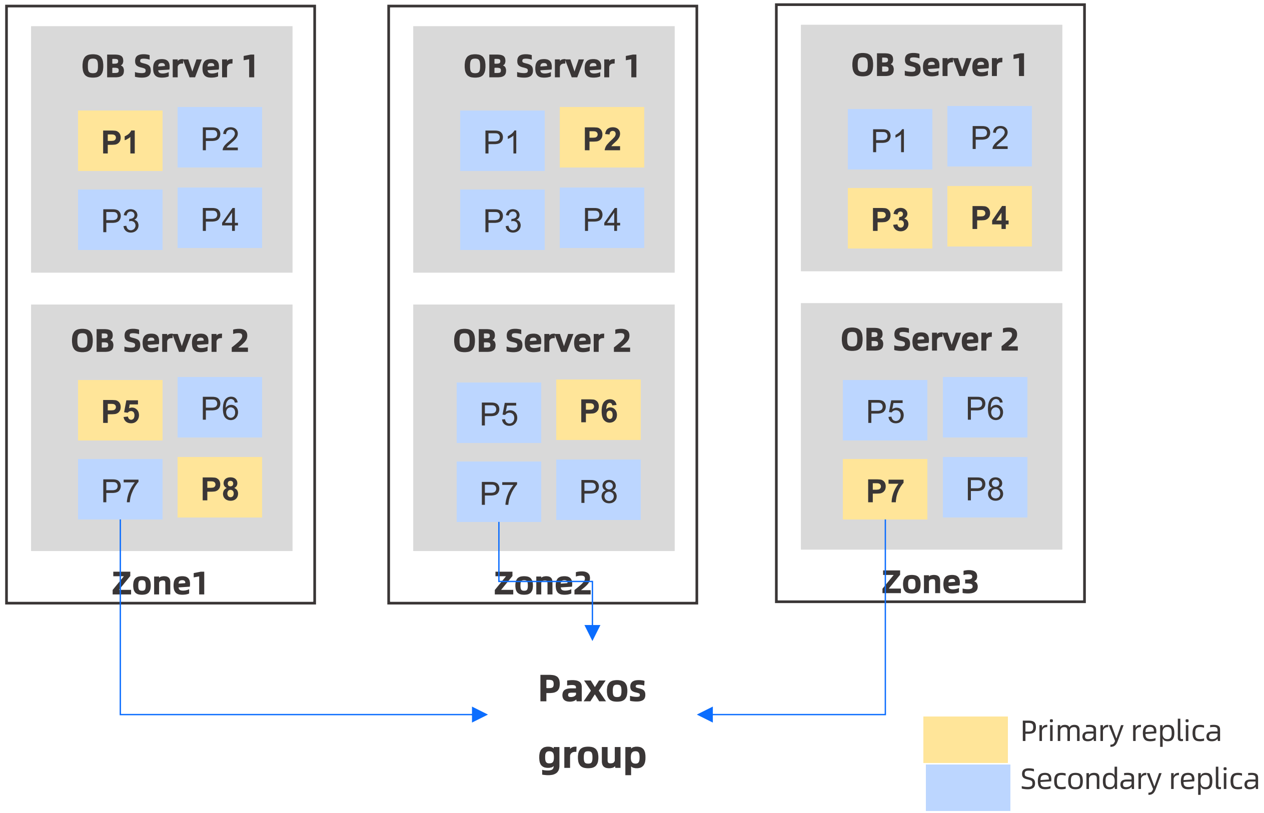
Depending on the type of data stored, there are several different types of copies to enable different businesses to make trade-offs between data security, performance scalability, availability, and cost

- **Full-featured replica:** This is the generic replica type, with complete data and functionality for transaction logging, MemTable and SSTable. It can quickly switch to the Leader to provide services externally at any time
- **Log type replica:** Contains only copies of logs, without MemTable and SSTable. It participates in log vote and provides logging services externally, and can participate in the recovery of other replicas, but cannot become the primary database service itself. Because log replicas consume fewer physical resources (CPU, memory, disk), they can effectively reduce the hardware cost, which in turn reduces the overall cost of the cluster
- **Read-only replica:** Contains complete logs, MemTable, SSTable, etc., but it's special log, which does not vote as PaxOS member, but as an observer which catch up with the paxOS member's logs in real time and apply them back locally. Such replicas can provide read-only access when apps does not require strong read consistency. Because it is not part of the PaxOS group member, hence it does not cause of additional transaction delay

Type	Log	MemTable	SSTable	Data security
Full-featured	Yes, participating in vote	Yes	Yes	High
Log	Yes, participating in vote	No	No	Low
Read-only	Yes, but only a listener, not part of paxos group member	Yes	Yes	Middle

Multi-replica consistency protocol

- **Paxos protocol group formed by partition:** Each partition has multiple replicas, which form up Paxos group automatically. On the partition level, multiple replicas use to ensure data reliability and availability, manage data more efficiently
- **Leader replica auto-selection** : OB automatically creates multiple replicas, also, automatically elect the leader. Primary replica provides apps access (as shown to the right, yellow replicas are accessed by applications, and blue replicas are used for redundancy purpose)



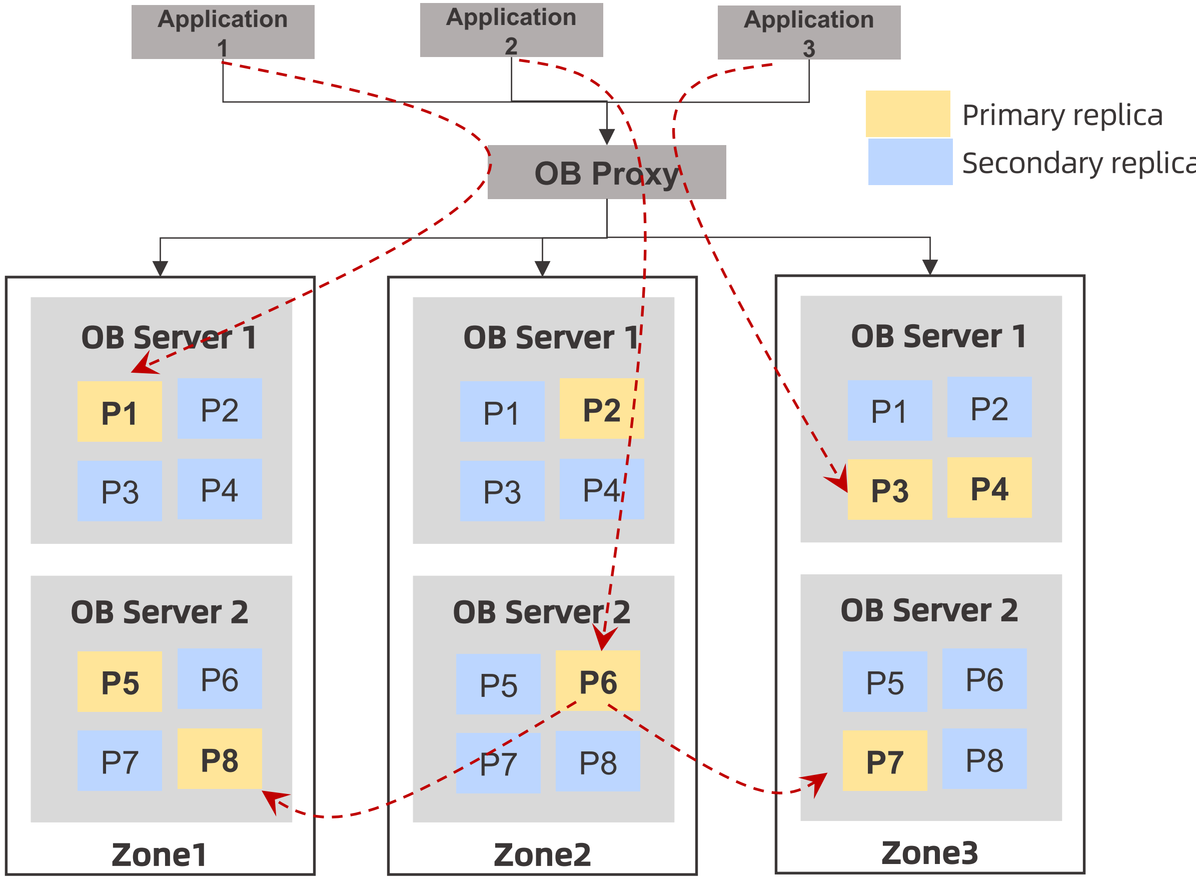
OCEANBASE

Copyright © Beijing OceanBase Technology

Load balancing and intelligent routing

- **Auto load balance**: The Primary replica is evenly distributed across each server which allows each server to carry workload. (As shown in diagram, apps 1/apps 2/apps 3, each apps read request routes to different OB server)

- **Each OB Server is independent**: Each OB Server can execute SQL independently. If an app need to access data on different machines, the OB Server will automatically route the request to the machine where the data resides, which is completely transparent to the apps (for example, application 2->P6->P7/P8).

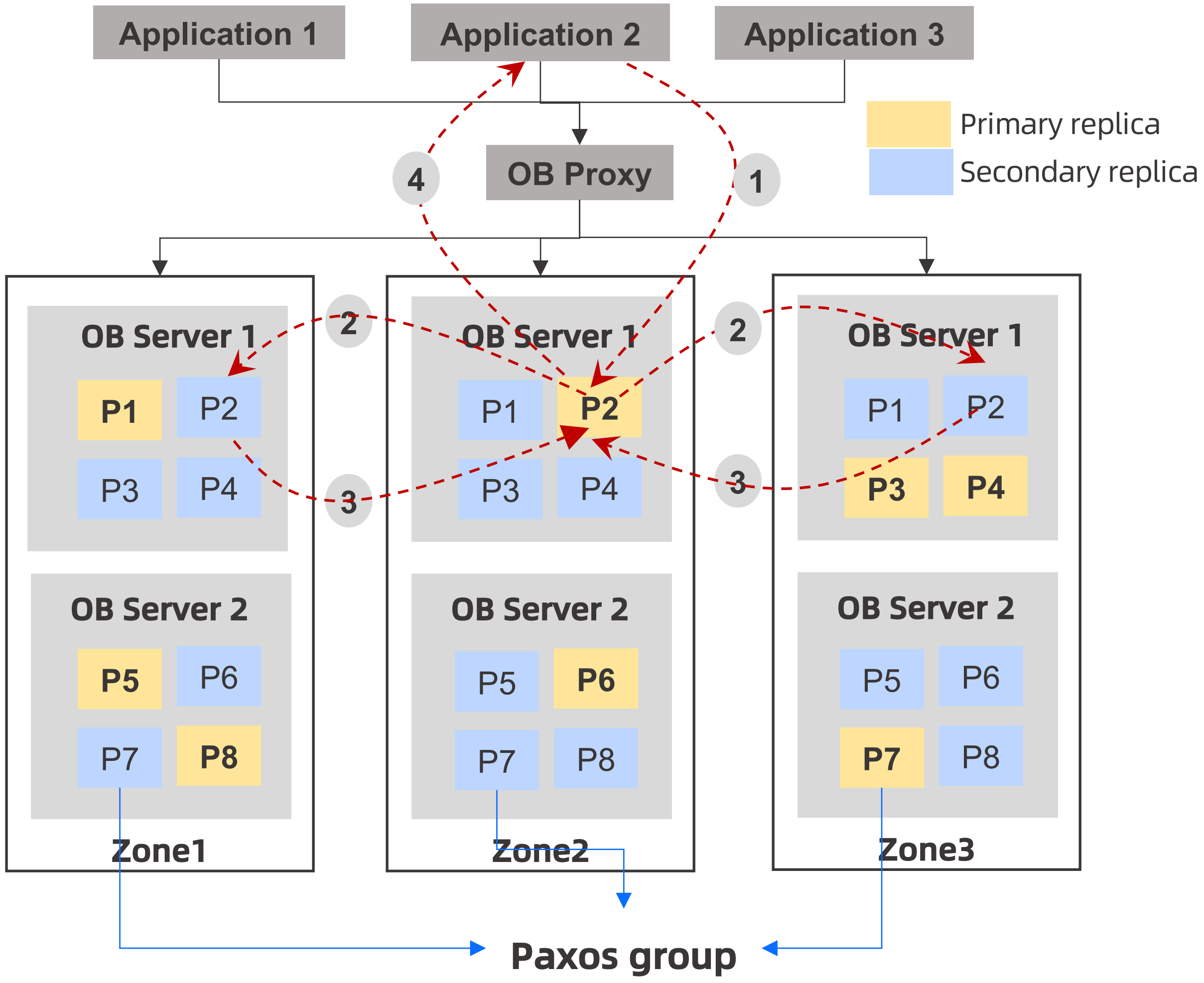


Synchronize Redo Log via multiple replicas to ensure data persistency

- **Paxos group members ensure data persistence via majority strong synchronization of redo-log**
- **The Leader does not need to wait for acknowledgment from all followers, as long as majority of followers sync completion is done, it signals apps a 'success'state**

Example

1. Apps write data to partition P2. P2 Primary replica in Zone2-ob Server1 takes the request
2. Send redo-log synchronization requests to the P2 Follower replica in Zone1-OB Server1 and Zone3-OB Server1
3. As long as any Follower completes the redo-log disk write and sends the response back to the Leader, then Leader considers that the redo-log has been strongly synchronized and does not need to wait for acknowledgement from other followers
4. The Leader signals apps that operation is complete



Obtain table and partition distribution info

Related information of table and partitions is stored in the two system tables.

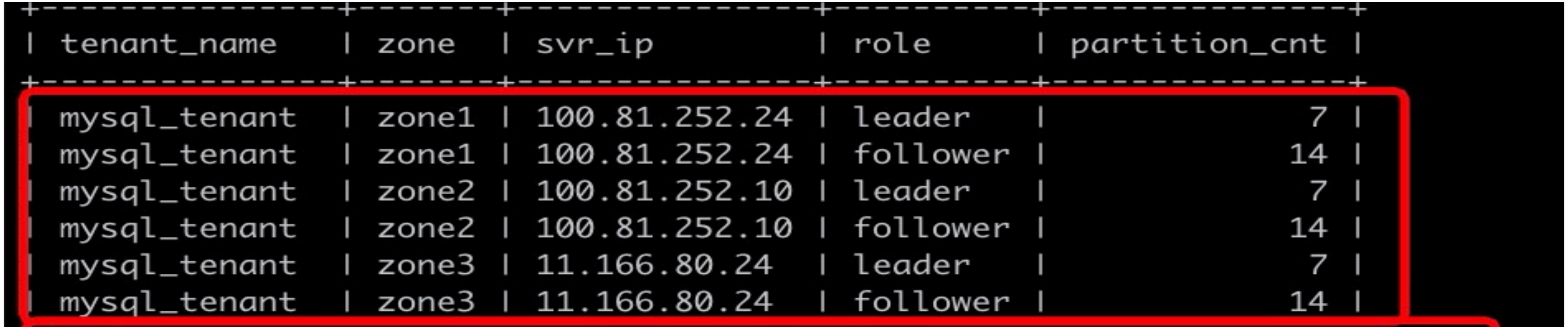
System table: __all_virtual_meta_table, key information

- tenant_id
- table_id
- partition_id
- svr_ip
- role: 1 – leader, 2 - follower

System table: __all_virtual_table, key information

- table_id
- table_name

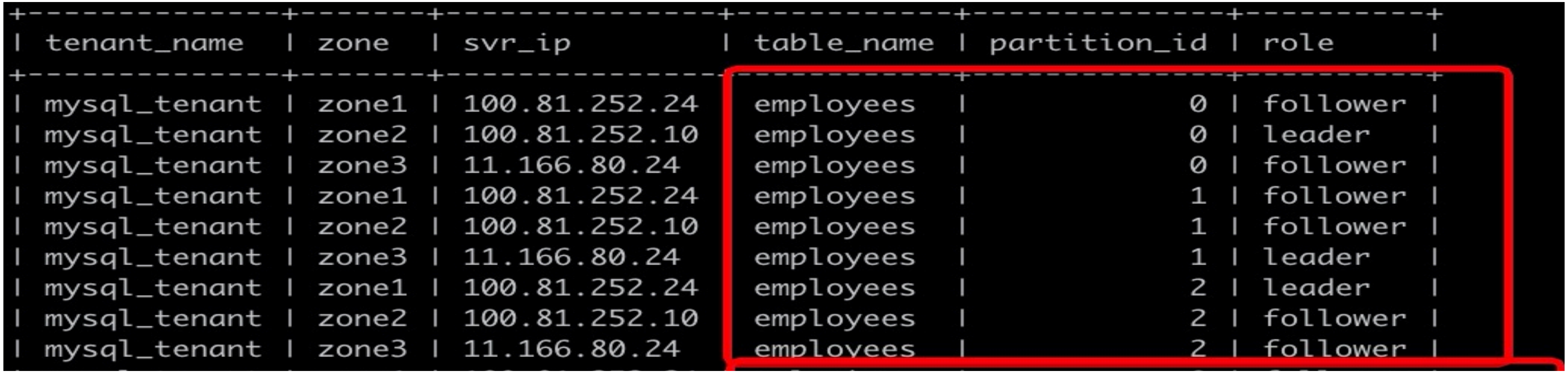
Obtain tenant partition distribution info



```
+--------------+-------+---------------+----------+---------------+
| tenant_name  | zone  | svr_ip        | role     | partition_cnt |
+--------------+-------+---------------+----------+---------------+
| mysql_tenant | zone1 | 100.81.252.24 | leader   |             7 |
| mysql_tenant | zone1 | 100.81.252.24 | follower |            14 |
| mysql_tenant | zone2 | 100.81.252.10 | leader   |             7 |
| mysql_tenant | zone2 | 100.81.252.10 | follower |            14 |
| mysql_tenant | zone3 | 11.166.80.24  | leader   |             7 |
| mysql_tenant | zone3 | 11.166.80.24  | follower |            14 |
```

- The tenant has a total of 21 partitions (21 Primary replicas and 42 Secondary replicas), which are spread across 6 servers of Zone1/2/3
- Each Zone has seven Primary replicas, which implements load balancing for three zones

Check table partition distribution



```
+--------------+-------+---------------+------------+--------------+----------+
| tenant_name  | zone  | svr_ip        | table_name | partition_id | role     |
+--------------+-------+---------------+------------+--------------+----------+
| mysql_tenant | zone1 | 100.81.252.24 | employees  |            0 | follower |
| mysql_tenant | zone2 | 100.81.252.10 | employees  |            0 | leader   |
| mysql_tenant | zone3 | 11.166.80.24  | employees  |            0 | follower |
| mysql_tenant | zone1 | 100.81.252.24 | employees  |            1 | follower |
| mysql_tenant | zone2 | 100.81.252.10 | employees  |            1 | follower |
| mysql_tenant | zone3 | 11.166.80.24  | employees  |            1 | leader   |
| mysql_tenant | zone1 | 100.81.252.24 | employees  |            2 | leader   |
| mysql_tenant | zone2 | 100.81.252.10 | employees  |            2 | follower |
| mysql_tenant | zone3 | 11.166.80.24  | employees  |            2 | follower |
```

- The table has 3 partitions, each contains 3 replicas, spread across each OB Server in 3 zones
- Each zone has a partition Primary replica for load balancing

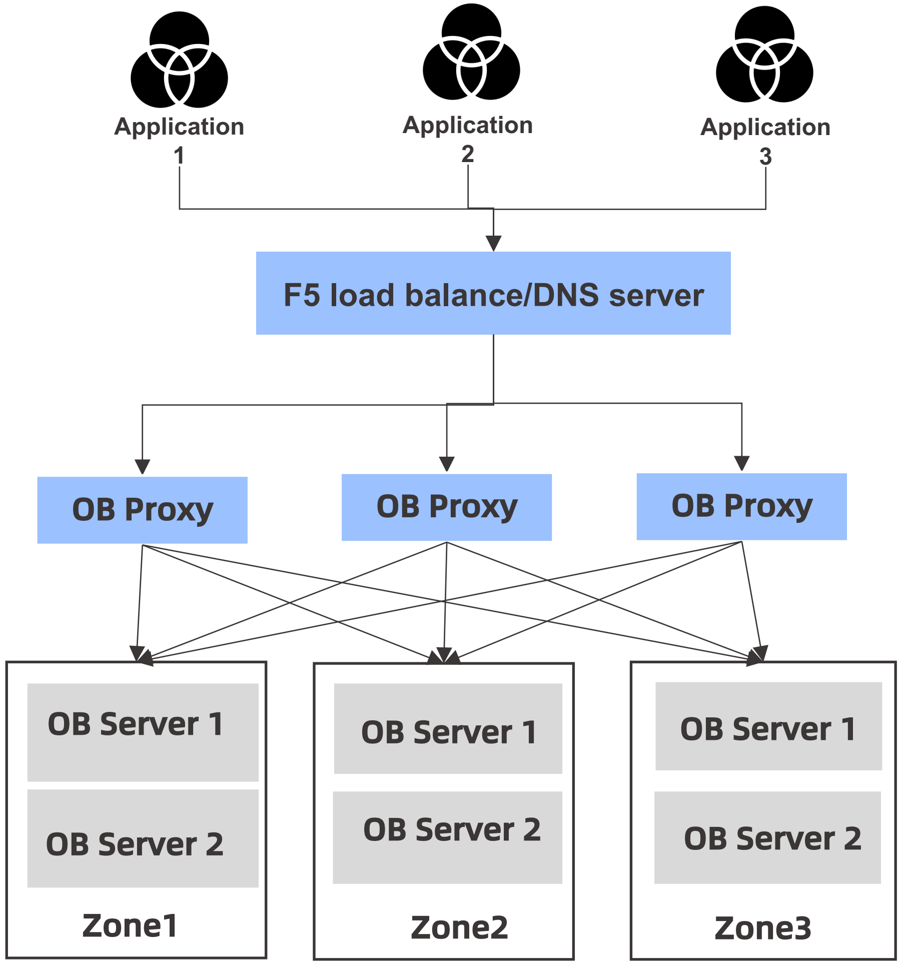
OBProxy: Intelligent routing for transparent apps access

Efficient routing & forwarding

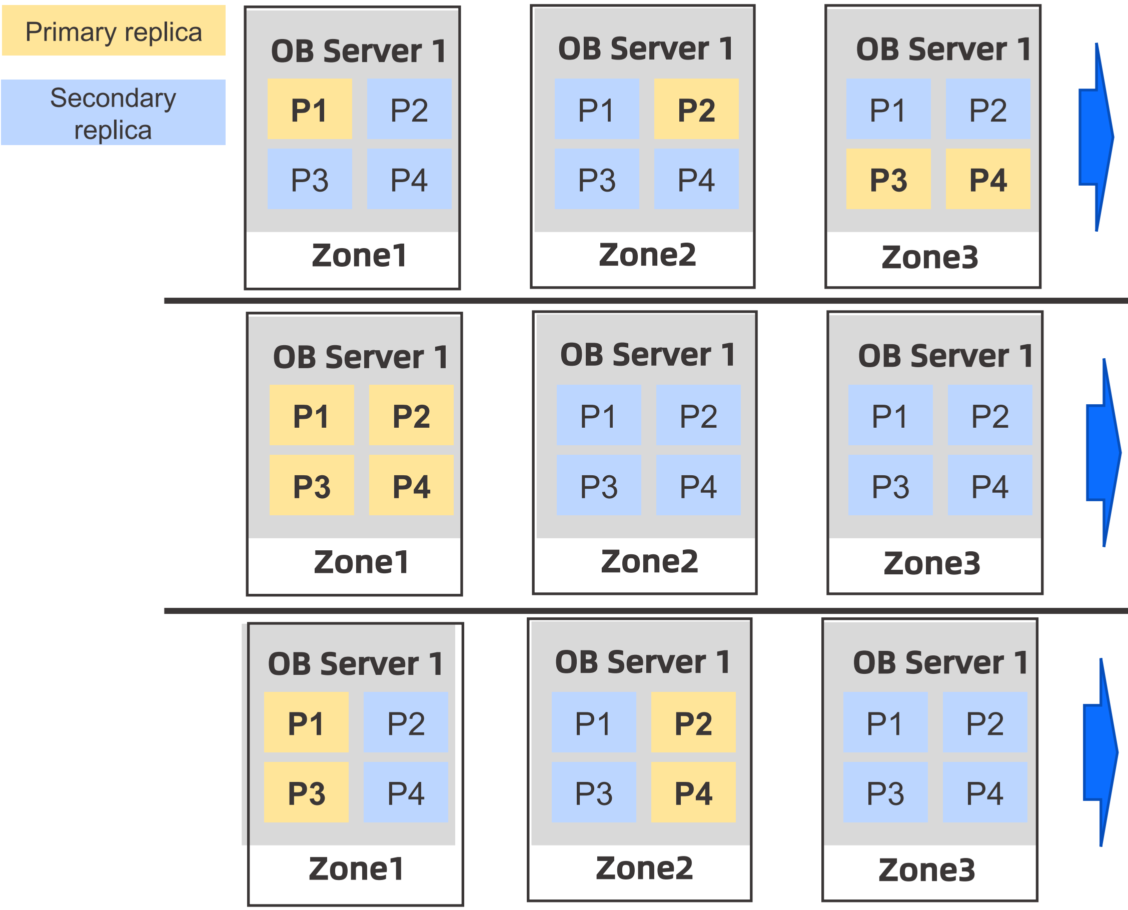
- Perform simple SQL analysis to determine the leader machine
- Forward route requests to corresponding Leader machine. If the Leader cannot be identified, the OBserver is randomly selected
- Lightweight SQL parsing + fast forwarding to ensure high performance, single OBProxy can forward millions of requests per second

Non-computing node, stateless

- Each OBProxy is a "stateless" that does not persist data and has no requirements for deployment location
- OBProxy does not participate computing tasks of database engine or transaction processing (single or distributed)
- Multiple OB proxies is isolated and can form up a cluster using F5/SLB, which provide load balancing capability
- Hardware server is not required. OBProxy can share the same hardware server with OB Server. If apps requires real-time performance, OBProxy can also be deployed in the middleware application Server



Primary Zone: Consolidate apps access to specific zone



Configuration: (z1,z2,z3) Meaning: ZONE_1 =ZONE_2 = ZONE_3

- Primary replicas are distributed evenly across machines
- Applicable for batch processing scenario where it desires to complete the whole task ASAP without focusing on particular SQL execution time

Configuration 2: (z1;z2;z3) Meaning : ZONE_1 >ZONE_2 > ZONE_3

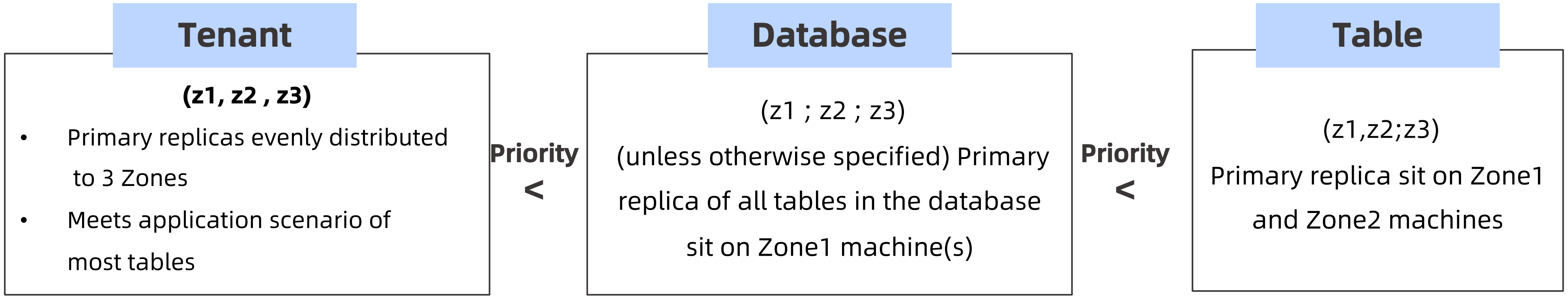
- Primary replica exists in Zone1 only while Zone2 and Zone3 have only follower replicas
- Applicable for delay-sensitive online processing business where the workload is not big and does not exceed the processing capability of single machine. This minimizes the possibility of across machine access, reducing time delay.

Configuration 3: (z1,z2;z3) Meaning : (ZONE_1 = ZONE_2) > ZONE_3

- Primary replicas are evenly distributed across Zone1 and Zone2, while Zone3 has follower replica only
- Applicable for the 3-location 5-IDC plan, centralize business to the nearest cities, which farther cities play the role of follower replica

By configuring different Primary zones for different tenants, workload can be prioritized into multiple zones, reducing cross-zone and cross-server operations. In the Zone List, comma (,) means the priority of both sides are identical, semicolon (;) means priority of left side is higher than that of the right side

Primary Zone have different levels : tenant, database and table



Tenant

(z1, z2 , z3)

- Primary replicas evenly distributed to 3 Zones
- Meets application scenario of most tables

Priority <

Database

(z1 ; z2 ; z3)

(unless otherwise specified) Primary replica of all tables in the database sit on Zone1 machine(s)

Priority <

Table

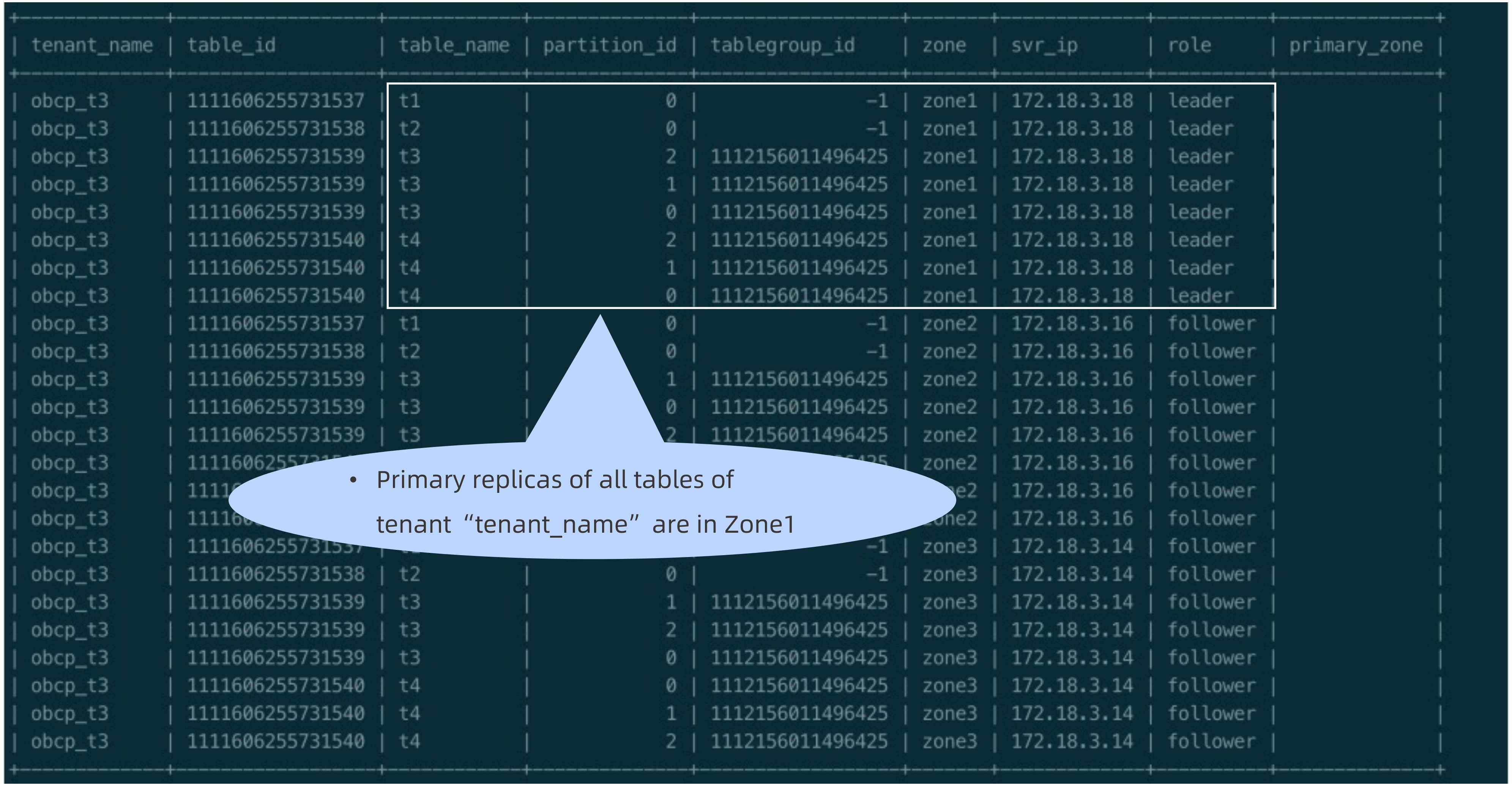
(z1,z2;z3)

Primary replica sit on Zone1 and Zone2 machines

- **If no special parameter is specified, primary zone of the upper-level object is automatically inherited. Database inherits the primary zone settings of tenant, and table inherits the primary zone settings of database**

- **The database and table can specify their own primary zone, which does not have to be the same as that of the upper-level, hence provide flexible load balancing policies**

Primary Zone example



```
+-------------+------------------+------------+--------------+------------------+-------+-------------+----------+--------------+
| tenant_name | table_id         | table_name | partition_id | tablegroup_id    | zone  | svr_ip      | role     | primary_zone |
+-------------+------------------+------------+--------------+------------------+-------+-------------+----------+--------------+
| obcp_t3     | 1111606255731537 | t1         |            0 |               -1 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731538 | t2         |            0 |               -1 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731539 | t3         |            2 | 1112156011496425 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731539 | t3         |            1 | 1112156011496425 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731539 | t3         |            0 | 1112156011496425 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731540 | t4         |            2 | 1112156011496425 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731540 | t4         |            1 | 1112156011496425 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731540 | t4         |            0 | 1112156011496425 | zone1 | 172.18.3.18 | leader   |              |
| obcp_t3     | 1111606255731537 | t1         |            0 |               -1 | zone2 | 172.18.3.16 | follower |              |
| obcp_t3     | 1111606255731538 | t2         |            0 |               -1 | zone2 | 172.18.3.16 | follower |              |
| obcp_t3     | 1111606255731539 | t3         |            1 | 1112156011496425 | zone2 | 172.18.3.16 | follower |              |
| obcp_t3     | 1111606255731539 | t3         |            0 | 1112156011496425 | zone2 | 172.18.3.16 | follower |              |
| obcp_t3     | 1111606255731539 | t3         |            2 | 1112156011496425 | zone2 | 172.18.3.16 | follower |              |
| obcp_t3     | 11116062557                                              6425 | zone2 | 172.18.3.16 | follower |              |
| obcp_t3     | 1111                                                            e2 | 172.18.3.16 | follower |              |
| obcp_t3     | 11116                                                          one2 | 172.18.3.16 | follower |              |
| obcp_t3     | 1111606255731537                                           -1 | zone3 | 172.18.3.14 | follower |              |
| obcp_t3     | 1111606255731538 | t2         |            0 |               -1 | zone3 | 172.18.3.14 | follower |              |
| obcp_t3     | 1111606255731539 | t3         |            1 | 1112156011496425 | zone3 | 172.18.3.14 | follower |              |
| obcp_t3     | 1111606255731539 | t3         |            2 | 1112156011496425 | zone3 | 172.18.3.14 | follower |              |
| obcp_t3     | 1111606255731539 | t3         |            0 | 1112156011496425 | zone3 | 172.18.3.14 | follower |              |
| obcp_t3     | 1111606255731540 | t4         |            0 | 1112156011496425 | zone3 | 172.18.3.14 | follower |              |
| obcp_t3     | 1111606255731540 | t4         |            1 | 1112156011496425 | zone3 | 172.18.3.14 | follower |              |
| obcp_t3     | 1111606255731540 | t4         |            2 | 1112156011496425 | zone3 | 172.18.3.14 | follower |              |
+-------------+------------------+------------+--------------+------------------+-------+-------------+----------+--------------+
```

- Primary replicas of all tables of tenant “tenant_name” are in Zone1

Table Group gathers the primary replica with the same partition ID of multiple tables in one OB Server, reducing the overhead introduced by distributed transactions

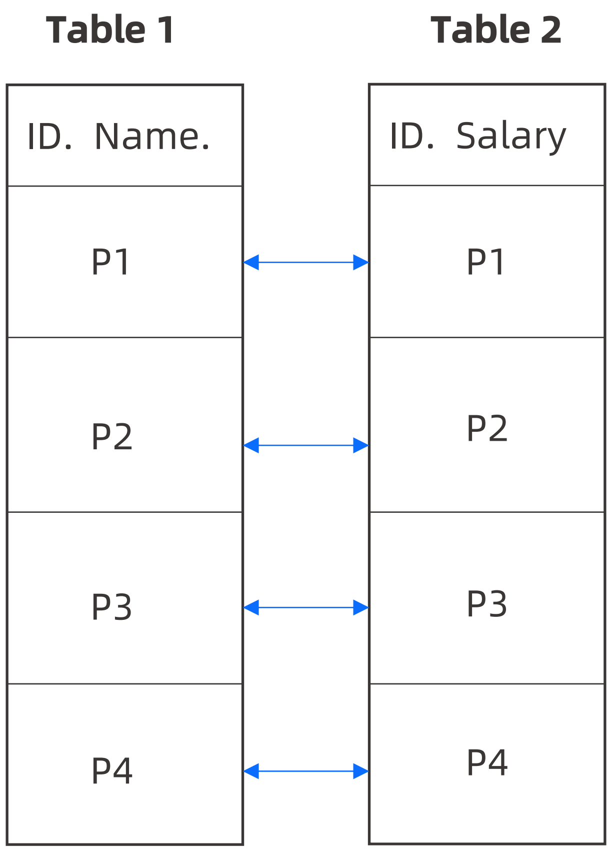


Table 1

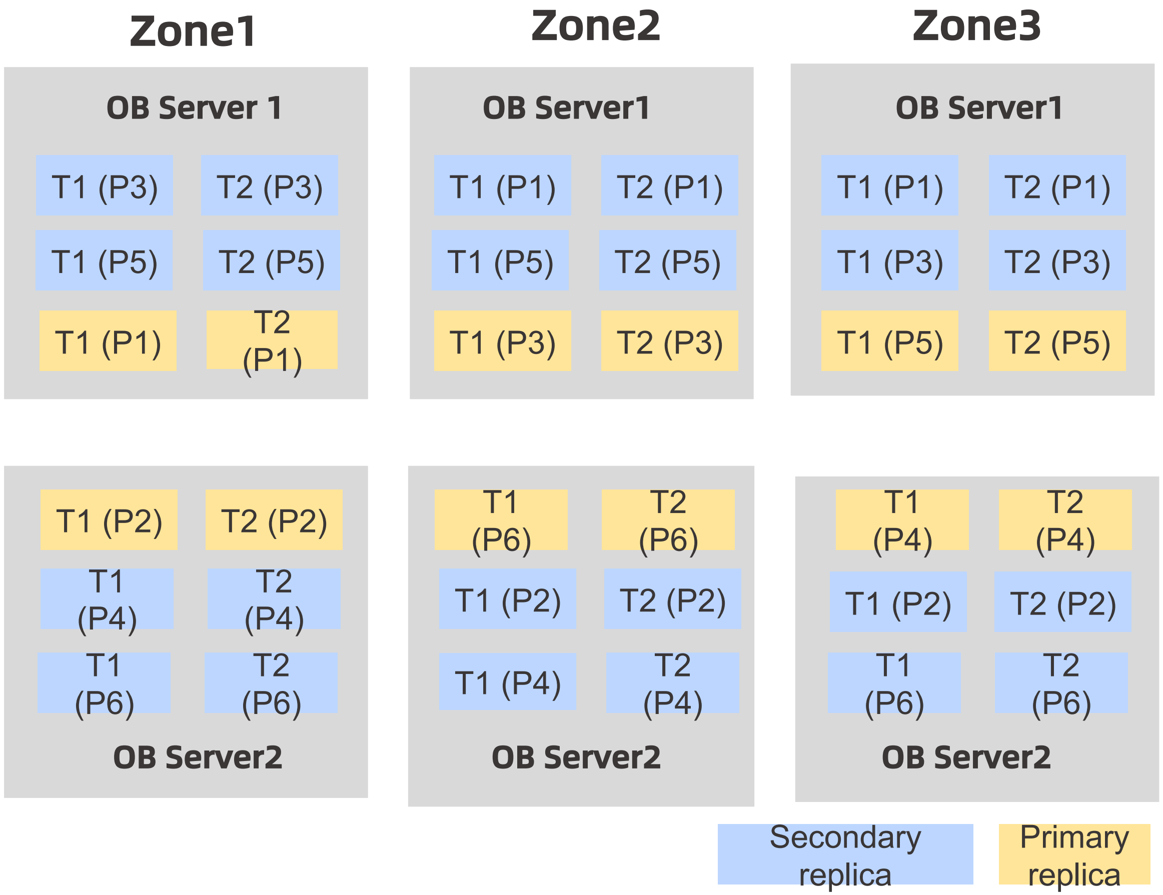
ID. Name.
P1
P2
P3
P4

Table 2

ID. Salary
P1
P2
P3
P4

- If multiple tables have exactly the same partitioning method (partition type, number of partition keys, number of partitions, etc.), then these tables can be put into one logical Table Group, which could affect dynamic load balancing policy
- For every table in the Table Group, Leader partitions with identical partition ID (partition_id) are on the same OBServer. Table Group effectively reduce the cross machine overhead introduced by distributed transaction without affecting overall load balancing mechanism
- If load balance is broken (e.g., hardware failure, scale in/out, etc.), all tables in the Table Group are treated together during partition/leader re-distribution
- can be created or removed dynamically, completely transparent to apps
- If tenant's unit_num=1 or primary_zone contains only one zone, the tablegroup is not needed.

