



OBCP V4.0 数据库架构与原理 实验指导手册

声明

蚂蚁集团版权所有 © 2020 ，并保留一切权利。

未经蚂蚁集团事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

商标声明

 **OCEANBASE** 及其他蚂蚁集团相关的商标均为蚂蚁集团所有。本文档涉及的第三方的注册商标，依法由权利人所有。

免责声明

由于产品版本升级、调整或其他原因，本文档内容有可能变更。蚂蚁集团保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在蚂蚁集团授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过蚂蚁集团授权渠道下载、获取最新版的用户文档。如因文档使用不当造成的直接或间接损失，本公司不承担任何责任。

目录

声明	I
目录	II
实验概述	1
前置实验准备	3
准备 1:修改 OCP 默认 IP	3
准备 2:初始化 OB 服务器	5
实验 1: 扩缩容	10
1.1.集群扩容	10
1 实验目的	10
2 实验环境	10
3 实验前提	10
4 实验步骤	10
5 思考讨论	15
1.2.租户扩容	16
1 实验目的	16
2 实验环境	16
3 实验前提	16
4 实验步骤	16
5 思考讨论	18
实验 2: 运维管理	19
2.1.集群运维	19
1 实验目的	19
2 实验环境	19
3 实验前提	19
4 实验步骤	19
5 思考讨论	21

2.2.配置项管理	22
1 实验目的.....	22
2 实验环境.....	22
3 实验前提.....	22
4 实验步骤.....	22
5 思考讨论.....	24
实验 3：容灾管理	25
3.1.备份恢复.....	25
1 实验目的.....	25
2 实验环境.....	25
3 实验前提.....	25
4 实验步骤.....	25
5 思考讨论.....	30
3.2.主备租户	31
1 实验目的.....	31
2 实验环境.....	31
3 实验前提.....	31
4 实验步骤.....	31
5 思考讨论.....	37
实验 4：监控诊断	38
4.1.全链路追踪	38
1 实验目的.....	38
2 实验环境.....	38
3 实验前提.....	38
4 实验步骤.....	38
5 思考讨论.....	40

4.2.SQL 监控.....	41
1 实验目的.....	41
2 实验环境.....	41
3 实验前提.....	41
4 实验步骤.....	41
5 思考讨论.....	44

实验概述

本手册是 OBCP V4 认证课程《DBA2:数据库运维与监控》的配套实验，帮助和指导学员完成课程主要章节的实验练习，通过实验让学员熟悉 OceanBase 数据库的常见运维操作，掌握 OceanBase 数据库监控与问题诊断的方法。

配套实验包含 DBA2 课程的重点内容，主要包含以下几个实验：

章节	实验名称
实验一：扩缩容	集群扩容
	租户扩容
实验二：运维管理	集群运维
	配置项管理
实验三：容灾管理	备份恢复
	主备租户
实验四：监控诊断	全链路追踪
	SQL 监控

本实验最低资源配置要求如下

服务器类型	服务器数量	最低资源要求
OCP	1	16C/32G/100G+200G
OBServer	3	4C/16G/100G+200G

在进行本实验练习前，要求学员完成以下课程学习：

OBCA V4.0 认证培训课程

OBCP V4.0 认证课程——《DBA1：数据库架构与原理》

OBCP V4.0 认证课程——《DBA2：数据库运维与监控》

前置实验准备

实验镜像中预装了 OAT (OceanBase 管理者工具) 和 OCP 云平台, 实验中需要使用 OAT 进行 OBServer 服务器的初始化, 使用 OCP 进行 OceanBase 集群的安装和操作。

准备 1: 修改 OCP 默认 IP

实验镜像中 OCP 及其相关组件均使用了本机地址: 127.0.0.1。在开展实验之前, 需要修改 OCP 及相关组件的对外服务 IP, 从 127.0.0.1 修改为 OCP 服务器的实际 IP, 操作步骤如下:

步骤 1 登录 OCP 服务器, 实验所用主机的 root 用户密码请以实验环境提供的为准。

步骤 2 在 OCP 服务器中, 使用 mysql 客户端登录 ocp_meta 租户, 实验镜像的 ocp_meta 租户的 root 用户密码默认为 bbBB22__ (注意最后是两个下划线)。

```
obclient -h<OCP 所在 IP 地址> -P2883 -uroot@ocp_meta -pbbBB22__ -c -A -D ocp
```

步骤 3 执行以下语句, 将 OCP、MetaDB、ES 服务的 IP 修改为 OCP 服务器的实际 IP。例如 OCP 服务器的 IP 地址为 **172.28.15.211**, 则修改语句如下:

```
select * from config_properties where value like '%127.0.0.1%';

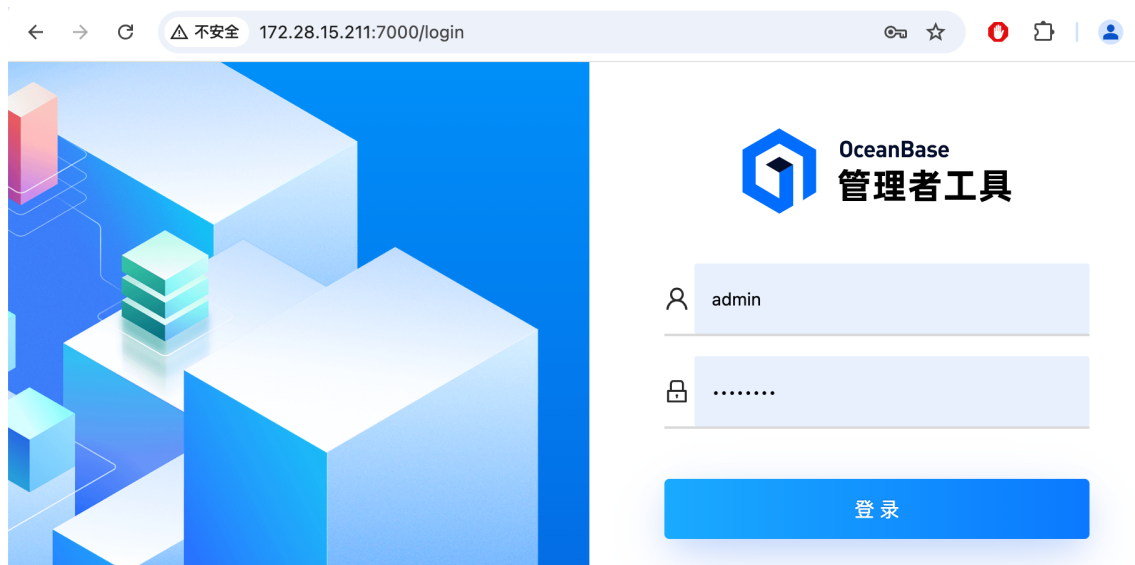
update config_properties set value='http://172.28.15.211:8080' where
`key`='ocp.site.url';

update config_properties set value='172.28.15.211' where
`key`='ocp.monitordb.host';

update config_properties set value='172.28.15.211:9200' where
`key`='ocp.analyze.es.client.addresses';
```

注意: 这几个参数的修改需要重启 OCP 才能生效

步骤 4 登录 OAT，登录地址为 `http://<ocp_server_ip>:7000/`，使用 OCP 服务器的 IP，用户名/密码为：`admin/bbBB22__`（注意最后是两个下划线）。



步骤 5 从左侧产品服务列表选择 OCP。



步骤 6 从 OCP 页面，点击右上角的**容器操作**，执行重启操作。（注意：需要在“服务器”-“凭据”中修改 ROOT 凭据密码，替换成实际实验环境的密码，否则容器重启无法成功）

✕ 容器操作

☒	容器名	服务器IP	运行状态	机房 (IDC)	地域 (Region)	操作
☒	ocp	127.0.0.1	● 运行中	IDC1	BJ	停止 重启

重启操作需要 1~2 分钟完成。如果能够登录 OCP 云平台，则表示重启完成。

准备 2:初始化 OB 服务器

在安装 OceanBase 集群前，需要对 OBServer 服务器进行相应的配置修改，包括磁盘初始化、时钟同步设置、属主用户创建，以及操作系统内核参数的调整。OAT 提供了一键式初始化服务器的功能，可以通过在 OAT 添加服务器并同步到 OCP 来完成 OBServer 服务器的初始化。

步骤 1 在 OAT 左侧功能列表中选择 **服务器-凭据**，执行添加服务器凭据的操作。比如 OBServer 服务器的 root 用户密码为 abc123 时，我们可以创建授权类型为密码认证、用户类型为 root 用户的凭据。

新建凭据

凭据名

授权类型

☒ 密码认证

☐ 公钥认证

☐ 已配置免密

用户类型

☒ root 用户

☐ 普通用户

密码



应用主机（可选）

步骤 2 在 OAT 左侧功能列表中选择 **服务器-服务器**，执行添加服务器的操作。

The screenshot shows the 'OceanBase 管理者工具' (OceanBase Manager Tools) interface in 'POC模式' (POC Mode). The left sidebar has a menu with '工作台' (Dashboard), '服务器' (Servers), '凭据' (Credentials), '产品服务' (Product Service), and '系统与安全' (System & Security). The '服务器' (Servers) option is selected. The main content area is titled '添加服务器' (Add Server) and contains a form with the following fields:

- 基础信息** (Basic Information)
- 服务器 IP** (Server IP): A text input field with a hint '按回车键可批量输入多个服务器 IP' (Press Enter to batch input multiple server IPs).
- SSH 端口** (SSH Port): A text input field with the value '22'.
- 机房 (IDC) ②** (IDC): A dropdown menu with the placeholder '请选择' (Please select).
- 凭据 ②** (Certificate): A dropdown menu with the placeholder '请选择' (Please select).
- 说明 (可选)** (Remarks): A text input field with the placeholder '请输入' (Please enter).

- 服务器 IP 中填入要进行初始化的 OBCServer IP，可以一次添加多个 IP。
- 机房（IDC）中选择已有的标识来标记 OBCServer 的位置，或者创建新的机房标识。
- 凭据中选择上一步为 OBCServer 创建的凭据。

步骤 3 在添加服务器时同时完成 OBCServer 的初始化配置。

The screenshot shows a web-based configuration interface for OBCServer initialization. At the top, there is a toggle switch for '初始化配置' (Initialization Configuration) which is turned on. Below this, the '服务器用途' (Server Purpose) section has two radio buttons: 'OB Server' (selected) and 'OB 产品服务' (OB Product Service). A blue tooltip for 'OB Server' explains it is for dedicated business clusters and OBProxy clusters, performing disk checks and setting startup scripts. The '操作系统属主用户' (OS Owner User) field contains 'admin'. The '用户密码' (User Password) field is empty, with a '随机密码' (Random Password) button and a note to remember or copy the password. The '用户 UID:GID' field contains '500:500'. The '时钟同步' (Clock Synchronization) toggle is off. The '自动同步至 OCP' (Auto sync to OCP) toggle is on. At the bottom, an 'OCP' dropdown menu shows 'OCP' selected.

