

# 阿里云

## 云数据库 PolarDB 最佳实践

文档版本：20220713

## 法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，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。
3. 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。阿里云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，并在阿里云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阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

# 通用约定

格式	说明	样例
 危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 危险 重置操作将丢失用户配置数据。
 警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	 警告 重启操作将导致业务中断，恢复业务时间约十分钟。
 注意	用于警示信息、补充说明等，是用户必须了解的内容。	 注意 权重设置为0，该服务器不会再接受新请求。
 说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击设置> 网络> 设置网络类型。
粗体	表示按键、菜单、页面名称等UI元素。	在结果确认页面，单击确定。
Courier字体	命令或代码。	执行 <code>cd /d C:/window</code> 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	<code>bae log list --instanceid</code> <code>Instance_ID</code>
[] 或者 [a b]	表示可选项，至多选择一个。	<code>ipconfig [-all -t]</code>
{ } 或者 {a b}	表示必选项，至多选择一个。	<code>switch {active stand}</code>

# 目录

1. 远程获取并解析PolarDB MySQL引擎Binlog日志	05
2. 订阅PolarDB MySQL引擎集群日志	10
3. 互联网行业高弹性系统架构最佳实践	15
4. 数据误操作时如何进行恢复	31
4.1. 不同场景下恢复方式概览	31
4.2. 使用DMS数据追踪进行恢复	31
4.3. 集群恢复的方式恢复整个集群	34
4.3.1. 整体流程和预估时间	34
4.3.2. 集群恢复：按时间点恢复	37
4.3.3. 集群恢复：按备份集恢复	44
4.4. 库表恢复的方式恢复对应表	49
4.4.1. 整体流程和预估时间	49
4.4.2. 库表恢复：按时间点恢复	50
4.4.3. 库表恢复：按备份集恢复	55
4.5. 从集群回收站恢复误删的集群	58
4.6. 从表回收站中恢复误删的表	61
5. 巡检问题分析最佳实践	64
5.1. PolarDB MySQL引擎CPU使用率高	64
5.2. PolarDB代理配置与流量异常	67
5.3. PolarDB死锁	71

# 1.远程获取并解析PolarDB MySQL引擎Binlog日志

本文介绍如何远程获取PolarDB MySQL引擎Binlog日志，并通过mysqlbinlog工具查看并解析Binlog日志。

## Binlog获取策略

连接方式	Binlog获取策略
<ul style="list-style-type: none"><li>主地址</li><li>默认集群地址</li><li>包含主节点的自定义集群地址</li></ul>	均从主节点获取。 <div><p>说明 如何设置集群连接地址，请参见配置数据库代理。</p></div>
仅包含只读节点的自定义集群地址	从任意只读节点获取。 <div><p>说明 集群版本需为以下版本之一：</p><ul style="list-style-type: none"><li>PolarDB MySQL引擎8.0版本且Revision version为8.0.1.1.12或以上</li><li>PolarDB MySQL引擎5.7版本且Revision version为5.7.1.0.12或以上</li></ul><p>否则，将返回 Only allow to dump binary log on primary instance 报错。</p><p>关于如何查看集群版本和升级集群版本，请参见版本管理。</p></div>

## 远程获取Binlog日志

1. 为PolarDB MySQL引擎集群开启Binlog，详细操作步骤，请参见开启Binlog。

说明 您需要先为集群开启Binlog才能查看并获取Binlog日志，否则将出现 You are not using binary logging 错误提示。

2. 在本地服务器上安装MySQL。
3. 通过MySQL客户端连接PolarDB集群，详细操作步骤，请参见使用客户端连接集群。本文以Linux系统为例。

```
root@~:~# mysql -h polardb.rwlb.rds.aliyuncs.com -P3306 -utest_api -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 
Server version: 8.0.13 Source distribution

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

4. 在已连接PolarDB集群的客户端中执行如下命令查看目标集群的Binlog文件列表：

```
show binary logs;
```

返回结果如下：

```
+-----+-----+
| Log_name          | File_size |
+-----+-----+
| mysql-bin.000005 |      2639 |
+-----+-----+
1 row in set (0.00 sec)
```

5. 在客户端执行如下命令，远程获取Binlog日志并保存至本地。

本文中以Linux系统为例，在执行如下语句前，需先执行exit退出MySQL后，才能远程获取Binlog日志并保存至本地。

```
mysql> exit
Bye
root@~:~# mysqlbinlog -utest_api -p -h -polardb.rwlb.rds.aliyuncs.com --read-from-remote-server --raw mysql-bin.000005
Enter password:
root@~:~#
```

```
mysqlbinlog -u<用户名> -p<密码> -h<连接地址> --read-from-remote-server --raw mysql-bin.**
****
```

示例：

```
mysqlbinlog -utest_api -p -h test-polardb.rwlb.rds.aliyuncs.com --read-from-remote-server --raw mysql-bin.000005
```

参数	说明	示例值
-u	PolarDB MySQL引擎集群中的账号名称。如何创建账号，请参见 <a href="#">创建数据库账号</a> 。	test_api
-p	以上账号的密码。如果此处留空，则在执行此命令后会被要求输入。	TestPwd123

参数	说明	示例值
<code>-h</code>	<p>PolarDB MySQL引擎集群的公网连接地址。</p> <div><p><b>说明</b></p><ul style="list-style-type: none"><li>若您的PolarDB MySQL引擎集群连接地址使用的是默认的端口号3306，端口号可省略不写，否则需要在连接地址后加上端口号。</li><li>当前仅支持通过主地址或集群地址（包括默认集群地址和自定义集群地址）的公网连接地址远程获取Binlog。如何申请公网连接地址，请参见<a href="#">申请集群地址和主地址</a>。</li></ul></div>	<code>test-polaradb.rwlb.rds.aliyuncs.com</code>
<code>--raw</code>	表示将获取到的Binlog文件按数据原有格式打印，而不会展示解析后的数据。	<code>--raw</code>
<code>mysql-bin.*****</code>	通过 <code>show binary logs;</code> 命令获取的目标Binlog文件的名字，即 <code>Log_name</code> 。	<code>mysql-bin.000005</code>

使用mysqlbinlog工具查看、解析Binlog日志

- 执行如下命令，通过mysqlbinlog工具查看Binlog日志文件内容：

```
mysqlbinlog -vv --base64-output=decode-rows mysql-bin.***** | more
```

**说明**

- `-vv`：查看具体SQL语句及备注。
- `--base64-output=decode-rows`：解析Binlog日志文件。

具体的Binlog日志内容如下图中红框所示部分：

```
[root@iz... ~]# mysqlbinlog -vv --base64-output=decode-rows mysql-bin.000110 | more
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET @@OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#160217 23:04:37 server id 2802943055 end_log_pos 107 Start: binlog v 4, server v 5.5.18.1-log created 160217 23:04:37
# at 107
#160217 23:04:38 server id 2802943055 end_log_pos 171 Query thread id=584632 exec time=0 error code=0
SET TIMESTAMP=1455721478/*!*/;
SET @@session.pseudo_thread_id=584632/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=0, @@session.unique_checks=1, @@session.autocommit=1/*!*/;
SET @@session.sql_mode=2097152/*!*/;
SET @@session.auto_increment_increment=1, @@session.auto_increment_offset=1/*!*/;
/*!\C utf8 *//*!*/;
SET @@session.character_set_client=33,@@session.collation_connection=33,@@session.collation_server=33/*!*/;
SET @@session.lc_time_names=0/*!*/;
SET @@session.collation_database=DEFAULT/*!*/;
BEGIN
/*!*/;
# at 171
```

- 解析Binlog日志

关于如何解析Binlog日志，请参见[解析Binlog日志](#)。

## 常见问题

- Q: 执行 `show binary logs;` 后，为什么会提示 `You are not using binary logging` 错误信息？

A: PolarDB集群默认关闭Binlog参数，请确保您已[开启Binlog](#)。

❓ 说明 开启或关闭Binlog后，集群会自动重启使参数新配置生效。建议您在业务低谷期进行操作并确保应用程序具备重连机制。

- Q: 为什么远程获取Binlog日志失败，并出现如下错误提示？

```
ERROR: Error in Log_event::read_log_event(): 'Sanity check failed', data_len: 151, event_type: 35
ERROR: Could not read entry at offset 120: Error in log format or read error.
ERROR: Got error reading packet from server: 'Slave can not handle replication events with the checksum that master is configured to log;
```

A: 当您使用的mysqlbinlog工具版本过低时，可能会出现以上错误提示中的任意一种。请检查您使用的mysqlbinlog工具版本，建议您升级到较高版本的mysqlbinlog工具再尝试远程获取Binlog日志。例如使用Ver 3.3版本出现如上错误提示，您可以将mysqlbinlog工具升级至Ver 3.4版本进行查看。

- Q: 为什么我看到的Binlog日志内容未经过解析？

A: 若在查看Binlog日志时，未使用 `--base64-output=decode-rows` 参数，导出的Binlog日志将会显示未经解析的日志（如下图所示）。请确保在使用mysqlbinlog工具查看Binlog日志时，已使用 `--base64-output=decode-rows` 参数。