Table Group example



Zone1

OB Server 1

T1 (P3) T2 (P3)

T1 (P5) T2 (P5)

T1 (P1) T2 (P1)

T1 (P2) T2 (P2)

T1 (P4) T2 (P4)

T1 (P6) T2 (P6)

OB Server2

Zone2

OB Server1

T1 (P1) T2 (P1)

T1 (P5) T2 (P5)

T1 (P3) T2 (P3)

T1 (P6) T2 (P6)

T1 (P2) T2 (P2)

T1 (P4) T2 (P4)

OB Server2

Zone3

OB Server1

T1 (P1) T2 (P1)

T1 (P3) T2 (P3)

T1 (P5) T2 (P5)

T1 (P4) T2 (P4)

T1 (P2) T2 (P2)

T1 (P6) T2 (P6)

OB Server2

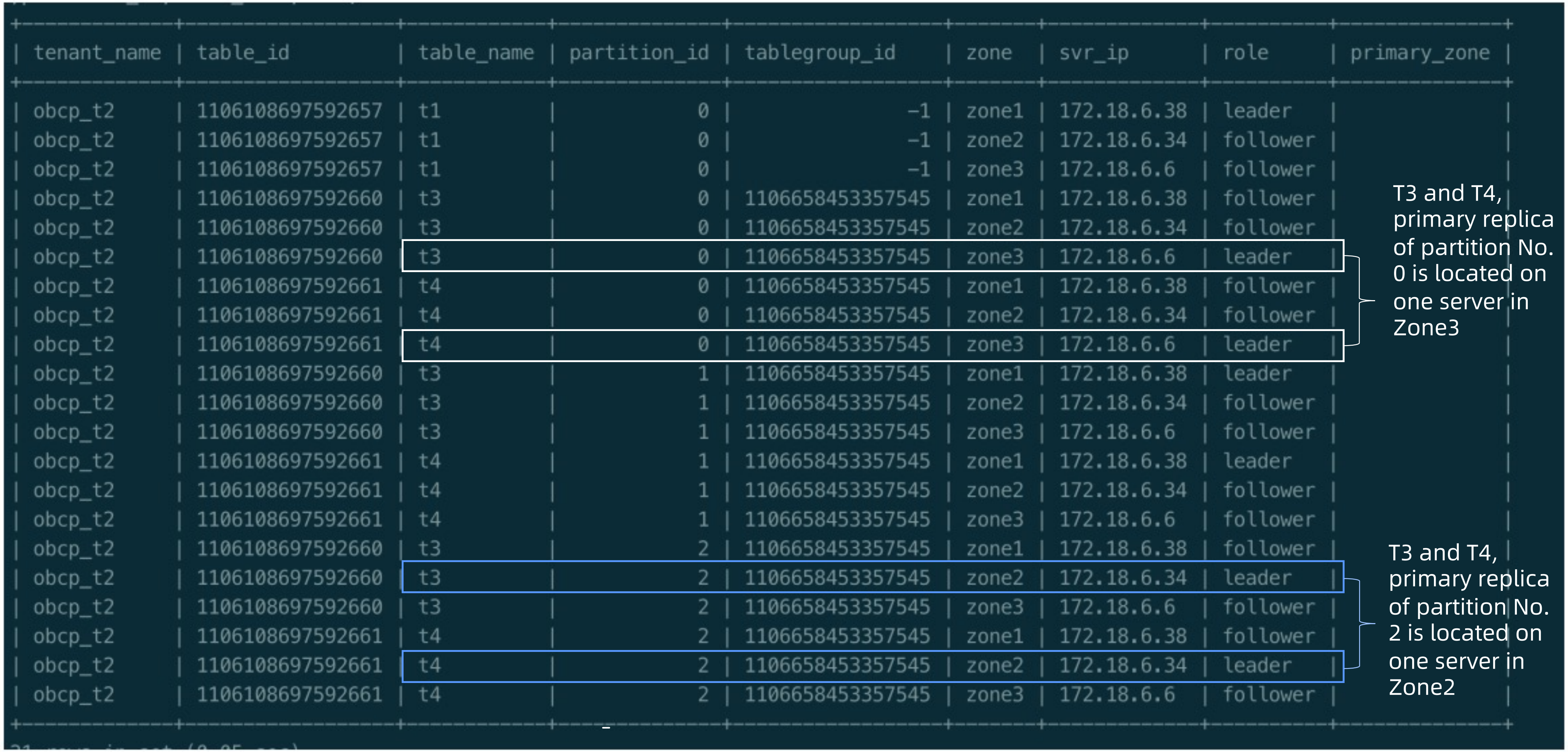
Secondary replica

Primary replica

- Assume that a cluster has three zones, the tenant's Unit Num is 2, and the Primary zones in Table 1 and Table 2 are load balanced
- For table Employee (emp_id), employee’s information and salary must be served on the same OBServer, e.g. Employees (P1) and Salaries (p1)
- Employees and Salaries tables form a Table Group
- The primary replica of the same partition ID of the Table Group is on same OB Server
- The overall load balancing is not broken. Each OB Server has a Primary replica and a Secodnary replica
- If Zone1-OB Server1 fails, T1 (P1) +T2 (P1) are moved as a whole unit

```
create tablegroup emp_id_group partition by hash partitions 6;

create table employees  (emp_id integer, emp_name varchar (64) )
tablegroup = 'emp_id_group' partition by hash  (emp_id)  partitions 6;

create table salaries  (emp_id integer, emp_salary number (10,2) )
tablegroup = 'emp_id_group' partition by hash  (emp_id)  partitions 6;
```

Table Group example



```
+-------------+------------------+------------+--------------+------------------+-------+-------------+----------+--------------+
| tenant_name | table_id         | table_name | partition_id | tablegroup_id    | zone  | svr_ip      | role     | primary_zone |
+-------------+------------------+------------+--------------+------------------+-------+-------------+----------+--------------+
| obcp_t2     | 1106108697592657 | t1         |            0 |               -1 | zone1 | 172.18.6.38 | leader   |              |
| obcp_t2     | 1106108697592657 | t1         |            0 |               -1 | zone2 | 172.18.6.34 | follower |              |
| obcp_t2     | 1106108697592657 | t1         |            0 |               -1 | zone3 | 172.18.6.6  | follower |              |
| obcp_t2     | 1106108697592660 | t3         |            0 | 1106658453357545 | zone1 | 172.18.6.38 | follower |              |
| obcp_t2     | 1106108697592660 | t3         |            0 | 1106658453357545 | zone2 | 172.18.6.34 | follower |              |
| obcp_t2     | 1106108697592660 | t3         |            0 | 1106658453357545 | zone3 | 172.18.6.6  | leader   |              |
| obcp_t2     | 1106108697592661 | t4         |            0 | 1106658453357545 | zone1 | 172.18.6.38 | follower |              |
| obcp_t2     | 1106108697592661 | t4         |            0 | 1106658453357545 | zone2 | 172.18.6.34 | follower |              |
| obcp_t2     | 1106108697592661 | t4         |            0 | 1106658453357545 | zone3 | 172.18.6.6  | leader   |              |
| obcp_t2     | 1106108697592660 | t3         |            1 | 1106658453357545 | zone1 | 172.18.6.38 | leader   |              |
| obcp_t2     | 1106108697592660 | t3         |            1 | 1106658453357545 | zone2 | 172.18.6.34 | follower |              |
| obcp_t2     | 1106108697592660 | t3         |            1 | 1106658453357545 | zone3 | 172.18.6.6  | follower |              |
| obcp_t2     | 1106108697592661 | t4         |            1 | 1106658453357545 | zone1 | 172.18.6.38 | leader   |              |
| obcp_t2     | 1106108697592661 | t4         |            1 | 1106658453357545 | zone2 | 172.18.6.34 | follower |              |
| obcp_t2     | 1106108697592661 | t4         |            1 | 1106658453357545 | zone3 | 172.18.6.6  | follower |              |
| obcp_t2     | 1106108697592660 | t3         |            2 | 1106658453357545 | zone1 | 172.18.6.38 | follower |              |
| obcp_t2     | 1106108697592660 | t3         |            2 | 1106658453357545 | zone2 | 172.18.6.34 | leader   |              |
| obcp_t2     | 1106108697592660 | t3         |            2 | 1106658453357545 | zone3 | 172.18.6.6  | follower |              |
| obcp_t2     | 1106108697592661 | t4         |            2 | 1106658453357545 | zone1 | 172.18.6.38 | follower |              |
| obcp_t2     | 1106108697592661 | t4         |            2 | 1106658453357545 | zone2 | 172.18.6.34 | leader   |              |
| obcp_t2     | 1106108697592661 | t4         |            2 | 1106658453357545 | zone3 | 172.18.6.6  | follower |              |
+-------------+------------------+------------+--------------+------------------+-------+-------------+----------+--------------+
21 rows in set (0.05 sec)
```

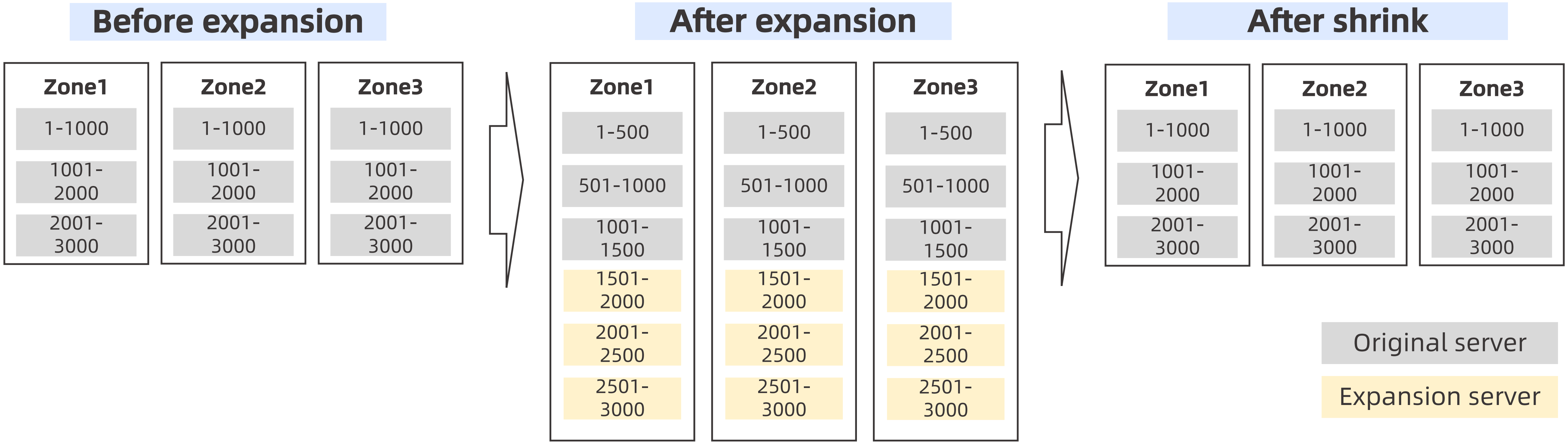
T3 and T4, primary replica of partition No. 0 is located on one server in Zone3

T3 and T4, primary replica of partition No. 2 is located on one server in Zone2

Chapter 4: OceanBase cluster technology architecture

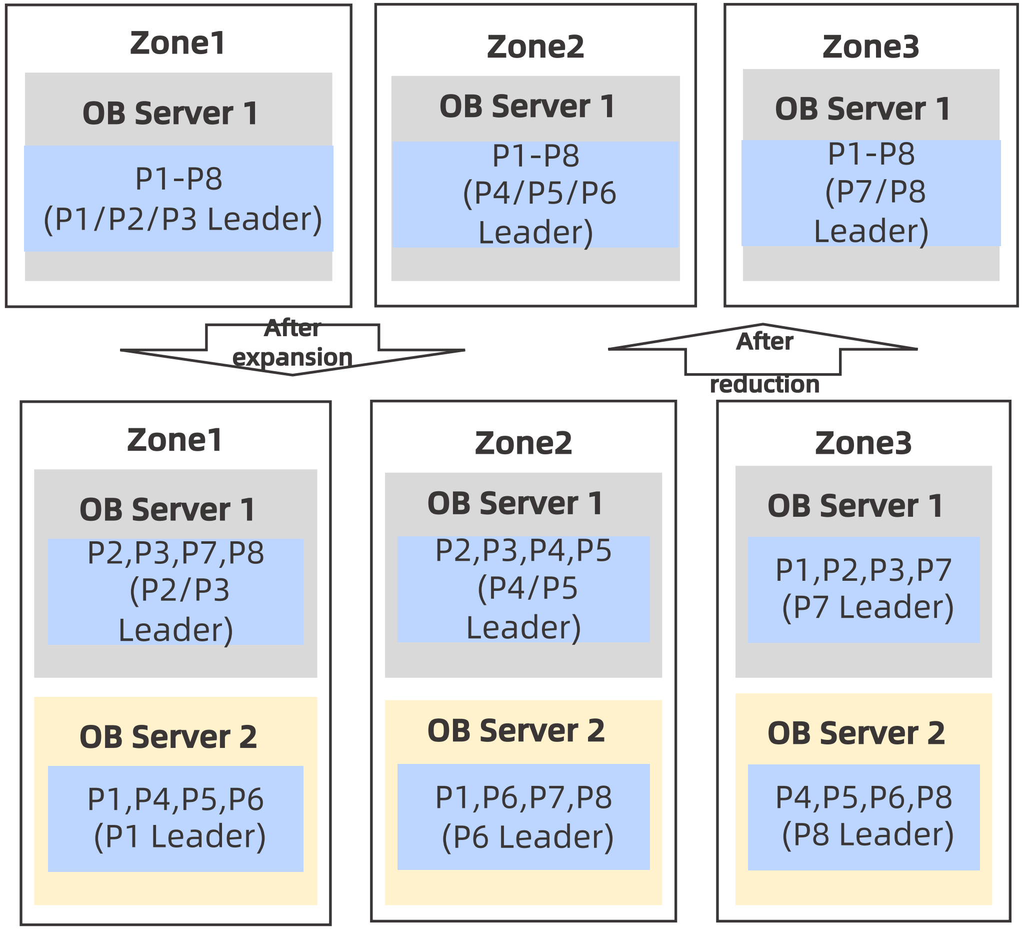
1. Paxos protocol and load balance
2. **Dynamic scale-out and scale-in**
3. Data reliability and availability
4. Distributed transaction, MVCC, transaction isolation level

Dynamic horizontal expansion scenario



- **This diagram shows an example of a three-zone cluster. Each Zone has 3000 replicas of partitions. After scale-out, OceanBase automatically move the 1500 partitions replicas (including the primary replica) to resource Unit on the new server**
- **OceanBase help businesses capacity expand and shrink linearly and deal with large-scale sales promotions and other activities at a lower cost**

Technical implementation of dynamic scale-out/in



Step1 **purchase resources**	Before sale promotion, prepare new servers and install the OceanBase software
Step2 **Catch up data**	After OB Server2 is added to the cluster, OB automatically selects replicas and dynamically copies them from OB Server1 to OB Server2 (existing data +redo LOGO incremental replication). This process has no impact to apps
Step3 **Switch service**	After data is balanced, the primary replica in OB Server1 stop service. The new primary replica in OB Server2 takes over services and deletes the old replica in OB Server1. This process has no impact on services
Step4 **shrink**	After peak hour is over, perform reverse operations to migrate the replica in OB Server2 back to OB Server1, and the apps accesses replica in OB Server1 again
Step5 **Return resource**	Release the resources of the purchased server

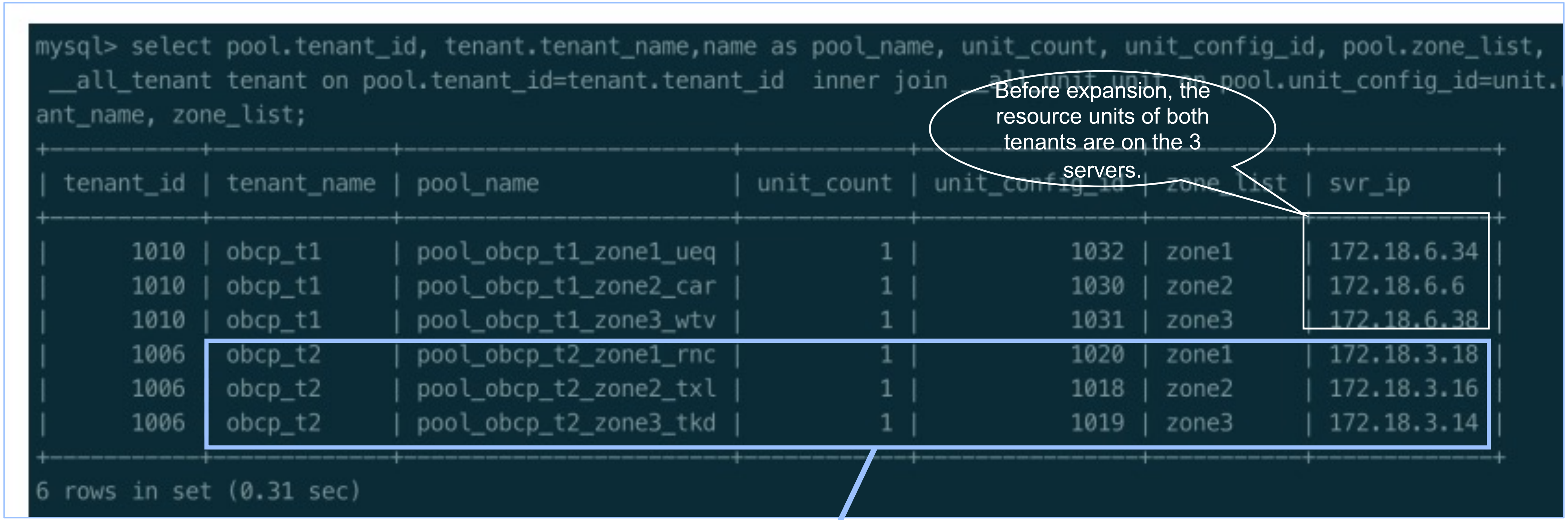
In the diagram, it assumes that a table has 8 partitions. After expansion, OceanBase automatically moves the copies of the four partitions (including the primary replica) to the tenant’s resource unit of the new Server in each Zone to achieve load balancing . The expansion process is dynamic and online, which is business un-awareness, reducing maintenance hassle

Basic step of expansion

1. Add new physical machines for each zone
2. On each newly added machine, start the Observer service
3. Run the ‘alter system add server …; ‘ , add new observer into cluster
4. Run ‘alter resource pool <pool name> unit_num = <bigger number>; ‘, command to increase the number of unit in the resource pool
5. OceanBase starts “rebalance” process automatically, and copy data from old unit to the new unit online. This process may take tens of minutes, depend on data size.
6. After completion of data duplication, OceanBase automatically switch the leader to new unit and delete the data on old partition.

Check __all_virtual_sys_task_status table to see whether there is outstanding task of comment as '%partition migration%‘, confirm data duplication is done

Cluster expansion

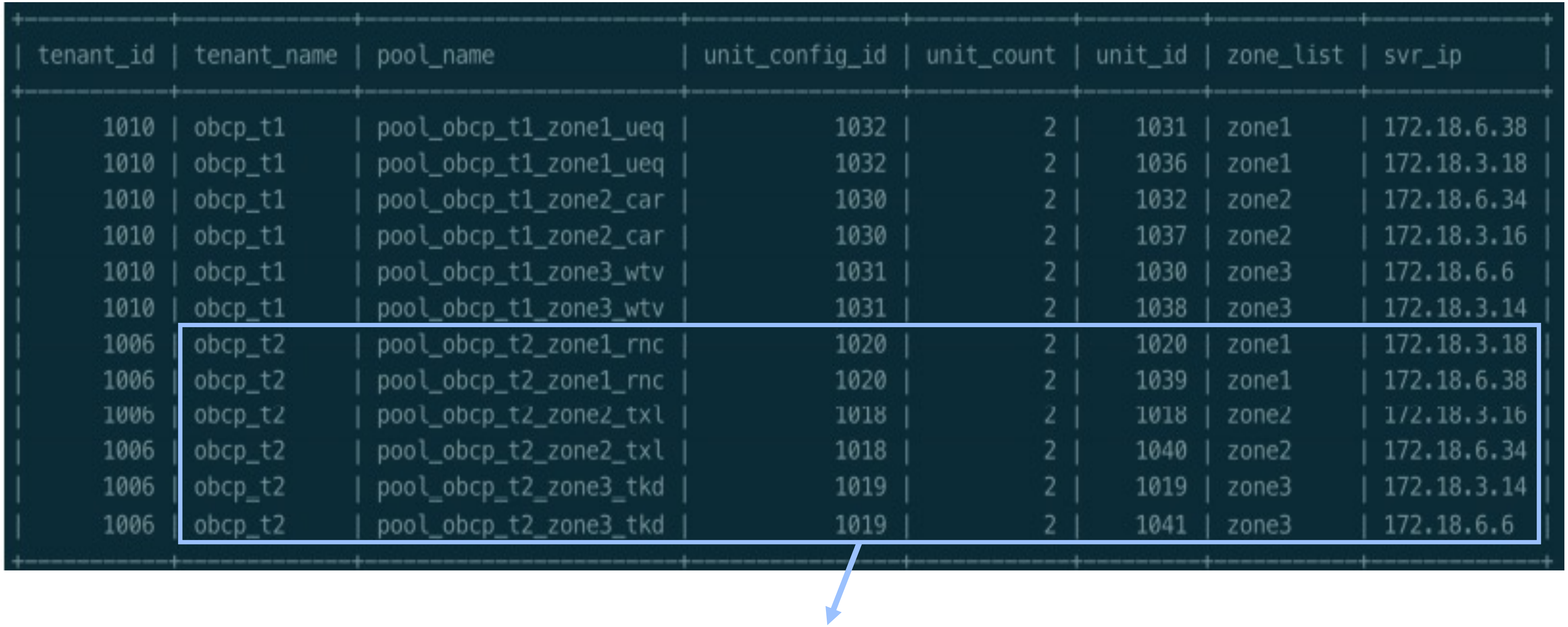


```
mysql> select pool.tenant_id, tenant.tenant_name,name as pool_name, unit_count, unit_config_id, pool.zone_list,
 __all_tenant tenant on pool.tenant_id=tenant.tenant_id  inner join  __all_unit unit on pool.unit_config_id=unit.
ant_name, zone_list;
+-----------+-------------+------------------------+------------+----------------+-----------+-------------+
| tenant_id | tenant_name | pool_name              | unit_count | unit_config_id | zone_list | svr_ip      |
+-----------+-------------+------------------------+------------+----------------+-----------+-------------+
|      1010 | obcp_t1     | pool_obcp_t1_zone1_ueq |          1 |           1032 | zone1     | 172.18.6.34 |
|      1010 | obcp_t1     | pool_obcp_t1_zone2_car |          1 |           1030 | zone2     | 172.18.6.6  |
|      1010 | obcp_t1     | pool_obcp_t1_zone3_wtv |          1 |           1031 | zone3     | 172.18.6.38 |
|      1006 | obcp_t2     | pool_obcp_t2_zone1_rnc |          1 |           1020 | zone1     | 172.18.3.18 |
|      1006 | obcp_t2     | pool_obcp_t2_zone2_txl |          1 |           1018 | zone2     | 172.18.3.16 |
|      1006 | obcp_t2     | pool_obcp_t2_zone3_tkd |          1 |           1019 | zone3     | 172.18.3.14 |
+-----------+-------------+------------------------+------------+----------------+-----------+-------------+
6 rows in set (0.31 sec)
```

Before expansion, the resource units of both tenants are on the 3 servers.

- After cluster expansion, the resource unit of tenant obcp_T2 is migrated to host (172.18.3.18/16/14).
- However, the number of tenant obcp_T1 and obcp_T2 resource units is still three, one for each zone

Tenant expansion



```
+-----------+-------------+------------------------+----------------+------------+---------+-----------+--------------+
| tenant_id | tenant_name | pool_name              | unit_config_id | unit_count | unit_id | zone_list | svr_ip       |
+-----------+-------------+------------------------+----------------+------------+---------+-----------+--------------+
|      1010 | obcp_t1     | pool_obcp_t1_zone1_ueq |           1032 |          2 |    1031 | zone1     | 172.18.6.38 |
|      1010 | obcp_t1     | pool_obcp_t1_zone1_ueq |           1032 |          2 |    1036 | zone1     | 172.18.3.18 |
|      1010 | obcp_t1     | pool_obcp_t1_zone2_car |           1030 |          2 |    1032 | zone2     | 172.18.6.34 |
|      1010 | obcp_t1     | pool_obcp_t1_zone2_car |           1030 |          2 |    1037 | zone2     | 172.18.3.16 |
|      1010 | obcp_t1     | pool_obcp_t1_zone3_wtv |           1031 |          2 |    1030 | zone3     | 172.18.6.6   |
|      1010 | obcp_t1     | pool_obcp_t1_zone3_wtv |           1031 |          2 |    1038 | zone3     | 172.18.3.14 |
|      1006 | obcp_t2     | pool_obcp_t2_zone1_rnc |           1020 |          2 |    1020 | zone1     | 172.18.3.18 |
|      1006 | obcp_t2     | pool_obcp_t2_zone1_rnc |           1020 |          2 |    1039 | zone1     | 172.18.6.38 |
|      1006 | obcp_t2     | pool_obcp_t2_zone2_txl |           1018 |          2 |    1018 | zone2     | 172.18.3.16 |
|      1006 | obcp_t2     | pool_obcp_t2_zone2_txl |           1018 |          2 |    1040 | zone2     | 172.18.6.34 |
|      1006 | obcp_t2     | pool_obcp_t2_zone3_tkd |           1019 |          2 |    1019 | zone3     | 172.18.3.14 |
|      1006 | obcp_t2     | pool_obcp_t2_zone3_tkd |           1019 |          2 |    1041 | zone3     | 172.18.6.6   |
+-----------+-------------+------------------------+----------------+------------+---------+-----------+--------------+
```

- After tenants obcp_T1 and obcp_T2 are scaled out, the number of resource unit for tenants increases from three to six
- 6 resource units well distributed to 6 servers

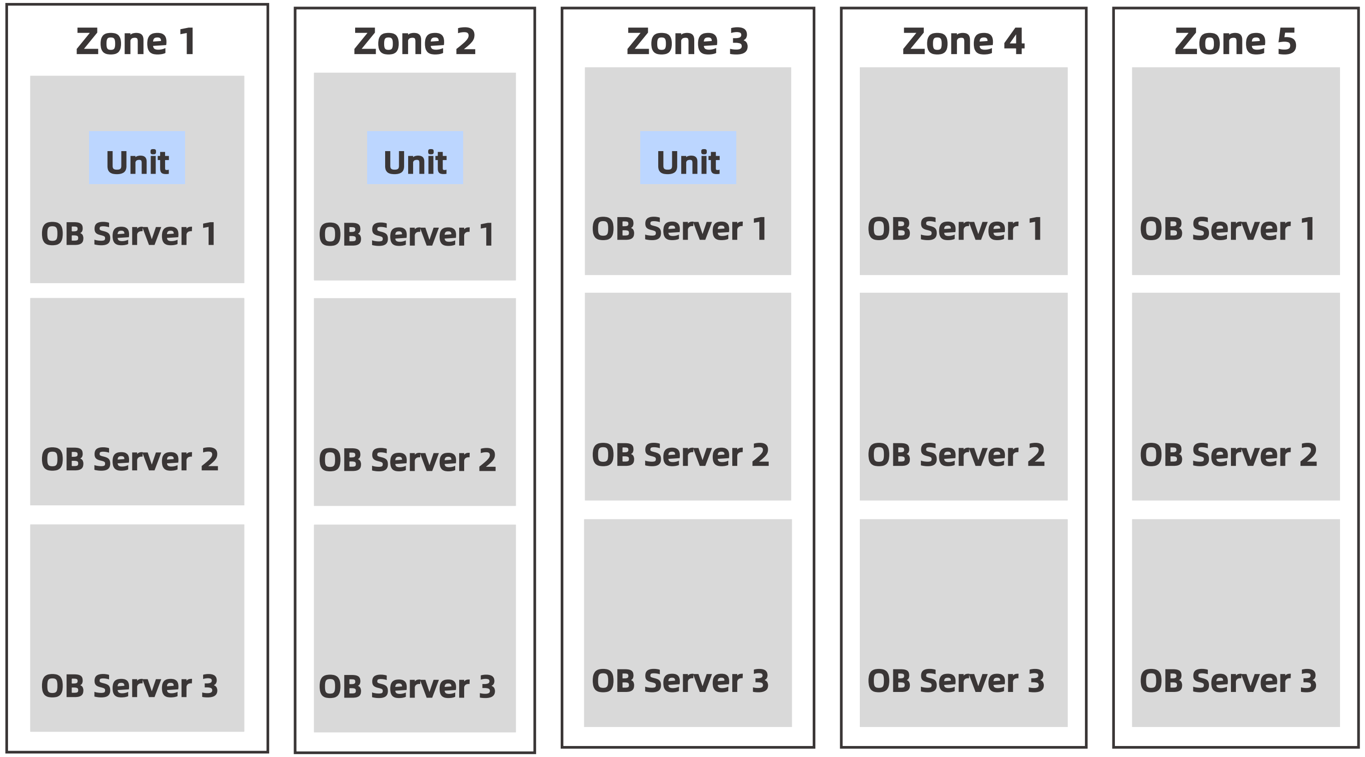
Tenant expansion – scale up by increasing resource pool specifications

Resources of cluster and tenants before expansion:

- 3-3-3-3-3 cluster, total 5 zones
- Tenant resource pool uses Zone1, Zone2, Zone3
- Unit Specs is 1C2G
- Number of resource units 1 (Unit Num=1)

Resources of cluster and tenants after expansion :

- Expand tenant resource pool specs from 1C2G to **2C8G**



Tenant expansion - scale up by increasing resource pool unit number

Resources of cluster and tenants before expansion:

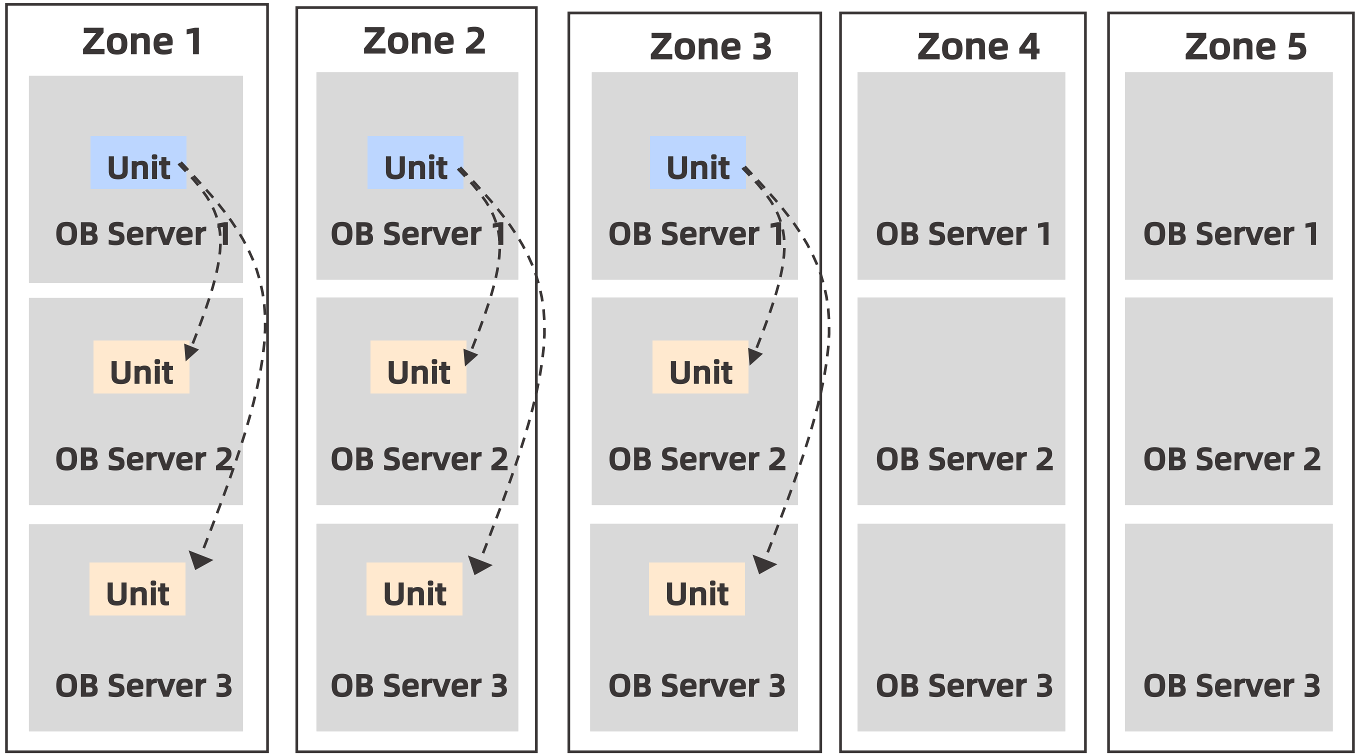
- 3-3-3-3-3 cluster, total 5 zones
- Tenant resource pool uses Zone1, Zone2, Zone3
- Unit specs is 1C2G
- Number of resource units 1

Resources of cluster and tenants after expansion:

- Unit Num of tenant resource pool changes from 1 to **3**

Load balance:

- Based on load balancing policies, data is automatically moved to the new Unit (within the Zone).
- After the data is balanced, the new Unit elects a Leader, balancing workload



Tenant expansion – increase fault tolerance level from 3 replicas to 5

Resources of cluster and tenants before expansion:

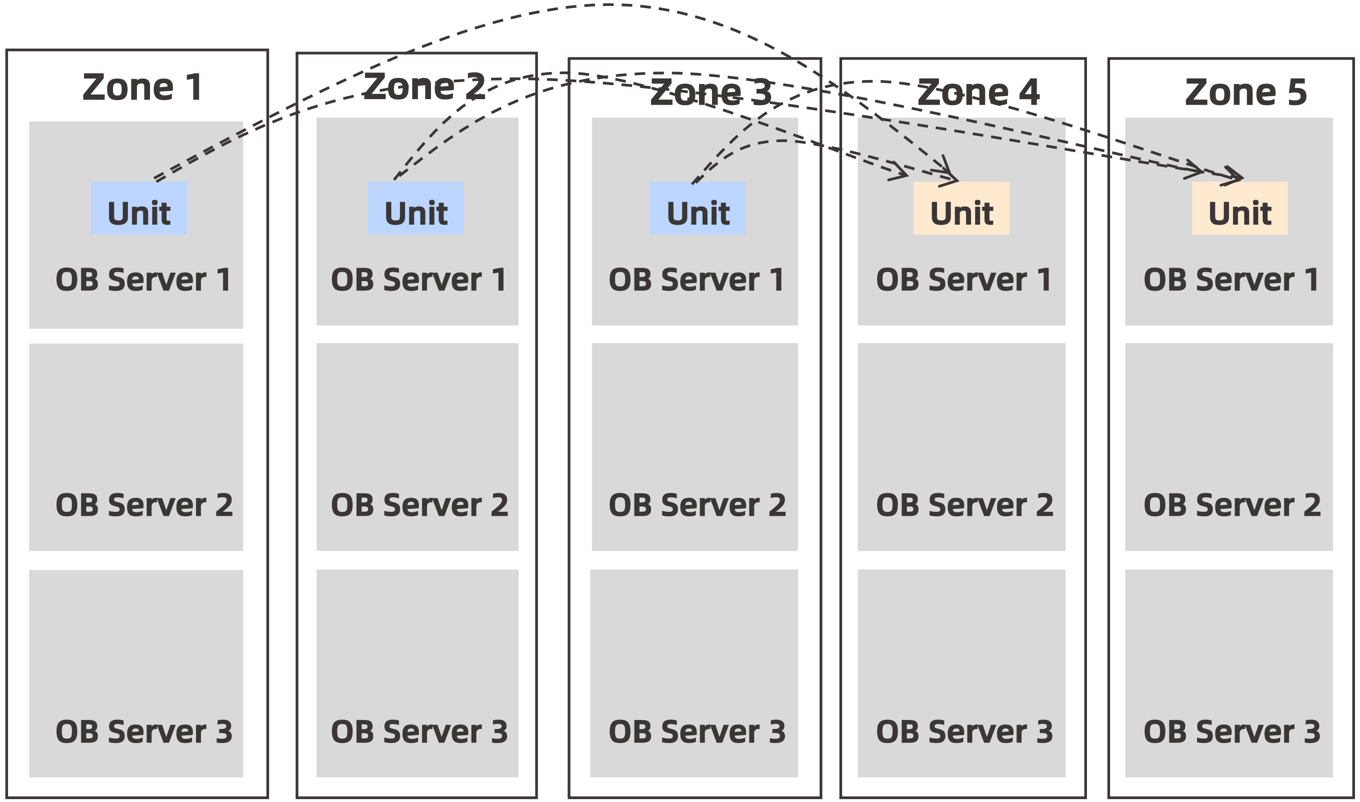
- 3-3-3-3-3 cluster, total 5 zones
- Tenant resource pool uses Zone1, Zone2, Zone3
- Specification is 1C2G
- Number of resource units 1

Resources of cluster and tenants after expansion:

- Tenant resource pool changes 3 zones to **5** Zones

Load balance

- Copy data from primary replicas within the original 3 zones (Paxos group members increased from 3 to 5)
- After the data is balanced, the new Unit elects a Leader to balance business workload (depending on the Primary Zone configuration, etc.)