- 服务器用途：选择 OB Server。
- 操作系统属主用户：使用默认 admin，并指定用户密码。
- 自动同步至 OCP：完成初始化的服务器信息同步添加到 OCP 云平台，方便随后使用 OCP 安装 OceanBase 集群。

步骤 4 时钟同步（可选配置）

时钟同步



时钟源 IP ?

按回车键可批量添加多个服务器 IP

minpoll ?

maxpoll ?

maxslewrage ?

4

6

500

同时把此服务器作为次级时钟源 ?（可选）

☐ 是 ☒ 否

- 如果在阿里云环境，所有机器均与云上时钟源同步，这里可以不做配置。
- 如果为线下独立部署的服务器，建议配置时钟同步，可以将 OCP 服务器作为 OBC 的时钟源。配置时钟同步时，需要修改时钟源服务器的 chrony 配置文件，以允许 OBC 的时钟同步请求。

```
[root@OCP ~]# vim /etc/chrony.conf

# 被允许与之同步的地址，修改为实际的 observer ip
allow 172.28.15.0/24

# 初始时间偏差超过 5 秒时，在前三次同步周期内完成时钟校准
makestep 5 3
```

步骤 5 在添加服务器时同时完成 OBase 的磁盘初始化。

磁盘初始化 ☒

● 为保障 OceanBase 数据库及相关产品服务的稳定性，通常建议将服务所用的目录配置在独立的磁盘上。选择初始化磁盘后，程序将自动通过 lvm 创建对应 vg，并将对应大小的 lv 挂载到指定的挂载目录上。

选择磁盘或分区

/dev/vdb ×

仅支持选择未做分区的整块磁盘或未被使用的磁盘分区；已选设备空间总和为：199.98 GB

挂载用途	挂载点	空间 (GB)	
☒ 软件安装路径	/home/admin	20	GB 使用剩余空间
☒ 数据盘路径	/data/1	60	GB 使用剩余空间
☒ 日志盘路径	/data/log1	119.98	GB 使用剩余空间

- 选择磁盘或分区：选择预留给 observer 的磁盘分区，比如 /dev/vdb。
- 挂载用途：使用默认的挂载点，按照磁盘空间容量分配容量，比如将 200G 磁盘分别划分 20G、60G、120G 给软件安装盘、数据盘、日志盘。

注意：在执行服务器初始化时，可能会出现以下报错，需要手动处理！

- 任务执行时报错：precheck 不通过

Task List:

- config_docker (ID: config_docker) 开始: 18:12:16 耗时: 0.14秒
- config_os (ID: config_os) 开始: 18:12:16 耗时: 8秒
- precheck (ID: precheck) 开始: 18:19:12 耗时: 8秒
- sync_server_to_oap (ID: sync_server_to_oap) 开始: - 耗时: -
- update_server (ID: update_server) 开始: - 耗时: -

Terminal Log:

```
112 [2024-10-16T18:19:17.810+0800] INFO - check RPM: bind-utils-9.11.4-26.P2.el7_9.16.x86_64 is installed ... PASS
113 [2024-10-16T18:19:18.409+0800] INFO - check RPM: libaio-0.3.109-13.el7.x86_64 is installed ... PASS
114 [2024-10-16T18:19:19.002+0800] INFO - check RPM: python-pycurl-7.19.0-19.el7.x86_64 libcurl-7.29.0-59.el7_9.2.x86_64 curl-7.29.0-59.el7_9.2.x86_64 is installed ... PASS
115 [2024-10-16T18:19:19.594+0800] INFO - check RPM: libatomic-4.8.5-44.el7.x86_64 is installed ... PASS
116 [2024-10-16T18:19:20.189+0800] INFO - check RPM: iproute-4.11.0-30.el7.x86_64 is installed ... PASS
117 [2024-10-16T18:19:20.785+0800] INFO - check RPM: fio-3.7-2.el7.x86_64 is installed ... PASS
118 [2024-10-16T18:19:20.823+0800] INFO - check mysql client, working ... PASS
119 [2024-10-16T18:19:20.828+0800] INFO - checking irq affinity ...
120 [2024-10-16T18:19:20.831+0800] INFO - not phy machine, skip check irq affinity, pass ... PASS
121 [2024-10-16T18:19:20.919+0800] INFO - NIC speed of eth0 check pass ... PASS
122 [2024-10-16T18:19:20.919+0800] ERROR -
123 [2024-10-16T18:19:20.919+0800] ERROR -
124 [2024-10-16T18:19:20.919+0800] ERROR - ### SUMMARY OF ISSUES IN PRE-CHECK ###
125 [2024-10-16T18:19:20.920+0800] ERROR - check CPU count: 4 < 8 ... EXPECT >= 8 ... FAIL
126 [2024-10-16T18:19:20.920+0800] ERROR - TIPS: replace another machine with more CPU
127 [2024-10-16T18:19:20.920+0800] ERROR - check total MEM: 15 GB < 30 GB ... EXPECT >= 30 GB ... FAIL
128 [2024-10-16T18:19:20.920+0800] ERROR - TIPS: replace another machine with more MEM
```

- 报错原因：服务器资源规格小于 OAT 建议的最低要求。
- 解决办法：忽略报错，将 precheck 结果设置为成功。

实验 1：扩缩容

1.1. 集群扩容

1 实验目的

在本实验中，我们将一个单可用区 OceanBase（单机版）扩展为 3 可用区 OceanBase 集群（分布式），实现从单机到分布式的扩容。

2 实验环境

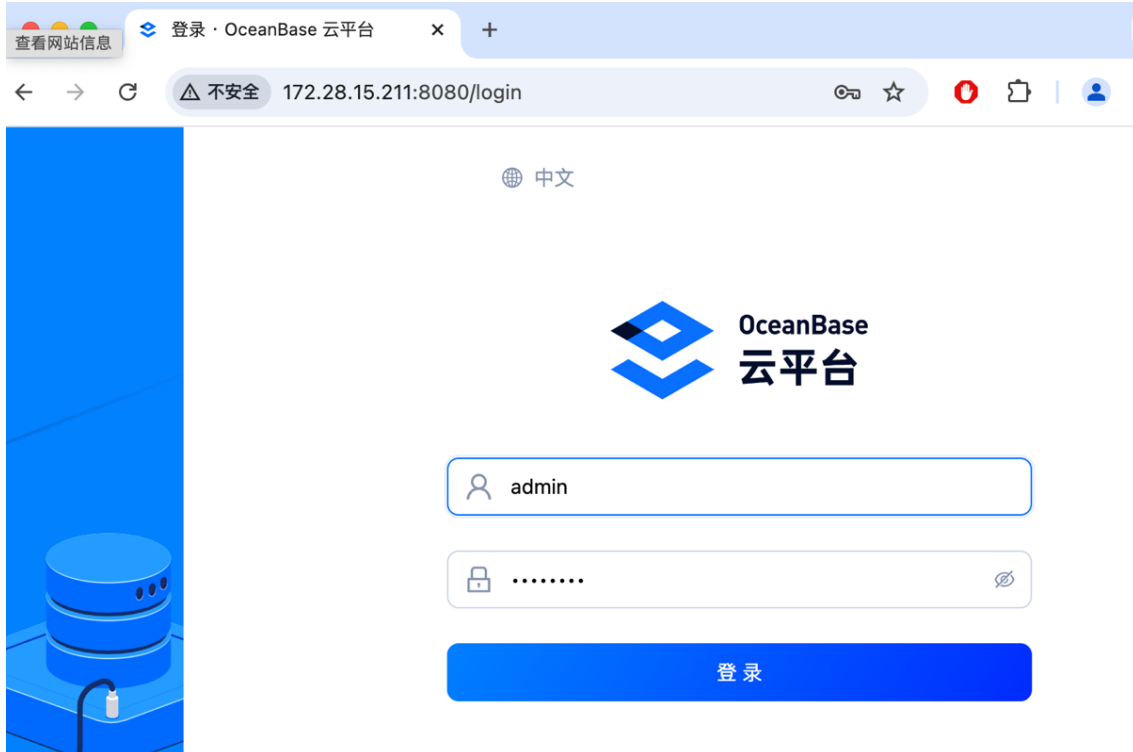
1 台 OCP 机器 + 3 台 OBServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。

4 实验步骤

步骤 1 登录 OCP，登录地址为 `http://<ocp_server_ip>:8080/`，使用 OCP 服务器的 IP，用户名/密码为：`admin/bbBB22__`（注意最后是两个下划线）。



步骤 2 查看主机列表，确认在 OAT 初始化完成的 3 台 OBCServer 服务器已经同步到 OCP。



IP 地址	主机名	标签	主机类型	机型	机房	地域	所属集群	所属服务	状态
172.28.15.214	OBServer	+ 标签	物理机	observer	IDC1	BJ	对象: obcp_test:1729067258 类型: OceanBase 集群	OceanBase	在线
172.28.15.213	OBServer	+ 标签	物理机	observer	IDC1	BJ	对象: obcp_test:1729067258 类型: OceanBase 集群	OceanBase	在线
172.28.15.212	OBServer	+ 标签	物理机	observer	IDC1	BJ	对象: obcp_test:1729067258 类型: OceanBase 集群	OceanBase	在线
172.0.0.1	OCP	+ 标签	物理机	MetaDB_server	IDC1	BJ	对象: ocp_meta:1729135356 类型: OceanBase 集群	OceanBase	在线

步骤 3 从集群列表页面中创建集群，指定集群的部署方式为单机。



创建集群

分布式集群 | 单机集中式

基本设置

集群类型: ☒ 主集群 ☐ 备集群

集群名: obcp_test

root@sys 密码: bbBB22__ (随机生成)

OceanBase 版本: oceanbase-4.2.1.7-107030032024062709-el7

关联 OBProxy 集群: ☐ (仅可选择与此 OceanBase 集群网络一致且非空的 OBProxy 集群)

部署模式

Zone 名称: zone1 | 机房: IDC1 | 机型 (可选): observer | CPU 架构: x86_64 | 主机: 172.28.15.212 | Root Service 位置: 172.28.15.212

+ 新增 Zone

● 单机 (集中式架构) 仅有一份数据副本，当发生存储或者硬件故障时可能存在数据丢失风险，建议您对于重要业务系统进行定期备份或者配置高可用方案，如租户主备库，多副本容灾等方式避免单点故障。