```
[root@iz... ~]# mysqlbinlog -vv mysql-bin.000110 | more
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET @@OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#160217 23:04:37 server id 2802943055 end_log_pos 107 Start: binlog v 4, server v 5.5.18.1-log created 160217 23:04:37
BINLOG '
BYzEVg9PhBgNzWAAAGsAAAAAAQANS41LjE4LjEtbG9nAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAEzgNAAGAEgAEBAQEEgAAVAAGggAAAAICAgCAA==
'/*!*/;
# at 107
#160217 23:04:38 server id 2802943055 end_log_pos 171 Query thread_id=584632 exec_time=0 error_code=0
```

- Q: 从只读节点获取Binlog时，为什么出现 `Only allow to dump binary log on primary instance` 错误提示？



A：当集群版本为以下版本之一时：

- PolarDB MySQL引擎8.0版本且Revision version为8.0.1.1.12或以上
- PolarDB MySQL引擎5.7版本且Revision version为5.7.1.0.12或以上

支持从只读节点获取Binlog。如何查看集群版本和升级集群版本，请参见[版本管理](#)。

- Q：从只读节点获取的Binlog时，为什么有时候结果会与从主节点上获取的不一致？

A：PolarDB支持在主节点和只读节点间共享Binlog，当主节点上的Binlog源信息同步到只读节点后，才能从只读节点获取Binlog。当出现同步延迟（源信息数据量很小，延迟一般不会很大）时，只读节点可能会读不到主节点上新写入的Binlog。

- Q：为什么在超出保存时长参数（binlog\_expire\_logs\_seconds）设置的时间后，Binlog并没有被删除？

A：若只读节点上有线程正在获取Binlog，该Binlog不会被立即删除。当主节点写入新的Binlog后，会再次检查是否有超时的Binlog，若此时该Binlog未参与任何只读节点上的线程，即被删除。

## 2. 订阅PolarDB MySQL引擎集群日志

为满足企业自定义管理日志的需求，PolarDB联合日志服务SLS推出了数据订阅功能，可以将您拥有的所有PolarDB MySQL引擎集群日志投递到日志服务SLS的中心化项目中统一进行管理。本文将介绍PolarDB MySQL引擎集群日志订阅和日志查看的方法。

### 注意事项

- PolarDB需为PolarDB MySQL引擎集群。
- PolarDB当前仅支持采集如下三种日志：
  - 审计日志
  - 慢日志
  - 性能日志
- 如需采集审计日志，请确保已为目标集群开启SQL洞察功能。SQL洞察功能按照审计日志存储的容量收取费用，详情请参见[SQL洞察费用说明](#)。

 说明 如仅需采集慢日志或性能日志，无需开启SQL洞察功能。

### 开启PolarDB日志采集

- 1.
2. 在日志应用区域，单击日志审计服务。
3. 配置日志采集同步授权。
  - 如果是同一账号下的日志采集，根据页面提示完成授权。具体操作请参见[首次配置](#)。
  - 如果是跨账号日志采集，具体操作请参见[自定义鉴权管理模式](#)。
4. 在云产品接入 > 全局配置页面，配置如下信息。
  - i. 在中心项目Project所在区域下拉列表中，选择日志中心化存储的目标地域。



全局配置 当前全局配置需要重新配置

中心项目Project所在区域: 请选择

区域Project: 暂无

云产品	审计相关日志	采集策略	存储方式	同步到中心
暂无数据				

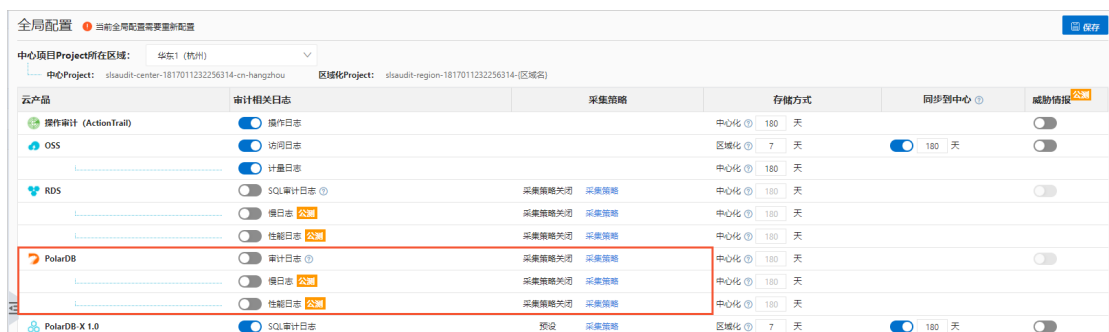
请先选择中心项目Project所在区域，并授权日志服务采集同步日志

系统会自动创建一个中心化项目，您在后面步骤中配置的采集数据都会投递到该中心化项目中。

#### 说明

- 更多关于中心化项目的信息，请参见[项目（Project）](#)。
- 目前支持的地域包括华北2（北京）、华北5（呼和浩特）、华东1（杭州）、华东2（上海）、华南1（深圳）、日本（东京）、新加坡。

- ii. 在云产品列表中，根据需要开启需要采集的PolarDB日志，您可以选择采集审计日志、慢日志或性能日志。



- iii. 单击目标日志右侧的采集策略，在采集策略配置对话框中将集群所在的地域添加到策略中，若不添加，系统会为所有地域创建日志库。更多信息，请参见[日志库 \(Logstore\)](#)和[采集策略](#)。

**说明** 您还可根据实际需要配置其他采集策略。PolarDB MySQL引擎集群可配置的采集策略如下：

- account.id：PolarDB MySQL引擎集群所属的阿里云账号ID。
- region：PolarDB MySQL引擎集群所属的地域，例如：cn-hangzhou。
- cluster.id：PolarDB MySQL引擎集群的ID。
- cluster.name：PolarDB MySQL引擎集群的名称。
- cluster.db\_type：PolarDB MySQL引擎集群兼容的数据库类型，取值当前仅支持MySQL。
- cluster.db\_version：数据库版本号，取值为8.0、5.7或5.6。
- tag.\*：自定义标签的名称。请将星号（\*）替换为您自定义的标签名。更多关于PolarDB MySQL引擎集群标签的信息，请参见[绑定标签](#)。

- iv. 在右上角单击**保存**。

**说明** 配置完成后，需要2分钟左右完成初始同步。如果出现异常，请根据页面提示信息进行调整。更多信息，请参见[开启日志采集功能](#)。

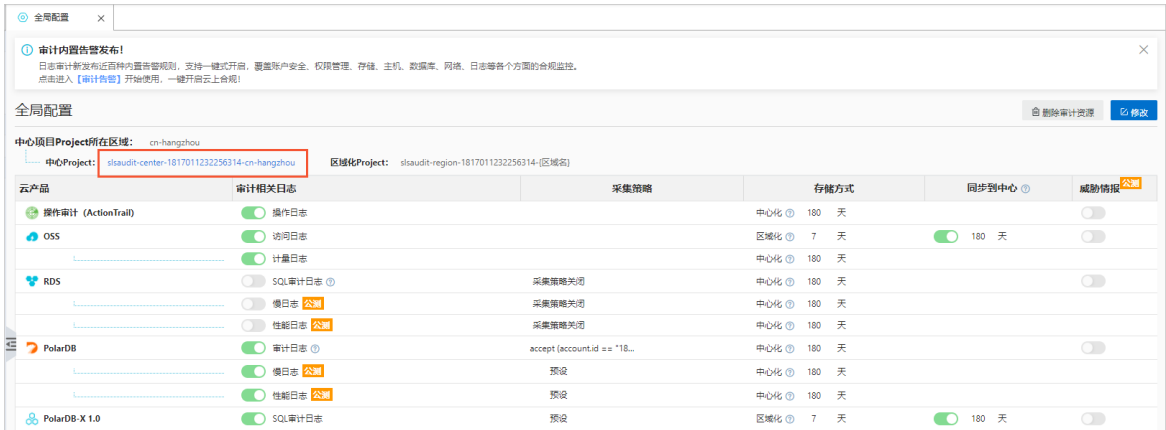
5. 在弹出的提示框中单击**前往接入状态**，查看日志的接入状态。

您也可以在左侧导航栏中选择云产品接入 > 接入状态进行查看。


## 查看采集的PolarDB日志

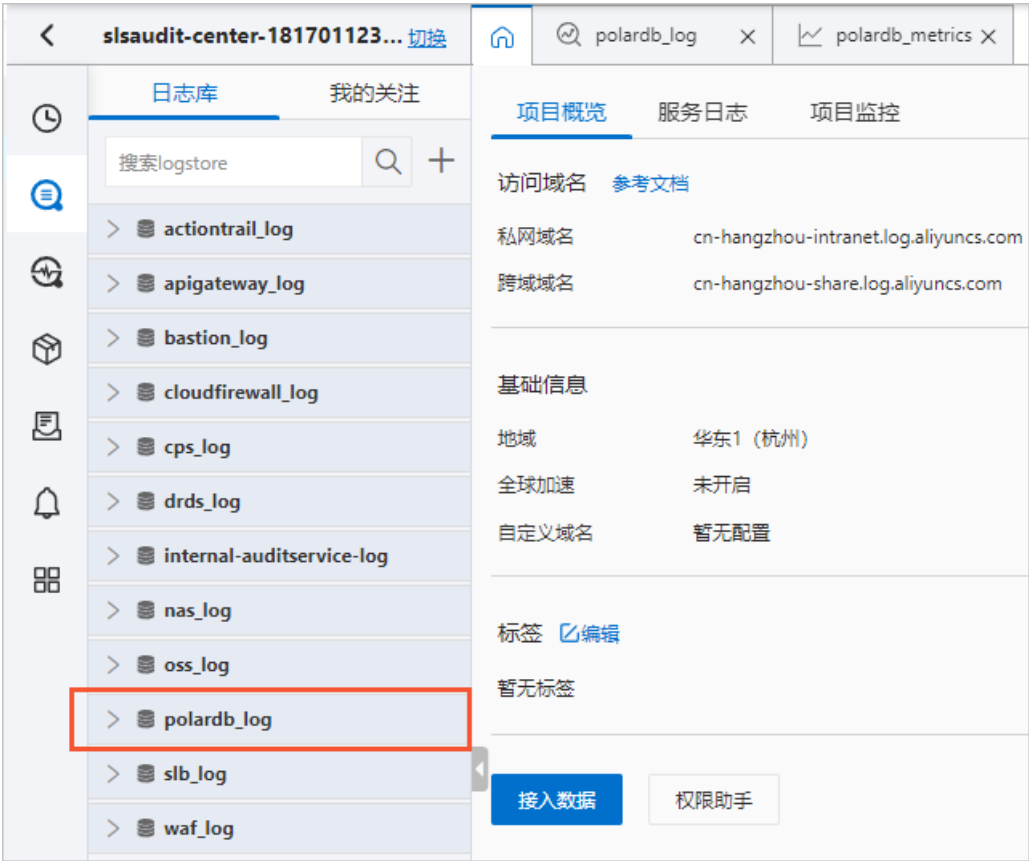
- 1.


2. 在**全局配置**页签中，单击已经创建的中心Project。Project名称格式为 `slsaudit-center-<阿里云账号ID>-<地域ID>`。

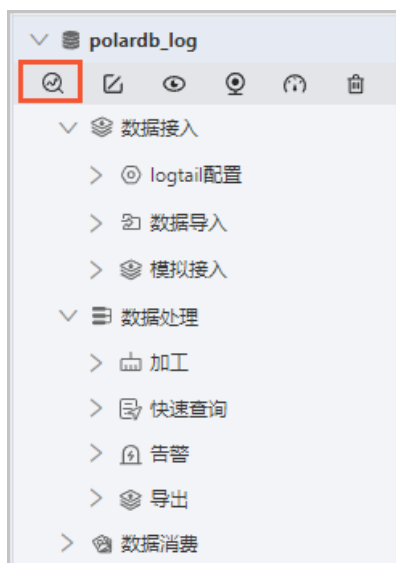


3. 查看日志信息。


- 查看PolarDB SQL审计日志或慢日志
  - 在日志库页签中，单击polaradb\_log左侧的图标展开该日志库。

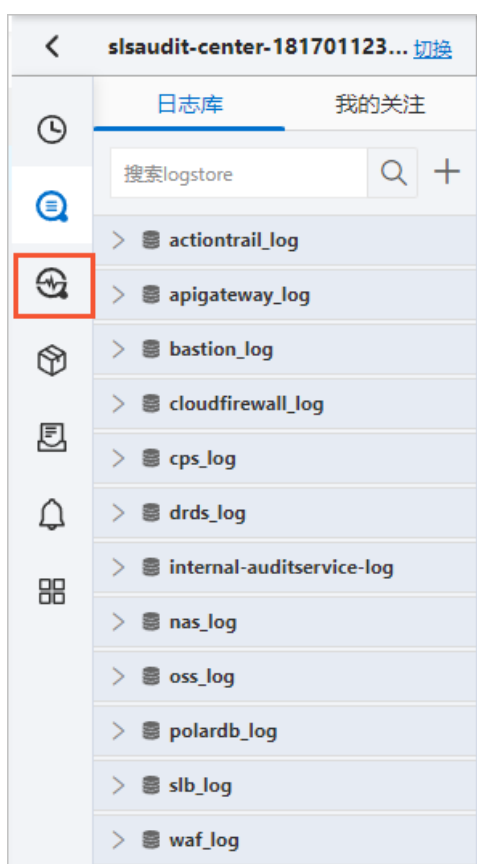



b. 单击图标，即可在页面右侧查看日志详情。




o 查看PolarDB性能日志

a. 在页面左侧导航栏中，单击图标。




b. 在时序库页签中，单击polardb\_metrics左侧的图标展开该时序库。

c. 单击图标，即可在页面右侧查看日志详情。


## 关闭PolarDB日志采集

1. 停止PolarDB日志的采集。具体操作请参见[相关操作](#)。

 **说明** 该操作需要等待2分钟左右完成同步。

2. 打开[云命令行（Cloud Shell）](#)。
3. 执行如下命令删除相关日志库。