Basic steps of scale down

1. Run 'the alter resource pool <pool name> unit_num = <smaller number>; ' , command to reduce the number of units in the resource pool
2. OceanBase automatically starts the rebalance process to copy data from the unit on to-be offline machine to the unit on other machines in the zone
3. After data is copied, OceanBase automatically switches services to the new Unit and deletes data from the unit on to-be offline machine
4. Execute 'ALTER System DELETE Server...;',; Command to complete machine offline operation
5. Manually terminate the Observer process on the offline machine and perform OS shutdown operation

Check __all_virtual_sys_task_status table to see whether there is outstanding task of comment as '%partition migration%' task, confirm data duplication is done

Chapter 4: OceanBase cluster technology architecture

1. Paxos protocol and load balance
2. Dynamic scale-out and scale-in
3. **Data reliability and availability**
4. Distributed transaction, MVCC, transaction isolation level

DR SLA Level

During system failure (such as power failure, downtime, or process abnormally killed), how apps observer the HA capability of the system

- **RTO (Recovery Time Objective)**: The maximum tolerable amount of time that a database stops working after a failure or disaster, which is the maximum tolerable time window which data must be recovered
- **RPO (Recovery Point Object):** This is a point in time in the past to which data can be recovered in the event of a disaster or emergency, it is actually the amount of data loss that a business can tolerate

RTO = service availability

RPO = data reliability

Disaster recovery capability level	RTO	RPO
1	Above 2 days	1~7 days
2	Above 24 hours	1~7 days
3	Above 12 hours	Hours to 1 day
4	Hours to 2 days	Hours to 1 day
5	Minutes to 2 days	0~30 minutes
6	**Minutes**	**0**

OceanBase RPO=0 RTO<30s, means that when a minority no. of servers fail, OceanBase can restore service within 30 seconds without any data loss.

OceanBase provides high availability based on generic PC server

Traditional database

- Traditional database runs on "luxurious" infrastructure, which lays a good foundation for ensuring high availability of database
- Expensive hardware

High-end servers, minicomputers

Dedicated storage, using RAID and other technologies

VS

OceanBase

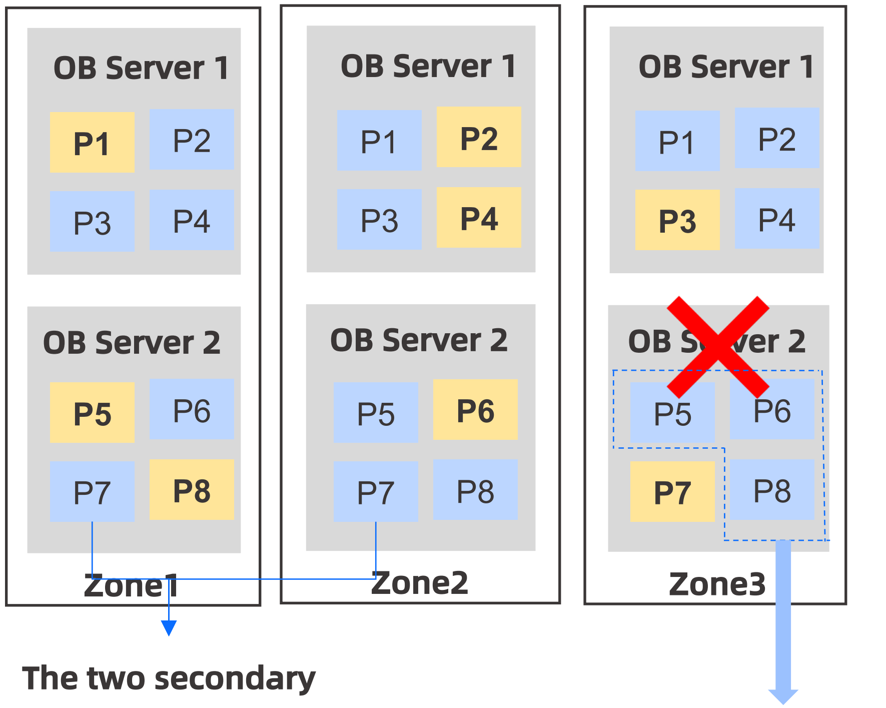
- OceanBase runs on less reliable PC servers and relies on its own software to ensure high availability
- Low hardware cost

A universal server that uses its own hard disks and does not require external storage or Raid

OceanBase relies on its software capabilities to provide higher availability under hardware's un-stability

OCEANBASE

Automatic service takeover in case of minority failure, ensuring high availability



The two secondary replicas negotiate and re-elect a new P7 primary replica and take over the apps operation

P5/P6/P8 are all secondary replicas. The remaining two replicas still form up the majority members and do not affect the cluster

Example: Leader failure

- Assume that Zone3-OB Server2 fails, and the P7 Leader primary replica cannot provide services
- The P7 follower in Zone1 and P7 follower in Zone2 will re-elect a new Leader and take over the service
- The entire process is completed automatically without manual intervention
- After zone3-OB Server2 recovers, the P7 replica first catch-up the data. After the data is equalized, the P7 replica continues to join the Paxos group. Generally, the P7 replica revert to primary replica (to achieve load balancing).

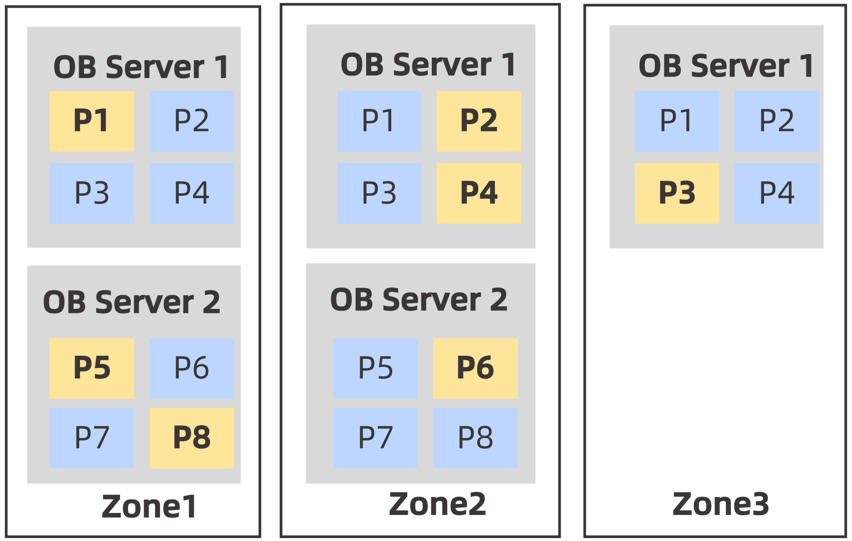
Example : Follower failure

- Assume that Zone3-OB Server2 fails and the secondary replicas (P5,P6,P8) do not accept app request initially. The remaining two replicas still meet the majority group condition and can still provide services normally without affecting apps
- After hardware server contains secondary replica recovers, the secondary replica catch up the data, re-join Paxos group, and the original ‘follower’ state is intact.

Handling policy in case of OB Server process abnormity

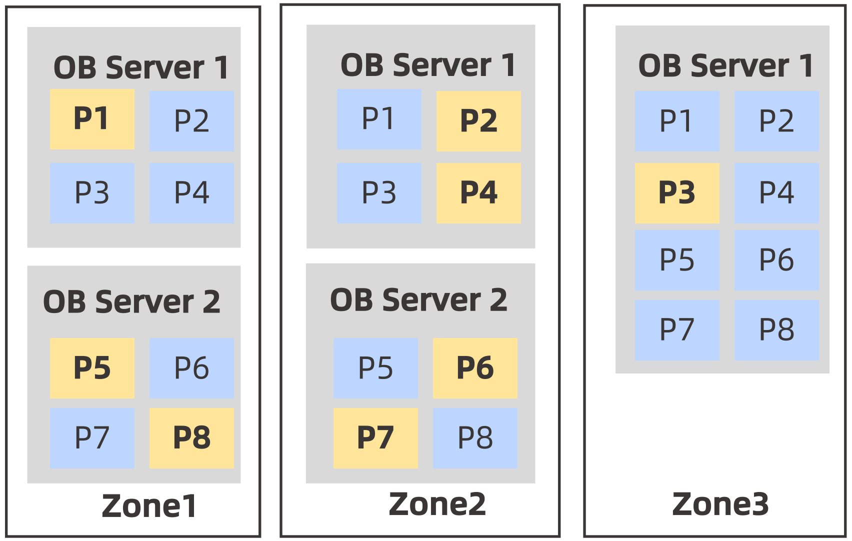
If the OB Server process terminates abnormally, parameter server_permanent_offline_time is used to determine whether to perform "temporary offline" operation.

Duration of Observer abnormal termination < server_permanent_offline_time



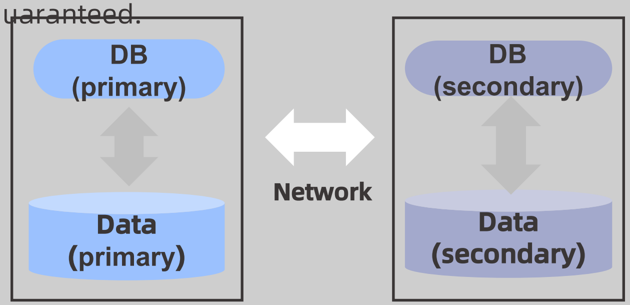
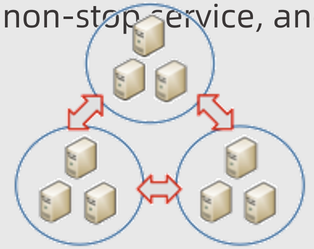
- Because the process termination takes not long, the abnormal process maybe recovered quickly. Therefore, OceanBase ignores the event to avoid frequent data migration
- At this time, P5-P8 only has two replicas. Although the majority group is still satisfied, RPO=0 can be guaranteed, but there are a risk (if there is 2nd server failure).

Duration of Observer abnormal termination > server_permanent_offline_time

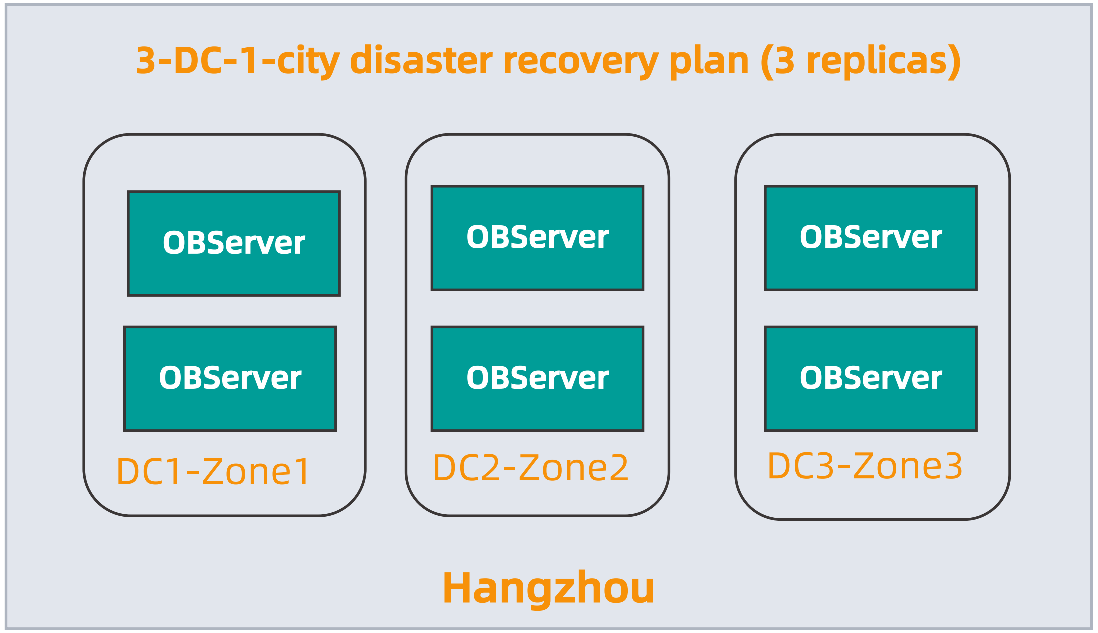


- OceanBase takes machines offline temporarily and copies the missing data from the primary replica of another zone to the remaining machines in the zone (with sufficient resources) to maintain the number of replicas
- After the OBServer process is recovered, it is automatically re-join to the cluster. If "temporary offline operation" has been done, then data in Unit need to be migrated from another machine of current zone or from another zone

HA solution comparison between traditional database and OceanBase

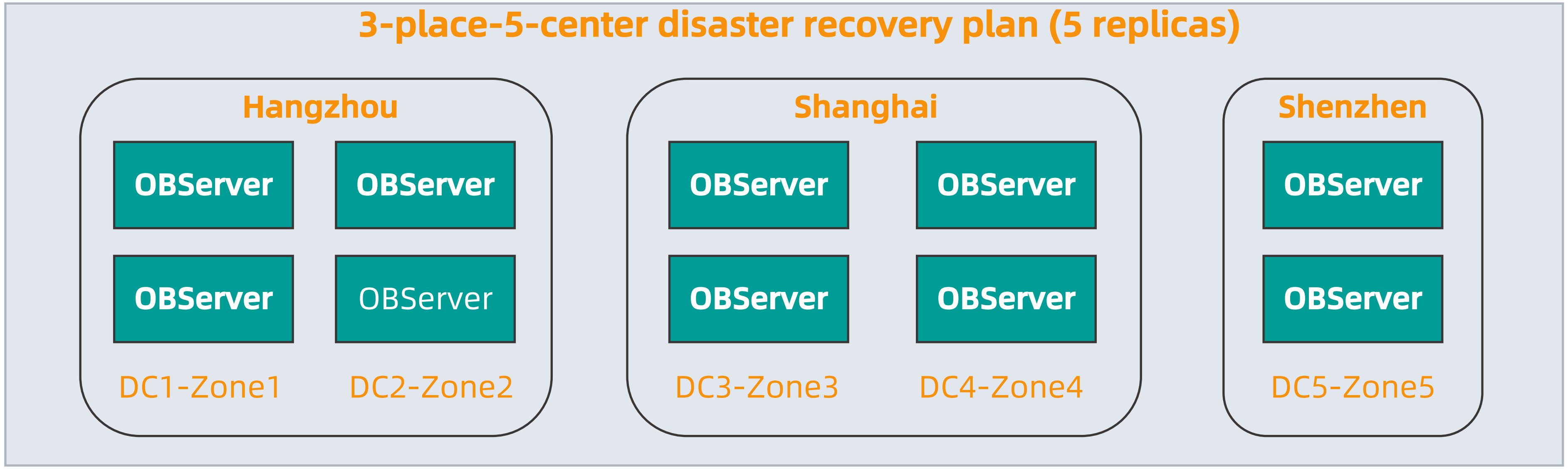
	Technical implementation	Standby server failure criteria	Primary server failure criteria
Traditional database	Primary + secondary databases, Redo log is used between them for synchronization. Generally, "maximum available mode" is used and RPO=0 is not guaranteed.  DB (primary) Data (primary) Network DB (secondary) Data (secondary)	**Normally, it is in maximum protection mode. If the standby server is faulty or the network is faulty, the primary redo-log cannot be synchronized to the standby server. As a result, applications cannot be executed properly and users cannot accept this mode. In this case, the system switches to the maximum performance mode, that is, the redo-log mode is asynchronous, but RPO>0;**	**Once the master fails, the master cannot automatically switch to the master (to avoid the brain split between the two masters). It takes at least half an hour to manually change the master to the master. RTO>30 minutes;**
OceanBase	Typical three-replica deployment based on Paxos: strong data consistency, continuous availability, automatic active/standby switchover, non-stop service, and no data loss. 	**If the machine contains minority secondary replica is failed, the primary replica cannot receive the response from the secondary replica, but can receive the response from other standby replicas. In this way, the majority is consistent and services are not affected. RPO=0;**	**If the machine contains primary replica is failed, the remaining two replicas can automatically elect a new leader. The new leader automatically takes over services without manual intervention. RTO<30 seconds;**

OceanBase DR topology: 3 DCs in 1 city



- Three data centers in the same city form a cluster (each DC holds a Zone). The inter-DC network delay ranges from 0.5 ms to 2ms
- In the event of a DC disaster, the remaining two replicas are still in the majority group and can still synchronize redo-log logs, ensuring RPO=0
- Do not faulty-tolerance for city-level disaster

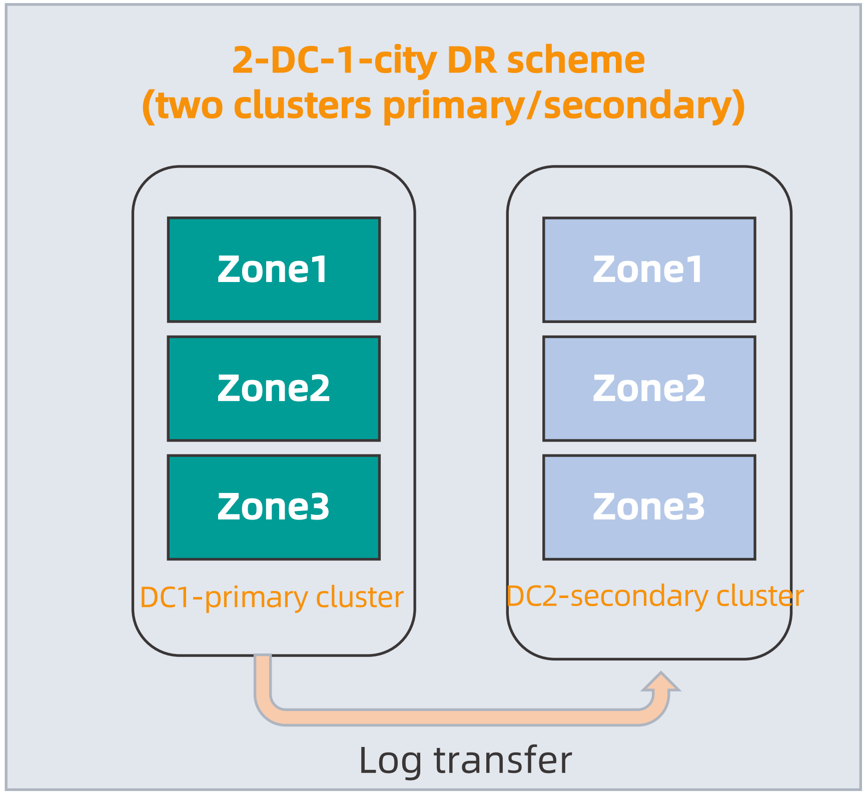
OceanBase DR topology: 3 Cities, 5 DCs & 5 Replicas



- Three cities form a cluster of 5 replicas
- The failure of any DC or city still constitutes a majority group, ensuring RPO=0
- Since more than three replicas are required to form a majority, but each city has a maximum of two replicas, 1st cities and 2nd should be close to each other to reduce the delay of redo sync
- To reduce costs, 3rd City can deploy only log-only replica

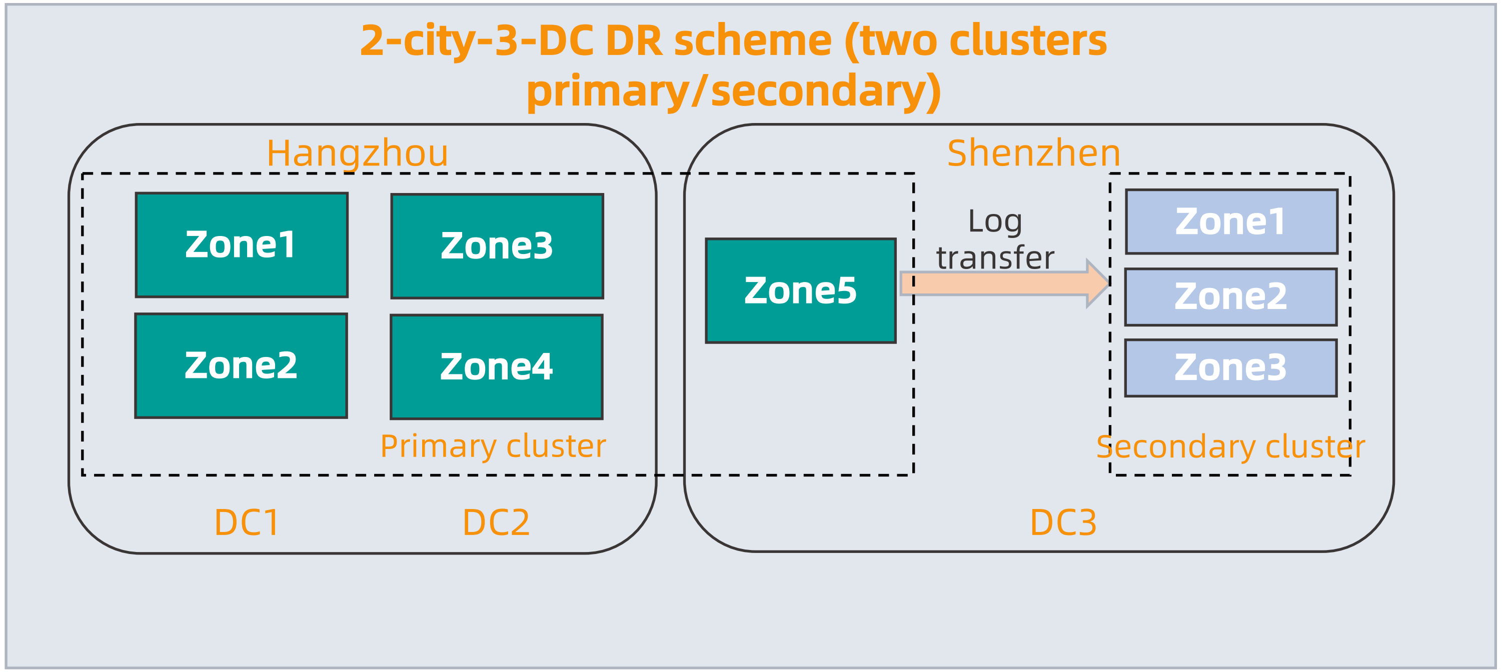
OceanBase DR topology: hot-standby scheme in 2 DCs of 1 city

The 3-DC-1-city or 3-city-5-DC scheme require expensive infrastructure. To leverage existing enterprise network infrastructure, OceanBase provides two solutions: 2-DC-1-city, and 2-city-3-DC



- Each city deploys an OceanBase cluster, with one primary cluster and one secondary cluster. Each cluster has its own individual Paxos group with multi-replica consistency
- Data is synchronized between clusters via redo-log, which is similar to the primary/standby replication mode of traditional database. There are two data synchronization modes: Asynchronous and Strong sync, similar to the Maximum performance and Maximum protection mode in ODG

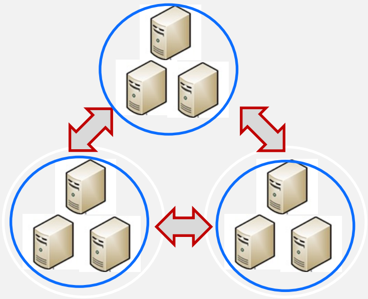
OceanBase DR topology: hot-standby scheme 3 DC 2 cities



- The primary and secondary cities form a 5-replica cluster. If any DC fails, max. up to 2 replicas can be lost, and the remaining 3 replicas will still satisfy the majority group
- The secondary city creates an independent 3-replica cluster, data is sync via async or strong sync mode
- In the event of a disaster in the main city, the secondary city can take over operations

Summary: Industrial implementation of Paxos ensures data reliability and availability

Strict Paxos protocol



- Multi-replica (ZONE) consistency protocol, one primary (leader) multiple secondary (follower)
- “majority” (>1/2) strong consistency

Special benefits

- Majority data consistency, tolerating any minority failure
- When the leader fails, services are automatically switched over without manual intervention
- It can flexibly respond to single-machine failures, DC disasters, and city-level disasters to achieve overall disaster recovery

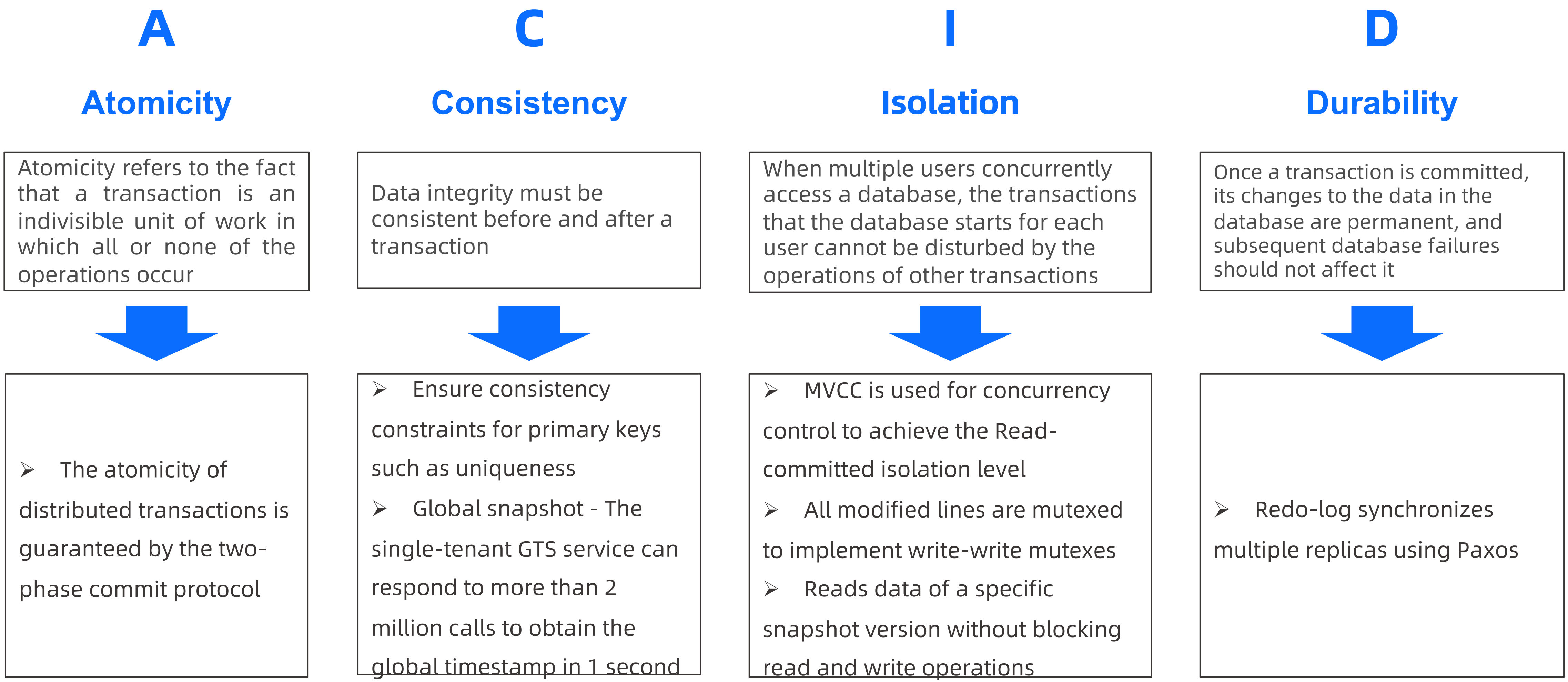
Technology value

- Any minority failure guarantees RPO = 0; service not downgraded during high workload stress test
- The leader failure automatically recovers within RTO < 30 seconds
- RTO and RPO are significantly better than traditional primary and standby database
- It has reached international DR SLA level 6

Chapter 4: OceanBase cluster technology architecture

1. Paxos protocol and load balance
2. Dynamic expansion and reduction
3. Data reliability and availability
4. **Distributed transaction, MVCC, transaction isolation level**

OceanBase guarantees ACID through a number of mechanisms for distributed transactions executed across machines



A

Atomicity

Atomicity refers to the fact that a transaction is an indivisible unit of work in which all or none of the operations occur

- The atomicity of distributed transactions is guaranteed by the two-phase commit protocol

C

Consistency

Data integrity must be consistent before and after a transaction

- Ensure consistency constraints for primary keys such as uniqueness
- Global snapshot - The single-tenant GTS service can respond to more than 2 million calls to obtain the global timestamp in 1 second

I

Isolation

When multiple users concurrently access a database, the transactions that the database starts for each user cannot be disturbed by the operations of other transactions

- MVCC is used for concurrency control to achieve the Read-committed isolation level
- All modified lines are mutexed to implement write-write mutexes
- Reads data of a specific snapshot version without blocking read and write operations

D

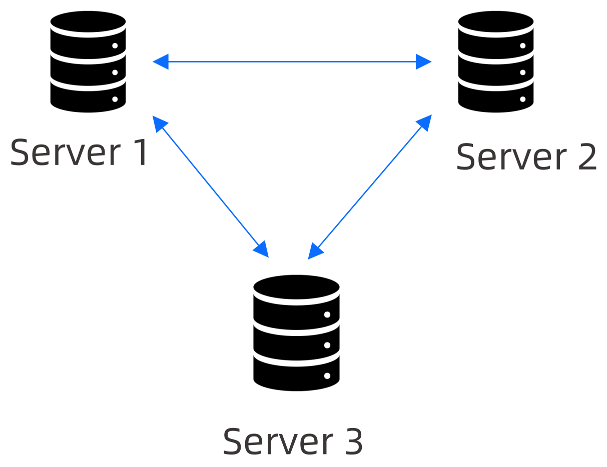
Durability

Once a transaction is committed, its changes to the data in the database are permanent, and subsequent database failures should not affect it

- Redo-log synchronizes multiple replicas using Paxos

Big challenge to distributed database ACID : Server Clock Discrepancy

Discrepancy of physical clock of machines



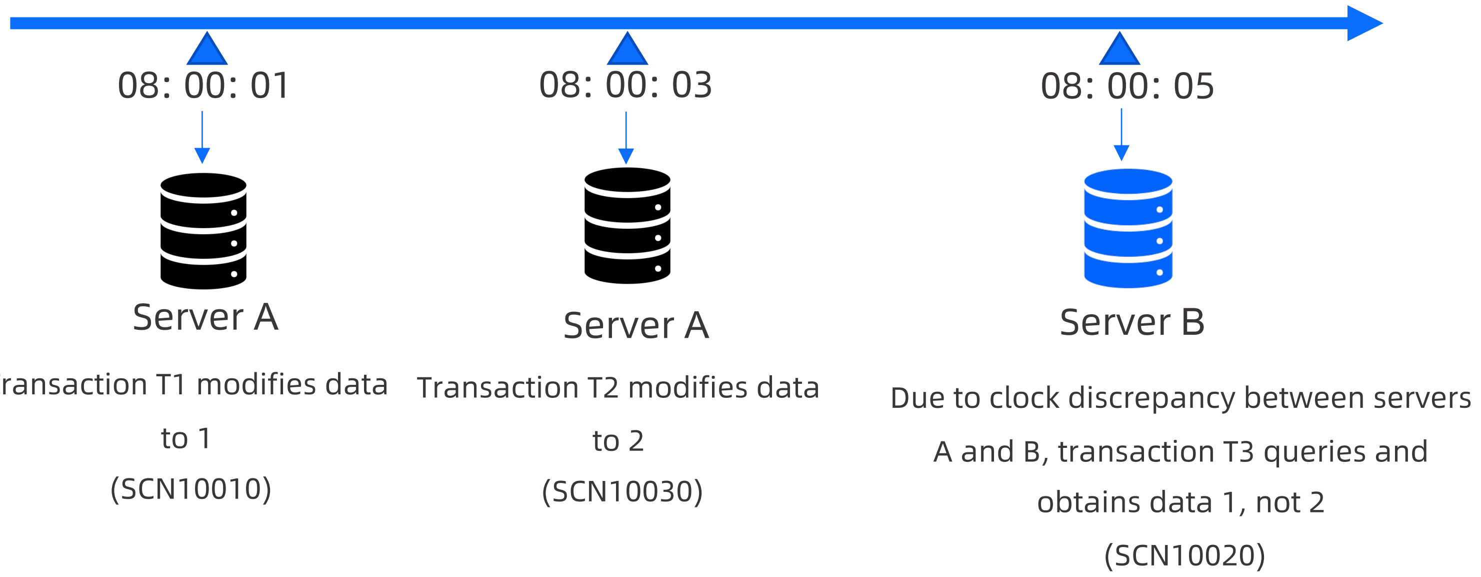
Server 1

Server 2

Server 3

Physical clock discrepancy between the servers

Technical challenges



08: 00: 01

Server A

Transaction T1 modifies data to 1

(SCN10010)

08: 00: 03

Server A

Transaction T2 modifies data to 2

(SCN10030)

08: 00: 05

Server B

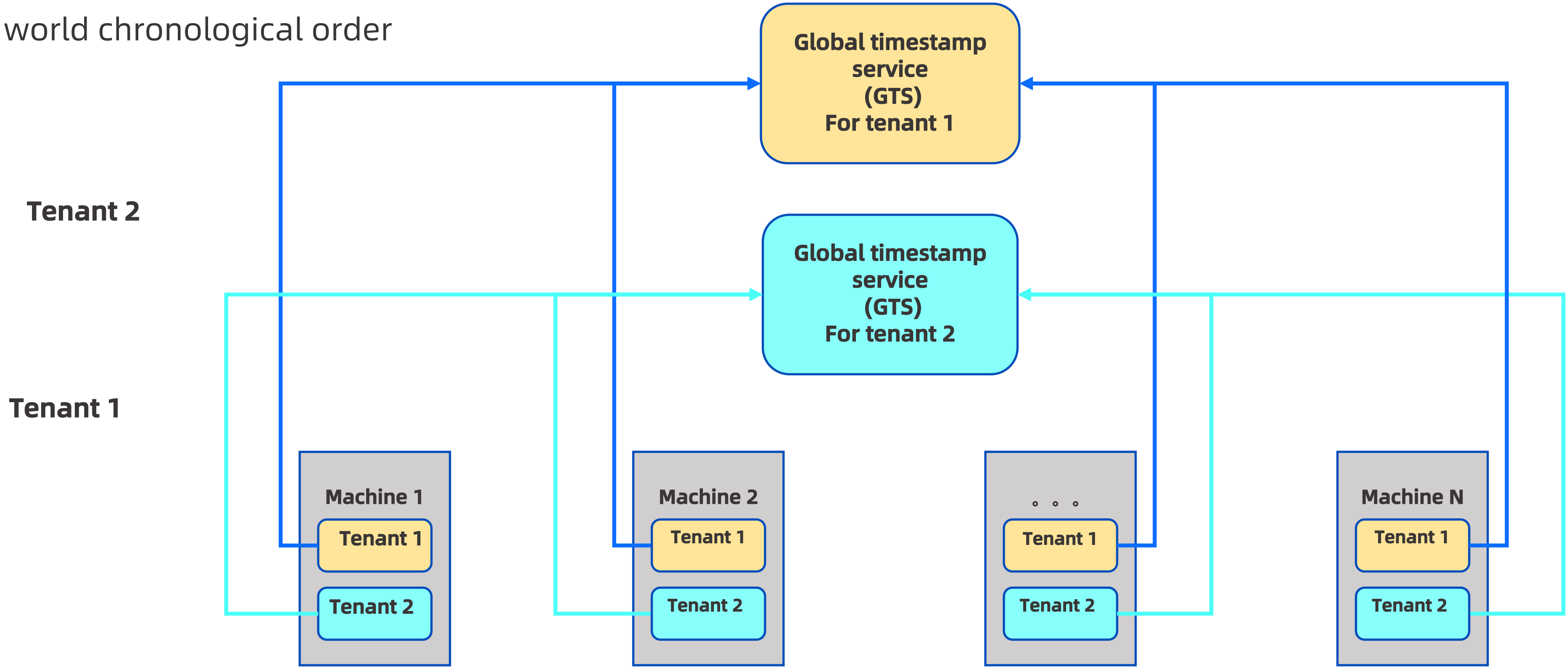
Due to clock discrepancy between servers A and B, transaction T3 queries and obtains data 1, not 2

(SCN10020)

Unlike the traditional shared-everything database architecture, OceanBase is a native distributed architecture with a shared-nothing architecture. There are technical challenges associated with distributed architecture when it comes to achieving globally (across machines) consistent snapshot isolation and multi-version-concurrency-control (MVCC)

OceanBase global consistency snapshot technology

OceanBase database utilizes a centralized service to provide globally consistent version numbers. Transactions that modify or query data, regardless of the physical machine from which the request originated, get a version number from this centralized service, ensuring that all version number are monotonically forward and consistent with real-world chronological order