Zone 优先级排序: ☐

添加仲裁服务: ☐

- **集群类型：**选择**主集群**
- **集群名：**要创建的 OceanBase 集群名称，比如 obcp_test
- **root@sys 密码：**OceanBase 集群 sys 租户的 root 用户的密码，比如 bbBB22__
- **OceanBase 版本：**从列表中选择 4.2.1 版本软件包。
- **部署模式：**单机集中式，即只部署一个 Zone，Zone 内只有一台 OBServer。

步骤 4 在集群创建页面，打开参数设置，设置 observer 进程的资源分配以及默认参数。

CPU 超卖设置

参数设置

如您对集群级别的 CPU、内存、数据盘以及日志盘的使用有定制化变更诉求，请设置参数：cpu_count(默认值：自动调整)、memory_limit_percentage(默认值：80%)、system_memory(默认值：30GB)、data_disk_usage_limit_percenta

参数模板 (可选)

选择参数模板

参数修改

参数名	参数值	取值类型	取值范围	操作
system_memory	4G	CAPACITY	[0M,]	删除
datafile_size	20G	CAPACITY	[0M,]	删除
__min_full_resource_pool_memory	1073741824	INT	[1073741824,]	删除

+添加启动参数

自定义设置

自定义配置支持您执行集群级别用户配置（如操作系统属主用户）、路径设置（如软件安装路径、数据盘路径、日志盘路径）以及端口设置（如SQL 端口、RPC 端口）

取消

提交

- system_memory：系统内部内存的预留大小。本实验为小规格安装，将其从 30G 修改为 4G。
- datafile_size：数据磁盘的大小。为了后续实验的进行，这里只分配数据盘空间的一部分，比如 20G。
- __min_full_resource_pool_memory：资源单元的内存下限，即创建租户时可设置的租户最小内存大小。本实验为小规格安装，将其从 5368709120（5G）修改为 1073741824（1G）。

步骤 5 提交集群创建任务并确认。在任务中心查看任务的执行详情。正常情况下，该任务在 10 分钟内完成。

任务 / Create primary OB cluster

Create primary OB cluster

任务 ID: 3107 任务状态: 执行中 操作对象: obcp_test:1729135357 操作对象类型: OceanBase 集群 任务发起人: admin

当前进度: 20/56 开始执行时间: 2024年10月21日 11:30:58 任务累计耗时: 2分钟

Prepare install cluster

Pre check for install ob

Pre check for install ob

Pre check for install ob

执行成功

执行成功

执行成功

执行成功

开始: 11:30:58

开始: 11:31:04

开始: 11:31:04

开始: 11:31:04

耗时: 5秒

耗时: 0.21秒

耗时: 0.21秒

耗时: 0.22秒

Bootstrap ob

EXECUTE 11:31:40

HttpTemplate : POST request to agent, url:http://172.28.15.212:62888/api/v1/task/status, request body:GetTaskStatusRequest(taskToken=615aa4c-f19c-4dc6-9628-8646961c55fe), params:null

69 2024-10-21 11:32:09.267 INFO [pool-manual-subtask-executor14,8c7383e12dff4575,8f701b615584] c.o.o.s.task.util.AgentAsyncTaskHelper : try to request task result(EXECUTE), result:false,null,<null>

70

71 2024-10-21 11:32:09.269 INFO [pool-manual-subtask-executor14,8c7383e12dff4575,8f701b615584] c.o.o.s.common.Lang.pattern.Retry : wait for 15 seconds

72

73 2024-10-21 11:32:24.271 INFO [pool-manual-subtask-executor14,8c7383e12dff4575,8f701b615584] c.o.o.e.internal.template.

步骤 6 在集群列表中打开新建的单机集群，进入集群概览页面，可以看到我们已经创建了一个单机集中式的 OceanBase 集群。

概览

功能介绍

扩展分布式集群 新建租户

单机（集中式架构）仅有一份数据副本，当发生存储或者硬件故障时可能存在数据丢失风险，建议您对于重要业务系统进行定期备份或者配置高可用方案，如租户主备库，多副本容灾等方式避免单点故障。

基本信息

集群名: obcp_test

部署模式: 单机

租户数量: 1

集群 ID: 1729135359

CPU 架构: x86_64

创建者: admin

OceanBase 版本号: 4.2.1.7

CPU 超卖额度: 未开启

OBProxy 集群: 关联 OBProxy 集群

Region 数量: 1

机器数量: 1

步骤 7 新建租户：指定租户名 mysql，Unit 规格 2C4G。

新建租户

基本设置

租户类型
☒ 主租户 ☐ 备租户

选择集群
obcp_test-1729135358
只有运行中的主集群才能新建租户

租户模式
MySQL

租户名
mysql

管理员初始密码
bbB22_
请牢记密码，也可复制密码并妥善保存

字符集
utf8mb4

Collation
utf8mb4_general_ci

备注
请输入

副本设置

Zone 名称
☒ zone1

副本类型
全能型副本

Unit 规格
u3C7G

Unit 数量
CPU/内存: 3C7G 1

- 租户类型：选择主租户。
- 管理员初始密码：MySQL 模式的租户管理员用户为 root，Oracle 模式的租户管理员用户为 sys，请牢记该初始密码。
- 副本设置：因为集群是单机集中式，这里只能选择一个全功能型副本。

■ 新增 Unit 规格 2C4G：

✕ 新增 Unit 规格

基础配置

规格名称

u2C4G

CPU 高级设置 ☐

CPU (核)

2.0

CPU 的可选范围为 ≥ 0.5 核，请输入 0.5 的整数倍。

内存 (GiB)

4

内存的可选范围为 ≥ 1 GiB，请输入 1 的整数倍。

更多配置 ☐

确定

取消

- 在安全设置中，设置允许访问租户的 IP 地址，实验环境可以允许所有 IP 访问。

安全设置

IP 地址白名单

☐ 自定义

☒ 所有 IP 都可访问

存在访问安全风险，请谨慎操作

参数设置

取消

提交

步骤 8 租户创建任务执行成功后（约 1 分钟），回到集群概览页面，将集群扩展为分布式。

✕ 扩展分布式集群

基本信息

所属集群: obcp_test:1729135358 OceanBase 版本号: 4.2.1.7

扩展配置

扩展方式

新增 Zone

新增 OBCServer

Zone 名称	机房	机型（可选）	CPU 架构	主机	
zone2	IDC1	observer	x86_64	172.28.15.213:22	✕
zone3	IDC1	observer	x86_64	172.28.15.214:22	✕
+ 新增Zone					

自定义设置 >

参数设置 >

租户配置

☐ 同步扩展所有租户副本

取消

确定

➤ 扩展方式：选择新增 Zone。

➤ 租户配置：为了后续实验的进行，此处不选择同步扩展所有租户副本。

步骤 9 新增 Zone 任务执行成功后（约 5 分钟），回到集群概览页面，可以看到集群已经从单机扩展为 1-1-1 的分布式。

基本信息

集群名: obcp_test

集群 ID: 1729135358

部署模式: 集群 (1-1-1)

CPU 架构: x86_64

步骤 10 使用 OCP 平台为 OceanBase 集群（obcp_test）安装 obproxy 集群。

OBProxy / 创建 OBProxy 集群

创建 OBProxy 集群

基本设置

集群名: obcp_test

root@proxysys 密码: bbB22____ 随机生成

软件版本: obproxy-4.3.1.0-412024070511.el7.x86_64.rpm

负载均衡管理: ☐

访问地址: 172.28.15.99

访问端口: 2883

启动方式: ConfigUrl

选择可连接 OceanBase 集群: obcp_test

部署 OBProxy

仅可选择 CPU 架构为 x86_64 的主机

机房	机型 (可选)	主机
IDC1	observer	172.28.15.212:22
IDC1	observer	172.28.15.213:22
IDC1	observer	172.28.15.214:22

+ 添加 OBProxy

- 集群名: obproxy 集群的名字, 可以随意指定, 也可以与关联的 OceanBase 集群同名。
- 访问地址: 仅用于生成 obproxy 的连接串, 地址可以随意填写, 不影响使用。
- 启动方式: 选择 ConfigUrl, 使用 OCP 来获取 OceanBase 的地址信息。
- 选择可连接 OceanBase 集群: 选择实验 1.1 中新建的 OceanBase 集群 (obcp_test)
- 我们选择将 OBProxy 与 OBServer 共用服务器的方式, 将 OBProxy 安装到 OBServer 服务器上。

5 思考讨论

思考题 1: OceanBase 作为一个单机分布式一体化的数据库, 可以在单机集中式、单副本多节点、多副本多节点的形态之间转换, 这样的产品特性能够为用户带来什么好处?

1.2.租户扩容

1 实验目的

在本实验中，我们使用分别使用黑屏（SQL 命令）和白屏（OCP 云平台）的方式对租户执行扩容操作，让学员熟悉租户资源管理以及扩缩容的相关命令和视图，熟悉 OCP 云平台便捷运维的方法。

2 实验环境

1 台 OCP 机器 + 3 台 OBServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。
- 完成实验 1.1 的操作。

4 实验步骤

步骤 1 在 OCP 服务器上，使用 obclient 客户端工具连接 OceanBase 集群的 sys 租户。

```
obclient -h<OBProxy ip 地址> -P2883 -uroot@sys#obcp_test -p -c -A -D oceanbase
```

步骤 2 查看当前集群中的租户定义。

```
SELECT TENANT_ID, TENANT_NAME, TENANT_TYPE, PRIMARY_ZONE, LOCALITY FROM  
DBA_OB_TENANTS;
```

TENANT_ID	TENANT_NAME	TENANT_TYPE	PRIMARY_ZONE	LOCALITY
1	sys	SYS	RANDOM	FULL{1}@zone1
1001	META\$1002	META	RANDOM	FULL{1}@zone1
1002	mysql	USER	RANDOM	FULL{1}@zone1

- PRIMARY_ZONE: RANDOM 表示 Primary Zone 为所有 Zone。
- LOCALITY: FULL{1}@zone1 表明租户只有一个全功能副本，分布在 zone1 内。