```
aliyunlog log delete_project --project_name=<项目名> --region-endpoint=<地域ID>.log.aliyuncs.com
```

 **注意** 上述操作将删除项目中的所有数据，请谨慎操作。

## 日志字段说明

请参见[日志字段](#)。

## 3. 互联网行业高弹性系统架构最佳实践

本方案通过弹性伸缩和PolarDB的配合，实现应用和数据库两个层面的弹性。

原文请参见[互联网行业高弹性系统架构最佳实践](#)。

### 前提条件

- 所有云产品均需在同一VPC中，如何创建VPC请参见[搭建IPv4专有网络](#)。
- 已创建64位CentOS 7系统的ECS实例，创建步骤请参见[创建ECS实例](#)。
- 已创建PolarDB MySQL引擎集群，并在PolarDB白名单中添加VPC网段。
  - 创建步骤请参见[购买按量付费集群](#)和[购买包年包月集群](#)。
  - 设置PolarDB白名单请参见[设置白名单](#)。
- 已创建同城容灾版本的云数据库Redis，并在Redis白名单中添加VPC网段。
  - 创建步骤请参见[创建实例](#)。
  - 设置Redis白名单请参见[设置白名单](#)。
- 已开通ESS服务，开通方式请参见[开通并授权服务](#)。
- 已创建SLB实例，创建方式请参见[创建负载均衡实例](#)。
- 已创建EIP，申请操作请参见[申请新EIP](#)。

② 说明 需要申请两个弹性公网IP，一个用于绑定ECS实例，另一个用于绑定SLB实例。

- 如何绑定ECS请参见文档[绑定ECS实例](#)。
- 如何绑定SLB请参见文档[绑定SLB实例](#)。

- 已申请域名且已经完成域名备案。
  - 申请域名请参见[注册通用域名](#)。
  - 域名备案请参见[ICP备案快速入门](#)。

### 应用场景

在互联网行业的业务发展中，很多业务具有突发性特点。

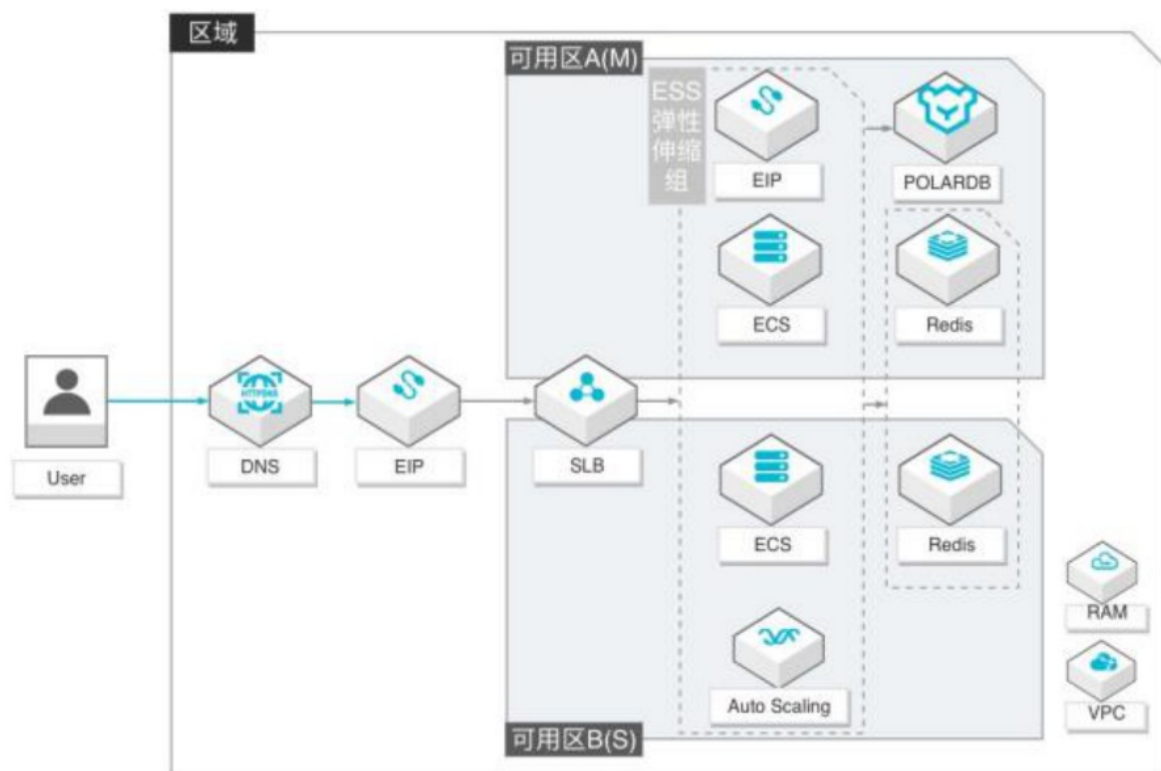
- 例如互联网电商的秒杀、促销等活动，这类业务的特点是时间固定，但访问量不固定。除了提前升级配置之外，客户往往希望系统本身也能有自动弹性伸缩的能力。
- 对于互联网教育的场景，由于存在放假和工作日的区别，系统也需要有一定的弹性伸缩能力去应对高出平时几倍的压力，等访问减少时，业务系统能释放冗余的资源达到节约成本的目标。

典型场景和需求：

- 业务系统波动大，以互联网行业为典型代表；
- 业务系统和数据库系统都要能实现弹性伸缩；
- 系统可用性高，弹性收缩用户感知小；
- 支持手工快速提升系统和数据库性能。

### 技术架构

本实践基于如下图所示的技术架构和主要流程。



## 方案优势

- 应用弹性：通过配置合理的弹性伸缩配置，业务高峰期到来业务压力上涨时自动增加ECS实例保障业务系统平稳运行。对于可预见的快速业务上涨，配置定时任务或者通过手动执行伸缩规则的方式可以预先备齐ECS资源。业务低谷期自动释放多余ECS资源节约成本。
- 数据库弹性：通过PolarDB在线节点配置弹性、只读节点弹性、同步克隆弹性和存储弹性四个维度的弹性，保障在大规模业务压力到来时快速应对业务压力。
- 应用高可用性：通过ESS收缩配置设置均衡分布策略，在主可用区宕机时，ESS会在备可用区开出相同资源，保障业务平稳运行。
- PolarDB、Redis、SLB高可用性：通过多可用区的主备策略保障跨可用区自动容灾。

## 注意事项

- 跨可用区访问可能会有1~2ms的延迟。
- 数据库不能实现自动弹性伸缩，需要人工干预升降配。
- 业务系统需无状态，有状态系统需要先进行业务改造。

## 操作步骤

本例以WordPress为业务系统演示。

1. 登录[云数据库PolarDB版控制台](#)。
2. 创建PolarDB集群数据库和拥有读写权限的账号，具体操作请参见[数据库管理](#)和[创建数据库账号](#)。
3. 登录[云服务器ECS控制台](#)。
4. 连接目标ECS实例，具体操作请参见[连接ECS实例](#)。
5. 在ECS上安装解压和压测工具。



```
yum install -y unzip zip sysbench dstat
```

#### ? 说明

- unzip和zip为压缩和解压缩程序，后续解压zip文件使用。
- sysbench和dstat为Linux系统压测和系统监控工具，后续验证弹性伸缩时使用。

## 6. 在ECS上安装Apache和PHP组件。

### i. 安装Apache和PHP组件。

```
yum install -y httpd php php-mysql php-gd php-imap php-ldap php-odbc php-pear php-xml php-xmlrpc
```

#### ? 说明 安装完成后使用如下命令查看PHP版本：

```
yum list installed | grep php
```

WordPress 5.2.2要求PHP最低版本为5.6.20，若PHP版本低于该版本，请使用如下命令重新安装PHP组件。