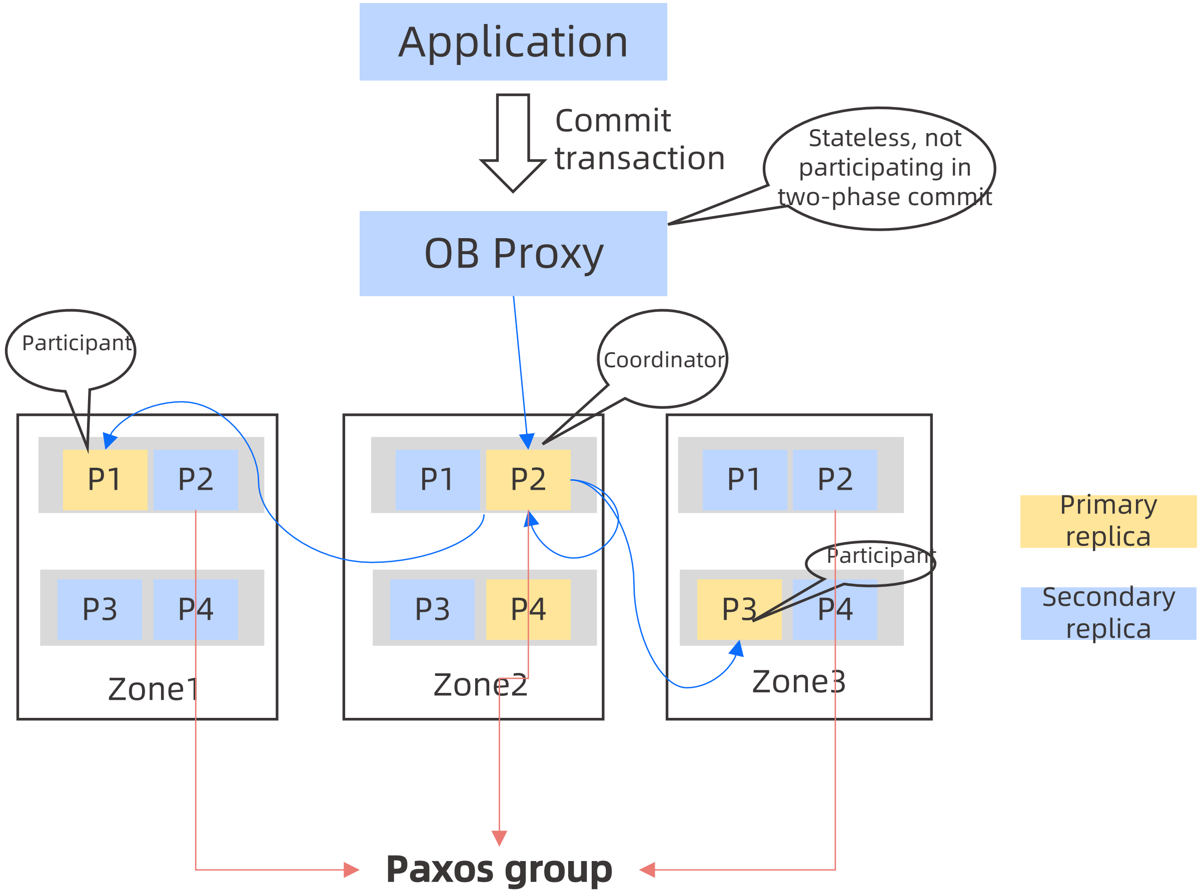
Innovative two-phase commit ensure atomicity of transaction on any failure

Technical challenges

- ✓ Rely on the two-phase commit protocol to ensure atomicity of transactions
- ✓ Two-phase commit: Multiple states, complex, and machines get stuck

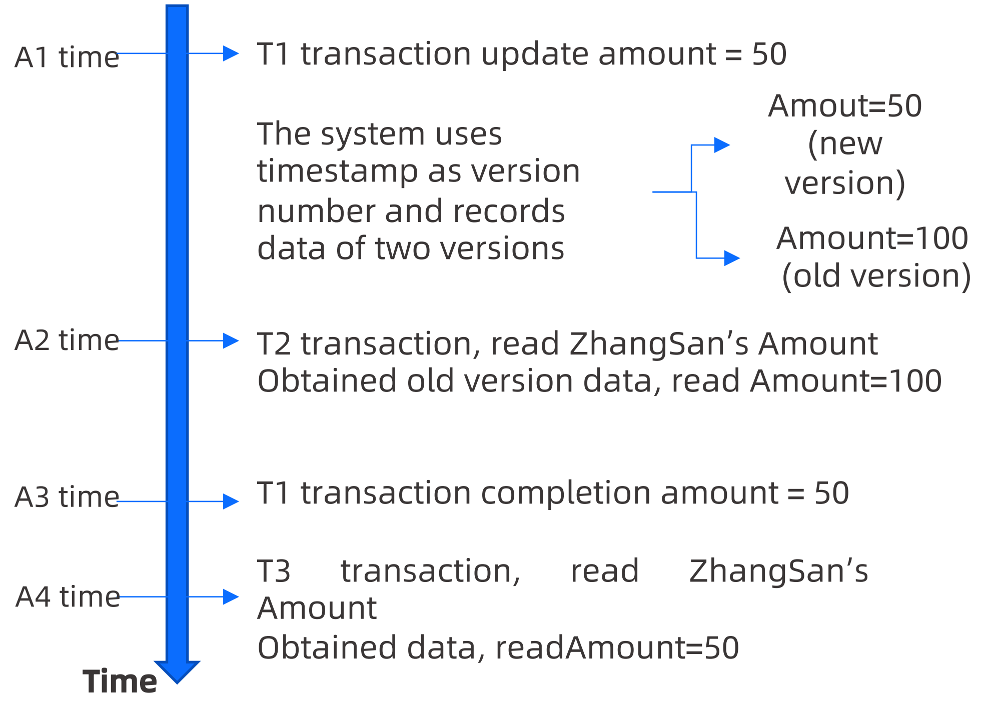
Features of OceanBase two-phase commit protocol

- ✓ The transaction coordinator and all participants are highly available
- ✓ For a single-node multi-partition transaction, if all participants prepare successfully, the transaction is considered to be in the commit state and the client immediately returns to commit
- ✓ Automatic handling of abnormal situations



Multi-version concurrency control (MVCC) to eliminate read/write conflict

Name	Amount
ZhangSan	100



A1 time → T1 transaction update amount = 50

The system uses timestamp as version number and records data of two versions
- Amout=50 (new version)
- Amount=100 (old version)

A2 time → T2 transaction, read ZhangSan's Amount
Obtained old version data, read Amount=100

A3 time → T1 transaction completion amount = 50

A4 time → T3 transaction, read ZhangSan's Amount
Obtained data, readAmount=50

Through MVCC, T1 write transaction and T2 read transaction are executed in parallel

MVCC core functions

Use data lock mechanism to control read/write conflicts, other transactions cannot read after row is locked, resulting in read/write competition and affecting read concurrency. MVCC can effectively solve this problem:

- Global unified data version number management from globally unique timestamp Service (GTS)
- Read and write operations must obtain the version number from GTS. One tenant has only one GTS service to maintain global (cross-machine) consistency
- Modify data: Before a transaction is committed, the old and new versions of the data coexist but have different version numbers
- Read data: Get the version number and then look for committed data that is less than or equal to the current version number
- Write operations obtain row locks, but read operations do not need locks, effectively avoiding read/write lock competition and improving read/write concurrency

Transaction isolation level

Isolation level that ensures global data consistency

Transaction concurrency problem

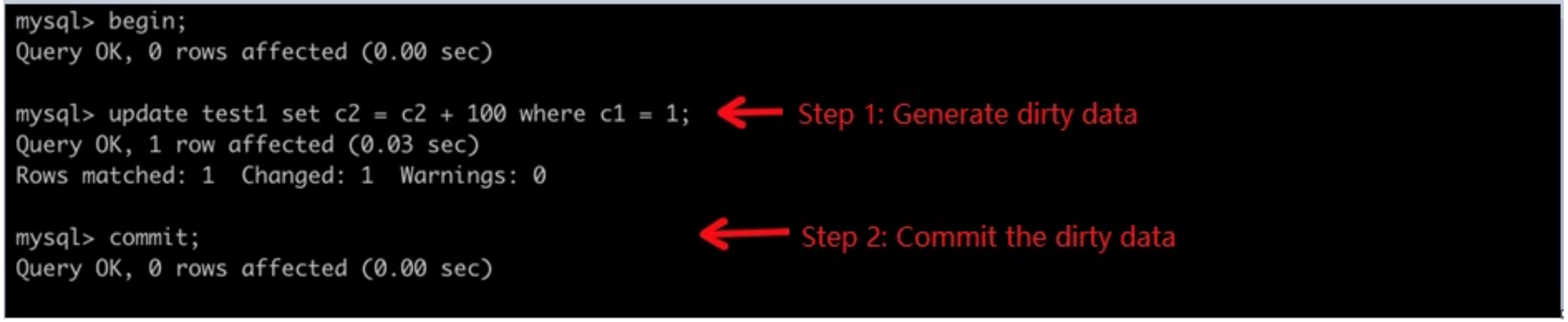
- Dirty read: Uncommitted data is read
- Non-repeatable read: the same batch of data may be read at different times in the same transaction (the data may be updated by other transactions in the meantime)
- Phantom read: When two queries are executed within the same transaction, and the second query results either contain data not present in the first query or are missing data present in the first query (data has been inserted or deleted by another transaction)

OceanBase supports multiple transaction isolation levels

- Based on globally consistent data version number management, different version number policies are used to achieve different isolation levels
- Oracle mode supports the following two isolation levels, and applications can use either level based on their requirements :
  - Read-Committed: Avoid dirty read, with unrepeatable and phantom read (default)
  - Serializable: Avoid dirty read, non-repeatable read, and phantom read
- MySQL mode supports read committed and repeatable isolation levels
- Dirty reads are not supported. Only committed data can be retrieved

MVCC examples

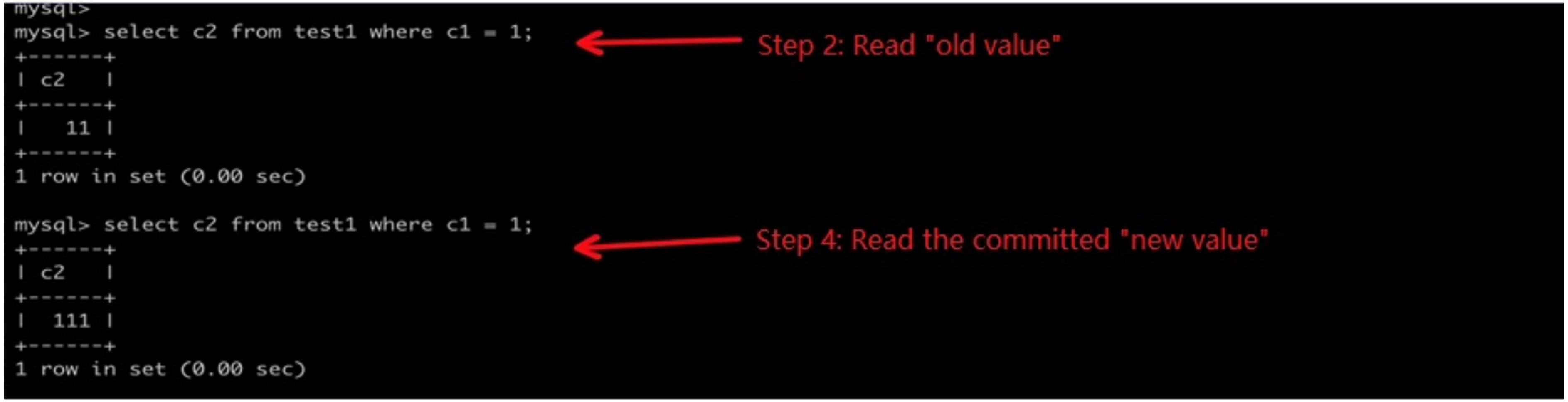
MVCC examples (results demonstration)



```
mysql> begin;
Query OK, 0 rows affected (0.00 sec)

mysql> update test1 set c2 = c2 + 100 where c1 = 1;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> commit;
Query OK, 0 rows affected (0.00 sec)
```

Step 1: Generate dirty data

Step 2: Commit the dirty data



```
mysql>
mysql> select c2 from test1 where c1 = 1;
+-------+
| c2    |
+-------+
|    11 |
+-------+
1 row in set (0.00 sec)

mysql> select c2 from test1 where c1 = 1;
+-------+
| c2    |
+-------+
|   111 |
+-------+
1 row in set (0.00 sec)
```

Step 2: Read "old value"

Step 4: Read the committed "new value"

Summary

This section describes how OB servers in the OceanBase cluster work together to achieve load balancing, high availability, and distributed transactions:

- Multiple copies of each partition form a Paxos group. Generally, the primary copy provides read and write services for services and synchronizes redo-log logs between the primary and secondary copies to ensure data consistency. The master copy does not have to wait for all redo-log logs from slave copies to fall, as long as a majority of them are satisfied, which provides better performance
- Generally, replicas and primary replicas are evenly distributed to each server in a Zone (consistent with the tenant resource pool) to implement automatic load balancing and prevent uneven work on each server
- Minority failure, the majority will automatically select a new master copy, to ensure no impact on business
- OB Proxy is a "stateless" service process that does not persist data and has no requirements for deployment location
- OceanBase can provide high availability with RPO=0 and RTO<30 seconds, which means that when a minority fails, OceanBase can restore services within 30 seconds without losing any data
- OceanBase provides three copies for three rooms in the same city and five copies for five centers in three places. In order to benefit the existing enterprises, OceanBase also provides the traditional two rooms in the same city and two places and three centers

Q&A

OCEANBASE

Simulation Questions (1)

1.[True/false] A partitioned replica contains only static data (SS Table) on the hard disk, no MemTable data and log data. ()

2.[True/false] The primary replica can only be scattered to all zones, but cannot be focused to one Zone.()

3.[True/false] Each OB Server is relatively independent and has its own independent SQL engine. If the data required by the application is not on the current OB Server, the OB Server will coordinate the data of other OB servers and uniformly feed back to the application, which is transparent to the application. ()

4.[True/false] The primary replica synchronizes redo-log logs to ensure reliability. The primary replica can respond to applications only after receiving messages indicating that all secondary replicas are successfully removed from disks. ()

5.[True/false] If an enterprise has two equipment rooms in a city, deploy two zones in one equipment room and the other zone in the other to provide room-level disaster recovery. ()

6. [Single choice] OceanBase fors the Paxos protocol group in unit of () .

A: tenant B: database C: table D: partition

7. [Single choice] Which of the following is correct about the expansion/reduction of OceanBase? ()

A: The administrator needs to stop the service. B: The service needs to be modified.

C: The system supports dynamic capacity expansion and reduction and has no service awareness

Simulation Questions (2)

8. [Single choice] OceanBase uses the two-phase commit protocol to ensure atomicity of transactions. In the two-phase commit protocol, who is the coordinator? ()

A: OB Proxy B: OB Server C: RootService general control service D: OCP cloud management platform

9. [Multiple choice] Which transaction isolation levels does the Oracle tenant of OceanBase support? ()

A: Dirty read B: Read-Committed C: Serializable

10. [Single choice] Which technology does OceanBase use to solve the problem of read/write exclusivity?()

A: MVCC B: Paxos protocol C: Global snapshot D: Exclusive lock

11. [Multiple choice] Which of the following is correct about the OB Proxy? ()

A: The OB Proxy is located between the application and the OB Server and routes the application requests to the appropriate OB Server.

B: OB Proxy must be deployed on an independent server to ensure its performance.

C: OB Proxy participates in the calculation tasks and transaction processing of the database engine.

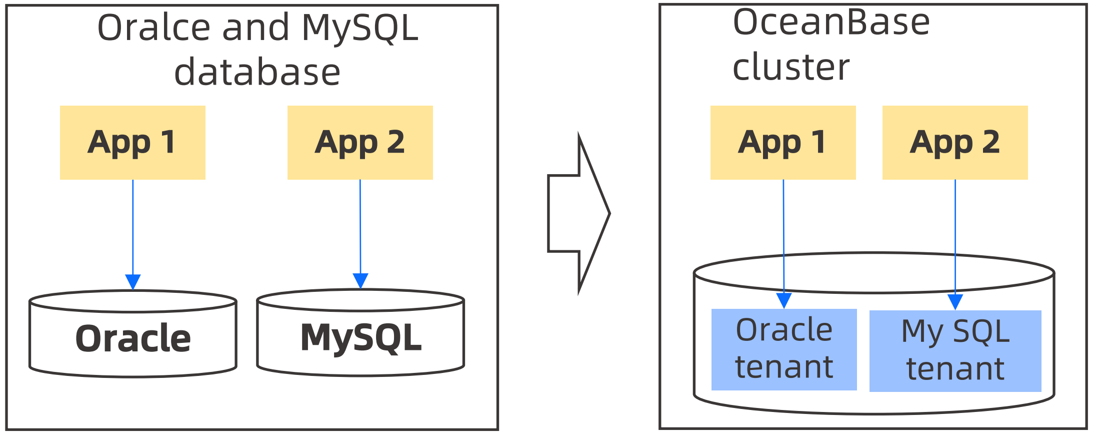
D: OB Proxy is a "stateless" service process that does not persist data.

Chapter 5: OBServer SQL engine and storage engine

1. **SQL engine**
2. Storage engine
3. Backup/recovery

SQL engine supports MySQL and Oracle compatible mode

Dual mode (MySQL + Oracle) coexistence: pioneer in the industry



- One cluster supports both MySQL and Oracle compatible DB features
- When creating a tenant, define it either MySQL or Oracle compatible mode
- DBA has shifted from maintaining "multiple database products" to maintaining a "unified database product". DBA can create tenants with different compatible modes based on application requirements

MySQL compatible mode

- Fully compatible with MySQL 5.6 syntax
- Compatible with MySQL communication protocol, MySQL application can directly migrate to OceanBase

Oracle compatible mode

- Compatible with Oracle 11g syntax
- Support 90% Oracle data type and internal functions, enhance with new software version releases
- Support stored procedure of distributed execution (PL/SQL)

SQL engine – Oracle compatibility

Basic features

- Data type: NUMBER/CHAR/LOB 10+
- Built-in functions: MATH/CAST/FORMAT 40+

DDL&DML

- ROWNUM/CTAS/MERGE INTO/...
- WINDOW FUNCTION: MIN/MAX/RANK 20+
- LOAD/DUPLICATED Table

Schema

- Sequence/Synonym
- CTE/Foreign Key
- GBK
- Static view: TABLES/INDEXES/OBJECTS 10+
- Dynamic view: V$SESSTAT/V$SESSION_WAIT 20+

PL

- PL/function/package basic syntax
- Anonymous blocks/self-defined types/cursor/ARRAY/dynamic SQL
- Common system package

Features and functions

- Recycle bin
- Hybrid Columnar Compression
- Outline
- Hierarchical Query
- Data Guard

Basic operations: Create, view and delete database

- **Use CREATE DATABASE statement to create database**

CREATE DATABASE [IF NOT EXISTS] dbname

[create_specification_list]

- **Use SHOW DATABASES statement to view database**

SHOW DATABASES;

- **Use DROP DATABASE statement to delete database**

DROP DATABASE my_db;

Basic operations: Create, view and delete table

- **Use CREATE TABLE statement to create table in database**

CREATE TABLE [IF NOT EXISTS] tblname
(create_definition,...)
[table_options]
[partition_options];

Example :

CREATE TABLE test (c1 int primary key, c2 VARCHAR (50)) ;

- **Use SHOW CREATE TABLE statement to view table creating statement**

SHOW CREATE TABLE test;

- **Use DROP TABLE statement to delete database**

DROP TABLE [IF EXISTS] table_list;
table_list:
tblname [, tblname ...]

Example:

DROP TABLE test;
DROP TABLE IF EXISTS test;

- **Use SHOWTABLES statement to view all tables in database**

Example

SHOW TABLES FROM my_db;

Basic operations: Create, view and delete index

An index is a structure created on a table to sort the values of one or more columns in a database table. Its main function is to improve the speed of query and reduce the performance overhead of database system

- **Create index**

CREATE [UNIQUE] INDEX indexname

ON tblname (index_col_name,...)

[index_type] [index_options]

- **View index**

SHOW INDEX FROM tblname;

- **Delete index**

DROP INDEX indexname

ON tblname;

Examples

- **Run the following command to create table test**

CREATE TABLE test (c1 int primary key, c2 VARCHAR (10)) ;

- **Run the following command to create index of test**

CREATE INDEX test_index ON test (c1, c2 DESC) ;

- **Run the following command to view index of test**

SHOW INDEX FROM test;

- **Run the following command to delete index of test**

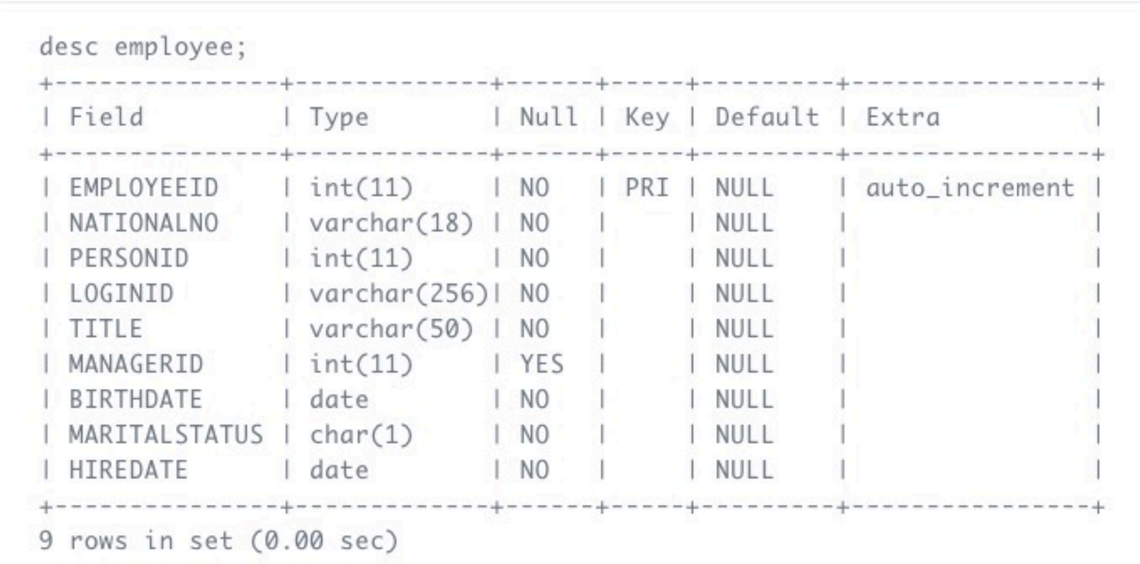
DROP INDEX test_index ON test;

Basic operations: Insert data

Insert data is the most commonly used syntax:

INSERT INTO table_name (list_of_columns) VALUES (list_of_values) ;

Example: Insert a row of data into table employee



```
desc employee;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| EMPLOYEEID    | int(11)     | NO   | PRI | NULL    | auto_increment |
| NATIONALNO    | varchar(18) | NO   |     | NULL    |                |
| PERSONID      | int(11)     | NO   |     | NULL    |                |
| LOGINID       | varchar(256)| NO   |     | NULL    |                |
| TITLE         | varchar(50) | NO   |     | NULL    |                |
| MANAGERID     | int(11)     | YES  |     | NULL    |                |
| BIRTHDATE     | date        | NO   |     | NULL    |                |
| MARITALSTATUS | char(1)     | NO   |     | NULL    |                |
| HIREDATE      | date        | NO   |     | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
9 rows in set (0.00 sec)
```

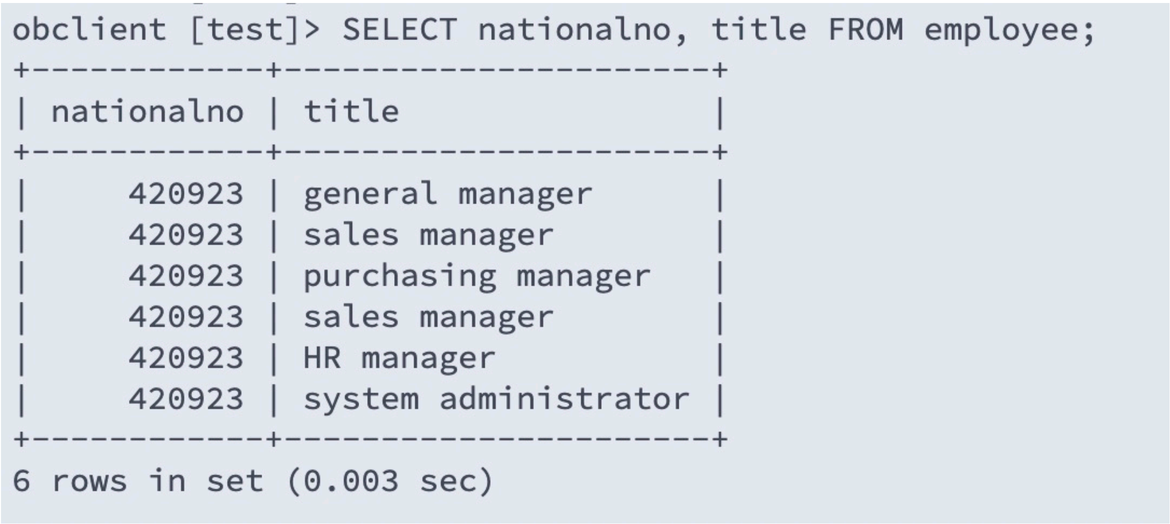
INSERT INTO resources.employee VALUES (99, ‘362202197011158871’,100,‘20’,’translation expert', 2, '1970-11-15', 's', '2015-9-28') ;

Basic operations: Query data

Use SELECT statement to retrieve table data

```
SELECT
[ALL | DISTINCT]
selectexpr [[AS] othername] [,selectexpr ...]
[FROM table_references]
[WHERE where_conditions]
[GROUP BY group_by_list]
[HAVING search_confitions]
[ORDER BY order_list]
 [LIMIT {[offset,] rowcount | rowcount OFFSET offset}]
[FOR UPDATE];
```

Example: Look for the ID card (nationalno) and title (title) from the table employee



```
obclient [test]> SELECT nationalno, title FROM employee;
+------------+----------------------+
| nationalno | title                |
+------------+----------------------+
|     420923 | general manager      |
|     420923 | sales manager        |
|     420923 | purchasing manager   |
|     420923 | sales manager        |
|     420923 | HR manager           |
|     420923 | system administrator |
+------------+----------------------+
6 rows in set (0.003 sec)
```

Basic operations: Update data

The common syntax for modifying table data:

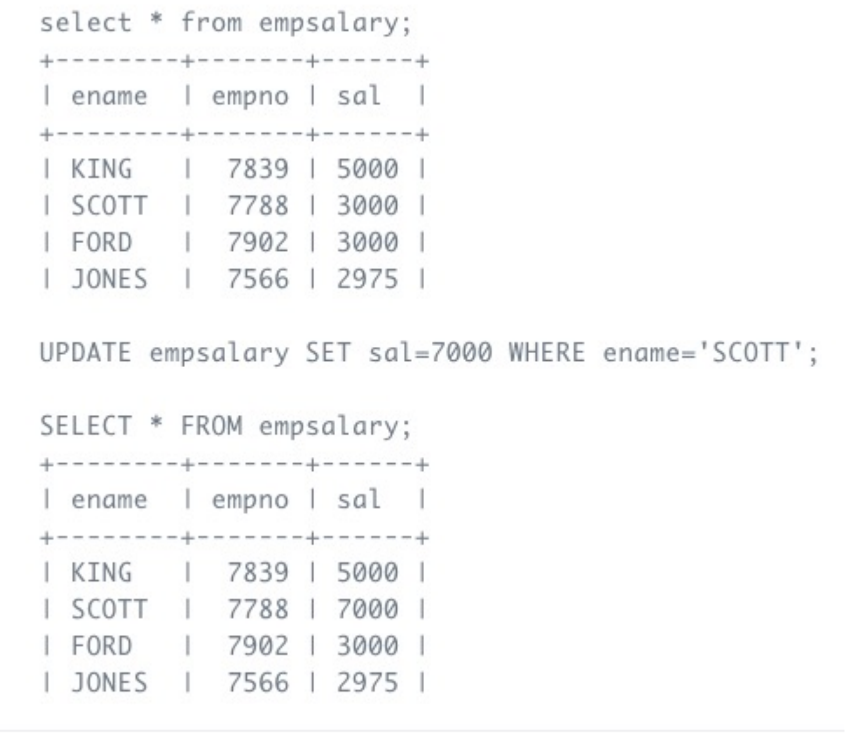
UPDATE tblname

SET colname=colvalues

[, colname=colvalues...]

[WHERE where_condition]

Example: Update a row of data in table empsalary



```
select * from empsalary;
+--------+-------+------+
| ename  | empno | sal  |
+--------+-------+------+
| KING   |  7839 | 5000 |
| SCOTT  |  7788 | 3000 |
| FORD   |  7902 | 3000 |
| JONES  |  7566 | 2975 |

UPDATE empsalary SET sal=7000 WHERE ename='SCOTT';

SELECT * FROM empsalary;
+--------+-------+------+
| ename  | empno | sal  |
+--------+-------+------+
| KING   |  7839 | 5000 |
| SCOTT  |  7788 | 7000 |
| FORD   |  7902 | 3000 |
| JONES  |  7566 | 2975 |
```

Basic operations: Transaction introduction

- **Database Transaction is a series of operations that are performed as a single logical unit of work. Transaction processing can be used to maintain database integrity by ensuring that batch SQL operations are executed at all or none**

- **Transaction starts with BEGIN TRANSACTION, or BEGIN and BEGIN WORK (supported as an alias of STARTTRANSACTION) statements, and ends with COMMIT or ROLLBACK statement**

Transaction syntax is as below:

```
START TRANSACTION | [BEGIN [WORK]]
COMMIT [WORK] ;
ROLLBACK [WORK];
```

Basic operations: Commit transaction

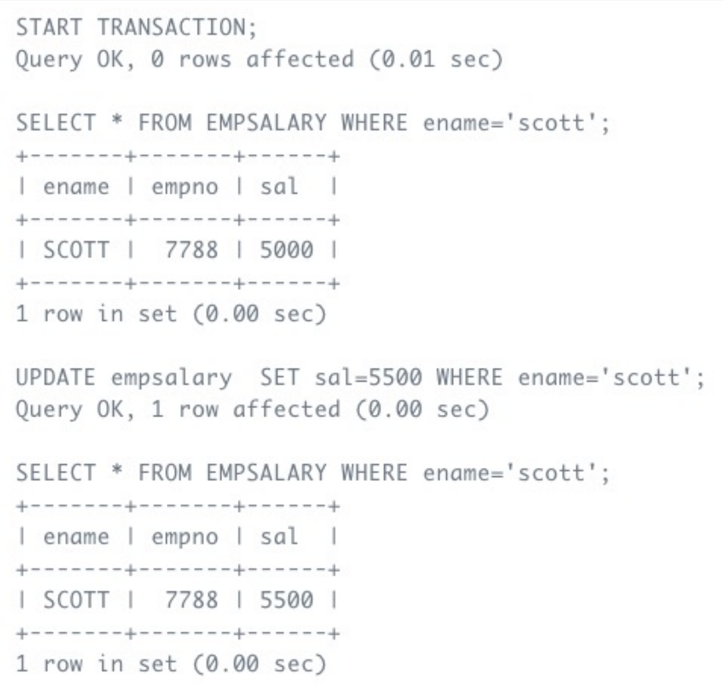
Before transaction commit:

- Changes are visible only to the current user and not to other users
- The changes are not final and can be undone by ROLLBACK statement

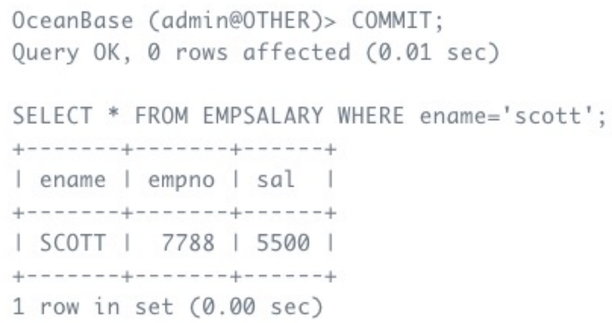
After transaction commit :

- Changes are visible to other users
- Changes are finalized and cannot be rolled back through the ROLLBACK statement

Example:



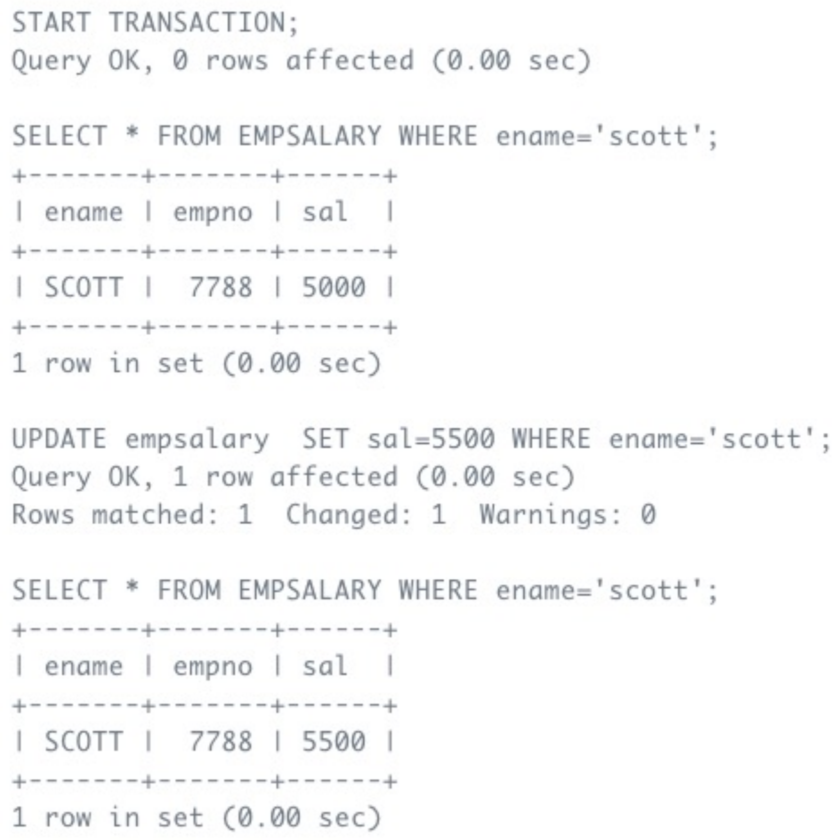
```
START TRANSACTION;
Query OK, 0 rows affected (0.01 sec)

SELECT * FROM EMPSALARY WHERE ename='scott';
+-------+-------+------+
| ename | empno | sal  |
+-------+-------+------+
| SCOTT |  7788 | 5000 |
+-------+-------+------+
1 row in set (0.00 sec)

UPDATE empsalary  SET sal=5500 WHERE ename='scott';
Query OK, 1 row affected (0.00 sec)

SELECT * FROM EMPSALARY WHERE ename='scott';
+-------+-------+------+
| ename | empno | sal  |
+-------+-------+------+
| SCOTT |  7788 | 5500 |
+-------+-------+------+
1 row in set (0.00 sec)
```



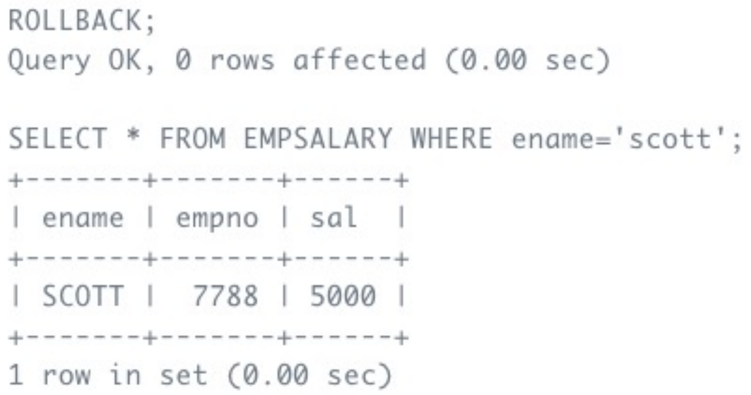
```
OceanBase (admin@OTHER)> COMMIT;
Query OK, 0 rows affected (0.01 sec)

SELECT * FROM EMPSALARY WHERE ename='scott';
+-------+-------+------+
| ename | empno | sal  |
+-------+-------+------+
| SCOTT |  7788 | 5500 |
+-------+-------+------+
1 row in set (0.00 sec)
```

Basic operations: Roll back transaction

Roll back transaction: ROLLBACK indicates that the transaction has ended. It rolls back all transaction operations and releases transaction locks.
Example:



```
START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

SELECT * FROM EMPSALARY WHERE ename='scott';
+-------+-------+------+
| ename | empno | sal  |
+-------+-------+------+
| SCOTT |  7788 | 5000 |
+-------+-------+------+
1 row in set (0.00 sec)

UPDATE empsalary  SET sal=5500 WHERE ename='scott';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

SELECT * FROM EMPSALARY WHERE ename='scott';
+-------+-------+------+
| ename | empno | sal  |
+-------+-------+------+
| SCOTT |  7788 | 5500 |
+-------+-------+------+
1 row in set (0.00 sec)
```



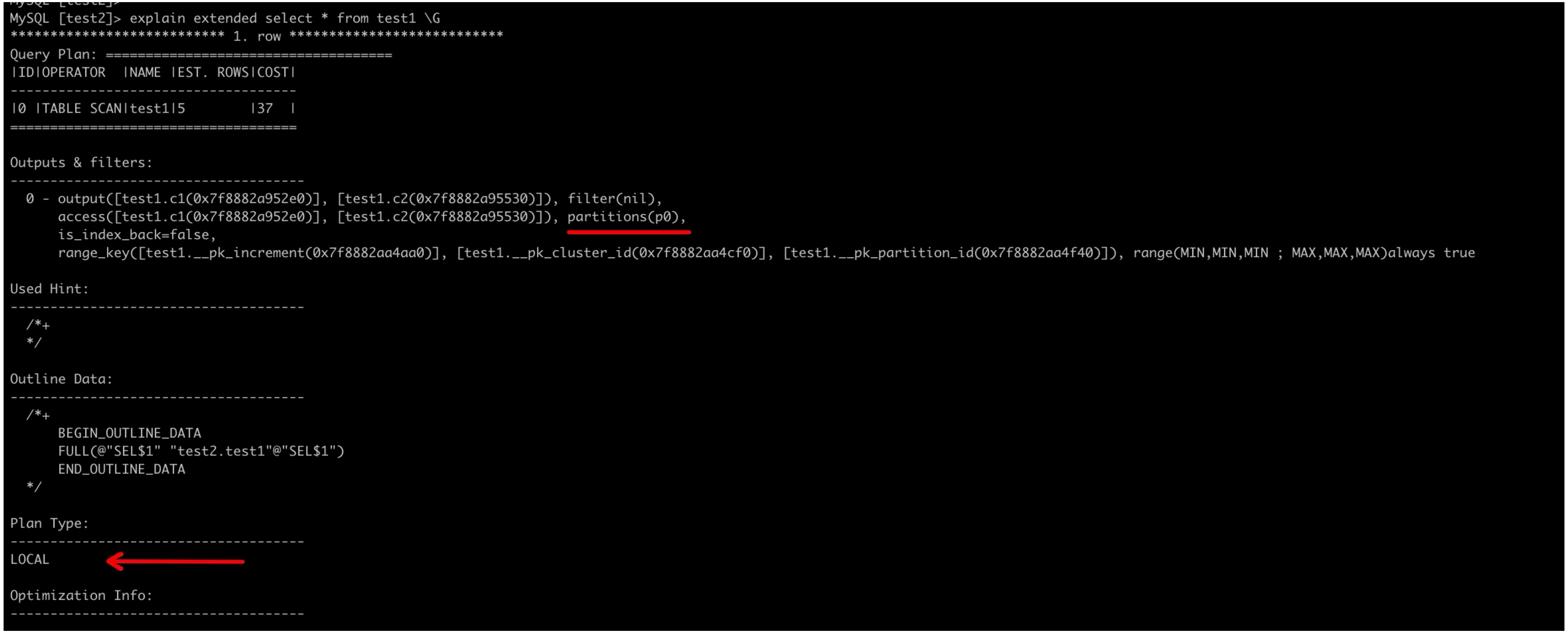
```
ROLLBACK;
Query OK, 0 rows affected (0.00 sec)

SELECT * FROM EMPSALARY WHERE ename='scott';
+-------+-------+------+
| ename | empno | sal  |
+-------+-------+------+
| SCOTT |  7788 | 5000 |
+-------+-------+------+
1 row in set (0.00 sec)
```