步骤 3 使用黑屏的方式对 sys 租户进行扩容，将其扩容为 3 副本。

- 查看 sys 租户当前的资源池与资源规格定义。

```
SELECT TENANT_ID, NAME, UNIT_COUNT, UNIT_CONFIG_ID, ZONE_LIST FROM  
DBA_OB_RESOURCE_POOLS;  
  
SELECT UNIT_CONFIG_ID, NAME, MAX_CPU, MIN_CPU, MEMORY_SIZE, LOG_DISK_SIZE FROM  
DBA_OB_UNIT_CONFIGS;
```

```
MySQL [oceanbase]> SELECT TENANT_ID, NAME, UNIT_COUNT, UNIT_CONFIG_ID, ZONE_LIST FROM DBA_OB_RESOURCE_POOLS;
+-----+-----+-----+-----+-----+
| TENANT_ID | NAME                | UNIT_COUNT | UNIT_CONFIG_ID | ZONE_LIST |
+-----+-----+-----+-----+-----+
| 1         | sys_pool            | 1         | 1              | zone1     |
| 1002      | pool_mysql_zone1_qch | 1         | 1001           | zone1     |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

MySQL [oceanbase]> SELECT UNIT_CONFIG_ID, NAME, MAX_CPU, MIN_CPU, MEMORY_SIZE, LOG_DISK_SIZE FROM DBA_OB_UNIT_CONFIGS;
+-----+-----+-----+-----+-----+-----+
| UNIT_CONFIG_ID | NAME                | MAX_CPU | MIN_CPU | MEMORY_SIZE | LOG_DISK_SIZE |
+-----+-----+-----+-----+-----+-----+
| 1              | sys_unit_config     | 1       | 1       | 1073741824  | 2147483648    |
| 1001           | config_mysql_zone1_u3C7G_qch | 2       | 2       | 4294967296  | 12884901888   |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

- sys 租户的 resource pool 为 sys_pool，当前只包含 zone1，对应的 unit_config_id=1。
- unit_config_id=1 对应的资源规格为 sys_unit_config，其资源定义为 CPU=1, MEMROY_SIZE=1G。

- 修改 resource pool 的定义，将其 zone list 扩展为 zone1,zone2,zone3。

```
ALTER RESOURCE POOL sys_pool ZONE_LIST=('zone1','zone2','zone3');
```

- 修改租户的副本数，将其 locality 扩展为 "FULL{1}@zone1, FULL{1}@zone2, FULL{1}@zone3"。

```
ALTER TENANT sys LOCALITY="FULL{1}@zone1, FULL{1}@zone2";
```

```
ALTER TENANT sys LOCALITY="FULL{1}@zone1, FULL{1}@zone2, FULL{1}@zone3";
```

- 从单副本扩展为多副本时，需要首先从单副本扩展为 2 副本，然后才能继续扩展。

- 查看当前集群中的租户定义。

```
SELECT TENANT_ID, TENANT_NAME, TENANT_TYPE, PRIMARY_ZONE, LOCALITY, PREVIOUS_LOCALITY
FROM DBA_OB_TENANTS;
```

```
+-----+-----+-----+-----+-----+-----+
| TENANT_ID | TENANT_NAME | TENANT_TYPE | PRIMARY_ZONE | LOCALITY                                | PREVIOUS_LOCALITY |
+-----+-----+-----+-----+-----+-----+
| 1         | sys         | SYS         | RANDOM      | FULL{1}@zone1, FULL{1}@zone2, FULL{1}@zone3 | NULL              |
| 1001      | META$1002  | META        | RANDOM      | FULL{1}@zone1                                | NULL              |
| 1002      | mysql       | USER        | RANDOM      | FULL{1}@zone1                                | NULL              |
+-----+-----+-----+-----+-----+-----+
```

- PREVIOUS_LOCALITY 为 Locality 变化前的定义，为 NULL 时表示租户没有在进行 Locality 变更（或者变更已经完成）。

步骤 4 使用白屏的方式对用户租户 mysql 进行扩容，将其扩容为 3 副本。

- 在 OCP 云平台中，从租户列表选择 mysql 租户。
- 在租户概览页面中，选择新增副本，将副本扩展到 zone2 和 zone3。（大约需要 2 分钟）

新增副本

操作对象: mysql

目标 Zone	副本类型	Unit 规格	Unit 数量	
zone2	全能型副本	u2C4G CPU/内存: 2C4G	1	
zone3	全能型副本	u2C4G CPU/内存: 2C4G	1	
+ 新增 Zone				

建议租户内 Zone 的 Unit 资源保持一致

取消 确定

- Unit 规格：建议选择与 zone1 相同的规格。

5 思考讨论

思考题 1：在 1-1-1 的集群架构中，对一个 3 副本的用户租户进行扩容的方法都有哪些？

实验 2：运维管理

2.1.集群运维

1 实验目的

在本实验中，我们使用 SQL 命令与系统视图来对集群以及租户的进行运维管理，让学员熟悉相关系统视图和运维操作。

2 实验环境

1 台 OCP 机器 + 3 台 OBCServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。
- 完成实验 1.1 的操作。

4 实验步骤

步骤 1 在 OCP 服务器上，使用 obclient 客户端工具连接 OceanBase 集群的 sys 租户。

```
obclient -h<OBProxy ip> -P2883 -uroot@sys#obcp_test -p -c -A -D oceanbase
```

步骤 2 查看集群运行状态。

- 查看 Zone 的状态。

```
SELECT * FROM DBA_OB_ZONES;
```

ZONE	CREATE_TIME	MODIFY_TIME	STATUS	IDC	REGION	TYPE
zone1	2024-10-31 17:02:46.939841	2024-10-31 17:03:11.257271	ACTIVE	IDC1	BJ	ReadWrite
zone2	2024-10-31 17:43:54.456564	2024-10-31 17:43:59.802394	ACTIVE	IDC1	BJ	ReadWrite
zone3	2024-10-31 17:43:57.670091	2024-10-31 17:43:59.811875	ACTIVE	IDC1	BJ	ReadWrite

- 查看 OBCServer 的状态。

```
SELECT SVR_IP,ZONE,WITH_ROOTSERVER,STATUS,START_SERVICE_TIME,STOP_TIME FROM  
DBA_OB_SERVERS;
```

SVR_IP	ZONE	WITH_ROOTSERVER	STATUS	START_SERVICE_TIME	STOP_TIME
172.28.15.212	zone1	YES	ACTIVE	2024-10-31 17:06:16.775863	NULL
172.28.15.213	zone2	NO	ACTIVE	2024-10-31 17:48:16.212602	NULL
172.28.15.214	zone3	NO	ACTIVE	2024-10-31 17:48:16.181789	NULL

步骤 3 在 OCP 的预案管理中禁用 OObserver 进程异常退出重新拉起的功能。



步骤 4 重启 zone3 内的 OObserver 节点。

- 停止该 OObserver 节点的服务，并查看 OObserver 的状态。

```
ALTER SYSTEM STOP SERVER '172.28.15.214:2882';  
  
SELECT SVR_IP,ZONE,WITH_ROOTSERVER,STATUS,START_SERVICE_TIME,STOP_TIME FROM  
DBA_OB_SERVERS;
```

SVR_IP	ZONE	WITH_ROOTSERVER	STATUS	START_SERVICE_TIME	STOP_TIME
172.28.15.212	zone1	YES	ACTIVE	2024-10-31 17:06:16.775863	NULL
172.28.15.213	zone2	NO	ACTIVE	2024-10-31 17:48:16.212602	NULL
172.28.15.214	zone3	NO	ACTIVE	2024-10-31 17:48:16.181789	2024-11-01 10:18:16.442629

- STATUS: ACTIVE 表示 observer 进程还处于活跃状态。
- STOP_TIME: 执行 STOP SERVER 命令的时间，表示 OObserver 停止服务。

- 登录到该 OObserver 所在的服务器，使用 kill 命令杀掉 observer 进程。

```
pgrep observer -a  
kill -9 $(pidof observer)
```

```
[root@OBSERVER ~]# pgrep observer -a  
3693 /home/admin/oceanbase/bin/observer  
[root@OBSERVER ~]# kill -9 $(pidof observer)  
[root@OBSERVER ~]# pgrep observer -a
```

- 查询 OObserver 的状态，发现 zone3 内的 OObserver 变为 INACTIVE。

```
SELECT SVR_IP,ZONE,WITH_ROOTSERVER,STATUS,START_SERVICE_TIME,STOP_TIME FROM  
DBA_OB_SERVERS;
```

SVR_IP	ZONE	WITH_ROOTSERVER	STATUS	START_SERVICE_TIME	STOP_TIME
172.28.15.212	zone1	YES	ACTIVE	2024-10-31 17:06:16.775863	NULL
172.28.15.213	zone2	NO	ACTIVE	2024-10-31 17:48:16.212602	NULL
172.28.15.214	zone3	NO	INACTIVE	NULL	2024-11-01 10:18:16.442629

- STATUS: INACTIVE 表示 observer 进程被杀掉。

➤ START_SERVICE_TIME: NULL 表示当前节点没有处于服务状态。

- 在 OObserver 服务器中，使用 admin 用户启动 observer 进程。

```
su - admin  
cd /home/admin/oceanbase && ./bin/observer
```

- 再次查询 OObserver 的状态，发现 zone3 内的 OObserver 的状态又变为 ACTIVE。

SVR_IP	ZONE	WITH_ROOTSERVER	STATUS	START_SERVICE_TIME	STOP_TIME
172.28.15.212	zone1	YES	ACTIVE	2024-10-31 17:06:16.775863	NULL
172.28.15.213	zone2	NO	ACTIVE	2024-10-31 17:48:16.212602	NULL
172.28.15.214	zone3	NO	ACTIVE	2024-11-01 10:35:54.742046	2024-11-01 10:18:16.442629

- 启动该 OObserver 节点的服务，并查看 OObserver 的状态。

```
ALTER SYSTEM START SERVER '172.28.15.214:2882';  
  
SELECT SVR_IP,ZONE,WITH_ROOTSERVER,STATUS,START_SERVICE_TIME,STOP_TIME FROM  
DBA_OB_SERVERS;
```

步骤 5 从历史事件视图中查看 OObserver 重启的记录。

```
SELECT * FROM DBA_OB_SERVER_EVENT_HISTORY ORDER BY TIMESTAMP DESC LIMIT 10;  
  
SELECT * FROM DBA_OB_ROOTSERVICE_EVENT_HISTORY ORDER BY TIMESTAMP DESC LIMIT 10;
```

5 思考讨论

思考题 1：MemStore 中的更新数据会在 kill observer 时被丢弃，重启 observer 时需要根据日志来重建丢失的数据，为了加快重启速度，应该在 kill observer 前采取什么操作？

2.2.配置项管理

1 实验目的

在本实验中，我们分别在 sys 租户和用户租户进行配置项的设置，让学员了解不同 scope 的配置项的设置方法。

2 实验环境

1 台 OCP 机器 + 3 台 OBServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。
- 完成实验 1.1 的操作。

4 实验步骤

步骤 1 在 OCP 服务器上，使用 obclient 客户端工具连接 OceanBase 集群的 sys 租户。

```
obclient -h<OBProxy ip> -P2883 -uroot@sys#obcp_test -p -c -A -D oceanbase
```

步骤 2 分别使用 SHOW 命令和系统视图查看 datafile 配置项的设置，确定配置项的 scope 是集群。

```
SHOW PARAMETERS LIKE 'datafile%';  
  