```
## 查看当前PHP版本
yum list installed | grep php
## 卸载低版本PHP组件
yum remove -y php.x86_64 php-cli.x86_64 php-common.x86_64 php-gd.x86_64 php-ldap.x86_64 php-mbstring.x86_64 php-mcrypt.x86_64 php-mysql.x86_64 php-pdo.x86_64
## 确认是否全都卸载完成
yum list installed | grep php
## 安装PHP组件的rpm包
rpm -Uvh https://mirror.webtatic.com/yum/el7/epel-release.rpm
rpm -Uvh https://mirror.webtatic.com/yum/el7/webtatic-release.rpm
## 重新安装PHP相关组件
yum install -y php56w.x86_64 php56w-cli.x86_64 php56w-common.x86_64 php56w-gd.x86_64 php56w-ldap.x86_64 php56w-mbstring.x86_64 php56w-mcrypt.x86_64 php56w-mysql.x86_64 php56w-pdo.x86_64
```

### ii. 启动Apache服务。

```
service httpd start
```

### iii. 设置Apache为开机默认启动。

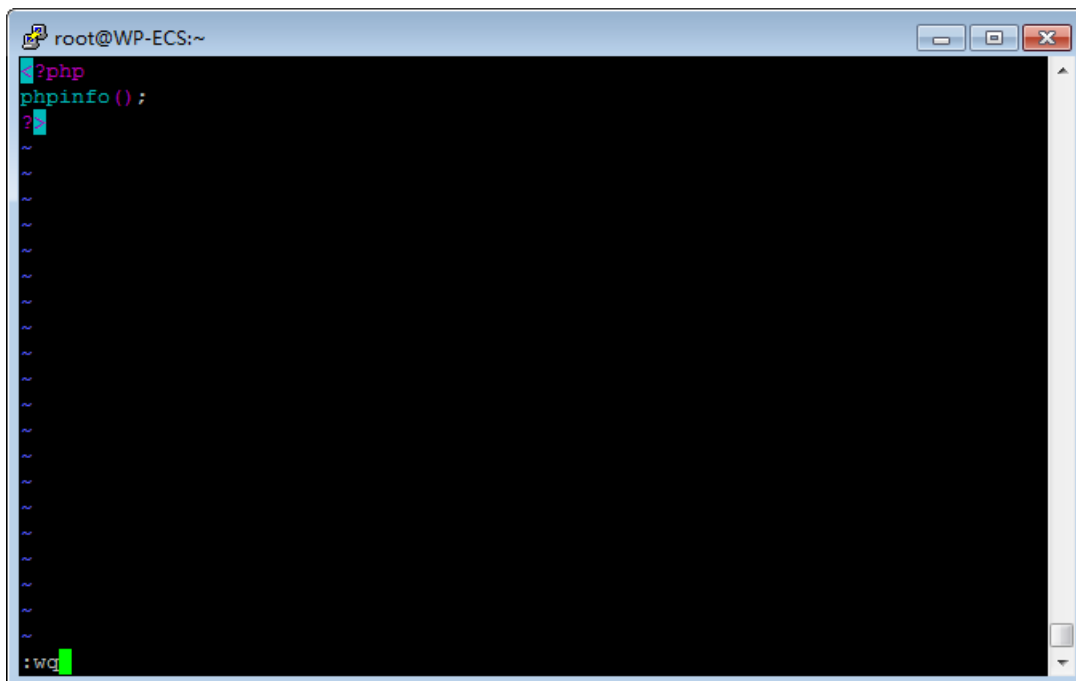
```
sudo chkconfig httpd on
```

### iv. 编辑 info.php 文件。

```
vim /var/www/html/info.php
```

- v. 按 `i` 进行编辑，在文件中输入如下内容。

```
<?php
phpinfo();
?>
```



- vi. 按Esc，输入 `:wq` 保存并退出。
- vii. 在浏览器输入 `http://<ECS的弹性公网IP>/info.php`，PHP安装正确会看到下图的PHP信息。

A screenshot of a web browser displaying the PHP configuration page. The page title is 'PHP Version 5.6.40' and it features the PHP logo. The content is a table with two columns: 'System' and 'Linux WP-ECS 3.10.0-957.21.3.el7.x86\_64 #1 SMP Tue Jun 18 16:35:19 UTC 2019 x86\_64'. The table lists various system and PHP configuration details.

System	Linux WP-ECS 3.10.0-957.21.3.el7.x86_64 #1 SMP Tue Jun 18 16:35:19 UTC 2019 x86_64
Build Date	Jan 12 2019 13:12:26
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/bz2.ini, /etc/php.d/calendar.ini, /etc/php.d/ctype.ini, /etc/php.d/curl.ini, /etc/php.d/exif.ini, /etc/php.d/fileinfo.ini, /etc/php.d/ftp.ini, /etc/php.d/gd.ini, /etc/php.d/gettext.ini, /etc/php.d/gmp.ini, /etc/php.d/iconv.ini, /etc/php.d/json.ini, /etc/php.d/ldap.ini, /etc/php.d/mbstring.ini, /etc/php.d/mcrypt.ini, /etc/php.d/mysql.ini, /etc/php.d/mysqli.ini, /etc/php.d/pdo.ini, /etc/php.d/pdo_mysql.ini, /etc/php.d/pdo_sqlite.ini, /etc/php.d/phar.ini, /etc/php.d/shmop.ini, /etc/php.d/simplexml.ini, /etc/php.d/sockets.ini, /etc/php.d/sqlite3.ini, /etc/php.d/tokenizer.ini, /etc/php.d/xml.ini, /etc/php.d/zip.ini
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API220131226,NTS
PHP Extension Build	API20131226,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled

7. 在ECS上安装WordPress。

## i. 创建WordPress安装目录。

```
mkdir -p /opt/WP
```

## ii. 打开WP文件夹。

```
cd /opt/WP
```

## iii. 下载并解压WordPress。

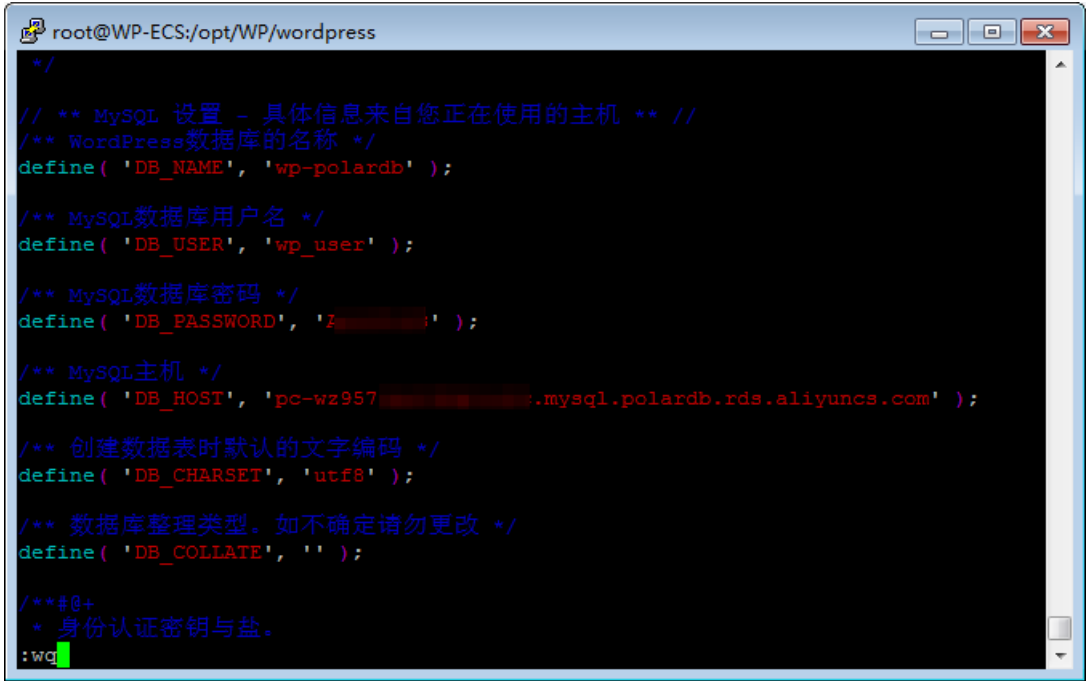
```
## 下载WordPress
wget https://cn.wordpress.org/latest-zh_CN.tar.gz
## 解压WordPress
tar -xzf latest-zh_CN.tar.gz
```

## iv. 配置WordPress访问PolarDB。

```
## 打开WordPress文件夹
cd /opt/WP/wordpress/
## 复制wp-config-sample.php文件，并将复制后的文件名称修改成为wp-config.php
cp wp-config-sample.php wp-config.php
## 编辑wp-config.php文件
vim wp-config.php
```

v. 按 `i` 进行编辑，配置如下参数。

```
/** WordPress数据库的名称 */
define( 'DB_NAME', 'wp-polardb' );
/** MySQL数据库用户名 */
define( 'DB_USER', 'wp_user' );
/** MySQL数据库密码 */
define( 'DB_PASSWORD', 'password' );
/** MySQL主机 */
define( 'DB_HOST', 'pc-wz957jq*****.mysql.polardb.rds.aliyuncs.com' );
```



参数	描述	示例
DB_NAME	PolarDB数据库名称。	wp-polardb
DB_USER	PolarDB数据库账户的用户名。	wp_user
DB_PASSWORD	PolarDB数据库账户对应的密码。	password
DB_HOST	PolarDB数据库内网连接地址， 查看内网连接地址方式请参见 <a href="#">申请集群地址和主地址</a> 。	pc-wz957jq*****.mysql.polardb.rds.aliyuncs.com

vi. 按Esc，输入 `:wq` 保存并退出。

vii. 打开`/var/www/html/`，并将WordPress目录复制到`/var/www/html/`路径下。