Check SQL execution plan – EXPLAIN command

- Syntax: explain [basic | extended | partition] <sql statement>
- easy to use, without creating a separate system table, you can directly obtain the execution plan of the statement
- The extended option generates more details and is recommended for troubleshooting execution plan issue
- The output format of the command is similar to that of the EXPLAIN tool of Oracle database
- Only get the execution plan, not actually execute it

observer SQL execution plan - EXPLAIN command

EXPLAIN use case: single partition table

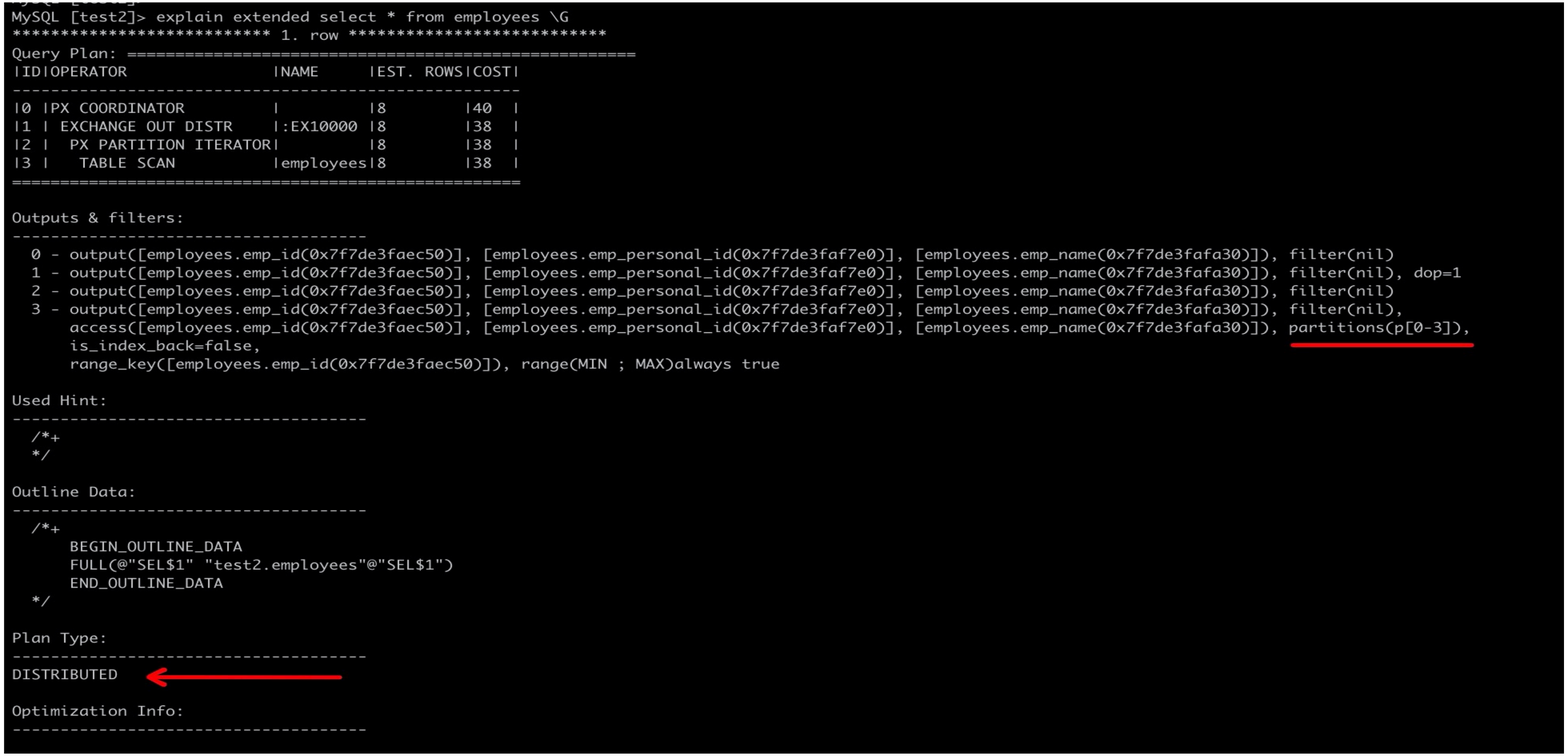


```
MySQL [test2]> explain extended select * from test1 \G
*************************** 1. row ***************************
Query Plan: ===================================
|ID|OPERATOR  |NAME |EST. ROWS|COST|
-----------------------------------
|0 |TABLE SCAN|test1|5        |37  |
===================================

Outputs & filters:
-------------------------------------
  0 - output([test1.c1(0x7f8882a952e0)], [test1.c2(0x7f8882a95530)]), filter(nil),
      access([test1.c1(0x7f8882a952e0)], [test1.c2(0x7f8882a95530)]), partitions(p0),
      is_index_back=false,
      range_key([test1.__pk_increment(0x7f8882aa4aa0)], [test1.__pk_cluster_id(0x7f8882aa4cf0)], [test1.__pk_partition_id(0x7f8882aa4f40)]), range(MIN,MIN,MIN ; MAX,MAX,MAX)always true

Used Hint:
-------------------------------------
  /*+
  */

Outline Data:
-------------------------------------
  /*+
      BEGIN_OUTLINE_DATA
      FULL(@"SEL$1" "test2.test1"@"SEL$1")
      END_OUTLINE_DATA
  */

Plan Type:
-------------------------------------
LOCAL

Optimization Info:
-------------------------------------
```

OCEANBASE

observer SQL execution plan - EXPLAIN command

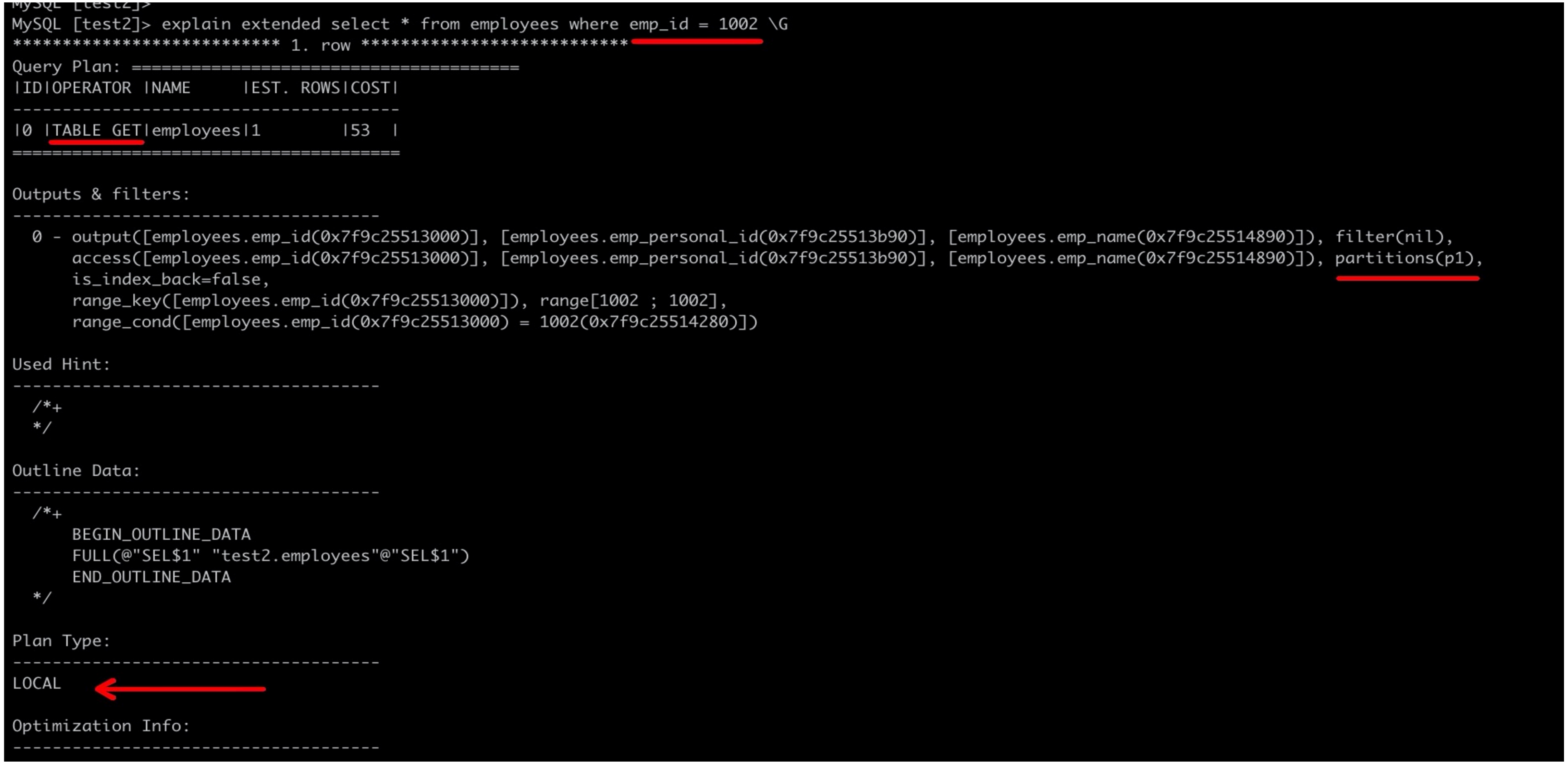
EXPLAIN use case: multi-partition concurrent query



```
MySQL [test2]> explain extended select * from employees \G
*************************** 1. row ***************************
Query Plan: =====================================================
|ID|OPERATOR               |NAME     |EST. ROWS|COST|
-----------------------------------------------------
|0 |PX COORDINATOR         |         |8        |40  |
|1 | EXCHANGE OUT DISTR    |:EX10000 |8        |38  |
|2 |  PX PARTITION ITERATOR|         |8        |38  |
|3 |   TABLE SCAN          |employees|8        |38  |
=====================================================

Outputs & filters:
-------------------------------------
  0 - output([employees.emp_id(0x7f7de3faec50)], [employees.emp_personal_id(0x7f7de3faf7e0)], [employees.emp_name(0x7f7de3fafa30)]), filter(nil)
  1 - output([employees.emp_id(0x7f7de3faec50)], [employees.emp_personal_id(0x7f7de3faf7e0)], [employees.emp_name(0x7f7de3fafa30)]), filter(nil), dop=1
  2 - output([employees.emp_id(0x7f7de3faec50)], [employees.emp_personal_id(0x7f7de3faf7e0)], [employees.emp_name(0x7f7de3fafa30)]), filter(nil)
  3 - output([employees.emp_id(0x7f7de3faec50)], [employees.emp_personal_id(0x7f7de3faf7e0)], [employees.emp_name(0x7f7de3fafa30)]), filter(nil),
      access([employees.emp_id(0x7f7de3faec50)], [employees.emp_personal_id(0x7f7de3faf7e0)], [employees.emp_name(0x7f7de3fafa30)]), partitions(p[0-3]),
      is_index_back=false,
      range_key([employees.emp_id(0x7f7de3faec50)]), range(MIN ; MAX)always true

Used Hint:
-------------------------------------
  /*+
  */

Outline Data:
-------------------------------------
  /*+
      BEGIN_OUTLINE_DATA
      FULL(@"SEL$1" "test2.employees"@"SEL$1")
      END_OUTLINE_DATA
  */

Plan Type:
-------------------------------------
DISTRIBUTED

Optimization Info:
-------------------------------------
```

observer SQL execution plan – EXPLAIN command

EXPLAIN use case: Partition table primary key query



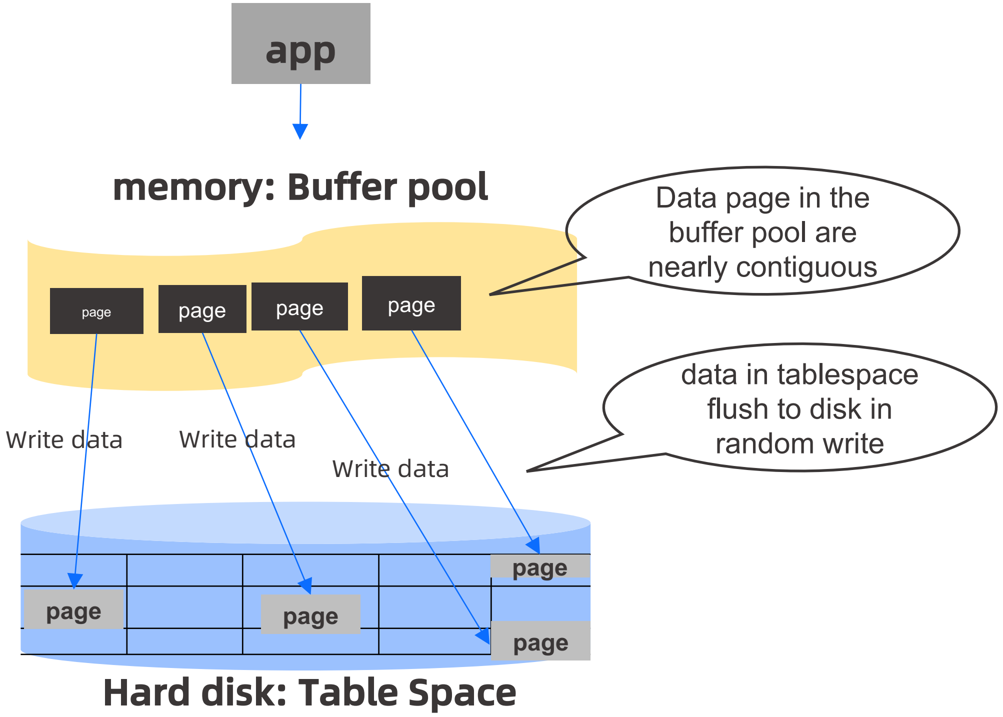
```
MySQL [test2]>
MySQL [test2]> explain extended select * from employees where emp_id = 1002 \G
*************************** 1. row ***************************
Query Plan: =======================================
|ID|OPERATOR |NAME     |EST. ROWS|COST|
---------------------------------------
|0 |TABLE GET|employees|1        |53  |
=======================================

Outputs & filters:
-------------------------------------
  0 - output([employees.emp_id(0x7f9c25513000)], [employees.emp_personal_id(0x7f9c25513b90)], [employees.emp_name(0x7f9c25514890)]), filter(nil),
      access([employees.emp_id(0x7f9c25513000)], [employees.emp_personal_id(0x7f9c25513b90)], [employees.emp_name(0x7f9c25514890)]), partitions(p1),
      is_index_back=false,
      range_key([employees.emp_id(0x7f9c25513000)]), range[1002 ; 1002],
      range_cond([employees.emp_id(0x7f9c25513000) = 1002(0x7f9c25514280)])

Used Hint:
-------------------------------------
  /*+
  */

Outline Data:
-------------------------------------
  /*+
      BEGIN_OUTLINE_DATA
      FULL(@"SEL$1" "test2.employees"@"SEL$1")
      END_OUTLINE_DATA
  */

Plan Type:
-------------------------------------
LOCAL

Optimization Info:
-------------------------------------
```

Chapter 5: OB Server SQL engine and storage engine

1. SQL engine
2. **Storage engine**
3. Backup/recovery

OCEANBASE

Random write and write amplification issue in traditional database



Massive random write: The buffer pool page maps to the tablespace page one by one, and frequent random write (check point) occur on disk when data is updated.

Write amplification: Random write causes SSD write magnification, affecting disk performance and lifespan

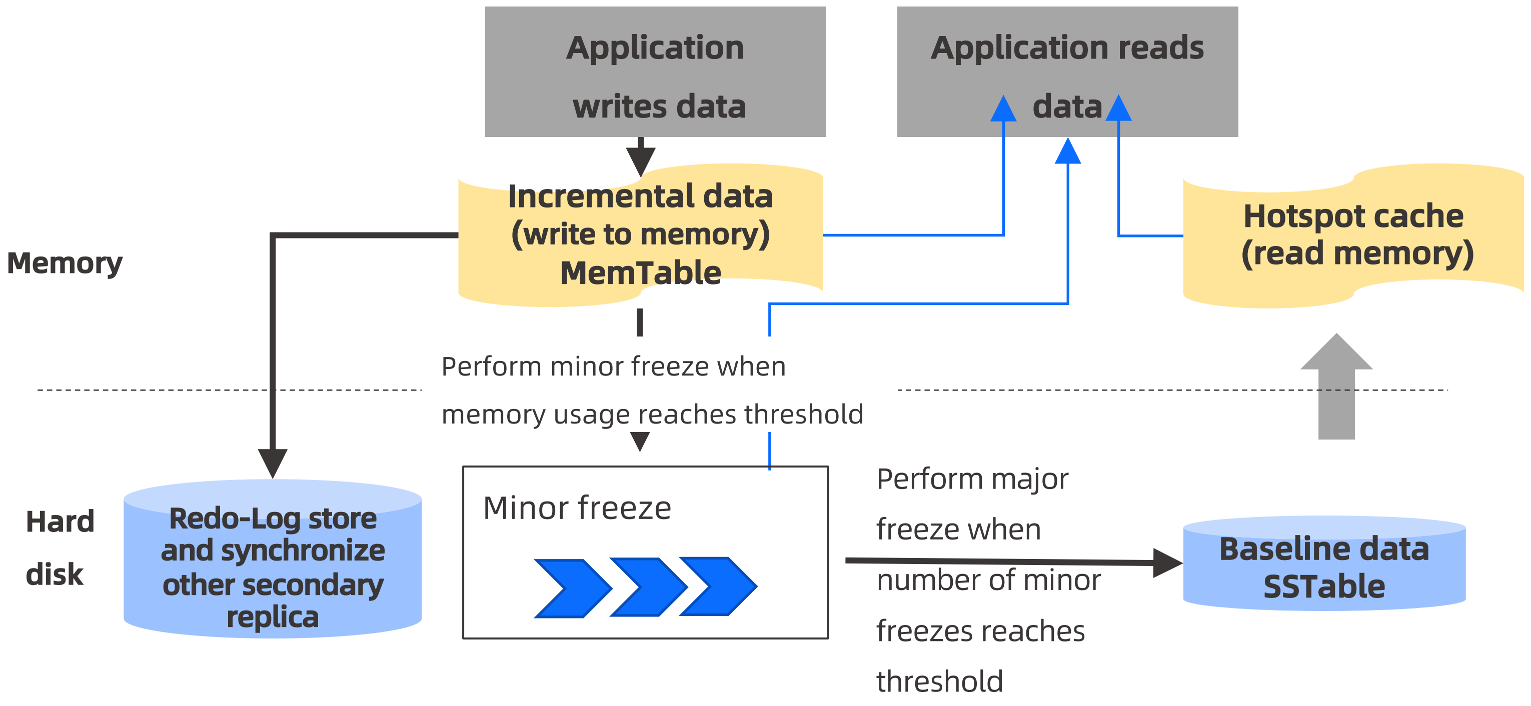
Data Read

- If data in buffer Pool, it is read from the memory. If not, it read from the disk to buffer Pool
- Improve read speed of hot data and reduce the delay

Data Write

- When modifying data, the data is written to the buffer pool and then flushed to disk
- The checkpoint is used to refresh dirty data to disk, resulting in random write and write magnification:
  - Data pages stored on disk are sporadic, resulting in a large number of random writes, large latency, and poor performance
  - Random write on SSD results in severe write magnification, which not only affects the write operation performance, but also reduces SSD lifespan significantly
  - High-end range read/write type of SSD are generally used

Semi-memory database+ LSM-Tree storage to avoid random write



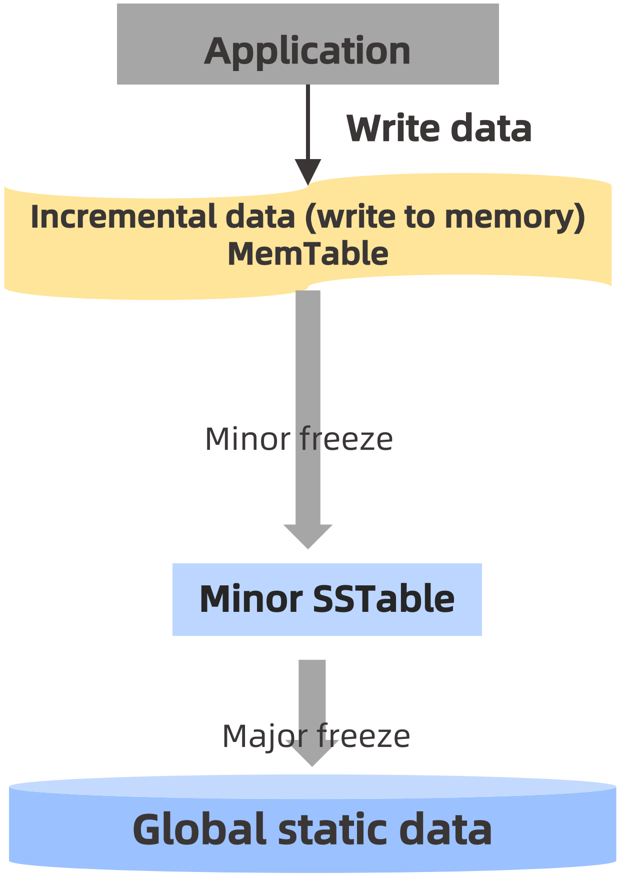
Application writes data
Application reads data
Memory
Incremental data (write to memory) MemTable
Hotspot cache (read memory)
Perform minor freeze when memory usage reaches threshold
Hard disk
Redo-Log store and synchronize other secondary replica
Minor freeze
Perform major freeze when number of minor freezes reaches threshold
Baseline data SSTable

- After the incremental data is written to the memory and redo-log is written to disk and synchronized to the secondary replica, it signal apps a ‘success’state
- When memory usage reaches the threshold, the MemTable is frozen, minor/major freeze operation is triggered to free up memory
- Incremental memory data is merged to disk in batches. Sequential data write instead of random data write
- Data is read from the hotspot cache, MemTable, and minor-SSTable to ensure data consistency

Technical advantages

- **Read/write separation**: Read memory is separated from write memory
- **Increase write speed**: semi-memory processing; Data modification is mainly performed in memory without frequent checkpoint operations, improving write performance
- **Avoid random write:** After dirty data is merged in batches, it is written to SSD in sequence to avoid random write, improve write performance and prolong SSD lifespan
- **Data persistence:** To avoid data loss in memory, redo-logs write to disk in real time via WAL to ensure data persistence
- **Lower cost:** Disk data is managed in order by primary key, reducing disk fragmentation and providing fast retrieval capability. Use generic read-intensive SSD
- The underlying storage is divided into micro blocks and macro blocks, which are managed internally by database

OceanBase minor/major freeze introduction



Minor freeze

- The goal is to consistently write MemTable to disk to free up memory
- Minor freeze process first freezes the MemTable (preventing new writes to current MemTable) and generate a new active MemTable
- The Partition data can perform minor freeze independently
- Data after minor freeze is rolled up with incremental data of the same version, not with global static data

Major freeze)

- consolidating static and dynamic data is time-consuming. When the incremental data generated by the minor-freeze accumulate to a certain extent, bigger version is created through a Major Freeze

- **Disk data is arranged according to the primary key, providing fast retrieval capability. Data in MemTable undergo multi-level sort-merge(minor-major) operations, ultimately, consolidate into SSTable, such activity has little impact on overall performance**

Difference between minor/major freeze

The biggest difference between minor freeze and major freeze is that the major freeze is a global operation performed by merging all partitions in a cluster with global static data at a unified snapshot point and finally creating a global snapshot. The comparison between minor and major freeze shown below:

Minor freeze	Major freeze
Partition level, materialization of MemTable	Global level, generating a global snapshot
Each Partition determines the freezing operation of the MemTable independently. The primary and secondary partitions do not need to be the same.	Freezing MemTable on all global partitions must be consistent between the primary and secondary partitions.
Minor freeze operation only merge data with the same version of Minor SSTables to produce new Minor SSTables, so only incremental data is contained, hence rows that are eventually deleted need special tag.	The major freeze operation merges latest SSTable and MemTable with the full static data of previous data version, generating a new full set of data.

Control memory data stored in disk (minor/major freeze)

Threshold triggering memstore minor freeze operation

- freeze_trigger_percentage parameter, default 70; indicate when the memstore is 70% full, it automatically triggers minor or major freeze operation, depending on the parameter setting.

minor freeze opportunity

- Automatically triggered when the memory reaches threshold
- Manual triggered by command ‘alter system minor freeze; ‘, via user root@sys

major freeze opportunity

- scheduled : Controlled by parameter major_freeze_duty_time. The default value is "02:00".
- Manual triggered by command ‘alter system major freeze;’, via user root@sys
- Maximum number of times reached: When the number of minor freeze operations specified by major_compact_trigger is reached, the major freeze operation is automatically triggered. When the value is 0, the minor freeze is disabled and the major freeze is activated

Major freeze operation support rotational type, which allows major freeze operation of multiple zones in turn.

Control memory data stored in disk (minor/major freeze) - Other notes

Can major freeze closed completely?

- Change parameter enable_major_freeze = False; the default value is True, which is recommended
- enable_manual_merge = True; Enable manual major freeze, all major freeze operations must be triggered manually. This parameter is not recommended, unless particular O&M cases

Concurrent major freeze threads

- Parameter merge_thread_count controls the degree of parallelism, the granularity is per partition
- The default value is 0 (system automatically determines the parallelism). If the value is too large, the apps performance may be affected
- In some rare scenarios, such as fast memory write operation (e.g., batch processing), this parameter can be increased to speed up memory dump

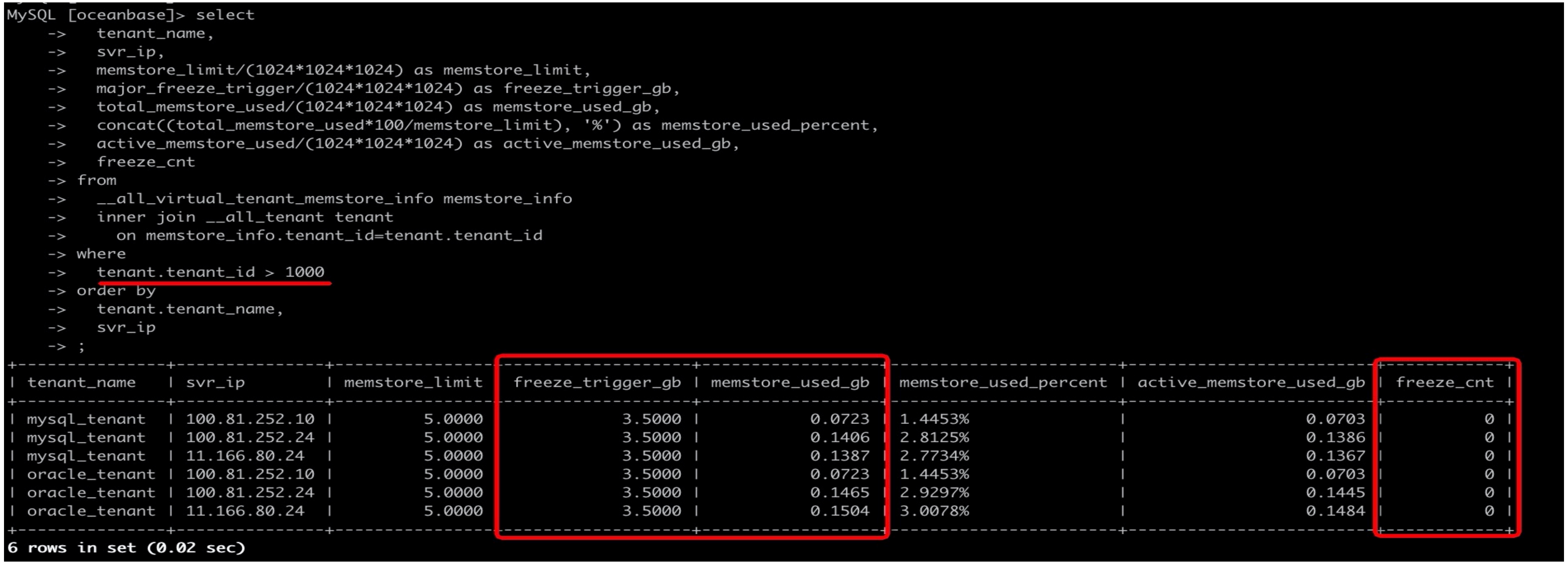
OCEANBASE

OBServer memory usage

- Check memstore usage

show parameters like 'memstore_limit_percentage';

__all_virtual_tenant_memstore_info, categorized by tenant

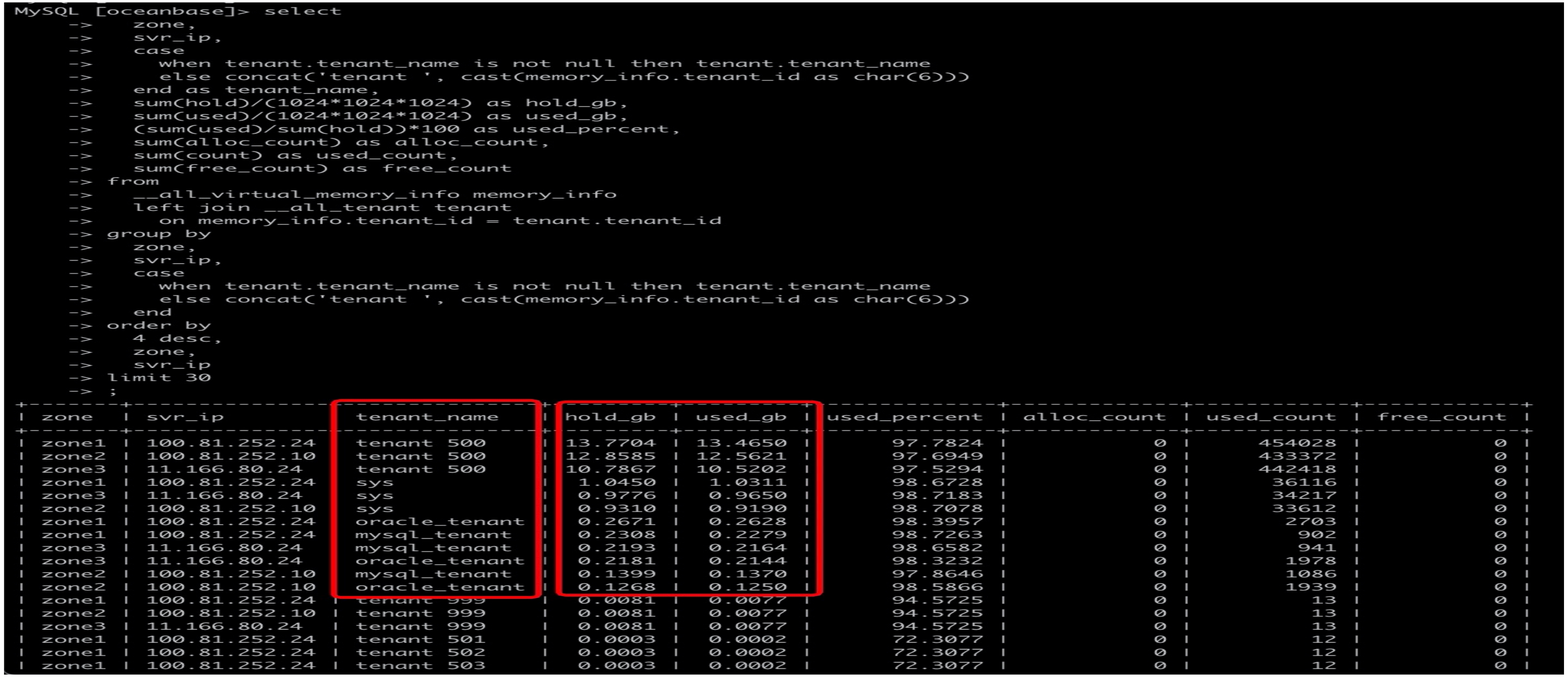


```
MySQL [oceanbase]> select
    ->   tenant_name,
    ->   svr_ip,
    ->   memstore_limit/(1024*1024*1024) as memstore_limit,
    ->   major_freeze_trigger/(1024*1024*1024) as freeze_trigger_gb,
    ->   total_memstore_used/(1024*1024*1024) as memstore_used_gb,
    ->   concat((total_memstore_used*100/memstore_limit), '%') as memstore_used_percent,
    ->   active_memstore_used/(1024*1024*1024) as active_memstore_used_gb,
    ->   freeze_cnt
    -> from
    ->   __all_virtual_tenant_memstore_info memstore_info
    ->   inner join __all_tenant tenant
    ->     on memstore_info.tenant_id=tenant.tenant_id
    -> where
    ->   tenant.tenant_id > 1000
    -> order by
    ->   tenant.tenant_name,
    ->   svr_ip
    -> ;
+---------------+---------------+----------------+-------------------+------------------+-----------------------+-------------------------+------------+
| tenant_name   | svr_ip        | memstore_limit | freeze_trigger_gb | memstore_used_gb | memstore_used_percent | active_memstore_used_gb | freeze_cnt |
+---------------+---------------+----------------+-------------------+------------------+-----------------------+-------------------------+------------+
| mysql_tenant  | 100.81.252.10 |         5.0000 |            3.5000 |           0.0723 | 1.4453%               |                  0.0703 |          0 |
| mysql_tenant  | 100.81.252.24 |         5.0000 |            3.5000 |           0.1406 | 2.8125%               |                  0.1386 |          0 |
| mysql_tenant  | 11.166.80.24  |         5.0000 |            3.5000 |           0.1387 | 2.7734%               |                  0.1367 |          0 |
| oracle_tenant | 100.81.252.10 |         5.0000 |            3.5000 |           0.0723 | 1.4453%               |                  0.0703 |          0 |
| oracle_tenant | 100.81.252.24 |         5.0000 |            3.5000 |           0.1465 | 2.9297%               |                  0.1445 |          0 |
| oracle_tenant | 11.166.80.24  |         5.0000 |            3.5000 |           0.1504 | 3.0078%               |                  0.1484 |          0 |
+---------------+---------------+----------------+-------------------+------------------+-----------------------+-------------------------+------------+
6 rows in set (0.02 sec)
```

OBServer memory usage

- Check usage of non-memstore area

__all_virtual_memory_info, categorized by tenant

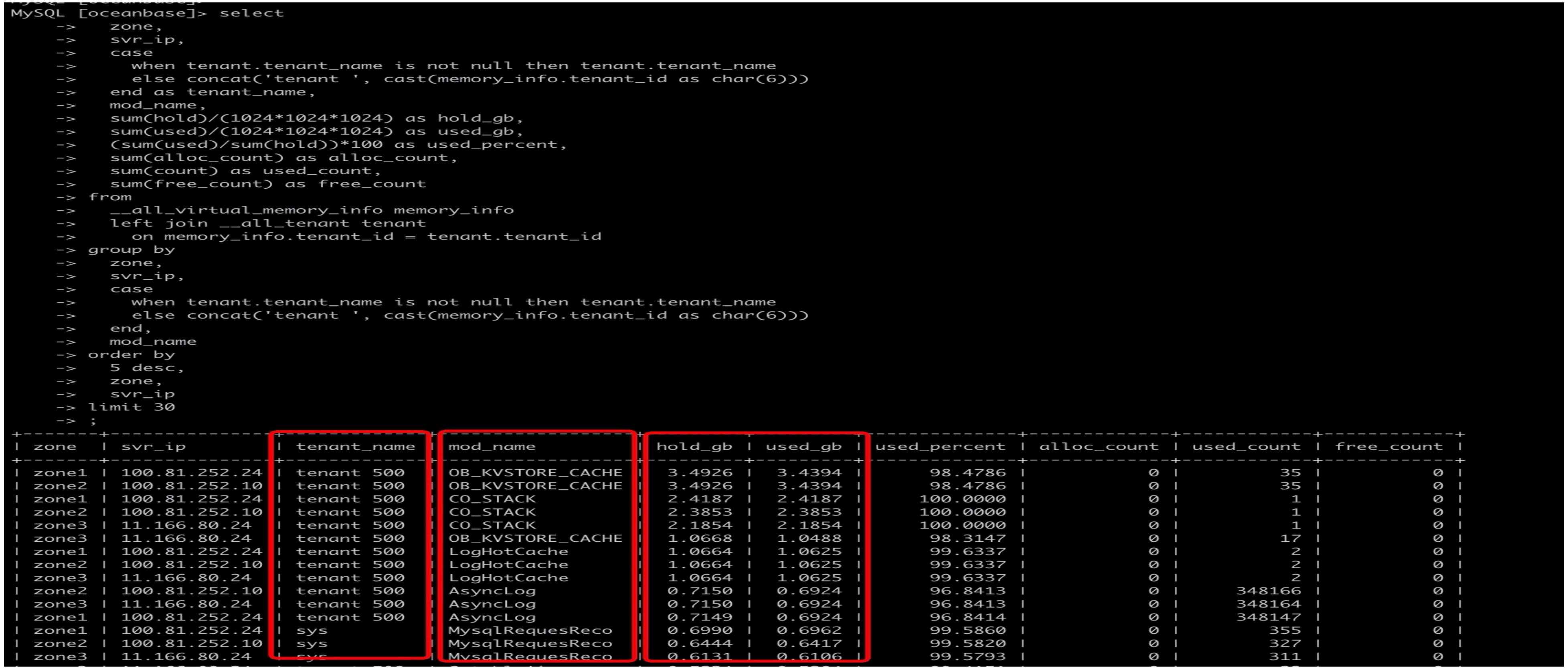


```
MySQL [oceanbase]> select
    ->     zone,
    ->     svr_ip,
    ->     case
    ->       when tenant.tenant_name is not null then tenant.tenant_name
    ->       else concat('tenant ', cast(memory_info.tenant_id as char(6)))
    ->     end as tenant_name,
    ->     sum(hold)/(1024*1024*1024) as hold_gb,
    ->     sum(used)/(1024*1024*1024) as used_gb,
    ->     (sum(used)/sum(hold))*100 as used_percent,
    ->     sum(alloc_count) as alloc_count,
    ->     sum(count) as used_count,
    ->     sum(free_count) as free_count
    -> from
    ->     __all_virtual_memory_info memory_info
    ->     left join __all_tenant tenant
    ->       on memory_info.tenant_id = tenant.tenant_id
    -> group by
    ->     zone,
    ->     svr_ip,
    ->     case
    ->       when tenant.tenant_name is not null then tenant.tenant_name
    ->       else concat('tenant ', cast(memory_info.tenant_id as char(6)))
    ->     end
    -> order by
    ->     4 desc,
    ->     zone,
    ->     svr_ip
    -> limit 30
    -> ;
+-------+----------------+-----------------+---------+---------+--------------+-------------+------------+------------+
| zone  | svr_ip         | tenant_name     | hold_gb | used_gb | used_percent | alloc_count | used_count | free_count |
+-------+----------------+-----------------+---------+---------+--------------+-------------+------------+------------+
| zone1 | 100.81.252.24  | tenant 500      | 13.7704 | 13.4650 |      97.7824 |           0 |     454028 |          0 |
| zone2 | 100.81.252.10  | tenant 500      | 12.8585 | 12.5621 |      97.6949 |           0 |     433372 |          0 |
| zone3 | 11.166.80.24   | tenant 500      | 10.7867 | 10.5202 |      97.5294 |           0 |     442418 |          0 |
| zone1 | 100.81.252.24  | sys             |  1.0450 |  1.0311 |      98.6728 |           0 |      36116 |          0 |
| zone3 | 11.166.80.24   | sys             |  0.9776 |  0.9650 |      98.7183 |           0 |      34217 |          0 |
| zone2 | 100.81.252.10  | sys             |  0.9310 |  0.9190 |      98.7078 |           0 |      33612 |          0 |
| zone1 | 100.81.252.24  | oracle_tenant   |  0.2671 |  0.2628 |      98.3957 |           0 |       2703 |          0 |
| zone1 | 100.81.252.24  | mysql_tenant    |  0.2308 |  0.2279 |      98.7263 |           0 |        902 |          0 |
| zone3 | 11.166.80.24   | mysql_tenant    |  0.2193 |  0.2164 |      98.6582 |           0 |        941 |          0 |
| zone3 | 11.166.80.24   | oracle_tenant   |  0.2181 |  0.2144 |      98.3232 |           0 |       1978 |          0 |
| zone2 | 100.81.252.10  | mysql_tenant    |  0.1399 |  0.1370 |      97.8646 |           0 |       1086 |          0 |
| zone2 | 100.81.252.10  | oracle_tenant   |  0.1268 |  0.1250 |      98.5866 |           0 |       1939 |          0 |
| zone1 | 100.81.252.24  | tenant 999      |  0.0081 |  0.0077 |      94.5725 |           0 |         13 |          0 |
| zone2 | 100.81.252.10  | tenant 999      |  0.0081 |  0.0077 |      94.5725 |           0 |         13 |          0 |
| zone3 | 11.166.80.24   | tenant 999      |  0.0081 |  0.0077 |      94.5725 |           0 |         13 |          0 |
| zone1 | 100.81.252.24  | tenant 501      |  0.0003 |  0.0002 |      72.3077 |           0 |         12 |          0 |
| zone1 | 100.81.252.24  | tenant 502      |  0.0003 |  0.0002 |      72.3077 |           0 |         12 |          0 |
| zone1 | 100.81.252.24  | tenant 503      |  0.0003 |  0.0002 |      72.3077 |           0 |         12 |          0 |
```