SELECT * FROM GV$OB_PARAMETERS WHERE NAME LIKE 'datafile%';
```

SVR_IP	SVR_PORT	ZONE	SCOPE	TENANT_ID	NAME	DATA_TYPE	VALUE	INFO
172.28.15.214	2882	zone3	CLUSTER	NULL	datafile_size	NULL	20G	size of the data file. Range: [0, +∞)
172.28.15.213	2882	zone2	CLUSTER	NULL	datafile_size	NULL	20G	size of the data file. Range: [0, +∞)
172.28.15.212	2882	zone1	CLUSTER	NULL	datafile_size	NULL	20G	size of the data file. Range: [0, +∞)

步骤 3 开启数据文件的动态扩容：

- 查看磁盘空间大小：

```
[root@OBServer /]# lsblk  
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT  
vda                                253:0    0  100G  0 disk  
└─vda1                            253:1    0  100G  0 part /  
vdb                                253:16   0  200G  0 disk  
├─oceanbase-ob_home               252:0    0   20G  0 lvm  /home/admin  
├─oceanbase-ob_data               252:1    0   60G  0 lvm  /data/1  
└─oceanbase-ob_log                252:2    0  120G  0 lvm  /data/log1
```

■ 查看 OBDServer 当前的磁盘分配情况：

```
SELECT SVR_IP, CONCAT(ROUND((DATA_DISK_CAPACITY)/1024/1024/1024,1),' GB') AS
DATA_DISK_CAPACITY, CONCAT(ROUND((DATA_DISK_ALLOCATED)/1024/1024/1024,1),' GB') AS
DATA_DISK_ALLOCATED, CONCAT(ROUND((DATA_DISK_IN_USE)/1024/1024/1024,1),' GB') AS
DATA_DISK_IN_USE FROM GV$OB_SERVERS;
```

SVR_IP	DATA_DISK_CAPACITY	DATA_DISK_ALLOCATED	DATA_DISK_IN_USE
172.28.15.212	20.0 GB	20.0 GB	0.2 GB
172.28.15.213	20.0 GB	20.0 GB	0.0 GB
172.28.15.214	20.0 GB	20.0 GB	0.0 GB

■ 将 datafile_maxsize 设置为一个比 DATA_DISK_ALLOCATED 大的值（小于磁盘/data/1 的大小），同时设置 datafile_next 为 datafile_maxsize 的 20%。

```
ALTER SYSTEM SET datafile_maxsize='50G';
ALTER SYSTEM SET datafile_next='10G';
```

■ 查看修改后的 OBDServer 磁盘空间上限：

```
SELECT SVR_IP, CONCAT(ROUND((DATA_DISK_CAPACITY)/1024/1024/1024,1),' GB') AS
DATA_DISK_CAPACITY, CONCAT(ROUND((DATA_DISK_ALLOCATED)/1024/1024/1024,1),' GB') AS
DATA_DISK_ALLOCATED, CONCAT(ROUND((DATA_DISK_IN_USE)/1024/1024/1024,1),' GB') AS
DATA_DISK_IN_USE FROM GV$OB_SERVERS;
```

SVR_IP	DATA_DISK_CAPACITY	DATA_DISK_ALLOCATED	DATA_DISK_IN_USE
172.28.15.212	50.0 GB	20.0 GB	0.2 GB
172.28.15.214	50.0 GB	20.0 GB	0.0 GB
172.28.15.213	50.0 GB	20.0 GB	0.0 GB

步骤 4 在 sys 租户内查看与设置用户租户的配置项：freeze_trigger_percentage。

■ 查看配置项：

```
SHOW PARAMETERS LIKE 'freeze_trigger_percentage' TENANT='mysql';
SELECT * FROM GV$OB_PARAMETERS WHERE NAME='freeze_trigger_percentage' AND
TENANT_ID=(SELECT TENANT_ID FROM DBA_OB_TENANTS WHERE TENANT_NAME='mysql');
```

SVR_IP	SVR_PORT	ZONE	SCOPE	TENANT_ID	NAME	DATA_TYPE	VALUE
		SECTION	EDIT_LEVEL				
172.28.15.212	2882	zone1	TENANT	1002	freeze_trigger_percentage	NULL	20
		triggered. Rang:(0,100)	TENANT	DYNAMIC_EFFECTIVE			

- 修改配置项：

```
ALTER SYSTEM SET freeze_trigger_percentage=30 TENANT='mysql';
```

步骤 5 在用户租户内查看与设置租户配置项：freeze_trigger_percentage。

- 登录用户租户 mysql：

```
obclient -h<OBProxy ip> -P2883 -uroot@mysql#obcp_test -p -c -A -D oceanbase
```

- 查看配置项：

```
SHOW PARAMETERS LIKE 'freeze_trigger_percentage';  
SELECT * FROM GV$OB_PARAMETERS WHERE NAME = 'freeze_trigger_percentage';
```

SVR_IP	SVR_PORT	ZONE	SCOPE	TENANT_ID	NAME	DATA_TYPE	VALUE
		SECTION	EDIT_LEVEL				
172.28.15.212	2882	zone1	TENANT	1002	freeze_trigger_percentage	NULL	30
triggered. Rang:(0,100)		TENANT	DYNAMIC_EFFECTIVE				

- 修改配置项：

```
ALTER SYSTEM SET freeze_trigger_percentage=20;
```

5 思考讨论

思考题 1：在用户租户 1 中是否可以查看、修改用户租户 2 的配置项？

实验 3：容灾管理

3.1.备份恢复

1 实验目的

在本实验中，我们部署 NFS 来做备份目的地，分别使用 SQL 命令行和 OCP 云平台进行物理备份与恢复的操作。学员通过本实验的练习，了解租户恢复的前提条件，熟悉 OceanBase 数据库物理备份与恢复的流程。

2 实验环境

1 台 OCP 机器 + 3 台 OBServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。
- 完成实验 1.1 与 1.2 的操作。

4 实验步骤

步骤 1 本实验使用 OCP 服务器作为 NFS 服务器，相关配置均已提前完成，请重新启动 NFS

```
systemctl restart nfs-config  
systemctl restart nfs-server
```

步骤 2 在所有 OBServer 服务器上，已提前部署 NFS 客户端，可以执行以下操作验证 NFS 服务器挂载：

- 查看远端 NFS 服务器的信息

```
# showmount -e 172.28.15.211 <---NFS 服务器地址
```

```
[root@OBServer ~]# showmount -e 172.28.15.211  
Export list for 172.28.15.211:  
/data/nfs_server *
```

常见问题：服务器上防火墙或安全策略可能会阻止了 NFS 服务，可以使用 “systemctl stop firewalld”，“systemctl stop iptables” 命令在服务器或客户端上停止这些服务。

- 创建一个本地 /data/ob_backup 目录，然后执行挂载

```
mkdir /data/ob_backup  
mount -tnfs4 -o  
rw,nfsvers=4.1, sync, lookupcache=positive, hard, timeo=600, wsize=1048576, rsize=104857  
6, namlen=255 172.28.15.211:/data/nfs_server /data/ob_backup
```

验证：可以简单的在 NFS 服务器的/data/nfs_server 目录下创建一个文件，然后在 NFS 的各个客户端的/data/nfs 目录下是否可以看到。

步骤 3 登录预装了 sysbench 工具的 OCP 服务器，使用 sysbench 对 mysql 租户创建 10 张表并写入数据。

```
cd /usr/sysbench/share/sysbench

./oltp_insert.lua --mysql-host=<实际 OBProxy 地址> --mysql-port=2883 --mysql-
user=root@mysql#obcp_test --mysql-password=<root 用户密码> --mysql-db=test --db-ps-
mode=disable --report-interval=10 --mysql-ignore-errors=6002,6004,4012,2013,4016 -
-tables=10 --table_size=100000 --threads=10 --time=60 prepare
```

步骤 4 使用黑屏方式（SQL 命令）执行物理备份。

- 登录用户租户 mysql，设置日志归档目的地，开启日志归档。

```
obclient -h<OBProxy ip> -P2883 -uroot@mysql#obcp_test -p -c -A -D oceanbase

ALTER SYSTEM SET LOG_ARCHIVE_DEST='LOCATION=file:///data/ob_backup/mysql/archive';
ALTER SYSTEM ARCHIVELOG;

SELECT ROUND_ID,DEST_ID,PATH,STATUS,START_SCN_DISPLAY,CHECKPOINT_SCN_DISPLAY FROM
DBA_OB_ARCHIVELOG;
```

ROUND_ID	DEST_ID	PATH	STATUS	START_SCN_DISPLAY	CHECKPOINT_SCN_DISPLAY
1	1001	file:///data/ob_backup/mysql/archive	PREPARE	NULL	NULL

➤ STATUS 一开始为 PREPARE，然后进入 BEGINNING 状态，最后变成 DOING。

- 设置数据备份目的地，发起全量备份，并查看任务执行状态。