```
cd /var/www/html
cp -rf /opt/WP/wordpress/* /var/www/html/
```

- viii. 在浏览器输入 `http://<ECS的弹性公网IP>`，在WordPress配置页面填写站点标题、用户名和密码等信息，如下图所示。



- ix. 单击左下角**安装WordPress**，完成安装。

## 8. 在WordPress上开通Redis缓存支持。

- i. 下载并解压redis-cache插件。

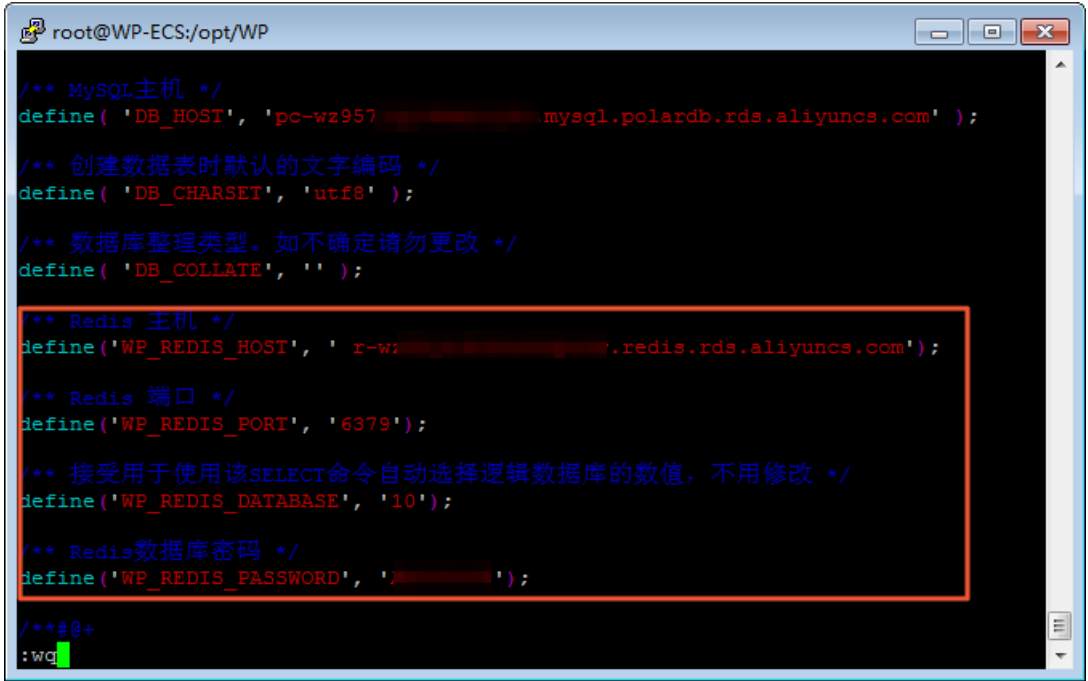
```
## 切换到WP目录
cd /opt/WP
## 下载redis-cache插件
wget https://downloads.wordpress.org/plugin/redis-cache.1.4.1.zip
## 解压redis-cache
unzip redis-cache.1.4.1.zip
## 复制到/var/www/html/wp-content/plugins/
cp -rf redis-cache /var/www/html/wp-content/plugins/
```

- ii. 编辑 `wp-config.php` 文件。

```
vim /var/www/html/wp-config.php
```

iii. 按 `i` 进行编辑，在PolarDB设置后增加Redis数据库信息。

```
/** Redis 主机 */
define('WP_REDIS_HOST', 'r-*****.redis.rds.aliyuncs.com');
/** Redis 端口 */
define('WP_REDIS_PORT', '6379');
/** 接受用于使用该SELECT命令自动选择逻辑数据库的数值，不用修改 */
define('WP_REDIS_DATABASE', '10');
/** Redis数据库密码 */
define('WP_REDIS_PASSWORD', 'password');
```



参数	描述	示例
WP_REDIS_HOST	云数据库Redis内网连接地址，查看内网连接地址方式请参见 <a href="#">查看连接地址</a> 。	r-*****.redis.rds.aliyuncs.com
WP_REDIS_PORT	Redis服务端口，默认为6379。	6379
WP_REDIS_DATABASE	指定缓存信息保存的DB，例如10就是保存到DB10。	10
WP_REDIS_PASSWORD	Redis的连接密码	password

iv. 按Esc，输入 `:wq` 保存并退出。

v. 复制配置文件到根目录下。

```
cp /var/www/html/wp-content/plugins/redis-cache/includes/object-cache.php /var/www/html/wp-content/
```

vi. 在浏览器输入 `http://<ECS的弹性公网IP>/wp-login.php`，登录WordPress管理页面。



vii. 在左侧导航单击插件 > 已安装插件

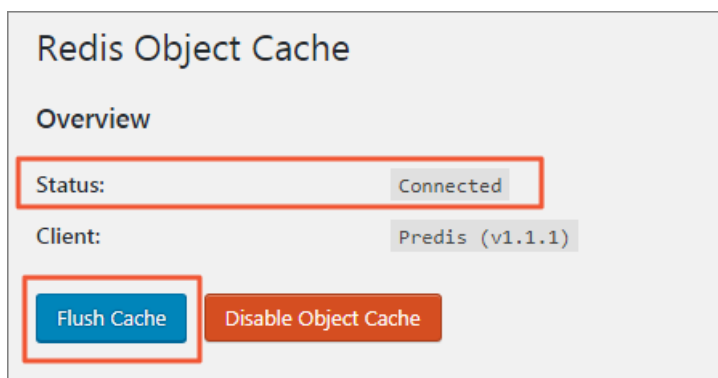
viii. 在插件页面，单击Redis Object Cache下的启动。



ix. 单击Settings，打开Redis Object Cache页面。



x. 当Status显示Connected时，表示Redis数据库连接正常。您可以点击Flush Cache将缓存数据导入Redis数据库。



9. 在WordPress管理页面左侧导航栏中单击设置 > 常规，在WordPress地址（URL）和站点地址（URL）选项中输入您预先申请并备案的域名。

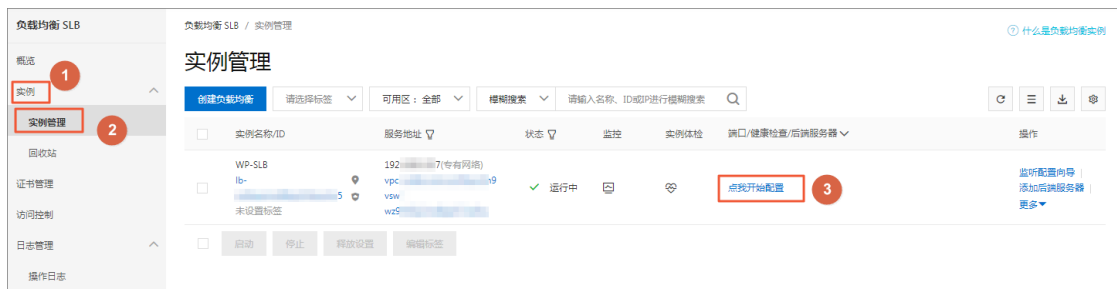


10. 配置负载均衡。

- 登录负载均衡SLB控制台。
- 在左侧导航栏单击实例 > 实例管理



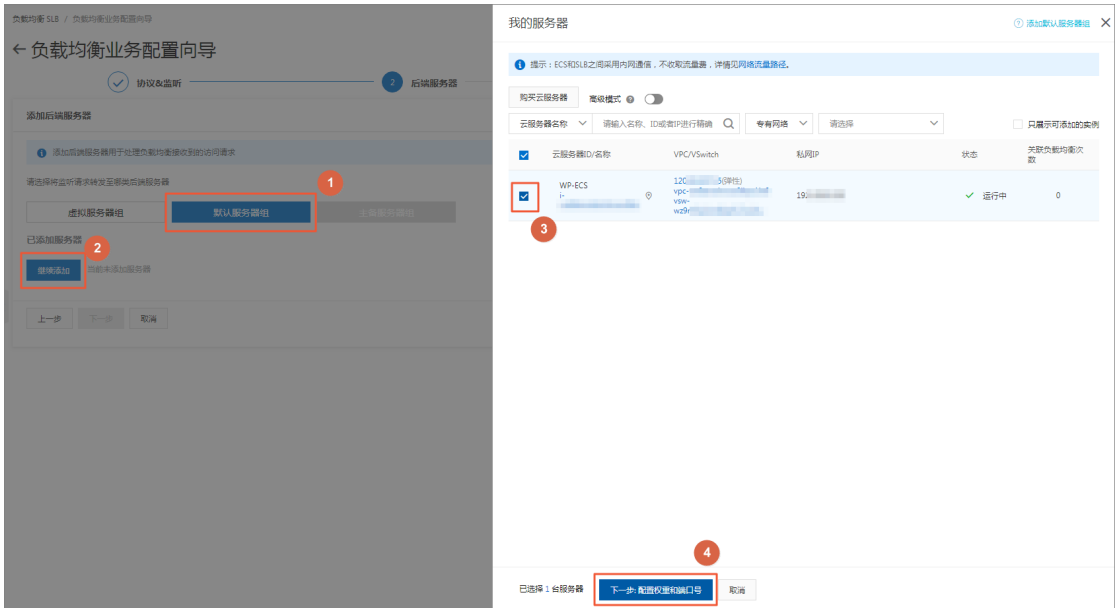
iii. 定位到目标SLB实例，单击右侧点我开始配置。



iv. 在负载均衡业务配置导航页面，选择HTTP协议，监听端口设置为80，开启开启会话保持功能，单击下一步。



v. 单击**默认服务器组** > **继续添加**，勾选之前创建的ECS实例，单击**下一步：配置权重和端口号**。



- vi. 设置端口号为80，权重为50，单击**下一步**。
- vii. 在**配置健康检查**页面，开启健康检查后单击**下一步**。
- viii. 在**配置审核**页面，单击**提交**。
- ix. 待到审核项目都显示**成功**后单击**确定**即可。

11. 登录**云解析DNS控制台**，将SLB弹性公网IP地址绑定预先申请并备案的域名。