OBServer memory usage

- Check non-memstore modules

__all_virtual_memory_info, categorized by tenant and modules (mod_name)

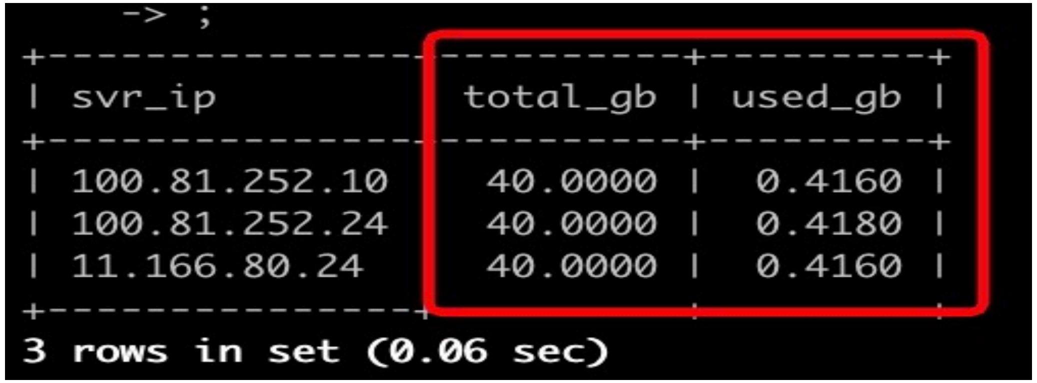


```
MySQL [oceanbase]> select
    ->   zone,
    ->   svr_ip,
    ->   case
    ->     when tenant.tenant_name is not null then tenant.tenant_name
    ->     else concat('tenant ', cast(memory_info.tenant_id as char(6)))
    ->   end as tenant_name,
    ->   mod_name,
    ->   sum(hold)/(1024*1024*1024) as hold_gb,
    ->   sum(used)/(1024*1024*1024) as used_gb,
    ->   (sum(used)/sum(hold))*100 as used_percent,
    ->   sum(alloc_count) as alloc_count,
    ->   sum(count) as used_count,
    ->   sum(free_count) as free_count
    -> from
    ->   __all_virtual_memory_info memory_info
    ->   left join __all_tenant tenant
    ->     on memory_info.tenant_id = tenant.tenant_id
    -> group by
    ->   zone,
    ->   svr_ip,
    ->   case
    ->     when tenant.tenant_name is not null then tenant.tenant_name
    ->     else concat('tenant ', cast(memory_info.tenant_id as char(6)))
    ->   end,
    ->   mod_name
    -> order by
    ->   5 desc,
    ->   zone,
    ->   svr_ip
    -> limit 30
    -> ;
+-------+---------------+-------------+-------------------+---------+---------+--------------+-------------+------------+------------+
| zone  | svr_ip        | tenant_name | mod_name          | hold_gb | used_gb | used_percent | alloc_count | used_count | free_count |
+-------+---------------+-------------+-------------------+---------+---------+--------------+-------------+------------+------------+
| zone1 | 100.81.252.24 | tenant 500  | OB_KVSTORE_CACHE  |  3.4926 |  3.4394 |      98.4786 |           0 |         35 |          0 |
| zone2 | 100.81.252.10 | tenant 500  | OB_KVSTORE_CACHE  |  3.4926 |  3.4394 |      98.4786 |           0 |         35 |          0 |
| zone1 | 100.81.252.24 | tenant 500  | CO_STACK          |  2.4187 |  2.4187 |     100.0000 |           0 |          1 |          0 |
| zone2 | 100.81.252.10 | tenant 500  | CO_STACK          |  2.3853 |  2.3853 |     100.0000 |           0 |          1 |          0 |
| zone3 | 11.166.80.24  | tenant 500  | CO_STACK          |  2.1854 |  2.1854 |     100.0000 |           0 |          1 |          0 |
| zone3 | 11.166.80.24  | tenant 500  | OB_KVSTORE_CACHE  |  1.0668 |  1.0488 |      98.3147 |           0 |         17 |          0 |
| zone1 | 100.81.252.24 | tenant 500  | LogHotCache       |  1.0664 |  1.0625 |      99.6337 |           0 |          2 |          0 |
| zone2 | 100.81.252.10 | tenant 500  | LogHotCache       |  1.0664 |  1.0625 |      99.6337 |           0 |          2 |          0 |
| zone3 | 11.166.80.24  | tenant 500  | LogHotCache       |  1.0664 |  1.0625 |      99.6337 |           0 |          2 |          0 |
| zone2 | 100.81.252.10 | tenant 500  | AsyncLog          |  0.7150 |  0.6924 |      96.8413 |           0 |     348166 |          0 |
| zone3 | 11.166.80.24  | tenant 500  | AsyncLog          |  0.7150 |  0.6924 |      96.8413 |           0 |     348164 |          0 |
| zone1 | 100.81.252.24 | tenant 500  | AsyncLog          |  0.7149 |  0.6924 |      96.8414 |           0 |     348147 |          0 |
| zone1 | 100.81.252.24 | sys         | MysqlRequesReco   |  0.6990 |  0.6962 |      99.5860 |           0 |        355 |          0 |
| zone2 | 100.81.252.10 | sys         | MysqlRequesReco   |  0.6444 |  0.6417 |      99.5820 |           0 |        327 |          0 |
| zone3 | 11.166.80.24  | sys         | MysqlRequesReco   |  0.6131 |  0.6106 |      99.5793 |           0 |        311 |          0 |
```

OBServer disk space usage

- Display disk usage on each machine

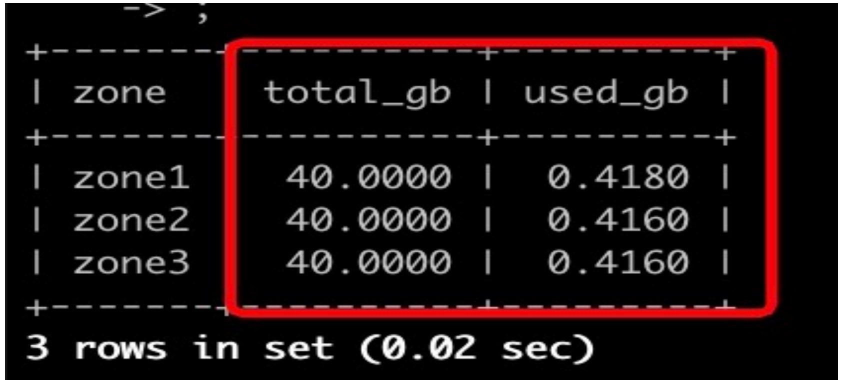
__all_virtual_disk_stat;



```
    -> ;
+--------------------------------+----------+
| svr_ip            total_gb | used_gb  |
+--------------------------------+----------+
| 100.81.252.10      40.0000 |  0.4160 |
| 100.81.252.24      40.0000 |  0.4180 |
| 11.166.80.24       40.0000 |  0.4160 |
+----------------
3 rows in set (0.06 sec)
```

- Display disk usage of each zone

__all_virtual_disk_stat,__all_server;

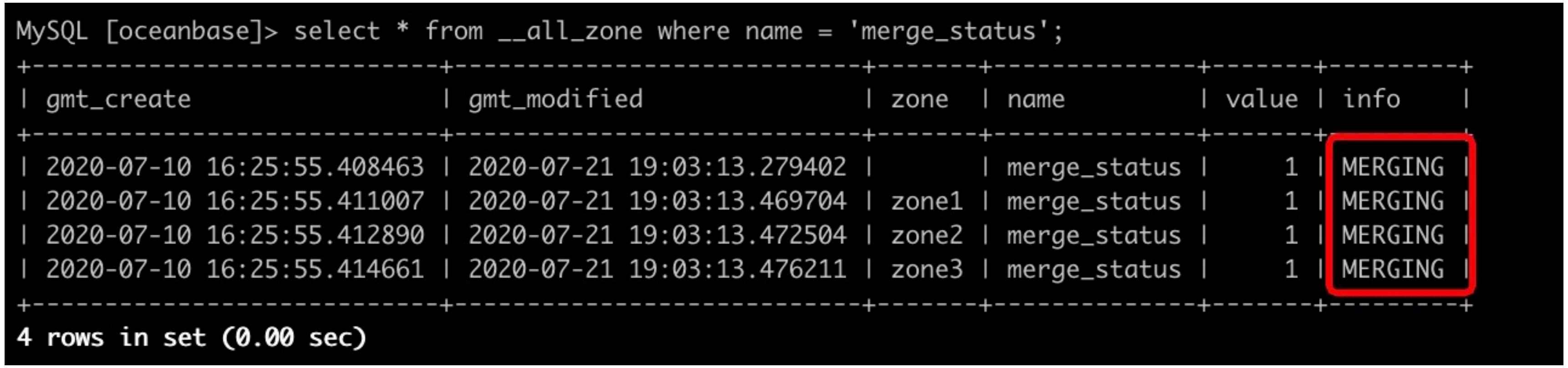


```
    -> ;
+---------------------------+----------+
| zone        total_gb | used_gb |
+---------------------------+----------+
| zone1        40.0000 |  0.4180 |
| zone2        40.0000 |  0.4160 |
| zone3        40.0000 |  0.4160 |
+---------------------------+----------+
3 rows in set (0.02 sec)
```

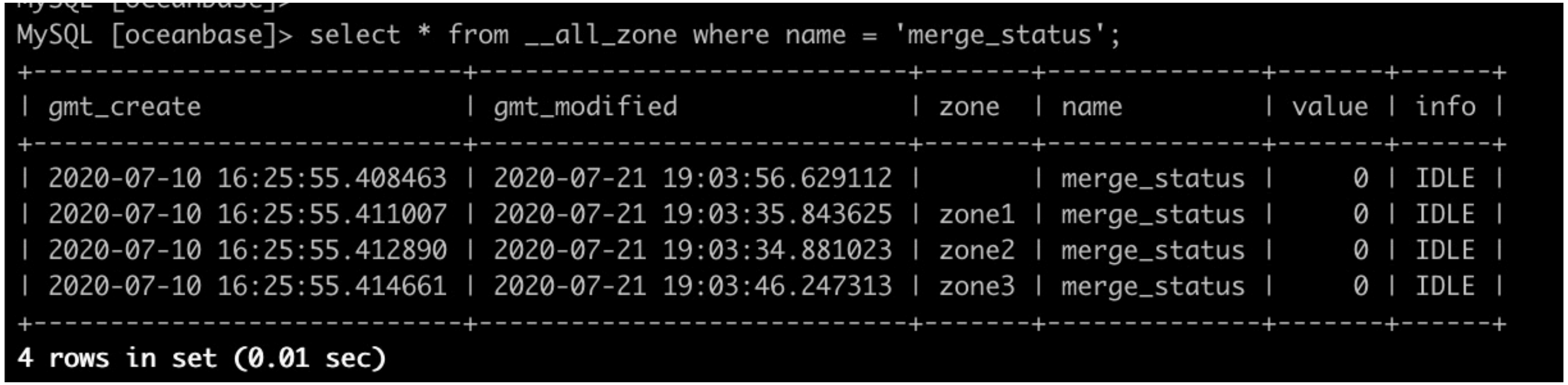
OBServer cluster major freeze operation status

- **Check cluster major freeze status**

```
select * from __all_zone where name = 'merge_status';
```

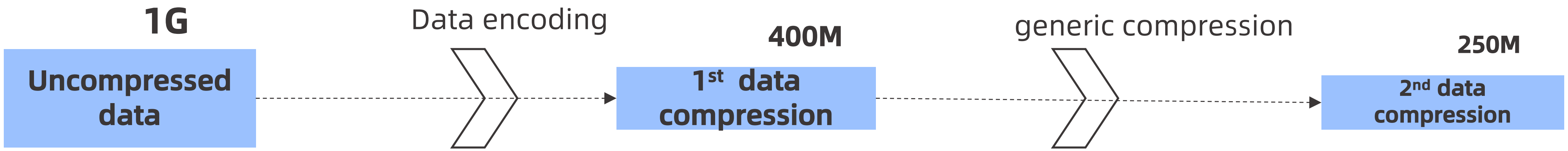


```
MySQL [oceanbase]> select * from __all_zone where name = 'merge_status';
+----------------------------+----------------------------+-------+--------------+-------+---------+
| gmt_create                 | gmt_modified               | zone  | name         | value | info    |
+----------------------------+----------------------------+-------+--------------+-------+---------+
| 2020-07-10 16:25:55.408463 | 2020-07-21 19:03:13.279402 |       | merge_status |     1 | MERGING |
| 2020-07-10 16:25:55.411007 | 2020-07-21 19:03:13.469704 | zone1 | merge_status |     1 | MERGING |
| 2020-07-10 16:25:55.412890 | 2020-07-21 19:03:13.472504 | zone2 | merge_status |     1 | MERGING |
| 2020-07-10 16:25:55.414661 | 2020-07-21 19:03:13.476211 | zone3 | merge_status |     1 | MERGING |
+----------------------------+----------------------------+-------+--------------+-------+---------+
4 rows in set (0.00 sec)
```



```
MySQL [oceanbase]> select * from __all_zone where name = 'merge_status';
+----------------------------+----------------------------+-------+--------------+-------+------+
| gmt_create                 | gmt_modified               | zone  | name         | value | info |
+----------------------------+----------------------------+-------+--------------+-------+------+
| 2020-07-10 16:25:55.408463 | 2020-07-21 19:03:56.629112 |       | merge_status |     0 | IDLE |
| 2020-07-10 16:25:55.411007 | 2020-07-21 19:03:35.843625 | zone1 | merge_status |     0 | IDLE |
| 2020-07-10 16:25:55.412890 | 2020-07-21 19:03:34.881023 | zone2 | merge_status |     0 | IDLE |
| 2020-07-10 16:25:55.414661 | 2020-07-21 19:03:46.247313 | zone3 | merge_status |     0 | IDLE |
+----------------------------+----------------------------+-------+--------------+-------+------+
4 rows in set (0.01 sec)
```

High data compression ratio of LSM Tree to reduce storage capacity



1G

Uncompressed data

Data encoding

400M

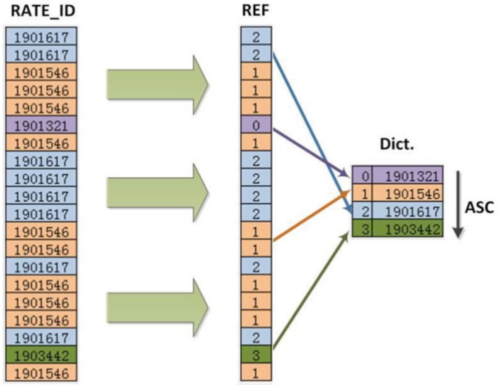
1st data compression

generic compression

250M

2nd data compression

Data coding compression technology for high compression, understand data better than generic compression algorithm, achieving higher data compression efficiency



RATE_ID	REF
1901617	2
1901617	2
1901546	1
1901546	1
1901546	1
1901321	0
1901546	1
1901617	2
1901617	2
1901617	2
1901617	2
1901546	1
1901546	1
1901617	2
1901546	1
1901546	1
1901546	1
1901617	2
1903442	3
1901546	1

Dict.	
0	1901321
1	1901546
2	1901617
3	1903442

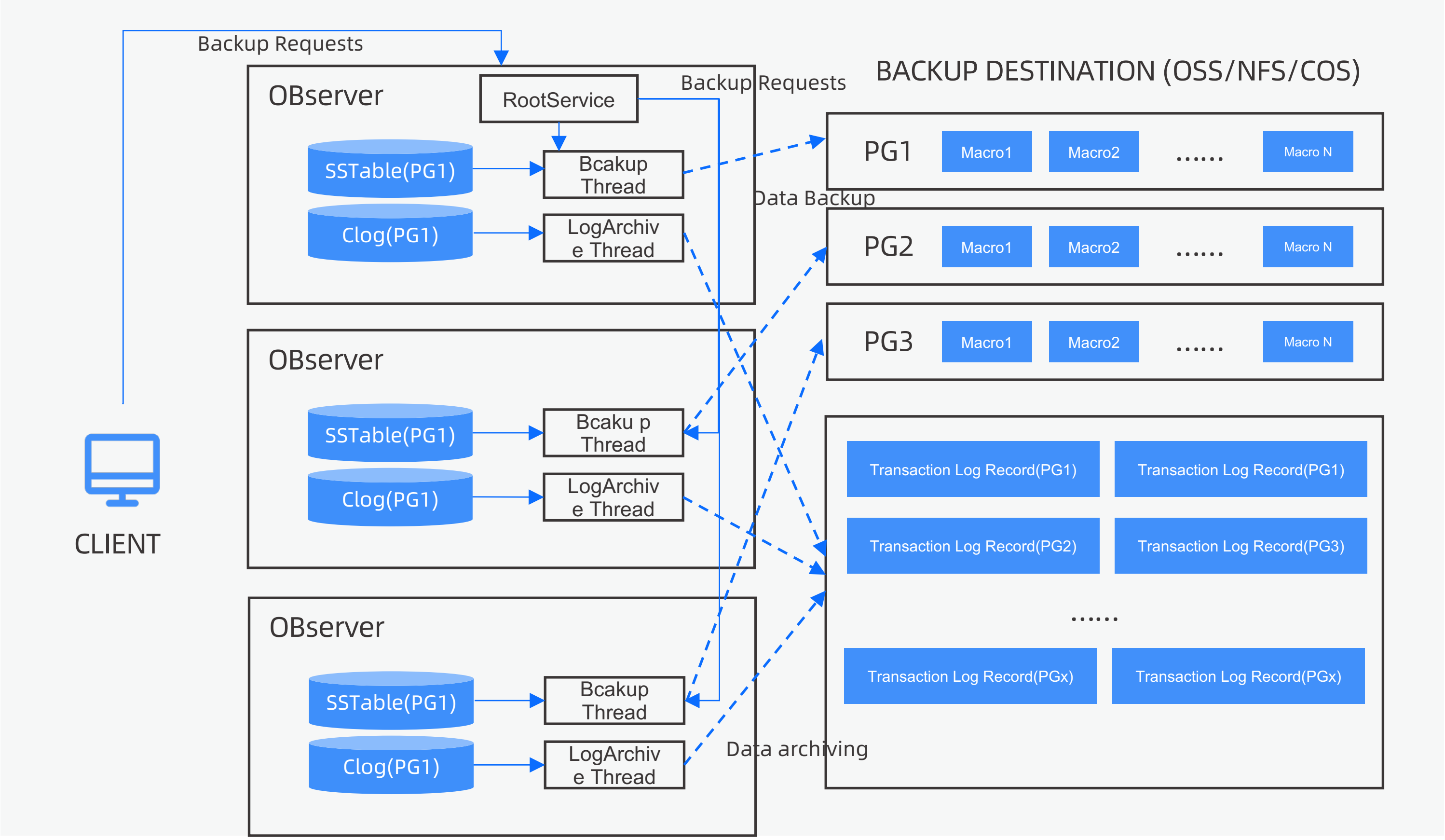
Dictionary: data with high duplication is de-duplicated, and the de-duplicated data is created as a dictionary, and the original data storage place is saved as a pointer to a specific dictionary mark. Data is accessed without decoding

- The 2nd compression is generic compression, using lZ4 and other compression algorithms to further compress the data after Encoding
- Support compression algorithm such as SNappy, LZ4, and ZSTD, allowing users to make trade-offs between compression rate and decompression time
- Assume with same block size (16KB) and the same compression algorithm (LZ4), same amount of data stored in OceanBase is half the size compared with that of MySQL 5.7 on average
- Query performance is basically intact, while write (merge) performance is significantly enhanced

Chapter 5: OB Server SQL engine and storage engine

1. SQL engine
2. Storage engine
3. **Backup/recovery**

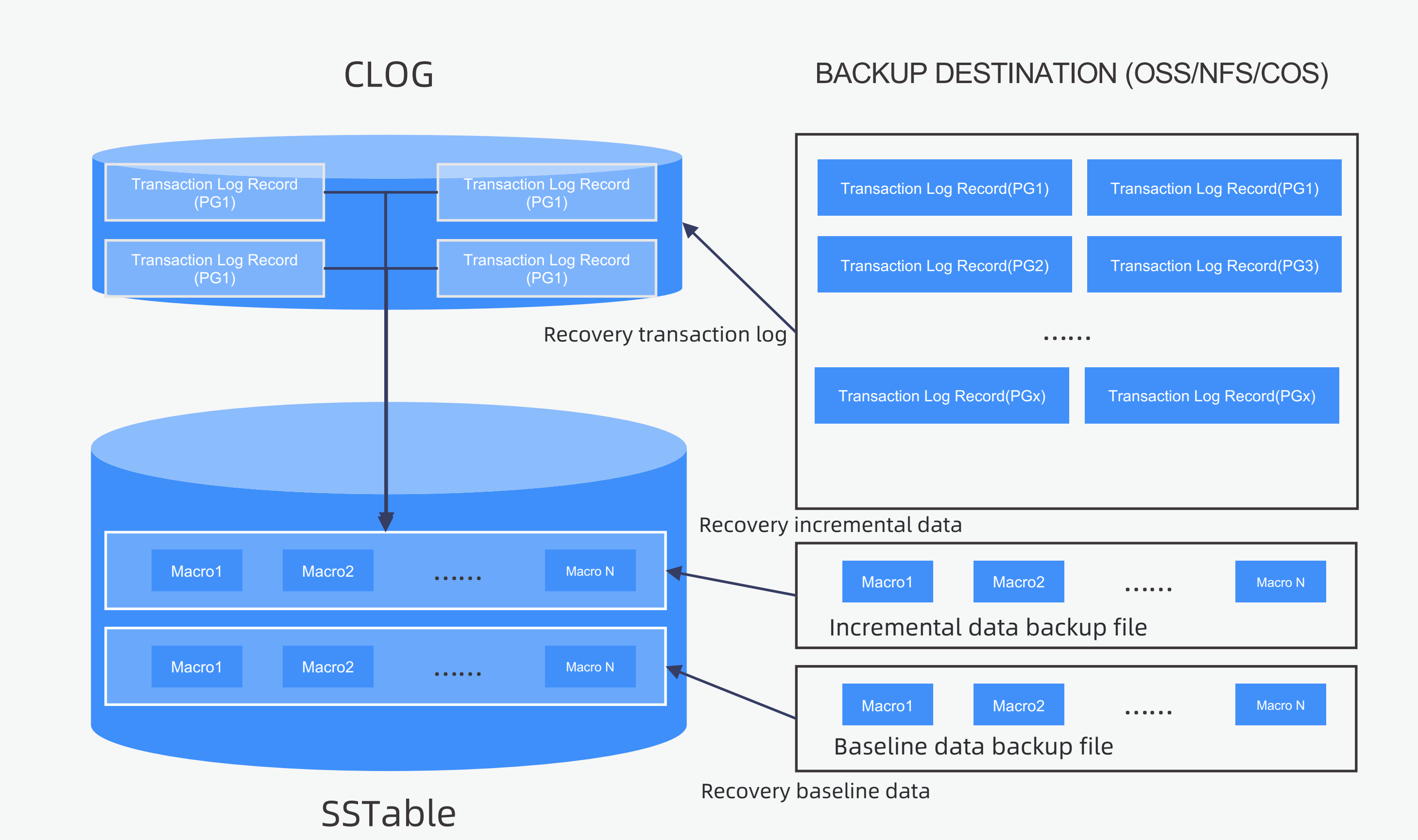
Physical backup: architecture



Support backup media such as OSS, NFS, and COS, and provides backup, recovery, and management functions. Supported functionality includes manually deleting a specified backup and automatically expiration backup

Physical backup includes: data backup and log archive operation. Data backup refers to backing up of baseline data, including full backup(Level 0) and incremental backup (Level 1). Log archive refers to the automatic log data archival

Physical recovery : architecture


CLOG
BACKUP DESTINATION (OSS/NFS/COS)
Transaction Log Record (PG1)
Transaction Log Record (PG1)
Transaction Log Record (PG1)
Transaction Log Record (PG1)
Transaction Log Record(PG1)
Transaction Log Record(PG1)
Transaction Log Record(PG2)
Transaction Log Record(PG3)
......
Transaction Log Record(PGx)
Transaction Log Record(PGx)
Recovery transaction log
Recovery incremental data
Macro1
Macro2
......
Macro N
Incremental data backup file
Baseline data backup file
Recovery baseline data
SSTable

- Tenant level recovery is supported. Recovery is a process of rebuilding new tenants based on existing backup data. User only issue one command 'alter system restore tenant…' to complete recovery process
- The recovery process includes the restore and recover of tenant system tables and user tables. Restore operation is to restore baseline data to target OBServer tenant. Recover operation is to restore archivelogs to OBServer corresponding to the baseline data

Summary

This chapter analyzes the basic technical architecture of SQL engine and storage engine in OB Server from the macro view of cluster to the micro view of OB Server:

- OceanBase can support both MySQL and Oracle tenants in one database. Fully compatible with MySQL 5.6 syntax and Oracle 11g syntax
- You can use the Explain command to view the SQL execution plan
- OceanBase is a quasi-in-memory database and uses LSMTree storage, which can effectively solve the problem of random write and write amplification
- To avoid data loss in memory, redo-logs are dropped in real time using WAL to ensure data persistence
- OceanBase uses dump and merge mechanisms to avoid frequent check points, improve write performance, compress data, and reduce costs

Q&A

Simulation Questions

1.[True/false] OceanBase can support both MySQL and Oracle tenants in a cluster.()

2.[True/false] When you use the Explain command to view the SQL execution plan, the SQL is actually executed. ()

3.[True/false] Merge must be done automatically by OceanBase and cannot be started manually. ()

4.[True/false] Disk data in OceanBase is ordered by primary key. ()

5. [Single choice] Which JDBC driver is required to connect to Oracle tenants using JDBC? ()

A: MySQL standard JDBC driver B: Oracle standard JDBC driver C: OceanBase developed JDBC driver

6. [Single choice] How many times does OceanBase typically compress for better compression? ()

A: Once B: Twice C: 3 times D: 4 times

8. [Multiple choice] Which storage media are supported by OceanBase backup and recovery services? ()

A: NFS B: IP-SAN C: FC-SAN D: AliYUN OSS

Chapter 6: Parameters and Variables

Parameter management

admin range

- Set cluster level parameters to control functions such as load balancing, major freeze time, merge mode, resource allocation, and module enable/disable etc.

Parameter effectiveness

- Parameter may take effect dynamically or upon cluster restart. Most are dynamic effect

Parameter level

- Parameters are classified into cluster level and tenant level, majority are cluster level parameters

admin permission

- System tenant can view and set parameters for all other tenants (including the SYS tenant). A user created tenant can set parameters only for its own

After the OBServer is started, if no parameter is specified, the default value specified by the system take effect. After OBServer process successfully started, parameter value persists in file /home/admin/oceanbase/etc/observer.config.bin, which can be viewed through Linux strings command

View cluster parameter

Syntax

view cluster parameter from system tenant :

show parameters [SHOW_PARAM_OPTS] [tenant='tenant’]

View cluster parameter from user created tenant:

show parameters [SHOW_PARAM_OPTS]

- [SHOW_PARAM_OPTS] : **Value can be specified as** [LIKE 'pattern' | WHERE expr
- [tenant='tenant'] : **Tenant name is required when view cluster parameter from system tenant**

Example

System tenant

- show parameters like 'sql_work_area' tenant=t1
- show parameters where edit_level='static_effective' and name='sql_work_area' tenant=t1;

User created tenant

- show parameters like 'sql_work_area'

Output definition

Column	**Meaning**
Zone	zone
svr_ip	Machine IP
svr_port	Machine port number
name	Configuration item name
value	Configuration item value
data_type	Configuration item data type (NUMBER,STRING,CAPACITY...)
info	Configuration item explanation (describing meaning and value range of the Parameter)
scope	Configuration item range attribute (Tenant|/Cluster)
source	Value source (Tenant|Cluster|CommandLine|ObAdmin|File)
edit_level	dynamic_effective (dynamically effective) / static_effective (effective upon restart)

OCEANBASE

Cluster parameter settings

Syntax

The syntax of modifying cluster parameters is as below. When modifying multiple system configuration items at the same time, separate them with commas (,):

```
ALTER SYSTEM SET param_name = expr
[COMMENT 'text’]
[PARAM_OPTS]
[TENANT = 'tenantname’]

PARAM_OPTS:
[ZONE='zone' | SERVER='server_ip: rpc_port’]
```

If no option is specified, values on all OBserver will be modified.

Sample

cluster config setup by sys tenant

```
alter system set sql_port=8888 –
```

tenant config setup by sys tenant

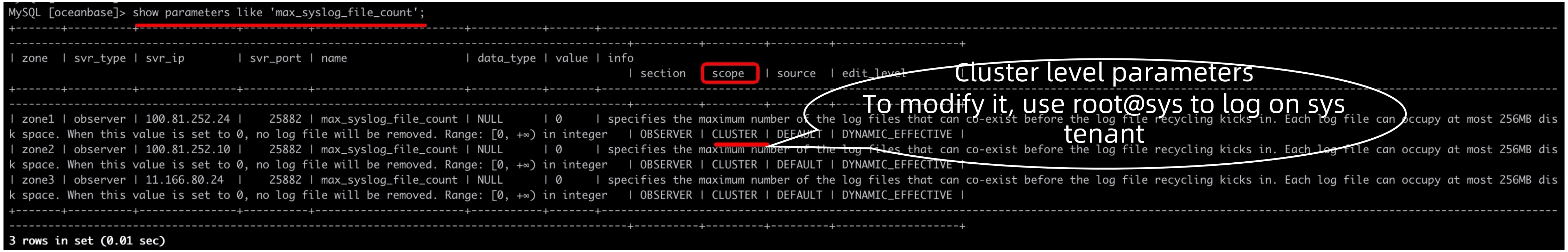
```
alter system set sql_work_area='1G' tenant='test_tenant’
alter system set max_syslog_file_count=100 zone='ZONE_1’
ALTER SYSTEM SET max_syslog_file_count = 100 server='100.88.8.54: 2882’;
```

tenant config setup by user created tenant

```
alter system set sql_work_area='2G’
alter system set memory_limit = '100G' SERVER='192.168.100.1: 2882’;
alter system set memory_limit = '100G' ZONE='z1';
```

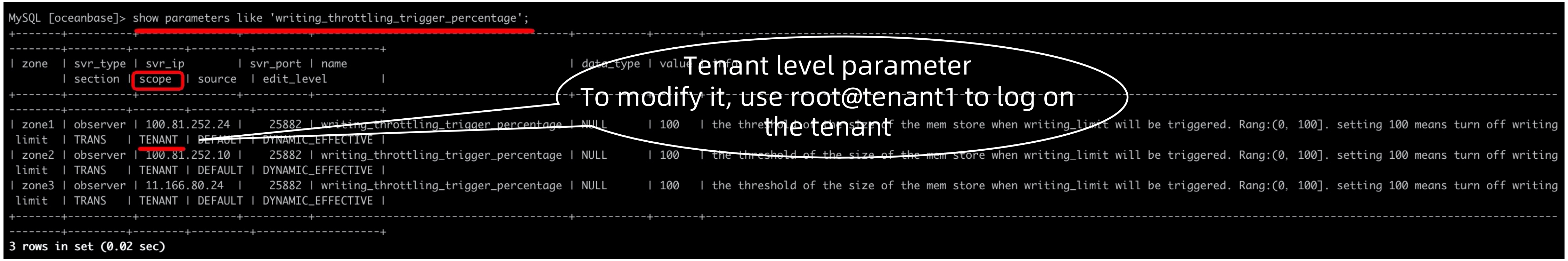
- **ALTER SYSTEM command cannot specify Zone and Server option at the same time. when specifying Zone, only designate one zone; same rule apply to config Server**
- **Cluster-level configuration item (Scope) cannot be set up by user created tenant, nor by created tenant designated by sys tenant. For example,** *alter system set memory_limit=‘100G’ tenant=‘test_tenant’* **causes an error, because memory_limit is a cluster level (scope) configuration item**

OBServer cluster level and tenant level parameters



```
MySQL [oceanbase]> show parameters like 'max_syslog_file_count';
+-------+----------+----------------+----------+-----------------------+-----------+-------+----------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------+----------+---------+---------+-------------------+
| zone  | svr_type | svr_ip         | svr_port | name                  | data_type | value | info
                                                                                                 | section  | scope   | source  | edit_level        |
+-------+----------+----------------+----------+-----------------------+-----------+-------+----------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------+----------+---------+---------+-------------------+
| zone1 | observer | 100.81.252.24 |    25882 | max_syslog_file_count | NULL      | 0     | specifies the maximum number of the log files that can co-exist before the log file recycling kicks in. Each log file can occupy at most 256MB dis
k space. When this value is set to 0, no log file will be removed. Range: [0, +∞) in integer   | OBSERVER | CLUSTER | DEFAULT | DYNAMIC_EFFECTIVE |
| zone2 | observer | 100.81.252.10 |    25882 | max_syslog_file_count | NULL      | 0     | specifies the maximum number of the log files that can co-exist before the log file recycling kicks in. Each log file can occupy at most 256MB dis
k space. When this value is set to 0, no log file will be removed. Range: [0, +∞) in integer   | OBSERVER | CLUSTER | DEFAULT | DYNAMIC_EFFECTIVE |
| zone3 | observer | 11.166.80.24  |    25882 | max_syslog_file_count | NULL      | 0     | specifies the maximum number of the log files that can co-exist before the log file recycling kicks in. Each log file can occupy at most 256MB dis
k space. When this value is set to 0, no log file will be removed. Range: [0, +∞) in integer   | OBSERVER | CLUSTER | DEFAULT | DYNAMIC_EFFECTIVE |
+-------+----------+----------------+----------+-----------------------+-----------+-------+----------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------+----------+---------+---------+-------------------+
3 rows in set (0.01 sec)
```

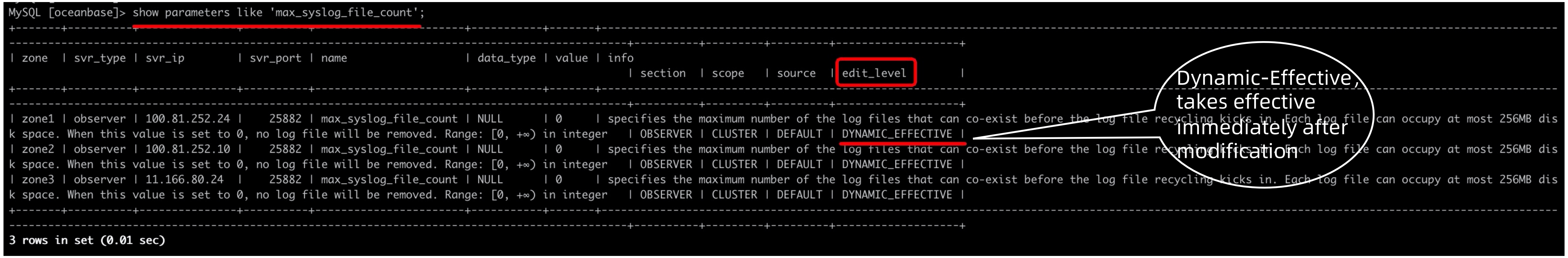
Cluster level parameters
To modify it, use root@sys to log on sys tenant



```
MySQL [oceanbase]> show parameters like 'writing_throttling_trigger_percentage';
+-------+----------+----------------+----------+---------------------------------------+-----------+-------+--------------------------------------------------------------------------------------------------------------------------
--------+--------+---------+---------+-------------------+
| zone  | svr_type | svr_ip         | svr_port | name                                  | data_type | value | info
        | section | scope  | source  | edit_level        |
+-------+----------+----------------+----------+---------------------------------------+-----------+-------+--------------------------------------------------------------------------------------------------------------------------
--------+--------+---------+---------+-------------------+
| zone1 | observer | 100.81.252.24 |    25882 | writing_throttling_trigger_percentage | NULL      | 100   | the threshold of the size of the mem store when writing_limit will be triggered. Rang:(0, 100]. setting 100 means turn off writing
 limit  | TRANS    | TENANT | DEFAULT | DYNAMIC_EFFECTIVE |
| zone2 | observer | 100.81.252.10 |    25882 | writing_throttling_trigger_percentage | NULL      | 100   | the threshold of the size of the mem store when writing_limit will be triggered. Rang:(0, 100]. setting 100 means turn off writing
 limit  | TRANS    | TENANT | DEFAULT | DYNAMIC_EFFECTIVE |
| zone3 | observer | 11.166.80.24  |    25882 | writing_throttling_trigger_percentage | NULL      | 100   | the threshold of the size of the mem store when writing_limit will be triggered. Rang:(0, 100]. setting 100 means turn off writing
 limit  | TRANS    | TENANT | DEFAULT | DYNAMIC_EFFECTIVE |
+-------+----------+----------------+----------+---------------------------------------+-----------+-------+--------------------------------------------------------------------------------------------------------------------------
--------+--------+---------+---------+-------------------+
3 rows in set (0.02 sec)
```