```
ALTER SYSTEM SET DATA_BACKUP_DEST='file:///data/ob_backup/mysql/data';
ALTER SYSTEM BACKUP DATABASE;
SELECT * FROM DBA_OB_BACKUP_JOBS;
SELECT * FROM DBA_OB_BACKUP_JOB_HISTORY;
```

TASK_ID	JOB_ID	INCARNATION	BACKUP_SET_ID	START_TIMESTAMP	END_TIMESTAMP	STATUS	ENCRYPTION_MODE	PASSWD	INPUT_BYTES	OUTPUT_BYTES	OUTPUT_RATE_BYTES	EXTRA_META_BYTES	TABLET_COUNT	LOCK_COUNT	FILE_COUNT	META_TURN_ID	DATA_TURN_ID	RESULT	COMMENT	PATH
1	1	1	1	2024-11-01 17:50:10.704154	NULL	BACKUP_DATA_MINO	NONE		354865845	159558552	0.0000	0	622	169	0	1	0	0		file:///data/ob_backup/mysql/data

步骤 5 使用白屏方式（OCP 云平台）执行物理备份并设置备份策略，访问路径：租户--备份恢复。

- OCP 会按照自己的命名规则在备份介质上创建日志归档和数据备份目录，在设置租户备份策略之前，需要首先**停止租户的日志归档**，以便 OCP 将日志归档写入到正确的归档目录。



- 新建租户的备份策略。

存储配置：设置有效的 NFS 存储目录，比如 /data/ob_backup/mysql，点击测试验证。

存储配置 选择已有配置 ▾

存储配置名称
daily_backup

存储类型
File OSS COS

存储目录
file:// /data/ob_backup/mysql 测试

备份存储容量报警阈值
80 %

调度周期：设置备份的频率以及全量与增量的类型。

调度时间
04:00:00

数据备份方式
建议至少配置 1 个全量备份

星期二: 全量 增量
星期四: 全量 增量
星期六: 全量 增量

自动合并 关闭时，在备份调度的过程中，始终不会自动发起合并，可能导致备份调度失败

同时发起日志备份 物理备份必须打开日志备份，且不可关闭

- 在备份策略触发备份前，手动执行一次全量备份。执行**立即备份**，备份任务大约需要 5 分钟完成。

× 立即备份租户

基础信息

备份租户

obcp_test:1729135359 / mysql

OceanBase 版本

4.2.1.7

备份方式 ?

物理备份

OceanBase 版本在 3.2.2-20220304134610 及以上的租户才能进行备份

存储配置

存储配置

OceanBase | daily_backup | file:///data/ob_backup...

测试

数据备份方式

数据备份方式

☒ 全量备份 ☐ 增量备份

取消

立即备份

步骤 6 使用白屏方式（OCP 云平台）执行恢复租户的操作。

- 在租户 mysql 的备份恢复管理页面执行**发起恢复**。

OceanBase 云平台

告警 任务中心 帮助 中文 admin

obcp_test:17... mysql 租户 ID: 1002 正常运行

备份恢复 功能介绍

发起恢复 立即备份 新建租户级备份策略

详情 备份策略

基本信息

状态: ● 等待备份调度 备份方式: 物理备份 备份文件: -

最近一次数据备份: 2024年11月1日 17:50:10 日志备份位点: 2024年11月2日 10:05:48 日志延时: 110s

数据备份调度历史: 查看

可恢复时间区间 < 2024年10月27日 ~ 2024年11月2日 >

10月27日 00:00:00 10月28日 00:00:00 10月29日 00:00:00 10月30日 00:00:00 10月31日 00:00:00 11月1日 00:00:00 11月2日 00:00:00 11月2日 23:59:59

● 可恢复时间区间 ● 数据备份时间点

- 设置恢复源信息，选择恢复日期为 mysql 租户的可恢复范围内的时间。

备份恢复 / 发起恢复

< 发起恢复

恢复源信息

发起恢复

源集群

源租户

obcp_test:1729135359

mysql:1002

恢复对象

租户

表

恢复日期

时分秒

微秒

2024年11月2日

11:22:00

0

所选恢复时刻: 2024年11月2日 11:22:00.000000

mysql 最近 7 天的可恢复时间: 2024年11月2日 11:20:40.417339 ~ 2024年11月2日 11:25:48.025240

物理备份

mysql 在 2024年11月2日的可恢复时间: 11:20:40.417339 ~ 11:25:48.025240

物理备份

- 设置恢复目标信息，将租户恢复到当前集群的 mysql_back 租户中，设定 Unit 规格为 1C3G。

恢复目标信息

目标集群

目标租户

obcp_test:1729135359

mysql_back

只有运行中单集群才能新建恢复租户

各 Zone 的 Unit 数量

1

当前可设最大个数为 1 (Zone 中最少 OBCServer 数决定 Unit 可设最大个数)

副本设置

	Zone 名称	副本类型	Unit 规格	Unit 数量
☒	zone1	全能型副本	u1C3G CPU/内存: 1C3G	1
☒	zone2	全能型副本	u1C3G CPU/内存: 1C3G	1
☒	zone3	全能型副本	u1C3G CPU/内存: 1C3G	1

- 提交租户恢复任务，大约需要 5 分钟完成。任务完成后，到集群的租户管理页面查看租户。

租户管理	租户列表				
性能监控	租户名	标签	OceanBase 版本号	租户模式	Zone 优先级
性能报告	mysql_back	+ 标签	4.2.1.7	MySQL	zone1,zone2,zone3
资源管理	mysql	+ 标签	4.2.1.7	MySQL	RANDOM
合并管理	sys	+ 标签	4.2.1.7	MySQL	RANDOM

步骤 7 使用 obclient 连接恢复出来的租户 mysql_back，查看其中包含的表和数据，并尝试执行更新操作。

```
obclient -h<OBProxy ip> -P2883 -uroot@mysql_back#obcp_test -p -c -A -D test

SHOW TABLES;

SELECT COUNT(*) FROM sbtest1;

UPDATE sbtest1 SET k=k+1 WHERE id=1;
```

```
+-----+
| Tables_in_test |
+-----+
| sbtest1        |
| sbtest10       |
| sbtest2        |
| sbtest3        |
| sbtest4        |
| sbtest5        |
| sbtest6        |
| sbtest7        |
| sbtest8        |
| sbtest9        |
+-----+
```

```
MySQL [test]> select count(*) from sbtest1;
+-----+
| count(*) |
+-----+
| 100000   |
+-----+
1 row in set (0.56 sec)
```

```
MySQL [test]> UPDATE sbtest1 SET k=k+1 WHERE id=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

5 思考讨论

思考题 1：为什么在开启数据备份前必须开启日志归档？

思考题 2：请从任务中心查看恢复任务的执行步骤，指出是什么操作导致了 OCP 恢复出来的 mysql_back 租户是一个独立的主租户，而不是 mysql 租户的备租户？

3.2.主备租户

1 实验目的

在 OceanBase V4 版本中，以租户为单位建立主备关系，主备库角色从主备集群变为了主备租户。我们可以为某一集群的主租户在任一集群内创建其对应的备租户，也可以使用主租户的物理备份数据恢复出一个备租户。

在本实验中，我们使用 OCP 云平台来简便地创建备租户，并执行主备租户的切换操作。

2 实验环境

1 台 OCP 机器 + 3 台 OBServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。
- 完成实验 1.1、1.2、3.1 的操作。

4 实验步骤

步骤 1 鉴于实验资源的限制，在创建新的租户之前，我们需要先将使用 3.1 中恢复出的租户 mysql_back 删掉。在 OCP 云平台的租户列表中选择 mysql_back，进入租户概览页面，执行删除租户的操作。



步骤 2 在集群的租户管理页面，执行新建租户的操作，为 mysql 租户在本集群中创建备租户 mysql_back。

- 在基本设置中，选择新建**备租户**，并指定备租户的同步放松为**基于归档**。
 - 网络直连：备租户即可以直接同步主租户的活跃日志，也可以同步主租户的归档日志。
 - 基于归档：备租户不读取主租户的活跃日志，只同步主租户的归档日志。

新建租户

基本设置

租户类型

主租户

备租户

主租户所属集群

obcp_test:1729135359

仅支持 OB 4.2 及以上版本配置租户级主备库

主租户

mysql

备租户所属集群

obcp_test:1729135359

仅支持 OB 4.2 及以上版本配置租户级主备库

备租户名称

mysql_back

主备同步方式

基于网络

基于归档

- 恢复备租户设置：OCP 自动选择主租户的归档与备份目录，点击备份集校验来确认备份数据完整。

恢复备租户设置

独立存储目录

备份集存储类型

File

OSS

COS

存储目录

file:// /data/ob_backup/mysql

已自动配置平台保存的最新备份集的存储目录

备份集校验

校验完成

■ 备租户副本设置：选择 1C3G 的 Unit 规格，可以设置与主租户不同的 Zone 优先级次序。

备租户副本设置

建议与主租户副本规格保持一致

Zone 名称	副本类型	Unit 规格	Unit 数量
zone1	全能型副本	u1C3G CPU/内存: 1C3G	1
zone2	全能型副本	u1C3G CPU/内存: 1C3G	1
zone3	全能型副本	u1C3G CPU/内存: 1C3G	1

Zone 优先级排序

Zone

同时选择多个 Zone 穿梭到右侧后，可设置为同一优先级

优先级排序

优先级从上到下代表从高到低，可拖拽排序

恢复为默认

zone1,zone2,zone3

32

- 提交备租户创建任务，大约需要 5 分钟完成。
- 任务完成后，在租户的拓扑图页面可以看到主备租户的拓扑关系。



步骤 3 使用 obclient 连接备租户，查看备租户中的表和数据；尝试执行更新操作，SQL 报错不能更新。

```
obclient -h<OBProxy ip> -P2883 -uroot@mysql_back#obcp_test -p -c -A -D test

SHOW TABLES;

SELECT COUNT(*) FROM sbtest1;

UPDATE sbtest1 SET k=k+1 WHERE id=1;
```

```
MySQL [test]> select count(*) from sbtest1;
+-----+
| count(*) |
+-----+
|   100000 |
+-----+
1 row in set (0.00 sec)

MySQL [test]> update sbtest1 set k=k+1 where id=1;
ERROR 4688 (HY000): standby tenant is read only
```

步骤 4 执行主备租户的 SWITCHOVER。

- 点击主租户 mysql 的租户概览页面右上角的 ...，选择日常切换。



- 指定将主租户从 obcp_test.mysql 切换到 obcp_test.mysql_back。



- 主备库切换要求备库与主库的同步延迟小于 5s，如果归档的延迟大于 5s，可能会弹出以下报错：



- 在租户的参数管理页面，搜索参数 archive_lag_target，将其从默认值 120s 修改为 2s。修改参数后，再次重试，报错消失。



- 基于归档的主备库在执行切换时，需要首先为备租户配置日志归档。本实验中依然使用 NFS 服务器

/data/ob_backup 做为备租户的归档地址。点击测试，测试通过后即可执行切换操作。切换操作大约 30 秒内完成。

日常切换

原主租户: obcp_test.mysql

新主租户: obcp_test.mysql_back

obcp_tes...

原主租户 → 新备租户

>>

obcp_te...

原备租户 → 新主租户

备租户未配置日志归档，不支持切换，请配置归档地址:

存储类型

File

OSS

COS

日志备份目录

file://

请输入

/obcp_test/1729135359/tenant_incarnation_1/1006/clog

测试

/data/ob_backup

取消

开始切换

➤ 如果切换任务在 set log backup destination 时报如下错误:

Prepare switch... 执行中

ID: 14261 Prepare swi...

开始: 13:59:18 耗时: 0.04秒

Set log backup ... 执行中

ID: 14264 Set log bac...

开始: 13:59:19 耗时: 16分钟23秒

Start log bac... 执行成功

ID: 14260 Start log ba...

开始: 14:15:43 耗时: 10秒

Switch primary t... 执行中

ID: 14258 Switch prim...

开始: 14:15:54 耗时: 0.31秒

Wait primary ten... 执行中

ID: 14263 Wait primar...