**说明** 如何绑定域名请参见**添加网站解析**。

12. 登录**云服务器ECS控制台**，为之前创建好的ECS创建镜像。

**说明** 如何创建ECS实例镜像请参见**使用实例创建自定义镜像**。

13. 登录**弹性伸缩控制台**，配置弹性伸缩。

## i. 创建伸缩组。

创建伸缩组

\*伸缩组名称：

WP-ESS

名称为2-40个字符，以大小写字母，数字或中文开头，可包含“.”、“\_”或“-”

\*伸缩最大实例数（台）：

5

最小为0，最大为1000

\*伸缩最小实例数（台）：

2

最小为0，最大为1000

\*默认冷却时间（秒）：

300

最小为0，必须为整数

移出策略：

先筛选

最早伸缩配置对应的实例

在结果中再筛选

最早创建的实例

移出

如何保证手工添加的ECS实例不被移出伸缩组

\*组内实例配置信息来源：

自定义伸缩配置

启动模板

\*网络类型：

经典网络

专有网络

专有网络

伸缩组可支持选择多个虚拟交换机

\*专有网络：

专有网络ID：

vpc-

in9

虚拟交换机：

(cn-shenzhen-e)

(cn-shenzhen-d)

创建专有网络

多可用区伸缩策略：

优先级策略

均衡分布策略

成本优化策略

回收模式：

释放模式

停机回收模式

负载均衡：

-- 请选择负载均衡 --

管理我的负载均衡

数据库：

负载均衡 ID：

负载均衡名称：SLB

网络类型：专有网络

管理我的数据库

提交

取消

? 说明 创建伸缩组具体配置请参见[创建伸缩组](#)。

## ii. 创建伸缩配置并启用。

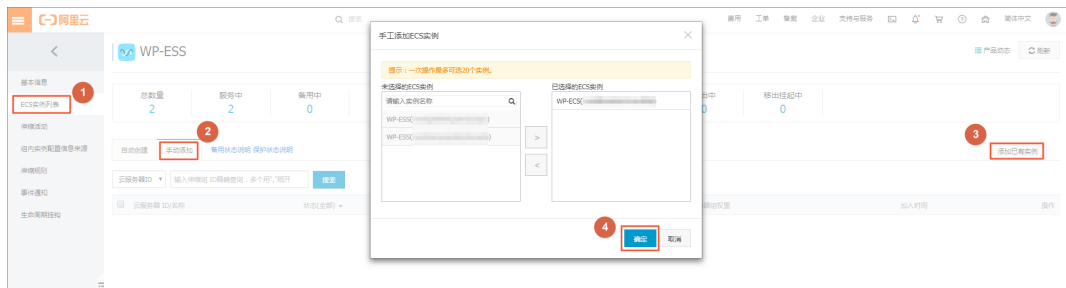
? 说明 创建伸缩配置具体配置请参见[创建伸缩配置](#)

## iii. 在该伸缩组中加入ECS实例。

## a. 登录弹性伸缩控制台，单击目标伸缩组后的管理。



## b. 单击ECS实例列表 &gt; 手动添加 &gt; 添加已有实例，将目标ECS实例移动到右侧后，单击确定。



## iv. 创建伸缩规则。

修改伸缩规则

\*规则名称：

ADD1

名称为2-64个字符，以大小写字母，数字或中文开头，可包含“.”、“\_”或“-”

\* 伸缩规则类型：

简单规则

\* 执行的操作：

增加

1

台

“增加”或“减少”选项最多只能支持500台，否则报错  
“调整至N台”以及“增加或减少N%”一次只能触发500台实例的伸缩活动

冷却时间（秒）：

修改伸缩规则

取消

修改伸缩规则

\*规则名称：

DROP1

名称为2-64个字符，以大小写字母，数字或中文开头，可包含“.”、“\_”或“-”

\* 伸缩规则类型：

简单规则

\* 执行的操作：

减少

1

台

“增加”或“减少”选项最多只能支持500台，否则报错  
“调整至N台”以及“增加或减少N%”一次只能触发500台实例的伸缩活动

冷却时间（秒）：

修改伸缩规则

取消

说明 创建伸缩规则具体配置请参见[创建伸缩规则](#)。

## v. 创建伸缩任务。

创建报警任务

在使用报警任务之前，需要在ECS的镜像里安装新版本的云监控Agent。 [查看帮助文档](#)

\*任务名称：

名称为2-64个字符，以大小写字母，数字或中文开头，可包含“.”，“\_”或“-”

描述：

最少需要2个字符

\*监控资源：

WP-

监控类型

☒ 系统监控 ☐ 自定义监控

\*监控项：

CPU

目前目标追踪类型的伸缩规则可以通过扩缩容将监控指标维持在目标值附近，且支持当前您选择的监控项的平均值作为监控指标，更加方便直观，推荐使用

统计周期(分钟)：

1

\*统计办法：

平均值

>=

阈值

80

%

重复几次后报警：

3次

\*报警触发规则：

ADD1

提交 取消

创建报警任务

在使用报警任务之前，需要在ECS的镜像里安装新版本的云监控Agent。 [查看帮助文档](#)

\*任务名称：

名称为2-64个字符，以大小写字母，数字或中文开头，可包含“.”，“\_”或“-”

描述：

最少需要2个字符

\*监控资源：

WP-

监控类型

☒ 系统监控 ☐ 自定义监控

\*监控项：

CPU

目前目标追踪类型的伸缩规则可以通过扩缩容将监控指标维持在目标值附近，且支持当前您选择的监控项的平均值作为监控指标，更加方便直观，推荐使用

统计周期(分钟)：

1

\*统计办法：

平均值

<=

阈值

50

%


重复几次后报警：

3次

\*报警触发规则：

DROP1

提交 取消

 说明 创建伸缩任务具体配置请参见[执行伸缩规则](#)。

14. 验证弹性伸缩。

- i. 通过控制台登录ECS，登录操作请参见[连接ECS实例](#)。
- ii. 通过压测命令提高ECS的CPU使用率。