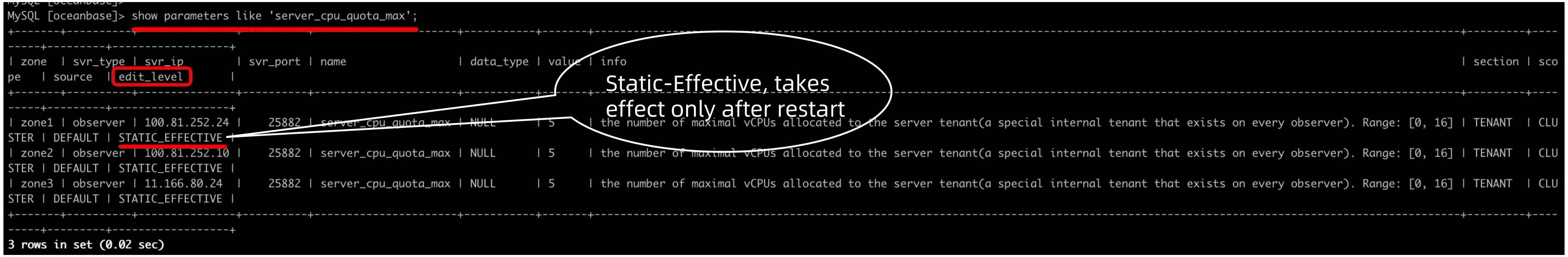
Tenant level parameter
To modify it, use root@tenant1 to log on the tenant

OBServer parameters that take effect dynamically or statically



```
MySQL [oceanbase]> show parameters like 'max_syslog_file_count';
+-------+----------+----------------+----------+-----------------------+-----------+-------+--------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------+----------+---------+---------+-------------------+
| zone  | svr_type | svr_ip         | svr_port | name                  | data_type | value | info
                                                                                                                        | section  | scope   | source  | edit_level        |
+-------+----------+----------------+----------+-----------------------+-----------+-------+--------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------+----------+---------+---------+-------------------+
| zone1 | observer | 100.81.252.24 |    25882 | max_syslog_file_count | NULL      | 0     | specifies the maximum number of the log files that can co-exist before the log file recycling kicks in. Each log file can occupy at most 256MB dis
k space. When this value is set to 0, no log file will be removed. Range: [0, +∞) in integer    | OBSERVER | CLUSTER | DEFAULT | DYNAMIC_EFFECTIVE |
| zone2 | observer | 100.81.252.10 |    25882 | max_syslog_file_count | NULL      | 0     | specifies the maximum number of the log files that can co-exist before the log file recycling kicks in. Each log file can occupy at most 256MB dis
k space. When this value is set to 0, no log file will be removed. Range: [0, +∞) in integer    | OBSERVER | CLUSTER | DEFAULT | DYNAMIC_EFFECTIVE |
| zone3 | observer | 11.166.80.24  |    25882 | max_syslog_file_count | NULL      | 0     | specifies the maximum number of the log files that can co-exist before the log file recycling kicks in. Each log file can occupy at most 256MB dis
k space. When this value is set to 0, no log file will be removed. Range: [0, +∞) in integer    | OBSERVER | CLUSTER | DEFAULT | DYNAMIC_EFFECTIVE |
+-------+----------+----------------+----------+-----------------------+-----------+-------+--------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------+----------+---------+---------+-------------------+
3 rows in set (0.01 sec)
```

Dynamic-Effective , takes effective immediately after modification



```
MySQL [oceanbase]> show parameters like 'server_cpu_quota_max';
+-------+----------+----------------+----------+----------------------+-----------+-------+------------------------------------------------------------------------------------------------------------------------------------------+---------+----
-----+---------+-------------------+
| zone  | svr_type | svr_ip         | svr_port | name                 | data_type | value | info                                                                                                                                     | section | sco
pe   | source  | edit_level        |
+-------+----------+----------------+----------+----------------------+-----------+-------+------------------------------------------------------------------------------------------------------------------------------------------+---------+----
-----+---------+-------------------+
| zone1 | observer | 100.81.252.24 |    25882 | server_cpu_quota_max | NULL      | 5     | the number of maximal vCPUs allocated to the server tenant(a special internal tenant that exists on every observer). Range: [0, 16] | TENANT  | CLU
STER | DEFAULT | STATIC_EFFECTIVE |
| zone2 | observer | 100.81.252.10 |    25882 | server_cpu_quota_max | NULL      | 5     | the number of maximal vCPUs allocated to the server tenant(a special internal tenant that exists on every observer). Range: [0, 16] | TENANT  | CLU
STER | DEFAULT | STATIC_EFFECTIVE |
| zone3 | observer | 11.166.80.24  |    25882 | server_cpu_quota_max | NULL      | 5     | the number of maximal vCPUs allocated to the server tenant(a special internal tenant that exists on every observer). Range: [0, 16] | TENANT  | CLU
STER | DEFAULT | STATIC_EFFECTIVE |
+-------+----------+----------------+----------+----------------------+-----------+-------+------------------------------------------------------------------------------------------------------------------------------------------+---------+----
-----+---------+-------------------+
3 rows in set (0.02 sec)
```

Static-Effective, takes effect only after restart

Common OB system configuration items (major freeze related)

Configuration item	Default	Explanation
zone_merge_timeout	3h	➢ Major freeze operation timeout of single zone ➢ range: [1s, +∞)
freeze_trigger_percentage	70	➢ memstore usage percentage when triggering major freeze ➢ range: （0， 100）
enable_manual_merge	FALSE	➢ Enable/disable manual merge ➢ • True: yes • False: no ➢ If the value is True, DBA disables major freeze operation
major_freeze_duty_time	02: 00	➢ Scheduled time of daily major freeze task ➢ Range: [00: 00, 24: 00)

Common OB system configuration items (syslog related)

Configuration item	Default	Explanation
syslog_level	INFO	➢ Log level: DEBUG, TRACE, INFO, WARN, ERROR
enable_syslog_recycle	FALSE	➢ Turn on switch for automatic log control or not, working with max_syslog_file_count to take effect
max_syslog_file_count	0	➢ Specify maximum log files that can exist once ➢ Each log file occupies 256M in size. ➢ When value is 0, no deletion occur. Range: [0, +∞)
trace_log_slow_query_watermark	100ms	➢ Print the trace log slow query control threshold. ➢ Range: [1ms,+∞)
syslog_io_bandwidth_limit	30MB	➢ System log (syslog) IO bandwidth limit
enable_syslog_wf	TRUE	➢ Separately save logs above Warning level to files or not

Common OB system configuration items (memory related)

Configuration item	Default	Explanation
minor_freeze_times	0	➢ Upper limit of dumps between two merges
large_query_threshold	100ms	➢ Large query decision condition ➢ Range: [1ms, +∞)
large_query_worker_percentage	30	➢ Large query resource allocation percentage ➢ Range: [0, 100] in percentage
memory_limit_percentage	80	➢ Upper limit of memory usage ➢ Range: [10， 90]

Common OB system configuration items (others)

Configuration item	Default	Explanation
server_permanent_offline_time	3600s	➢ Duration of the server going offline permanently ➢ Range: [20s,+∞)
enable_auto_leader_switch	TRUE	➢ Allow proactive system primary switchover
clog_sync_warn_threshold	100ms	➢ Report WARN log upon commitlog synchronization timeout ➢ Range: [1ms,1000ms]
enable_sql_audit	TRUE	➢ Enable SQL audit function or not, default TRUE indicating SQL audit function enabled ➢ Range TRUE,FALSE
sql_audit_memory_limit	10% memory	➢ When SQL audit function is enabled, maximum memory available for internal SQL audit table, default is 10% memory. Range: [64M, +∞)

Example: OceanBase 11-11 sales promotions parameter adjustment practice

- **Manual merge completed before 0 o’clock**
- **Adjust parameters**
  - Turn on minor feeze (above 10 times)
  - Turn off sql_audit
  - Reduce operation logIO
- **support memory only access in the first hour of 11-11 event**

OCEANBASE

Variables, apps tenant related

Controls the global level or session level attributes of a tenant, majority are dynamic effectiveness, minority take effect upon session re-connect

Check variable values

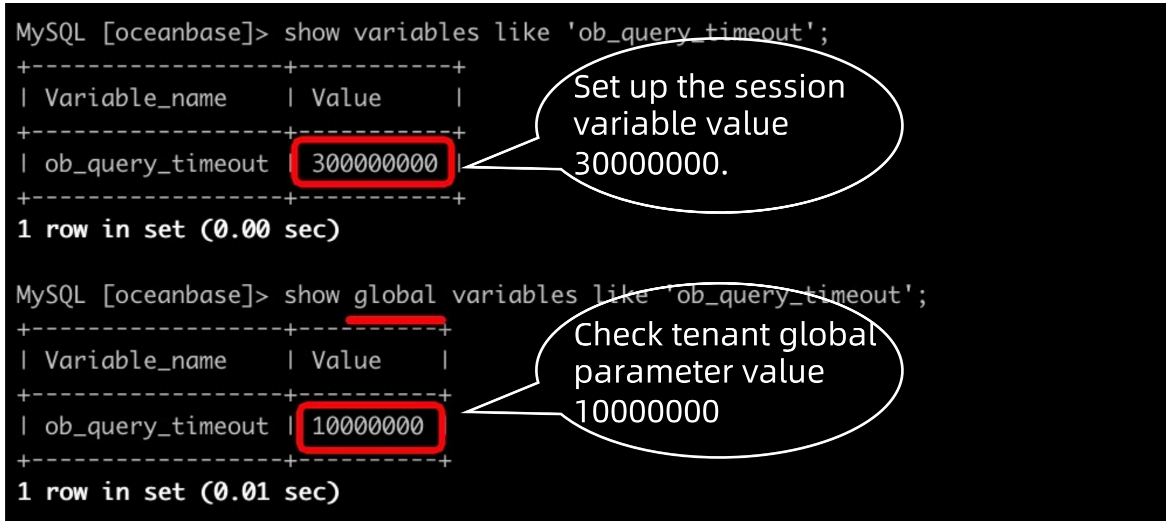
- *show variables;*
- *show variables like '%<pattern>%';*

Modify variables

- *set @@session.<name> = <value>;*
- *set @@global.<name> = <value>;*

Session variables vs Global variables

- **Session variables:** Session-level modified (only for this Session). When client connects to the database, the DB session inherits global variable value. Change made to session variable take effect only for current session.
  - set ob_trx_timeout = 200000000
- **Global variables:** Global level (tenant level) modified (not expired with session disconnection). Tenants are sharing Global variables, which means that these global variables are shared by different users, and tenant stores the changes you made to global variables, which are still valid when you disconnect and re-connect the tenant.
  - set global ob_trx_timeout = 200000000
  - Do not affect connected sessions
  - Affect new sessions



```
MySQL [oceanbase]> show variables like 'ob_query_timeout';
+------------------+-----------+
| Variable_name    | Value     |
+------------------+-----------+
| ob_query_timeout | 300000000 |
+------------------+-----------+
1 row in set (0.00 sec)

MySQL [oceanbase]> show global variables like 'ob_query_timeout';
+------------------+----------+
| Variable_name    | Value    |
+------------------+----------+
| ob_query_timeout | 10000000 |
+------------------+----------+
1 row in set (0.01 sec)
```

Set up the session variable value 30000000.

Check tenant global parameter value 10000000

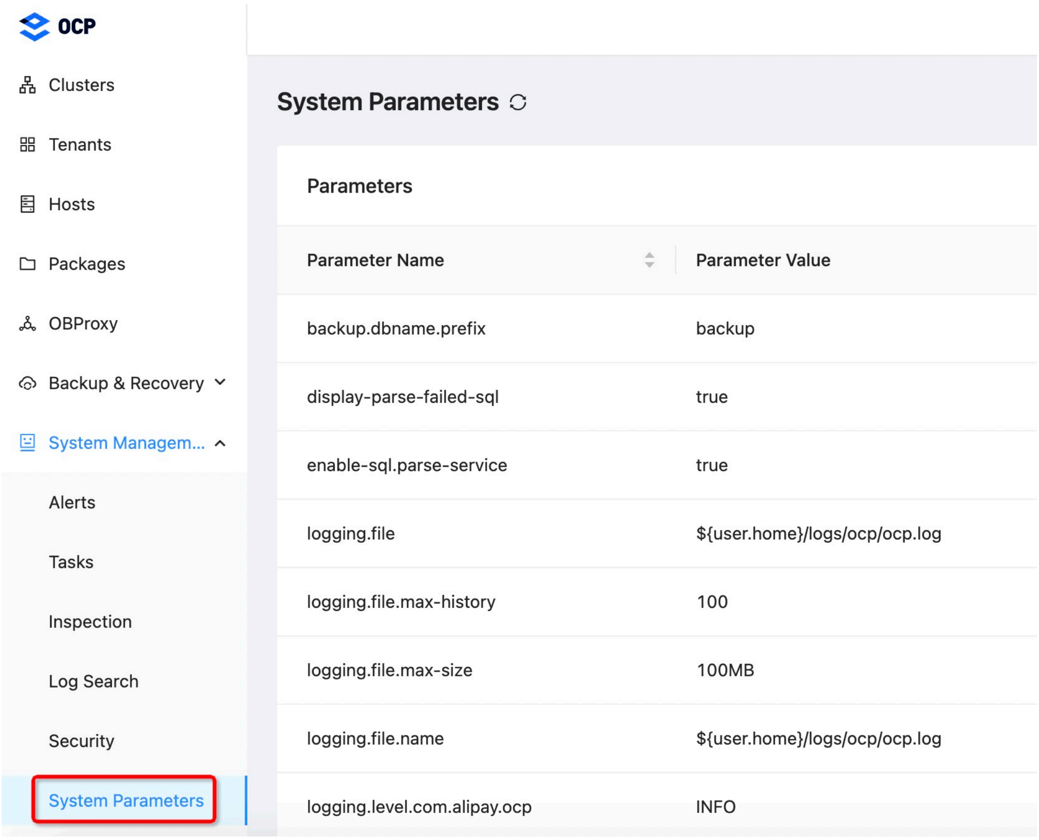
Common OB system variables

Configuration item	Default	Explanation	Attributes
ob_query_timeout	10000000	Query timeout	GLOBAL | SESSION
ob_trx_timeout	100000000	Transaction timeout	GLOBAL | SESSION
ob_read_consistency	STRONG	Read consistency level	GLOBAL | SESSION
ob_enable_truncate_flashback	ON	truncated table entering recycle bin or not	GLOBAL | SESSION
lower_case_table_names	1	Case insensitive 1: Case insensitive 0: Case sensitive	GLOBAL | SESSION | READONLY

- **ob_timestamp_service: Similar to the TrueTime of Google Spanner, through which OB implements global consistency on cluster tenant level**
  - set global ob_timestamp_service='GTS'
  - set global ob_timestamp_service='LTS'

OCEANBASE

ODC management session function supports checking and modifying session attributes



System Parameters

Parameters

Parameter Name	Parameter Value
backup.dbname.prefix	backup
display-parse-failed-sql	true
enable-sql.parse-service	true
logging.file	${user.home}/logs/ocp/ocp.log
logging.file.max-history	100
logging.file.max-size	100MB
logging.file.name	${user.home}/logs/ocp/ocp.log
logging.level.com.alipay.ocp	INFO

ODC provides a GUI interface, you can clearly view and modify supported variables. Make changes to variables :

- If the value of a variable is a character or number type, you can enter the modified value directly on variable editing page
- If the value of a variable is an enumeration type, ODC will list the set of values supported by the variable in the dropdown box, then user can modify it through GUI interface without memorizing the name and value of the variable, which reduces the cost and improves efficiency of variable modification
- ODC marks the modified database variables by orange color, manage variables intuitively.

Summary

OceanBase parameters and variables can be viewed and modified through commands, to meet the special requirements of different tenants or sessions.

- Parameters are on cluster level or tenant level, and take effect dynamically or upon restart
- Use *show parameters like '%<pattern>%'* to check parameter; Use *alter system set <name> = <value>* to modify parameter
- Variables are on the session level or global level. Session variables are modified on the session level, which takes effect on the session only; global variables are modified on the global level, which does not become ineffective due to session exit.
- For variables, use *show variables like '%<pattern>%'* to check the parameters, and use *set @@session.<name> = <value>;set @@global.<name> = <value>* to modify the parameters.

Q&A

Copyright © Beijing OceanBase Technology

Simulation Questions

1.[True/false] Session variables take effect only for the current session and do not affect other sessions of the tenant. ()

2. [Single choice] Which command can be used to query the properties of a parameter? ()

A: show parameters like '%<pattern>%’; B: alter system set <name> = <value>;

C: show variables like '%<pattern>%’; D: set @@global.<name> = <value>;

3. [Multiple choice] What are the two levels of parameter? ()

A: cluster level B: Zone level C: OB Server level D: tenant level

Chapter 7: introduction to OCP tools

OCEANBASE

OCP (OceanBase Cloud Platform) is enterprise -level database management platform

The OceanBase Cloud Platform (OCP) is an enterprise-class database management Platform. IT provides life-cycle management services not only for The OceanBase cluster and tenant components, but also for oceanBase-related resources (such as hosts, networks, and software packages), enabling DBA to manage the OceanBase cluster more efficiently and reducing cost of enterprise IT operation and maintenance

Effective OceanBase cluster management

- Supports high availability
- Second-level response time

Visual experience

- "Task Details" provides flow charts and natural/flexible canvas interaction, making task progress management more intuitive and efficient
- Newly added topology diagram of clusters and tenants to show distributed business logic, display running status, and key data

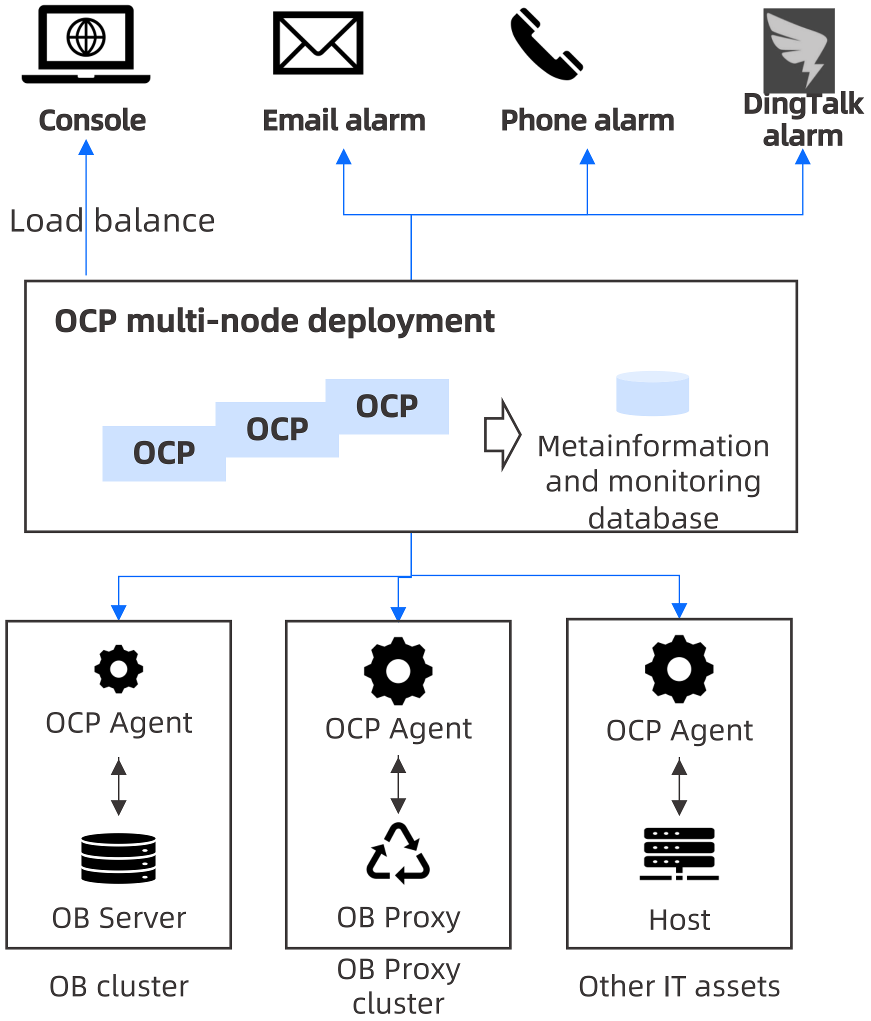
Smooth and effective menu navigation

- The task process of the core jobs are designed in serializable mode, which is clear and without breakpoints
- Provides shortcuts for O&M tasks. Quickly switch to tasks on any page

Professional functionality

- Provides full life cycle management for OceanBase clusters
- Specialized management functions designed for OceanBase
- Progressive management from cluster to session
- Well-designed topology function

OCP product architecture and function, one-stop management operation and maintenance tool



Product architecture

- install OCP Agent on mgt. target . The OCP manages and monitors mgt target through Agent
- OCP provides management, monitoring, and alarm functions for the administrator
- Each OCP node contains complete functions. A single node provide overall OCP capabilities. When an OCP node is unavailable, automatically switch over to available OCP node
- OCP Server supports multi-node deployment and implements load balancing using DNS, HAProxy, Nginx, or F5 to ensure high availability

Dependent software/hardware resources

- OCP Server can be installed on a dedicated physical machine or in a Docker container
- The X86 OCP Server supported OS contains RHEL, CentOS, AliOS, and OpenSUSE. It also supports AliOS, Kirin, and Huawei EulerOS operating systems based on the ARM architecture
- OCP-agent consumes less resources and has no special requirement on hardware
- Clients use Web browsers to access OCP service, such as Chrome, Firefox, Safari, and Edge

OCP core functions

01

Cluster management

Provides life-cycle management, including installation, o&M, performance monitoring, configuration, upgrade, and deletion

02

Host management

Add a host, delete a host, and display key host information

03

Tenant management

Tenant creation, tenant topology, performance monitoring, session management, and parameter management

04

Alarm management

Supports alarms of different dimensions, such as clusters, tenants, and hosts. The system generates alarms based on alarm rules

05

Backup/recovery management

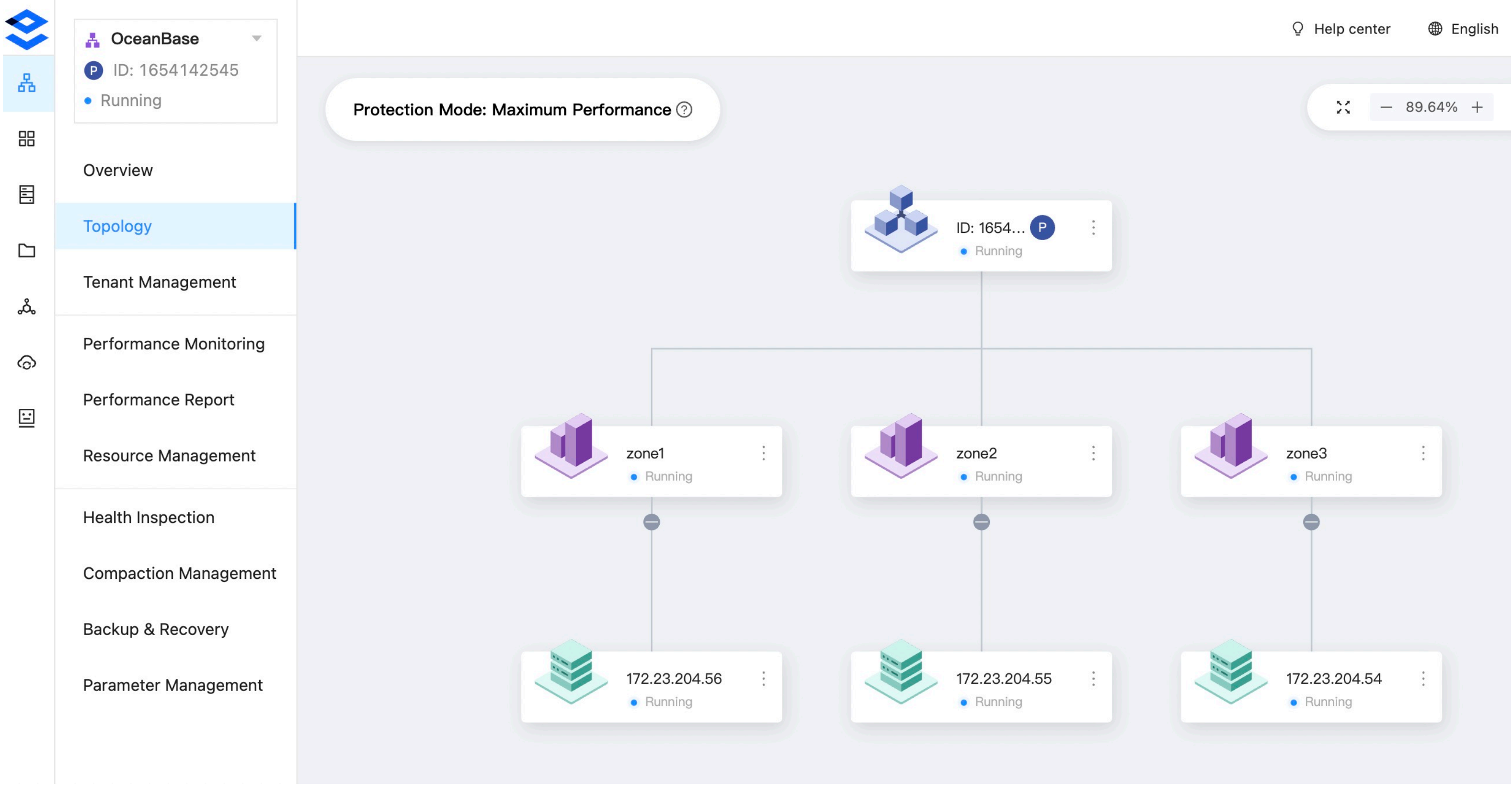
Supports full backup, incremental backup, redo-log backup, full recovery, and incomplete recovery for the OceanBase cluster and tenant level

06

User and permission management

Manage users and roles to ensure system security

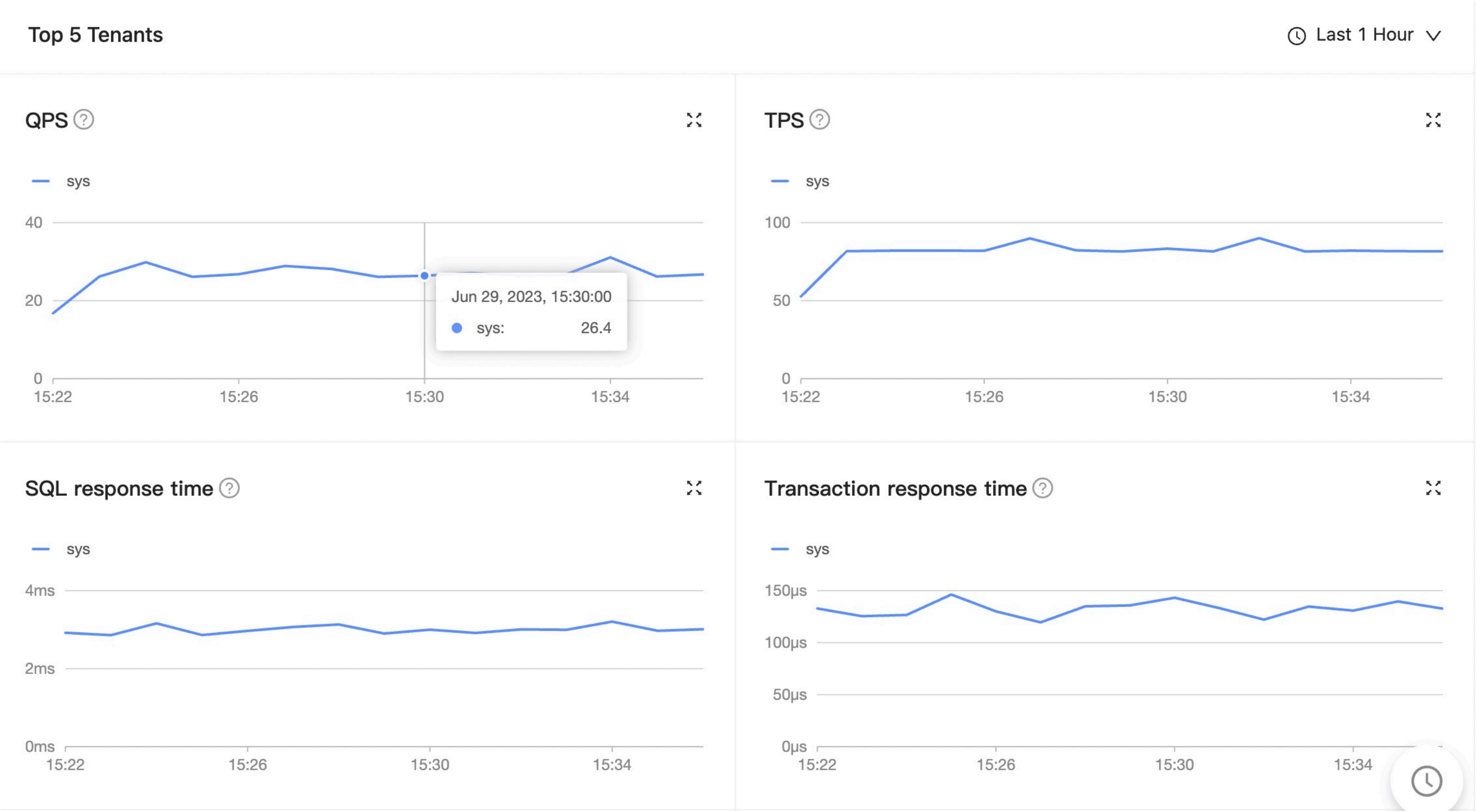
Cluster management – Cluster topology



Protection Mode: Maximum Performance

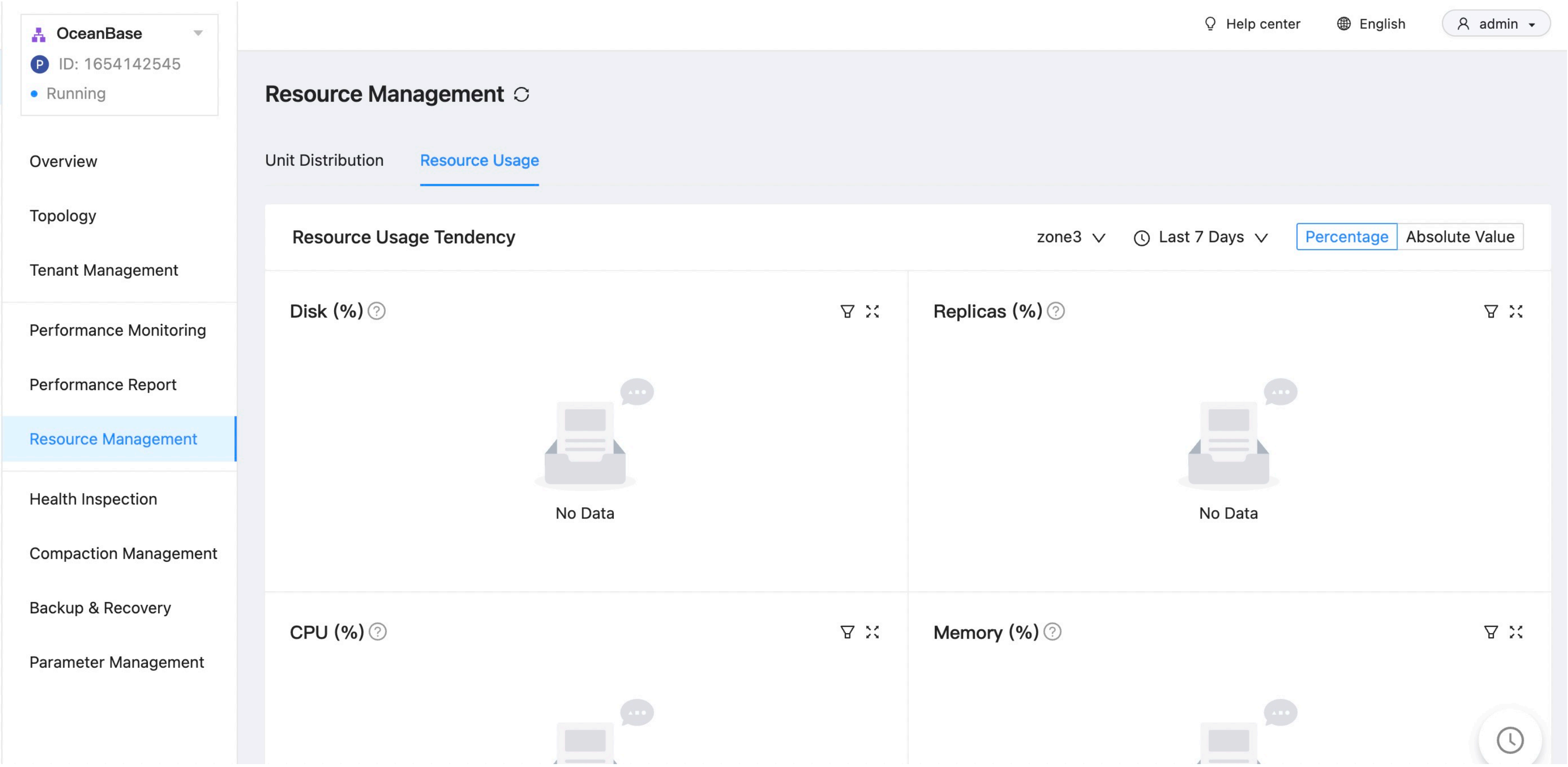
- Cluster: displays the Cluster ID, cluster type (active or standby), and cluster status. The standby DB delay time is also displayed on top of the standby cluster icon
- Zone: displays the Zone name and current status of the Zone, including QPS, no. of connections, no. of Unit etc.
- Server: displays the IP address and current status of the server, including QPS, no. of connections, and SQL port

Cluster management – tenant management



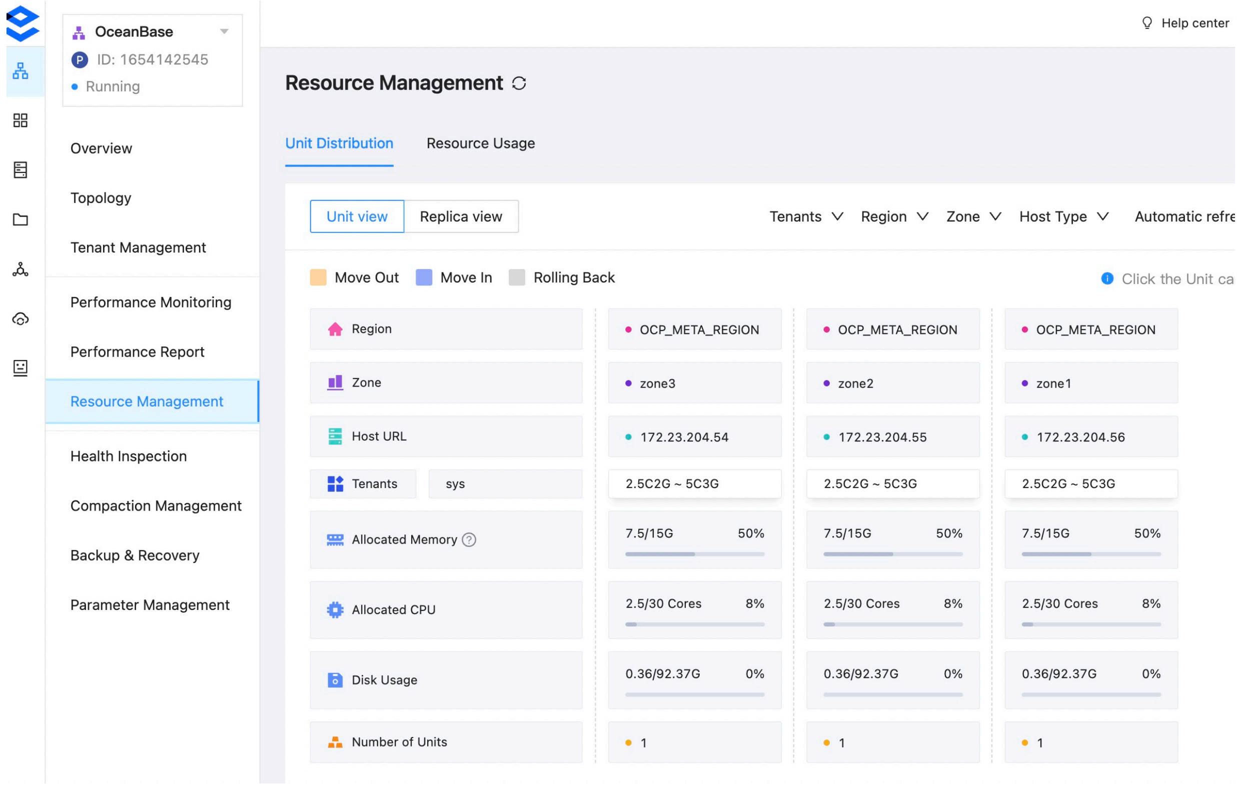
- View the performance information of TOP-5 tenants by highest workload in the current cluster, including TPS, QPS, SQL response time, transaction response time, number of active sessions, event waits_number, event waits_time, capacity_no. of tables, and capacity_no. of partitions
- View the monitoring information generated within the last hour, day, or week based on business requirement

Cluster management - resource management (1)



- View trending chart for disks, partition replicas, CPU, and memory
- View resources by Zone