开始: 14:15:55 耗时: 10秒

Set log backup destination ID: 14264

RETRY 14:15:41 RETRY 14:05:00 RETRY 14:00:47

异常信息

错误码: OBE10002

错误信息: 操作OceanBase失败, 错误信息: (conn=1056315) the format file does not exist under the destination

错误原因: 连接 mysql_back 执行 ALTER SYSTEM SET log_archive_dest = ? 失败, 错误信息: (conn=1056315) the format file does not exist under the destination

解决方案: 请联系技术支持

则需要在 NFS 服务器中删除 mysql_back 租户在 NFS 服务器中的日志归档目录。

```
cd /data/nfs_server/obcp_test/1729135359/tenant_incarnation_1;ll
total 8
drwx----- 4 nfsnobody nfsnobody 4096 Nov  2 10:49 1002
drwx----- 3 nfsnobody nfsnobody 4096 Nov  2 13:59 1006
rm -rf 1006
```

35

步骤 5 执行主备租户的 FAILOVER。

- 点击当前备租户 mysql 的租户概览页面右上角的 ...，选择容灾切换。



- 执行切换时提示主租户正在运行中，不允许执行。



- 转而选择主备解耦，在任务中心查看主备解耦的操作过程。



- 解耦操作完成后，到集群的租户管理页面查看当前的租户主备角色，发现租户 mysql 与 mysql_back 均为主租户。

租户列表

租户名	标签 	OceanBase 版本号	租户模式 	Zone 优先级 
mysql_back  	+ 标签	4.2.1.7	MySQL	zone1,zone2,zone3
mysql  	+ 标签	4.2.1.7	MySQL	RANDOM
sys  	+ 标签	4.2.1.7	MySQL	RANDOM

5 思考讨论

思考题 1： SWITCHOVER 与 FAILOVER 的区别是什么，分别应该在什么样的场景下使用？

思考题 2： 观察 OCP 主备解耦的操作过程，请说明主备解耦的具体流程是什么？

实验 4：监控诊断

4.1.全链路追踪

1 实验目的

本实验让学员体验 OceanBase V4 的全链路追踪的能力，使用全链路追踪查看 SQL 在执行过程中在各个网络节点、执行节点的耗时，并了解 SQL 解析、执行的过程以及分布式执行的特点。

2 实验环境

1 台 OCP 机器 + 3 台 OBServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。
- 完成实验 1.1、1.2、3.1 的操作。

4 实验步骤

步骤 1 使用 obclient 连接 mysql 租户。

```
obclient -h<OBProxy ip> -P2883 -uroot@mysql#obcp_test -p -c -A -D test
```

步骤 2 查看租户默认的全链路 Trace 收集策略。

```
SELECT TENANT_NAME,LEVEL,SAMPLE_PERCENTAGE,RECORD_POLICY
FROM GV$OB_FLT_TRACE_CONFIG;
```

```
+-----+-----+-----+-----+
| TENANT_NAME | LEVEL | SAMPLE_PERCENTAGE | RECORD_POLICY |
+-----+-----+-----+-----+
| mysql      | 1    | 10                | SAMPLE_AND_SLOW_QUERY |
+-----+-----+-----+-----+
```

步骤 3 设置当前 Session 的全链路 Trace 收集策略为搜集所有 SQL 执行的 Trace，并确认设置生效。

```
CALL DBMS_MONITOR.OB_SESSION_TRACE_ENABLE(NULL,3,1,'ALL');

SELECT SVR_IP, TENANT, DB, USER, ID, STATE, LEVEL, SAMPLE_PERCENTAGE,
RECORD_POLICY FROM GV$OB_PROCESSLIST WHERE ID=CONNECTION_ID();
```

```
+-----+-----+-----+-----+-----+-----+-----+-----+
| SVR_IP    | TENANT | DB      | USER | ID       | STATE | LEVEL | SAMPLE_PERCENTAGE | RECORD_POLICY |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 172.28.15.212 | mysql | oceanbase | root | 3221673852 | ACTIVE | 3    | 100              | ALL          |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

- 使用 CONNECTION_ID()函数返回当前 Session 的 ID。

步骤 4 执行以下 SQL 语句。

```
USE test;

SELECT count(*) FROM sbtest1 t1,sbtest2 t2;
```

步骤 5 在 SELECT 执行结束后，使用 SHOW TRACE 命令查看 SQL 执行的全链路 Trace，发现结果为空。

```
SHOW TRACE;
```

步骤 6 打开当前 Session 的在线 show trace 功能，并再次 SHOW TRACE，发现结果依然为空。

```
SET ob_enable_show_trace=on;

SHOW TRACE;
```

步骤 7 再次执行 SELECT，并在结束后立刻执行 SHOW TRACE 命令，终于可以看到全链路跟踪的 Trace。

```
SELECT count(*) FROM sbtest1 t1,sbtest2 t2;

SHOW TRACE;
```

Operation	StartTime	ElapseTime
com_query_process	2024-11-06 16:49:01.592724	10006.382 ms
└─ mpquery_single_stmt	2024-11-06 16:49:01.592728	10006.368 ms
└─ sql_compile	2024-11-06 16:49:01.592740	5.384 ms
└─ pc_get_plan	2024-11-06 16:49:01.592743	0.006 ms
└─ hard_parse	2024-11-06 16:49:01.592794	5.320 ms
└─ parse	2024-11-06 16:49:01.592794	0.046 ms
└─ resolve	2024-11-06 16:49:01.592857	0.227 ms
└─ rewrite	2024-11-06 16:49:01.593122	0.284 ms
└─ optimize	2024-11-06 16:49:01.593419	0.807 ms
└─ code_generate	2024-11-06 16:49:01.594242	0.122 ms
└─ pc_add_plan	2024-11-06 16:49:01.598012	0.082 ms
└─ sql_execute	2024-11-06 16:49:01.598135	10000.870 ms
└─ open	2024-11-06 16:49:01.598136	0.055 ms
└─ response_result	2024-11-06 16:49:01.598199	10000.247 ms
└─ get_das_id	2024-11-06 16:49:01.598207	0.000 ms
└─ do_local_das_task	2024-11-06 16:49:01.598218	0.092 ms
└─ get_das_id	2024-11-06 16:49:01.599994	0.000 ms
└─ do_local_das_task	2024-11-06 16:49:01.600020	0.494 ms
└─ close	2024-11-06 16:49:11.598508	0.375 ms
└─ close_das_task	2024-11-06 16:49:11.598511	0.171 ms
└─ close_das_task	2024-11-06 16:49:11.598694	0.062 ms
└─ end_transaction	2024-11-06 16:49:11.598868	0.003 ms

- 请从 Trace 信息中获取 SQL 编译（解析、生成执行计划）的耗时，和 SQL 真正的执行耗时。

步骤 8 登录 OCP 云平台，从左侧功能列表中选择：日志服务—链路查询，选择 mysql 租户最近 10 分钟内的链路信息。如果 SELECT 语句超时出错，也可选择“仅查看出错链路”。

日志服务

日志查询 [链路查询](#)

时间范围: 近 10 分钟 2024年11月6日 16:4 → 2024年11月6日 16:5 目标租户: obcp_test:1729135359 / mysql

搜索内容: sql_id 请输入 链路耗时: 大于 0ms 仅查看出错链路: ☒ 重置 查询

步骤 9 从结果中选择执行耗时与 SELECT 执行耗时接近的最近一条 trace id，查看其全链路 Trace 信息。

× 链路详情

Trace ID: 0006263a-96e1-a24f-1c7b-ff6f9671683c Span 数: 22 耗时: 10006.382ms 出错 正常

SQL ID: 18267A34B4FEB4B1B4ADAB3D279F4F51

Span 名称	节点	执行时间线
com_query_process	OBServer, 172.28.15.214	10006.382ms
mpquery_single_stmt	OBServer, 172.28.15.214	10006.368ms
sql_compile	OBServer, 172.28.15.214	5.384ms
pc_get_plan	OBServer, 172.28.15.214	0.006ms
hard_parse	OBServer, 172.28.15.214	5.32ms
parse	OBServer, 172.28.15.214	0.046ms
resolve	OBServer, 172.28.15.214	0.227ms
rewrite	OBServer, 172.28.15.214	0.284ms
optimize	OBServer, 172.28.15.214	0.807ms
code_generate	OBServer, 172.28.15.214	0.122ms
pc_add_plan	OBServer, 172.28.15.214	0.082ms
sql_execute	OBServer, 172.28.15.214	10000.87ms
open	OBServer, 172.28.15.214	0.055ms
response_result	OBServer, 172.28.15.214	10000.247ms
get_das_id	OBServer, 172.28.15.214	0ms
do_local_das_task	OBServer, 172.28.15.214	0.092ms
get_das_id	OBServer, 172.28.15.214	0ms
do_local_das_task	OBServer, 172.28.15.214	0.494ms
close	OBServer, 172.28.15.214	0.375ms
close_das_task	OBServer, 172.28.15.214	0.171ms
close_das_task	OBServer, 172.28.15.214	0.062ms
end_transaction	OBServer, 172.28.15.214	0.003ms

5 思考讨论

思考题 1: 如果设置 ob_enable_show_trace=off，租户和 Session 设置的全链路搜集策略是否还有效果，是否还能获得 SQL 执行的全链路 Trace？

思考题 2: 怎样正确使用 SHOW TRACE 命令查看 SQL 执行的全链路 Trace？

4.2.SQL 监控

1 实验目的

SQL 监控与调优是 DBA 日常工作的重要内容，本实验模拟一个慢 SQL 的场景，让学员熟悉实时查询慢 SQL 的执行计划、执行统计的操作，以及如何在 SQL 执行结束后追查慢 SQL 的执行情况。

2 实验环境

1 台 OCP 机器 + 3 台 OBServer 服务器

3 实验前提

- 完成前置实验准备中相关软件配置及服务器初始化。
- 完成实验 1.1、1.2、3.1 的操作。

4 实验步骤

步骤 1 打开两个连接 mysql 租户的会话。

```
obclient -h<OBProxy ip> -P2883 -uroot@mysql#obcp_test -p -c -A -D test
```

步骤 2 会话 1：执行一个慢 SQL，以便于会话 2 在 SQL 执行时进行监控。比如：

```
select /*+query_timeout(6000000000)*/ count(*) from sbtest1 t1,sbtest2 t2 where  
t1.id<>t2.id and t1.k>t2.k ;
```

步骤 3 会话 2：在慢 SQL 的执行过程中，查询会话 1 的 id(Session ID)、trace_id、sql_id 等信息。

```
SELECT svr_ip,svr_port, id, state, command, time, total_time, sql_id, trace_id,  
info FROM oceanbase.GV$OB_PROCESSLIST WHERE state = 'active';
```

MySQL [test]> SELECT svr_ip,svr_port, id, state, command, time, total_time, sql_id, trace_id, info FROM oceanbase.GV\$OB_PROCESSLIST WHERE state = 'active';									
svr_ip	svr_port	id	state	command	time	total_time	sql_id	trace_id	info
172.28.15.214	2882	3222267987	ACTIVE	Query	0.011064	0.011111	A1EF0243B2CAFBA5505C749DA0CC4311	YB42AC1C0FD6-000625D4B1BD0145-0-0	SELECT svr_ip,
172.28.15.214	2882	3222267850	ACTIVE	Query	8.596478	8.59653	34534DEC9B05F5FE15833AB4B2984462	YB42AC1C0FD6-000625D4B24D0185-0-0	select /**que

步骤 4 会话 2：查看会话 1 中当前执行 SQL 的执行计划和执行统计，比较估算代价与实际代 sql_id 价的差异。