```
sysbench cpu --cpu-max-prime=2000000 --threads=2 --time=1000 run
```

iii. 根据之前配置的报警任务。

- 当CPU使用率超过80%时，触发ESS的报警任务，ESS会自动创建ECS服务器。
- 当CPU使用率低于50%时，触发ESS的报警任务，ESS会自动减少ECS服务器。

至此配置完成。

## 4. 数据误操作时如何进行恢复

### 4.1. 不同场景下恢复方式概览

数据误操作有多种场景，根据场景的不同以及对应数据库引擎版本的不同，PolarDB MySQL引擎提供了多种方式，帮助您恢复数据。本章节列举了常见的误操作的场景，您可根据实际情况，选择对应的数据恢复方式。

场景	数据库版本		恢复方式
误删表	5.6		采用库表恢复的方式恢复误删的表，包括按时间点恢复或者按备份集恢复两种方法：
	5.7		<ul style="list-style-type: none"> <li>如果要恢复的时间点正好是某备份集（快照）的时间点，可以直接选择<b>库表恢复：按备份集恢复</b>。</li> <li>如果要恢复到的时间点在备份集（快照）之外的其他时间点，可以选择<b>库表恢复：按时间点恢复</b>。</li> </ul>
	8.0		若集群已开启表回收站功能，则可 <b>从表回收站中恢复误删的表</b> ；若未开启，则可通过库表恢复的方式恢复整个集群。
误删数据库	5.6		采用库表恢复的方式恢复误删的库，包括按时间点恢复或者按备份集恢复两种方法：
	5.7		<ul style="list-style-type: none"> <li>如果要恢复的时间点正好是某备份集（快照）的时间点，可以直接选择<b>库表恢复：按备份集恢复</b>。</li> <li>如果要恢复到的时间点在备份集（快照）之外的其他时间点，可以选择<b>库表恢复：按时间点恢复</b>。</li> </ul>
	8.0		<ul style="list-style-type: none"> <li>如果要恢复到的时间点在备份集（快照）之外的其他时间点，可以选择<b>库表恢复：按时间点恢复</b>。</li> </ul>
误删集群	5.6		<b>从集群回收站恢复误删的集群</b>
	5.7		
	8.0		
误操作表中数据，如整体覆盖、误删除/修改表中的列/行/数据	误操作受影响的数据量在10万以内		建议使用 <b>使用DMS数据追踪进行恢复</b> 方式，也可使用库表恢复和集群恢复方式。
	误操作受影响的数据量超过10万	5.6	采用库表恢复的方式恢复误操作的库或表，包括按时间点恢复或者按备份集恢复两种方法：
		5.7	<ul style="list-style-type: none"> <li>如果要恢复的时间点正好是某备份集（快照）的时间点，可以直接选择<b>库表恢复：按备份集恢复</b>。</li> <li>如果要恢复到的时间点在备份集（快照）之外的其他时间点，可以选择<b>库表恢复：按时间点恢复</b>。</li> </ul>
		8.0	<ul style="list-style-type: none"> <li>如果要恢复到的时间点在备份集（快照）之外的其他时间点，可以选择<b>库表恢复：按时间点恢复</b>。</li> </ul> <p>若无法判断受影响的库和表，则也可以使用集群恢复的方式进行恢复。</p>

### 4.2. 使用DMS数据追踪进行恢复

DMS的数据追踪功能可以快速按需找到目标时间段内的相关更新，并生成逆向回滚语句，追踪完的数据可批量生成回滚脚本，通过数据变更工单最终执行到数据库中完成数据的修复。当误操作受影响的数据量在10万以内时，您可使用该方法进行数据恢复。

前提条件

- 已为集群开启Binlog。详细操作步骤，请参见[开启Binlog](#)。
- 若在DMS中，该集群的管控模式为自由操作与稳定变更，请先通过DMS登录目标集群。

步骤一、查找需回滚记录，生成回滚脚本

首先，您需要在DMS控制台提交数据追踪工单申请，审批通过后，根据解析出的日志，选择需要回滚的变更记录，并生成回滚脚本。

1. 登录[新版DMS控制台](#)。
2. 在页面顶部，选择全部功能 > 数据方案 > 数据追踪。
3. 在页面右上角，单击+数据追踪。
4. 在数据追踪工单申请页面，根据下表填写工单申请。

工作台

数据追踪工单

SQL idb\_test

工单列表

工单申请

数据追踪工单申请

1

申请

\* 任务名称:

数据误更新追踪回滚

\* 时间范围:

2019年7月22日 13:18:00

2019年7月22日 15:18:00

\* 库名:

idb\_test@

[vp]

\* 表名:

zitang\_test

X

\* 追踪类型:

☒ 插入 ☒ 更新 ☒ 删除

变更相关人:

请选择

提交申请

2

文件获取

3

审批

4

日志处理

5


日志搜索

参数名	说明
任务名称	便于后续检索，并提供给审批人员清晰的操作意图。
库名	误操作对象的数据库名。
表名	误操作对象的表名。
追踪类型	支持多选，可按需勾选检索的操作。




参数名	说明
时间范围	默认为当前时间往前2小时。单次工单最长追踪时长跨度为6小时，若超过则可分段建多个工单处理。最长可追踪时间点为目标数据库服务器上binlog文件当前存储的最早时间。

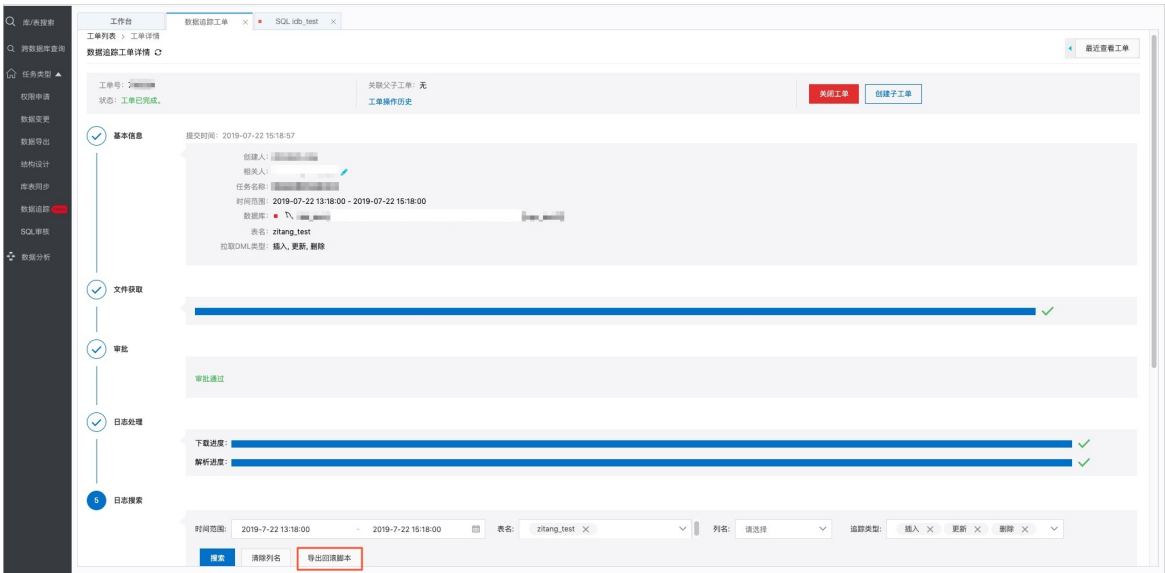
5. 单击**提交申请**，系统将获取日志文件。  
获取日志文件后，系统会进入审批环节。

 **说明** 若Binlog未开启或数据库未登录，系统将无法获取日志文件。

6. 等待审批通过。

 **说明** 数据追踪的默认安全审批规则为：数据库对应的DBA进行审批。

7. 审批通过后，系统会进行日志下载、日志解析等操作。  
8. 待解析完成后，您可以勾选需要回滚的变更记录，单击**导出回滚脚本**即可。



步骤二、执行回滚

导出回滚脚本后，您可以提交普通数据变更工单，将此回滚脚本作为附件上传至工单中，执行至目标数据库中。

- 1. 在DMS控制台页面顶部，选择**全部功能 > 数据方案 > 普通数据变更**。
- 2. 可参考下表填写**数据变更工单申请**，并单击**提交申请**。

数据变更工单申请

● 申请

申请的数据变更类别：

普通数据变更

无锁变更

历史数据清理

批量数据导入

可编程对象

说明：1.多条SQL之间, 请用英文分号隔开  
2.请不要编写对数据库不友好的SQL，以免影响线上业务运行

\* 数据库：

请输入数据库名搜索

\* 影响行数：

1

\* 变更SQL：

☐ 文本

☒ 附件

\* 附件

上传文件

仅支持.txt和.sql的文件类型, 最大不能超过15MB（如需更大附件导入请选择“批量数据导入”功能）

提交申请

数据变更参数说明

参数名	说明
数据库	要恢复的数据库名称。
影响行数	预估本次更新会影响的数据行数，可在SQLConsole以 <code>count</code> 统计。
变更SQL	选择附件，并将步骤一中生成的脚本上传至附件。

3. 工单提交后，系统会进行内容合法性检查。

4. 当检查通过后，需要您提交审批，审批通过后，就会生成执行任务。

🔗 说明 数据变更的默认安全审批规则为：数据库对应的DBA进行审批。

5. 单击执行变更。

## 4.3. 集群恢复的方式恢复整个集群

### 4.3.1. 整体流程和预估时间

通过集群恢复方式，可以恢复整个集群的数据。

集群恢复分为按时间点恢复和按备份集（快照）恢复，两者的差别在于要恢复至的时间点是否是备份集的时间点。如果是备份集的时间点，则可选择按备份集（快照）恢复，更方便。如果要恢复至的时间不是已有的备份集的时间点，则只能选择按时间点恢复。

#### 整体流程

34

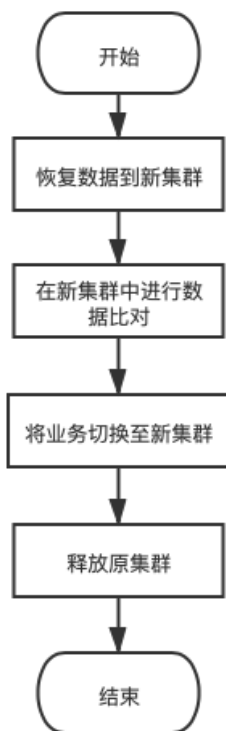
> 文档版本：20220713

不管是按时间点恢复还是按备份集（快照）恢复，两者的关键流程是一致的：先将全量数据恢复到一个新集群。在新集群验证数据并确认无误后，您可选择将恢复后的数据迁移回原集群，也可以直接将您的业务切换至新的集群。整体流程如下：

恢复至新集群，再迁移回原集群



恢复至新集群，并将业务切换至新集群



## 预估时间

集群恢复所需的时间与集群规格、大小以及备份的大小有关。预估耗时如下，供参考：

步骤		预估耗时
恢复数据到新集群	创建临时集群	5~10分钟
	恢复备份集（快照）数据	3 GB/分钟
	恢复Redo日志增量数据 <div>❓ 说明 仅按时间点恢复的方式需要恢复该类数据。</div>	1.5 GB/分钟
在新集群中进行数据比对		3~5分钟
将新集群的数据迁移到原集群		集群规格对应的最大 IOPS×15%×16 KB/秒

例如，若一个集群（4核16 GB，最大IOPS为32000），集群大小为100 GB，备份集（快照）数据大小为10 GB，Redo日志大小为3 GB，则恢复耗时预估为：

- 恢复备份集（快照）数据：10/3=3.3分钟
- 恢复Redo日志增量数据：3/1.5=2分钟
- 将新集群的数据迁移到原集群：100×1024×1024/(32000×15%×16)=22.7分钟

若您选择恢复至新集群后，再迁移回原集群，则整体耗时=5+3.3+2+3+22.7≈36分钟左右。

若您选择恢复至新集群后，直接将业务切换到该新集群，则整体耗时=5+3.3+2+3+业务切换耗时+集群配置耗时≈13分钟+业务切换耗时+集群配置耗时。

 **说明** 建议在业务低峰期进行数据恢复。

### 4.3.2. 集群恢复：按时间点恢复

通过一个完整的数据全量备份（快照）以及后续一段时间的Redo日志，就可以将PolarDB集群恢复到日志保存期限内的任意时间点。在知晓误操作发生的时间点后，可以通过此方式，将整个集群恢复到误操作发生之前。

#### 前提条件

- 已存在备份集。因为按时间点恢复是先将所选时间点前的一个全量数据备份恢复到集群，然后根据Redo日志恢复增量数据到所选时间点。若您希望提升恢复速度，可以开启增强备份，从而缩短备份周期，增加备份密度。
- 建议在业务低峰期进行数据恢复。

#### 注意事项


恢复后的集群包含原集群的数据和账号信息，不包含原集群的参数设置。

#### 步骤一、获取误操作的时间点

如果您已知误操作的时间点，可以跳过本步骤。如果您不记得误操作的具体时间点，可以通过此步骤查看。

##### 方式一：SQL洞察


对于PolarDB MySQL引擎集群版，若集群已开通SQL洞察功能，可通过该功能查看审计日志，获取误操作的时间点。

 **说明** SQL洞察会收费，费用与审计日志的存储容量和存储时长有关。关于更多SQL洞察的说明，请参见[SQL洞察](#)。

SQL洞察功能仅支持查看开通该功能后产生的SQL日志。因此若集群未开通SQL洞察，可通过方式二查看误操作时间点。

##### 方式二：远程获取并解析Binlog日志

- 为集群开启Binlog。详细操作步骤，请参见[开启Binlog](#)。

 **说明** 您需要先为集群开启Binlog才能查看并获取Binlog日志，否则将出现 `You are not using binary logging` 错误提示。

- 在本地服务器上安装MySQL，并通过MySQL客户端连接集群。详细操作步骤，请参见[使用客户端连接集群](#)。本文以Linux系统为例。