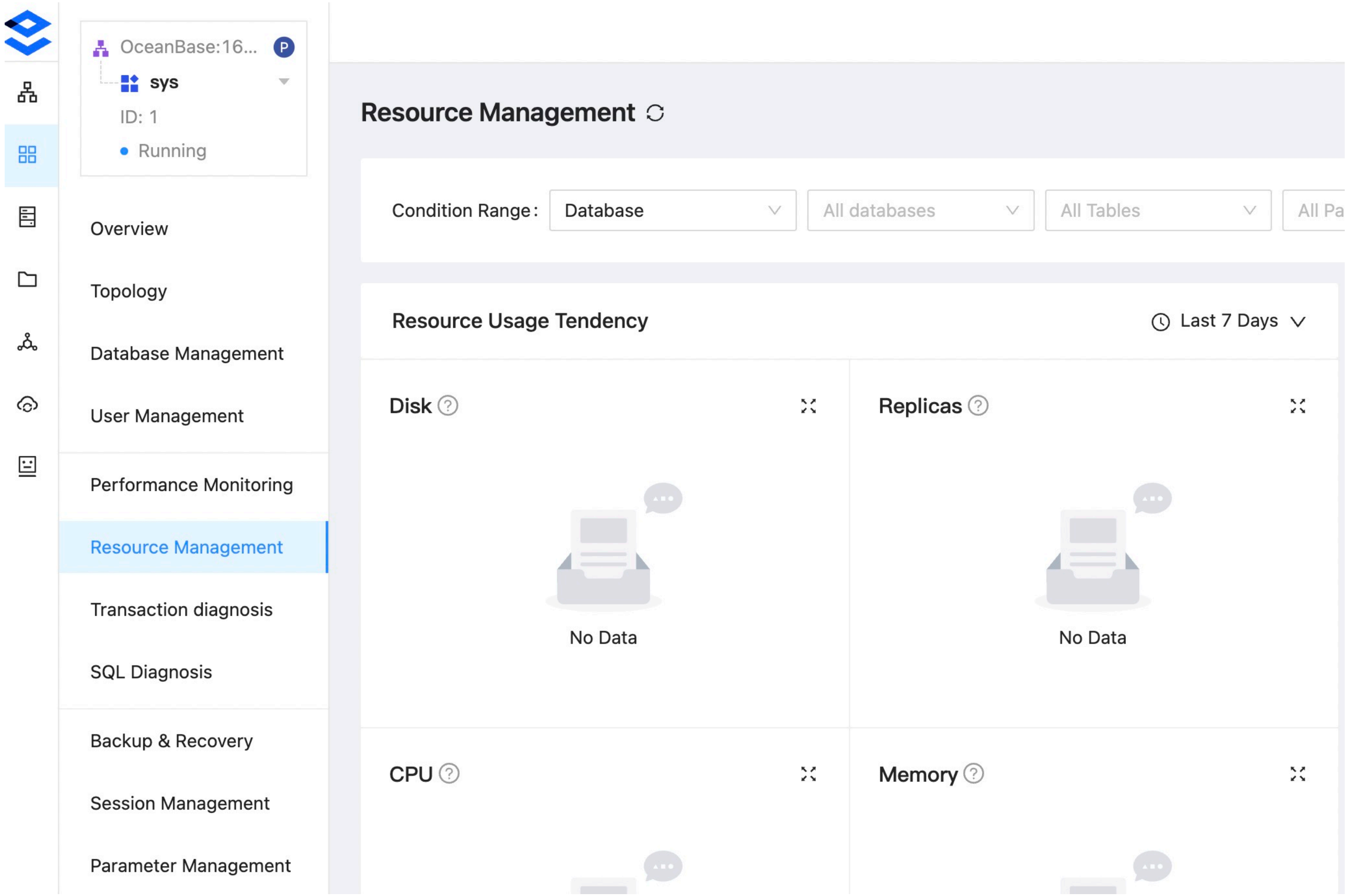
Cluster management - resource management（2）



Region		OCP_META_REGION	OCP_META_REGION	OCP_META_REGION
Zone		zone3	zone2	zone1
Host URL		172.23.204.54	172.23.204.55	172.23.204.56
Tenants	sys	2.5C2G ~ 5C3G	2.5C2G ~ 5C3G	2.5C2G ~ 5C3G
Allocated Memory		7.5/15G 50%	7.5/15G 50%	7.5/15G 50%
Allocated CPU		2.5/30 Cores 8%	2.5/30 Cores 8%	2.5/30 Cores 8%
Disk Usage		0.36/92.37G 0%	0.36/92.37G 0%	0.36/92.37G 0%
Number of Units		1	1	1

- View the allocated memory, allocated CPU, and used disk in each Zone to determine whether the usage of the same type of resources on machine in each Zone is basically the same. If the usage is basically the same, then it means workload is balanced
- Determine whether the disk capacity of the cluster is sufficient based on the percentage of used disks. If the disk usage reaches 80% of the disk capacity, expand the disk capacity by adding additional hardware servers to maintain a normal disk capacity

Tenant management - tenant resource management



Resource Management

Condition Range: Database | All databases | All Tables | All Pa

Resource Usage Tendency — Last 7 Days

Disk: No Data

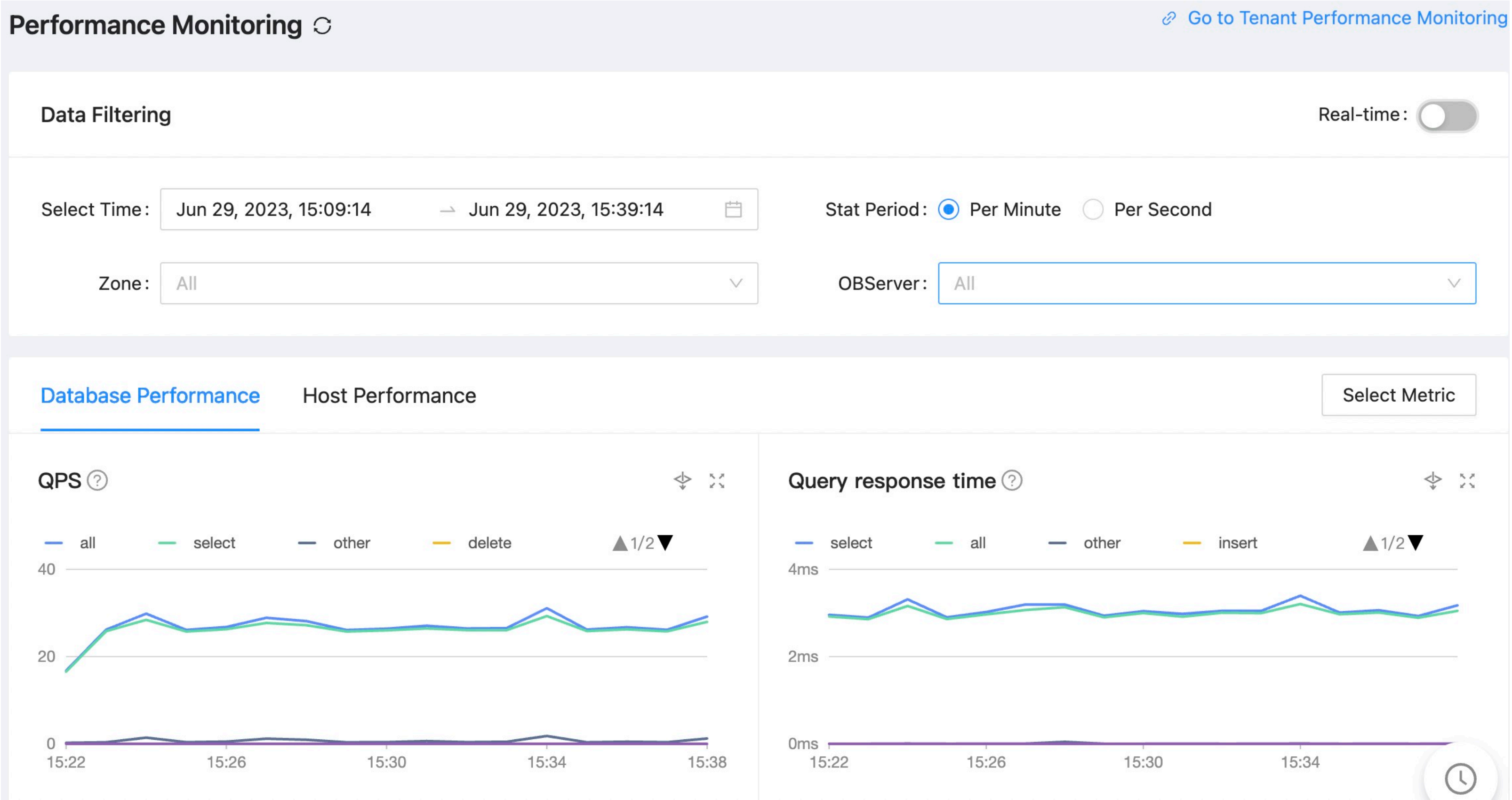
Replicas: No Data

CPU

Memory

- In the resource usage area, you can view information about disks, partition replicas, CPU, and memory in the latest week, 1 month, 6 months, 1 year, or a user-defined time range

Tenant management - tenant performance management



Performance Monitoring

Go to Tenant Performance Monitoring

Data Filtering

Real-time: ○

Select Time: Jun 29, 2023, 15:09:14 → Jun 29, 2023, 15:39:14

Stat Period: ● Per Minute ○ Per Second

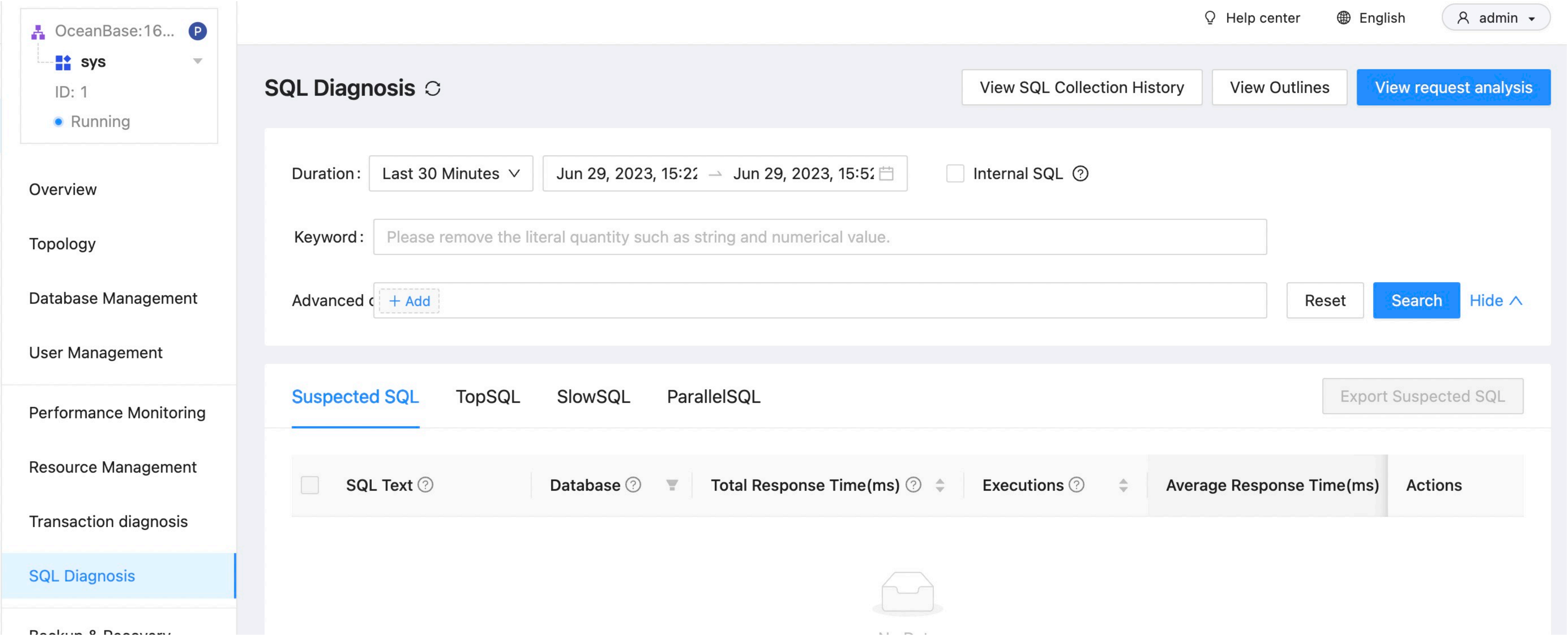
Zone: All

OBServer: All

- Support statistics and monitoring of "performance and SQL”，“Transactions”，“Storage and cache”
- Monitor data by statistical period or real-time
- View data by Zone or OBServer

Tenant management – SQL diagnosis

Diagnose suspicious SQL, TopSQL and SlowSQL to identify risky statements, avoiding business threats



SQL Diagnosis

Duration: Last 30 Minutes | Jun 29, 2023, 15:2: → Jun 29, 2023, 15:5:

□ Internal SQL

Keyword: Please remove the literal quantity such as string and numerical value.

Advanced c + Add

Suspected SQL | TopSQL | SlowSQL | ParallelSQL

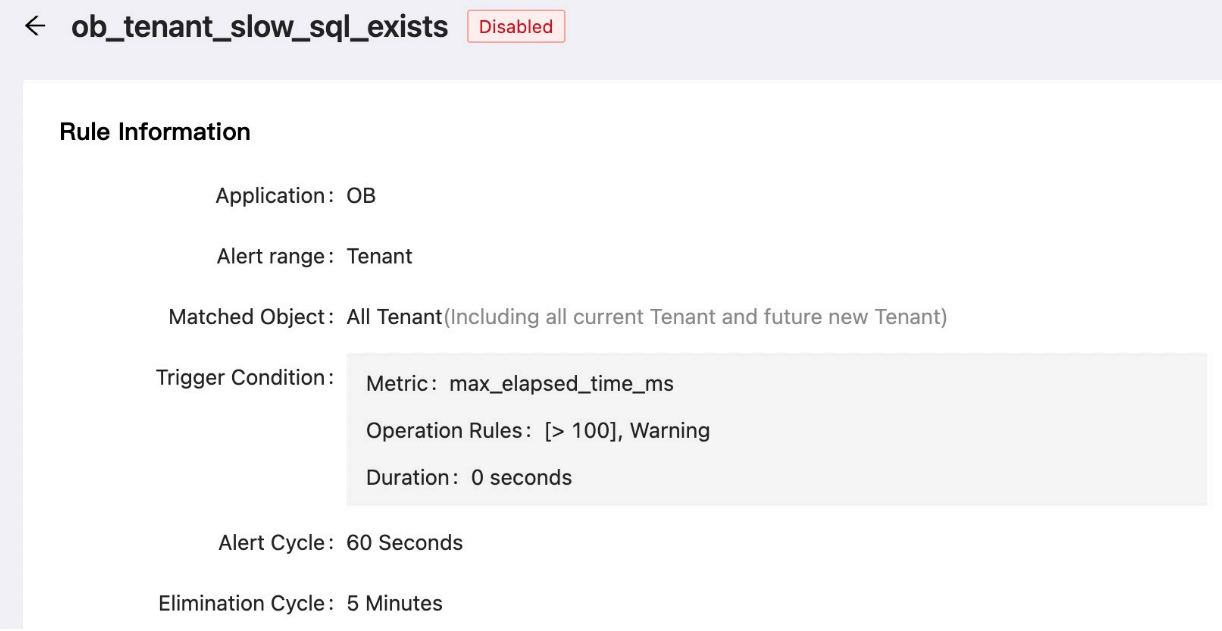
□	SQL Text	Database	Total Response Time(ms)	Executions	Average Response Time(ms)	Actions

Alarm management - OceanBase alarm level

level	Definition	Color	Notes
1	Down	Purple	Completely unavailable, requires immediate action
2	Critical	Red	System availability deteriorates and needs urgent repair to prevent total unavailability, such as memory utilization > 90% for 3 minutes
3	Alert	Orange	The system is still available, but is about to become unavailable. Measures should be taken to prevent the decrease of availability. For example, no. of OB tenant connections exceeds 80% of the upper limit
4	Caution	Blue	According to the trend, the key performance indicator of the system is decreasing, but it has not reached the level triggering a warning. You can find potential problems to avoid alarm
5	Info	Green	operation alert, not an alarm in nature. It is usually an important operation performed by the administrator, for example, a cluster goes offline

- The alarm scope includes OceanBase cluster, OceanBase tenant, service, and server host
- To avoid alarm storm caused by a large number of alarms, configure alarm channel aggregation

Alarm management - Alarm item



← ob_tenant_slow_sql_exists Disabled

Rule Information

Application: OB

Alert range: Tenant

Matched Object: All Tenant(Including all current Tenant and future new Tenant)

Trigger Condition:

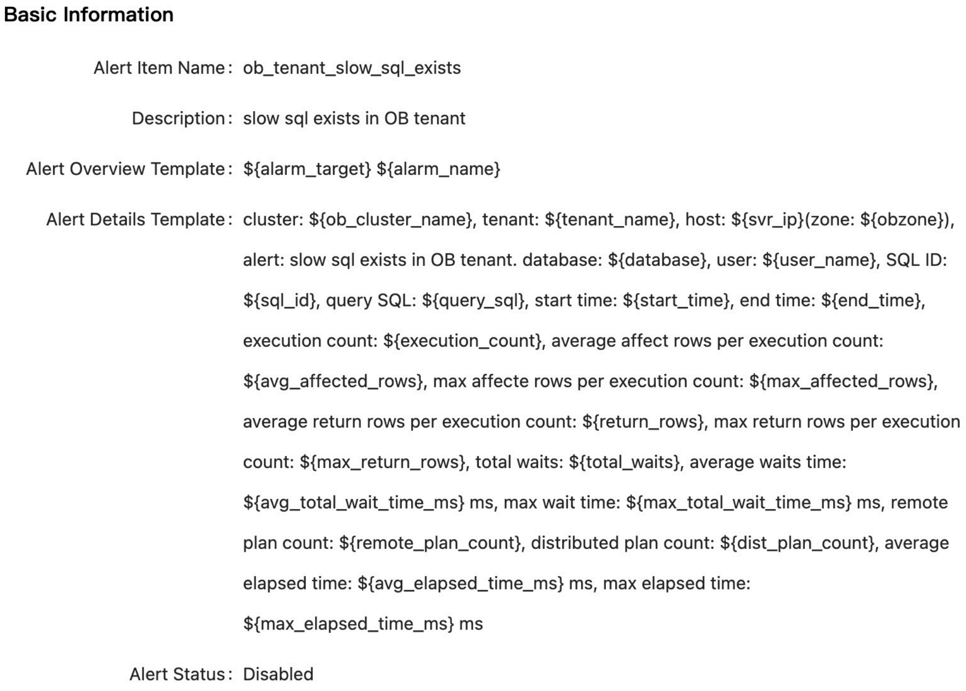
Metric: max_elapsed_time_ms

Operation Rules: [> 100], Warning

Duration: 0 seconds

Alert Cycle: 60 Seconds

Elimination Cycle: 5 Minutes



Basic Information

Alert Item Name: ob_tenant_slow_sql_exists

Description: slow sql exists in OB tenant

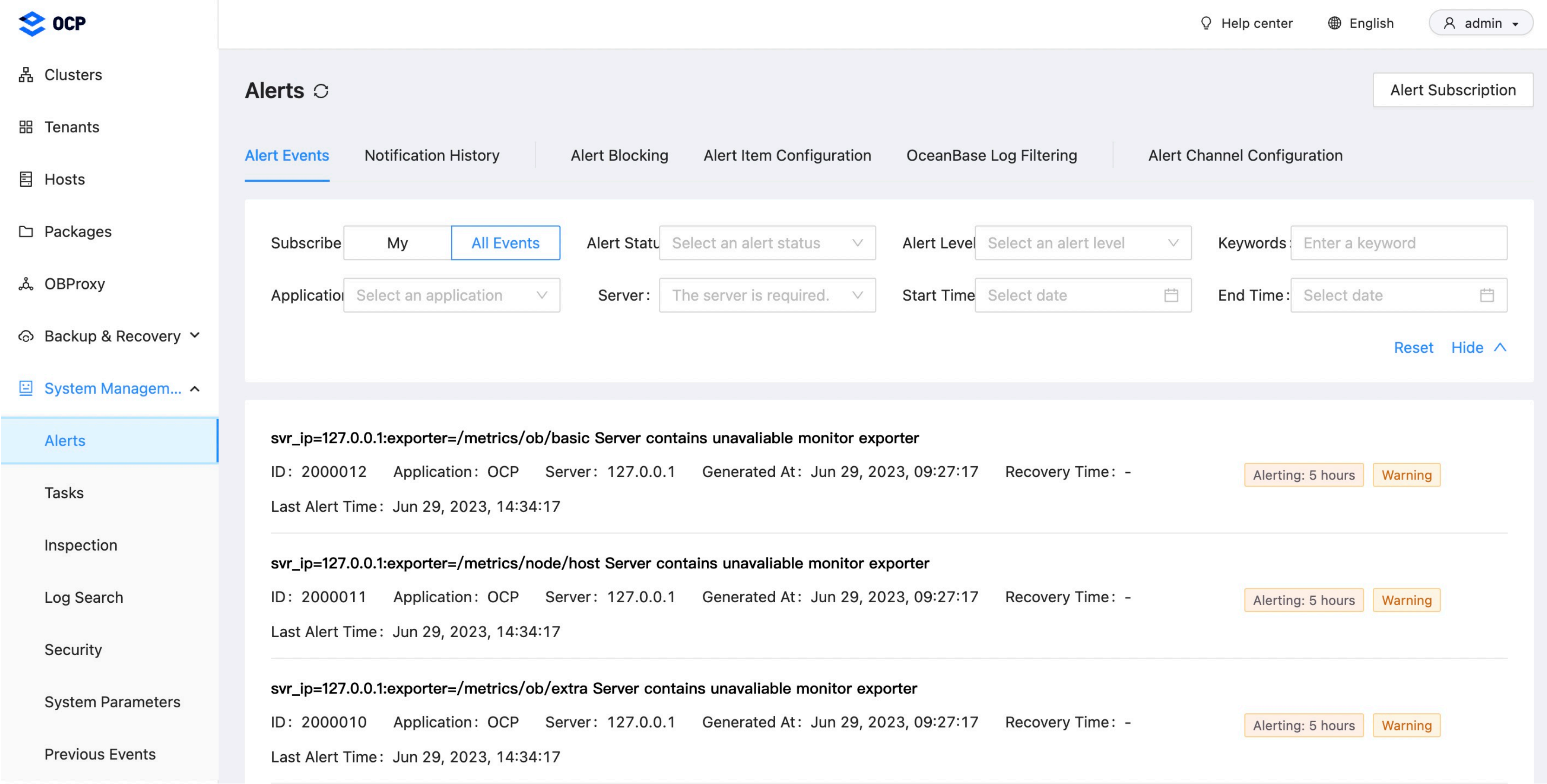
Alert Overview Template: ${alarm_target} ${alarm_name}

Alert Details Template: cluster: ${ob_cluster_name}, tenant: ${tenant_name}, host: ${svr_ip}(zone: ${obzone}), alert: slow sql exists in OB tenant. database: ${database}, user: ${user_name}, SQL ID: ${sql_id}, query SQL: ${query_sql}, start time: ${start_time}, end time: ${end_time}, execution count: ${execution_count}, average affect rows per execution count: ${avg_affected_rows}, max affecte rows per execution count: ${max_affected_rows}, average return rows per execution count: ${return_rows}, max return rows per execution count: ${max_return_rows}, total waits: ${total_waits}, average waits time: ${avg_total_wait_time_ms} ms, max wait time: ${max_total_wait_time_ms} ms, remote plan count: ${remote_plan_count}, distributed plan count: ${dist_plan_count}, average elapsed time: ${avg_elapsed_time_ms} ms, max elapsed time: ${max_elapsed_time_ms} ms

Alert Status: Disabled

Configure the range, trigger, checking period, level and other information of alarms

Alarm management - view alarm event



Subscribe: My | All Events Alert Statu: Select an alert status Alert Level: Select an alert level Keywords: Enter a keyword

Applicatioi: Select an application Server: The server is required. Start Time: Select date End Time: Select date

svr_ip=127.0.0.1:exporter=/metrics/ob/basic Server contains unavaliable monitor exporter

ID: 2000012 Application: OCP Server: 127.0.0.1 Generated At: Jun 29, 2023, 09:27:17 Recovery Time: - Alerting: 5 hours Warning

Last Alert Time: Jun 29, 2023, 14:34:17

svr_ip=127.0.0.1:exporter=/metrics/node/host Server contains unavaliable monitor exporter

ID: 2000011 Application: OCP Server: 127.0.0.1 Generated At: Jun 29, 2023, 09:27:17 Recovery Time: - Alerting: 5 hours Warning

Last Alert Time: Jun 29, 2023, 14:34:17

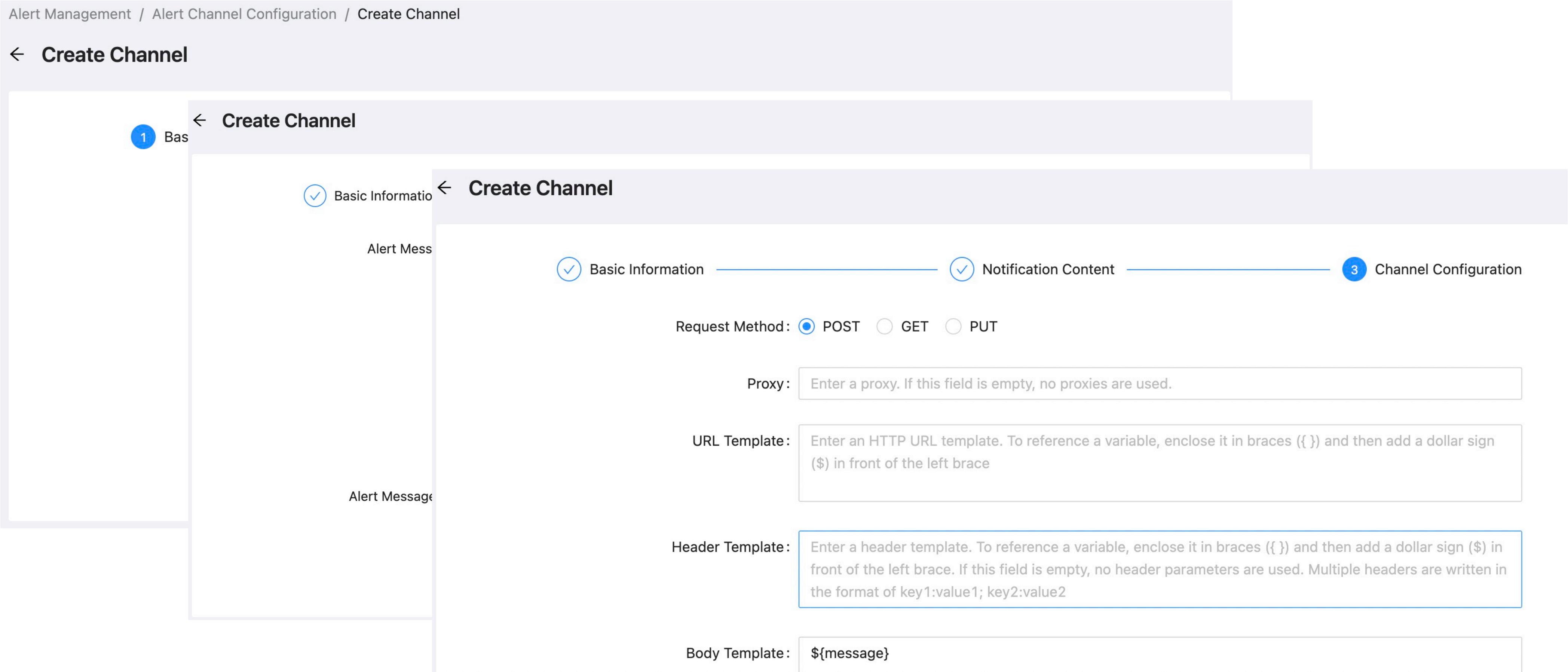
svr_ip=127.0.0.1:exporter=/metrics/ob/extra Server contains unavaliable monitor exporter

ID: 2000010 Application: OCP Server: 127.0.0.1 Generated At: Jun 29, 2023, 09:27:17 Recovery Time: - Alerting: 5 hours Warning

Last Alert Time: Jun 29, 2023, 14:34:17

The alarm event list on the console is used to view and retrieve all alarms. The alarm event list supports rich search criteria. The keyword search matches the alarm overview, alarm detail, and all label values

Alarm management - Alarm channel



Alert Management / Alert Channel Configuration / Create Channel

← Create Channel

← Create Channel

✓ Basic Information — ✓ Notification Content — 3 Channel Configuration

Request Method: ● POST ○ GET ○ PUT

Proxy: Enter a proxy. If this field is empty, no proxies are used.

URL Template: Enter an HTTP URL template. To reference a variable, enclose it in braces ({ }) and then add a dollar sign ($) in front of the left brace

Header Template: Enter a header template. To reference a variable, enclose it in braces ({ }) and then add a dollar sign ($) in front of the left brace. If this field is empty, no header parameters are used. Multiple headers are written in the format of key1:value1; key2:value2

Body Template: ${message}

Alarms are independent functions. If no channel or subscription is configured, you can view alarms only on the alarm event page of the console. By configuring alarm channels and alarm subscription, users can receive alarm notification messages

Q&A

Simulation Questions

1. [Single choice] Which of the following components provides a graphical user interface (GUI) for cluster management, tenant management, and alarm monitoring? ()

A: ODC developer center B: OCP cloud management platform

C: OB Proxy D: OB Server

Chapter 8: Simulation question answers

Chapter 1 Simulation question answers

1. [True/false] Although the architecture of split database and table solves the scalability problem of centralized database, it also brings new problems (no support for complex SQL, ACID is difficult to guarantee distributed transactions, etc.) (T)

2. [Multiple choice] What are the challenges of traditional centralized relational databases? (AC)

A. high cost: run on high-end servers, minicomputers, high-end storage and other proprietary hardware;

B. Lack of ecology: insufficient documentation, training, application, etc.

C. poor scalability: can not get rid of the stand-alone architecture, only vertical expansion, not horizontal expansion;

D. Poor performance: at any time, the performance of the traditional centralized database is worse than that of the distributed database;

Chapter 2 Simulation question answers

1.[True/false] TPC-C is a running test, there are no rules, as long as you can run high. (F)

2.[True/false] OceanBase database was incubated internally by Alibaba and Ant for 10 years before it was gradually rolled out to the outside market. (T)

3.[True/false] OceanBase database is a re-release based on open source databases. (F)

4. [Single choice] What type of database is OceanBase (C)

A: Centralized database;

B: NoSQL database ；

C: Distributed relational database;

5. [Multiple choice] What are the core features of OceanBase? (ABCD)

A: High scalability, can use ordinary PC server for horizontal expansion;

B: High performance, peak peak 61 million times/SEC, single table maximum 320 billion rows;

C: High availability. Paxos ensures strong consistency. RPO=0 and RTO<30 seconds.

D: High compatibility, supporting MySQL and Oracle modes, reducing the cost of business migration and transformation;

E: High cost, using minicomputers, high-end storage and other proprietary hardware;

Chapter 3 Simulation question answers

1.[True/false] OceanBase has been published in Aliyun public cloud and private cloud (T)

2.[True/false] OceanBase only supports CPUS of X86 architecture, not other CPUs (such as Kunpeng, Hygon, Phytium, etc.) (F)

3.[True/false] Zone is a logical concept that assigns the same tag to a group of servers in a cluster. Servers with the same tag belong to the same Zone (T)

4.[True/false] Zones can correspond to different cities, equipment rooms in a city, or racks in an equipment room (T)

5.[True/false] Once a tenant's resource pool is created completely, it cannot be changed (F)

6. [Single choice] OceanBase is a cluster. Which component manages the entire cluster and supports functions such as global DDL and cluster data consolidation? (B)

A: OB Proxy B: RootService general control service C: OCP management platform D: ODC developer center

7. [Single choice] OceanBase cluster can support both MySQL and Oracle tenants, which blank screen tool can connect to Oracle tenants? (A)

A: OceanBase client; B: Standard MySQL client

8. [Single choice] Which operating system does OceanBase not support? (B)

A: CentOS; B: Windows C: NeoKylin D: Galaxy Kylin

Chapter 3 Simulation question answers (2)

9. [Single choice] If an OceanBase cluster has three zones, each Zone has five OB serers. How many universal copies can a partition have at most? (B)

A: 10 B: 3 C: 6 D: 5

10. [Single choice] If a cluster has three zones, each Zone has five OB servers. Unit Num of the resource pool corresponding to a tenant is 3. How many resource units does the tenant have? (B)

A: 15 B: 9 C: 45 D: 30

11. [Multiple choice] Which products is OceanBase composed of? (ABCD)

A: Database kernel: SQL engine and storage engine, compatible with MySQL and Oracle modes; Use the Paxos protocol to ensure high availability;

B: OCP cloud management platform: provides management tools for administrators, such as cluster management, Zone management, and tenant management.

C: OMS data migration tool: synchronizes baseline data and incremental data, subscribes to data links from data warehouses, and migrates data from heterogeneous databases.

D: ODC Developer Center: provides daily database development, SQL diagnosis, session management and data import and export functions.

Chapter 4 Simulation question answers (1)

1.[True/false] A partitioned replica contains only static data (SS Table) on the hard disk, no MemTable data and log data. (F)

2.[True/false] The primary replica can only be scattered to all zones, but cannot be focused to one Zone.(F)

3.[True/false] Each OB Server is relatively independent and has its own independent SQL engine. If the data required by the application is not on the current OB Server, the OB Server will coordinate the data of other OB servers and uniformly feed back to the application, which is transparent to the application. (T)

4.[True/false] The primary replica synchronizes redo-log logs to ensure reliability. The primary replica can respond to applications only after receiving messages indicating that all secondary replicas are successfully removed from disks. (F)

5.[True/false] If an enterprise has two equipment rooms in a city, deploy two zones in one equipment room and the other zone in the other to provide room-level disaster recovery. (F)

6. [Single choice] OceanBase fors the Paxos protocol group in unit of (D) .

A: tenant   B: database   C: table   D: partition

7. [Single choice] Which of the following is correct about the expansion/reduction of OceanBase? (C)

A: The administrator needs to stop the service.   B: The service needs to be modified.

C: The system supports dynamic capacity expansion and reduction and has no service awareness

Chapter 4 Simulation question answers (2)

8. [Single choice] OceanBase uses the two-phase commit protocol to ensure atomicity of transactions. In the two-phase commit protocol, who is the coordinator? (B)

A: OB Proxy B: OB Server C: RootService general control service D: OCP cloud management platform

9. [Multiple choice] Which transaction isolation levels does the Oracle tenant of OceanBase support? (BC)

A: Dirty read B: Read-Committed C: Serializable

10. [Single choice] Which technology does OceanBase use to solve the problem of read/write exclusivity?(A)

A: MVCC B: Paxos protocol C: Global snapshot D: Exclusive lock

11. [Multiple choice] Which of the following is correct about the OB Proxy? (AD)

A: The OB Proxy is located between the application and the OB Server and routes the application requests to the appropriate OB Server.

B: OB Proxy must be deployed on an independent server to ensure its performance.

C: OB Proxy participates in the calculation tasks and transaction processing of the database engine.

D: OB Proxy is a "stateless" service process that does not persist data.

Chapter 5 Simulation question answers

1.[True/false] OceanBase can support both MySQL and Oracle tenants in a cluster.(T)

2.[True/false] When you use the Explain command to view the SQL execution plan, the SQL is actually executed. (F)

3.[True/false] Merge must be done automatically by OceanBase and cannot be started manually. (F)

4.[True/false] Disk data in OceanBase is ordered by primary key. (T)

5. [Single choice] Which JDBC driver is required to connect to Oracle tenants using JDBC? (C)

A: MySQL standard JDBC driver B: Oracle standard JDBC driver C: OceanBase developed JDBC driver

6. [Single choice] How many times does OceanBase typically compress for better compression? (B)

A: Once B: Twice C: 3 times D: 4 times

7. [Multiple choice] Which storage media are supported by OceanBase backup and recovery services? (AD)

A: NFS B: IP-SAN C: FC-SAN D: AliYUN OSS

Chapter 6 Simulation question answers

1.[True/false] Session variables take effect only for the current session and do not affect other sessions of the tenant. (T)

2. [Single choice] Which command can be used to query the properties of a parameter? (A)

A: show parameters like '%<pattern>%’; B: alter system set <name> = <value>;

C: show variables like '%<pattern>%’; D: set @@global.<name> = <value>;

3. [Multiple choice] What are the two levels of parameter? (AD)

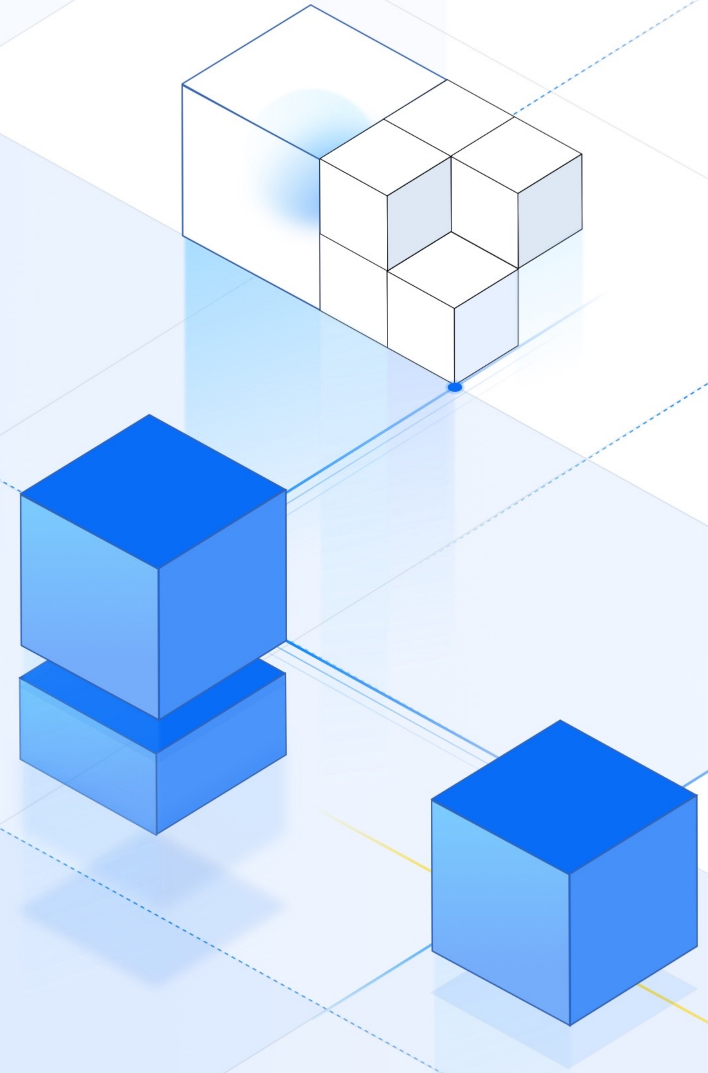
A: cluster level B: Zone level C: OB Server level D: tenant level

Chapter 7 Simulation question answers

1. [Single choice] Which of the following components provides a graphical user interface (GUI) for cluster management, tenant management, and alarm monitoring? (B)

A: ODC developer center B: OCP cloud management platform

C: OB Proxy D: OB Server



Thanks

OCEANBASE