```
SELECT DBMS_XPLAN.DISPLAY_ACTIVE_SESSION_PLAN(3222267850, 'typical',  
'172.28.15.214', '2882');
```

ID	OPERATOR	NAME	EST. ROWS	EST. TIME(us)	REAL. ROWS	REAL. TIME(us)	IO TIME(us)	CPU TIME(us)
0	EXCHANGE IN REMOTE		1	386314821	0	0	0	0
1	EXCHANGE OUT REMOTE		1	386314820	712403646	0	0	0
2	SCALAR GROUP BY		1	386314820	38501	0	0	0
3	NESTED-LOOP JOIN		3333300000	325901981	712410785	505055155	0	0
4	TABLE FULL SCAN	t1(k_1)	100000	6387	0	0	0	0
5	DISTRIBUTED TABLE RANGE SCAN	t2(k_2)	33334	2145	0	0	0	0

步骤 5 会话 2: 使用 GV\$OB_PROCESSLIST 中查到的 trace_id, 在 GV\$SQL_PLAN_MONITOR 查看会话 1 中慢 SQL 的执行算子的实时统计。

```
SELECT PLAN_LINE_ID, PLAN_OPERATION, SVR_IP, OUTPUT_ROWS, STARTS RESCAN_TIMES,
LAST_REFRESH_TIME - FIRST_REFRESH_TIME ELAPSED_TIME FROM
oceanbase.GV$SQL_PLAN_MONITOR WHERE TRACE_ID = 'YB42AC1C0FD6-000625D4B24D0185-0-0'
ORDER BY PLAN_LINE_ID,SVR_IP;
```

PLAN_LINE_ID	PLAN_OPERATION	SVR_IP	ELAPSED_TIME	OUTPUT_ROWS	RESCAN_TIMES
0	PHY_DIRECT_RECEIVE	172.28.15.212	5609.066862	0	0
0	PHY_SCALAR_AGGREGATE	172.28.15.214	NULL	0	0
1	PHY_NESTED_LOOP_JOIN	172.28.15.214	NULL	3926900921	0
2	PHY_TABLE_SCAN	172.28.15.214	NULL	89377	0
3	PHY_TABLE_SCAN	172.28.15.214	NULL	3926940189	88432

步骤 6 会话 1: 使用 Ctrl+C 取消当前慢 SQL 的执行。

步骤 7 任意会话: 在 GV\$OB_SQL_AUDIT 中查看慢 SQL 的整体执行情况。

```
SELECT SVR_IP, SVR_PORT, USEC_TO_TIME(REQUEST_TIME) AS REQUEST_TIME, TENANT_ID,
TRACE_ID, SQL_ID, PLAN_ID, PLAN_TYPE, ELAPSED_TIME, QUEUE_TIME, EXECUTE_TIME,
EXECUTE_TIME-TOTAL_WAIT_TIME_MICRO AS CPU_TIME, TOTAL_WAIT_TIME_MICRO,
RETURN_ROWS, AFFECTED_ROWS, RET_CODE, QUERY_SQL FROM oceanbase.GV$OB_SQL_AUDIT
WHERE SQL_ID= '34534DEC9BD5F5FE15833AB4B2984462' AND TRACE_ID='YB42AC1C0FD4-
000625D4ABAF865C-0-0';
```

SVR_IP	SVR_PORT	REQUEST_TIME	TENANT_ID	TRACE_ID	SQL_ID	PLAN_ID	PLAN_TYPE	ELAPSED_TIME
172.28.15.214	2882	2024-11-07 13:06:36.200016	1002	YB42AC1C0FD6-000625D4B24D0185-0-0	34534DEC9BD5F5FE15833AB4B2984462	408	1	156793849

步骤 8 任意会话: 使用上一步从 GV\$OB_SQL_AUDIT 查询得到的 svr_ip、svr_port、tenant_id、plan_id, 在 GV\$PLAN_CACHE_PLAN_EXPLAIN 中查看慢 SQL 的物理执行计划。

```
SELECT PLAN_ID, PLAN_LINE_ID, OPERATOR, NAME, ROWS, COST FROM
oceanbase.GV$OB_PLAN_CACHE_PLAN_EXPLAIN WHERE SVR_IP='172.28.15.214' AND
SVR_PORT=2882 AND TENANT_ID=1002 AND PLAN_ID=408 ORDER BY PLAN_LINE_ID;
```

PLAN_ID	PLAN_LINE_ID	OPERATOR	NAME	ROWS	COST
2696	0	PHY_DIRECT_RECEIVE	NULL	1	386314820
2696	1	PHY_DIRECT_TRANSMIT	NULL	1	386314819
2696	2	PHY_SCALAR_AGGREGATE	NULL	1	386314819
2696	3	PHY_NESTED_LOOP_JOIN	NULL	3333300000	325901980
2696	4	PHY_TABLE_SCAN	t1(k_1)	100000	6386
2696	5	PHY_TABLE_SCAN	t2(k_2)	33334	2144

步骤 9 登录到慢 SQL 执行所在的 OBC 服务器（示例中为 172.28.15.212），在对应时间的 observer 进程日志中检索慢 SQL 执行的 trace_id。

```
cd /home/admin/oceanbase/log;
```

```
ll -rt observer.log*

grep YB42AC1C0FD6-000625D4B24D0185-0-0 observer.log.20241107131212869

[2024-11-07 13:09:12.993858] WDIAG [SERVER] do_process (obmp_query.cpp:808) [11575][T1002_L0_G0][T1002][YB42AC1C0FD6-000625D4B24D0185-0-0] [lt=10][errcode=-5065] execute query fail(ret=-5065, timeout_timestamp=1730961996200016)
[2024-11-07 13:09:12.993982] TRACE [TRACE] after_process (obmp_base.cpp:145) [11575][T1002_L0_G0][T1002][YB42AC1C0FD6-000625D4B24D0185-0-0] [lt=7] [slow query](TRACE=begin_ts=1730955996200036 2024-11-07 05:06:36.200036|[process_begin] u=0 in_queue_time:19, receive_ts:1730955996200016, enqueue_ts:1730955996200020|[start_sql] u=0 addr:{ip:"172.28.15.212", port:31942}|[query_begin] u=1 trace_id:YB42AC1C0FD6-000625D4B24D0185-0-0|[before_processor_run] u=5 |[session] u=2 sid:3222267850, tenant_id:1002|[parse_begin] u=25 stmt:"select /*+query_timeout(6000000000)*/ count(*) from sbtest1 t1,sbtest2 t2 where t1.id<>t2.id and t1.k>t2.k", stmt_len:106|[transform_with_outline_begin] u=100 |[transform_with_outline_end] u=2 |[resolve_begin] u=8 |[resolve_end] u=337 |[transform_begin] u=40 |[tl_calc_part_id_end] u=1057 |[get_location_cache_begin] u=0 |[get_location_cache_end] u=7 |[get_location_cache_begin] u=217 |[tl_calc_part_id_end] u=15 |[storage_estimation_begin] u=3 |[storage_estimation_end] u=26 |[tl_calc_part_id_end] u=271 |[get_location_cache_begin] u=0 |[get_location_cache_end] u=6 |[tl_calc_part_id_end] u=156 |[storage_estimation_begin] u=3 |[storage_estimation_end] u=14 |[tl_calc_part_id_end] u=311 |[tl_calc_part_id_end] u=9 |[storage_estimation_begin] u=3 |[storage_estimation_end] u=37 |[tl_calc_part_id_end] u=176 |[tl_calc_part_id_end] u=830 |[storage_estimation_begin] u=3 |[storage_estimation_end] u=12 |[tl_calc_part_id_end] u=167 |[tl_calc_part_id_end] u=8 |[tl_calc_part_id_end] u=5 |[cg_end] u=287 |[plan_id] u=254 plan_id:408|[tl_calc_part_id_end] u=7 |[do_open_plan_begin] u=12 plan_id:408|[sql_start_stmt_begin] u=1 |[sql_start_stmt_end] u=602 |[exec_plan_begin] u=0 |[exec_plan_end] u=13 |[do_open_plan_end] u=0 |[storage_table_scan_begin] u=29 |[S_table_scan_begin] u=8 |[S_table_scan_end] u=27 |[storage_table_scan_end] u=1 |[storage_table_scan_begin] u=3305 |[S_table_scan_begin] u=7 |[S_table_scan_end] u=309 |[storage_table_scan_end] u=0 |[close_plan_begin] u=156783839 |[S_revert_iter_begin] u=4 |[S_revert_iter_end] u=179 |[S_revert_iter_begin] u=8 |[S_revert_iter_end] u=108 |[start_end_stmt] u=6 |[end_stmt] u=728 |[close_plan_end] u=49 |[auto_end_plan_begin] u=1 |[auto_end_plan_end] u=5 |[session] u=80 sid:3222267850, tenant_id:1002|[query_end] u=211 |[session] u=4 sid:3222267850, tenant_id:1002|[process_end] u=4 run_ts:1730955996200042|total_timeu=156793944)
```

➤ [slow query]日志里详细统计了 sql 各个执行阶段的执行耗时。

步骤 10 登录到 OCP 云平台，从租户页面选择“SQL 诊断”功能，使用 OCP 的智能 SQL 诊断功能查看刚刚执行的慢 SQL。

TopSQL

SlowSQL

可疑 SQL

ParallelSQL

☐ SQL 文本 ②	数据库 ②	总执行次数 ②	总响应时间 (ms) ②	最大响应时间 (ms) ②	最大返回行数 ②
☐ select /*+query_timeout...	test	1	186013.35	186,013.35	0
☐ KILL ?	NO_database	1	967.4	967.4	0
☐ SELECT * FROM oceanb...	test	1	297.85	297.85	1

×

Plan Hash 执行计划详情

基本信息

Plan Hash: 5685911676540044665

合并版本: 0

执行计划类型: 本地执行计划

计划生成时间: 2024年11月7日 11:53:36

CPU 时间: 2155590ms

执行步骤

算子	名称	预估行	代价 ②	输出 & 过滤
PHY_SCALAR_AGGREGATE	NULL	1	386314819	NULL
PHY_NESTED_LOOP_JOIN	NULL	3333300000	325901980	NULL
PHY_TABLE_SCAN	t1(k_1)	100000	6386	table_rows:100000, physical_range_rows:100000, logical_r...
PHY_TABLE_SCAN	t2(k_2)	33334	2144	table_rows:100000, physical_range_rows:33333, logical_ra...

5 思考讨论

思考题 1：对整个系统进行 SQL 执行的监控和优化是 DBA 日常的工作内容之一，OceanBase 数据库提供了哪些手段来管理 SQL 的执行计划，保证最优的执行计划可以被选用？