```
root@~:~# mysql -h .polardb.rwlb.rds.aliyuncs.com -P3306 -utest_api -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is
Server version: 8.0.13 Source distribution

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

3. 在已连接集群的客户端中执行如下命令查看目标集群的Binlog文件列表：

```
show binary logs;
```

返回结果如下：

```
+-----+-----+
| Log_name          | File_size |
+-----+-----+
| mysql-bin.000005 |      2639 |
+-----+-----+
1 row in set (0.00 sec)
```

4. 在客户端执行如下命令，远程获取Binlog日志并保存至本地。

本文中以Linux系统为例，需先执行exit退出MySQL后，才能远程获取Binlog日志并保存至本地。

```
mysql> exit
Bye
root@~:~# mysqlbinlog -utest_api -p -h .polardb.rwlb.rds.aliyuncs.com --read-from-remote-server --raw mysql-bin.000005
Enter password:
root@~:~#
```

执行如下语句：

```
mysqlbinlog -u<用户名> -p<密码> -h<连接地址> --read-from-remote-server --raw mysql-bin.**
****
```

参数	说明	示例值
-u	集群中的账号名称。	test_api
-p	以上账号的密码。如果此处留空，则在执行此命令后会被要求输入。	TestPwd123
-h	集群的公网连接地址。 <div><div>② 说明</div><div><ul style="list-style-type: none"><li>若您的集群连接地址使用的是默认的端口号3306，端口号可省略不写，否则需要在连接地址后加上端口号。</li><li>当前仅支持通过主地址或集群地址（包括默认集群地址和自定义集群地址）的公网连接地址远程获取Binlog。如何申请公网连接地址，请参见<a href="#">申请集群地址和主地址</a>。</li></ul></div></div>	test-polardb.rwlb.rds.aliyuncs.com

参数	说明	示例值
<code>--raw</code>	表示将获取到的Binlog文件按数据原有格式打印，而不会展示解析后的数据。	<code>--raw</code>
<code>mysql-bin.*****</code>	通过 <code>show binary logs;</code> 命令获取的目标Binlog文件的名字，即 <code>Log_name</code> 。	<code>mysql-bin.000005</code>

示例：

```
mysqlbinlog -utest_api -p -h test-polaradb.rwlb.rds.aliyuncs.com --read-from-remote-server --raw mysql-bin.000005
```

5. 执行如下命令，通过mysqlbinlog工具查看Binlog日志文件内容：

```
mysqlbinlog -vv --base64-output=decode-rows mysql-bin.***** | more
```

#### 说明

- `-vv`：查看具体SQL语句及备注。
- `--base64-output=decode-rows`：解析Binlog日志文件。

具体的Binlog日志内容如下图中红框所示部分：

```
[root@iz... ~]# mysqlbinlog -vv --base64-output=decode-rows mysql-bin.000110 | more
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET @OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#160217 23:04:37 server id 2802943055 end_log_pos 107 Start: binlog v 4, server v 5.5.18.1-log created 160217 23:04:37
# at 107
#160217 23:04:38 server id 2802943055 end_log_pos 171 Query thread id=584632 exec time=0 error code=0
SET TIMESTAMP=1455721478/*!*/;
SET @@session.pseudo_thread_id=584632/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=0, @@session.unique_checks=1, @@session.autocommit=1/*!*/;
SET @@session.sql_mode=2097152/*!*/;
SET @@session.auto_increment_increment=1, @@session.auto_increment_offset=1/*!*/;
/*!\C utf8 *//*!*/;
SET @@session.character_set_client=33,@@session.collation_connection=33,@@session.collation_server=33/*!*/;
SET @@session.lc_time_names=0/*!*/;
SET @@session.collation_database=DEFAULT/*!*/;
BEGIN
/*!*/;
# at 171
```

6. 在获取Binlog日志文件内容后，可参见[解析Binlog日志](#)进行Binlog日志解析。

## 步骤二、恢复数据到新集群

在获取误操作的时间点后，可执行如下操作，将整个集群恢复到一个新集群。

- 进入集群详情页，在左侧导航栏中，选择配置与管理 > 备份恢复。
- 在备份恢复页面，单击按时间点恢复。

创建备份

按时间点恢复

备份SQL文件及Binlog


2021年3月8日

-

2021年3月8日

📅

一级备份（快照）总大小 🔍 2.05 GB（其中免费额度为50%当前存储用量，大约为 1.82 GB）[备份功能FAQ](#)

备份集ID	备份开始时间	备份结束时间	状态	一致性快照时间点 <span>🔍</span>	备份方法	备份类型	大小 <span>🔍</span>	存储位置	是否有效	备份策略
	2021年3月8日 15:30:09	2021年3月8日 15:30:19	备份 完成	2021年3月8日 15:30:11	快照 备份	全量 备份	654.44 MB	二级 备份	有效	自动 备份

3. 在打开的克隆实例页面，设置要恢复至的时间点以及新集群的参数。

i. 选择新集群的付费模式。

#### 说明

若您恢复完成后不打算使用新的集群，而是继续使用原集群，那么新集群仅作临时用，付费模式选择按量付费即可。

若您恢复完成后打算直接使用这个新的集群，则付费模式和原集群保持一致，或者根据需要进行调整。

ii. 操作类型选择恢复到过去时间点，并在备份时间点中选择要将集群恢复到的过去的时间点，即误操作发生前的时间。

#### 操作类型

从备份集恢复数据

恢复到过去时间点

克隆一个独立集群

基于备份集和Redo日志，恢复到过去的某个时间点

#### 备份时间点

2021-03-14 15:15:15

4. 设置其他集群参数。

#### 说明

若您恢复完成后不打算使用新的集群，而是继续使用原集群，那么新集群仅作临时用，建议使用默认设置即可，其规格和设置不会影响整个恢复过程。

若您恢复完成后打算直接使用这个新的集群，则这些参数可以和原集群保持一致，或者根据需要进行调整。

5. 阅读并选中服务协议，单击立即购买。

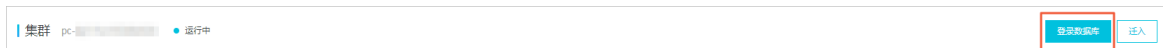
6. 在支付页面，确认待支付订单信息，单击订购即可。

## 步骤三、在新集群中进行数据比对

当全量集群数据恢复至新集群中完成后，您需要登录到新集群，并进行数据比对，验证数据。

您可通过DMS登录新集群，也可通过其他客户端或者命令行的方式。本文介绍通过DMS登录集群进行验证，关于其他方式，请参考[连接数据库集群](#)。

1. 在新集群的基本信息页面右上角，单击登录数据库。



2. 在弹出的对话框中，输入原集群中创建的数据库账号和数据库密码，单击登录。



3. 登录DMS后请刷新页面，在左侧导航栏中，单击已登录实例。
4. 在已登录实例列表中，单击目标集群名称，找到并双击目标数据库名称，即可切换到目标数据库进行管理。



5. 查找到误操作的数据，确认其是否已恢复到误操作前，且其他数据也和误操作发生前一致。

数据验证完成后，如果您想后续直接使用新集群，则将您的业务和应用切换至新集群的连接地址，并在新集群中完成相关配置（如白名单、参数配置等），确保您的业务可以和以前一样正常运行。若您还是想继续使用原集群，则需要继续执行**步骤四、将新集群的数据迁移到原集群**。

## 步骤四、将新集群的数据迁移到原集群

确认新集群的数据之后，若您还是想继续使用原集群，您可以使用DTS将数据从新集群迁移回原集群。

1. 登录**数据传输控制台**，在左侧导航栏，单击**数据迁移**。
2. 在**迁移任务列表**页面顶部，选择迁移的目标集群所属地域。

概览

迁移任务列表

华东1 (杭州)

华东2 (上海)

华北1 (青岛)

华北2 (北京)

华南1 (深圳)

华北3 (张家口)

香港

美国 (硅谷)

美国 (弗吉尼亚)

新加坡

阿联酋 (迪拜)

数据集成 New

德国 (法兰克福)

马来西亚 (吉隆坡)

澳大利亚 (悉尼)

印度 (孟买)

英国 (伦敦)

日本 (东京)

印度尼西亚 (雅加达)

华北5 (呼和浩特)

数据迁移

DTS常见问题

刷新

创建迁移任务

体验新版控制台

数据订阅

数据同步

迁移任务名

请输入迁移任务名进行搜索

搜索

排序: 默认排序

状态: 全部

标签

3. 单击页面右上角的创建迁移任务。
4. 配置迁移任务的源库和目标库连接信息。  
源库，即新集群；目标库信息，即误操作发生的原集群，也是此次要恢复的集群。

1.源库及目标库

2.迁移类型及列表

3.高级配置

4.预检查

\* 任务名称: PolarDB集群间迁移

源库信息

\* 实例类型: PolarDB

DTS支持链路类型

\* 实例地区: 华东1 (杭州)

\* PolarDB实例ID: pc-\*\*\*\*\*

\* 数据库账号: dtstest

\* 数据库密码: \*\*\*\*\*

测试连接

测试通过

目标库信息

\* 实例类型: PolarDB

\* 实例地区: 华东1 (杭州)

\* PolarDB实例ID: pc-\*\*\*\*\*

\* 数据库账号: dtstest

\* 数据库密码: \*\*\*\*\*

测试连接

测试通过

取消

授权白名单并进入下一步

5. 完成上述配置后，单击页面右下角的授权白名单并进入下一步。

说明

此步骤会将DTS服务器的IP地址自动添加到源和目标集群的白名单中，用于保障DTS服务器能够正常连接源和目标集群。

6. 选择迁移类型和迁移对象。

1.源库及目标库

2.迁移类型及列表

3.高级配置

4.预检查

\* 迁移类型：☒结构迁移 ☒全量数据迁移 ☒增量数据迁移 注：增量迁移不支持trigger的同步，详情请[参考文档](#)

注：DTS全量任务运行期间，不要清理DTS任务启动后源库产生的增量数据日志。源库如果过早清理日志，可能会导致DTS增量任务失败。

数据迁移适合于短期的数据迁移场景，主要应用于上云迁移、数据库扩容拆分及阿里云数据库之间的数据迁移。  
如果需要进行长期的数据实时同步，请使用数据同步功能。

迁移对象

若全局搜索，请先展开树 | 🔍

dtstestdata

Tables

Views

全选中

>

<

已选对象(鼠标移到对象行,点击编辑可修改对象名或过滤条件) [详情点我](#)

| 🔍

dtstestdata (2个对象)

customer

order

全移除

\* 映射名称更改：  
☒ 不进行库表名称批量更改 ☐ 要进行库表名称批量更改

\* 源库、目标库无法连接后的重试时间  分钟 ?

\* 源表DMS\_ONLINE\_DDL过程中是否复制临时表到目标库：  
☐ 是 ☒ 否 ?

注意：


1. 数据迁移只会将源库的数据（结构）复制一份到目标数据库,并不会对源数据库数据（结构）造成影响。  
2. 在做结构和全量迁移期间不要做DDL操作，否则可能导致任务失败。

取消

上一步

保存

预检查并启动

配置	说明
迁移类型	勾选 <b>结构迁移</b> 和 <b>全量数据迁移</b> 。
迁移对象	<p>在<b>迁移对象</b>框中单击待迁移的对象，然后单击  图标将其移动至<b>已选择对象</b>框。</p> <div>  <b>注意</b> 迁移对象选择的粒度为库、表、列。         </div>

7. 上述配置完成后，单击页面右下角的**预检查并启动**，预检查通过后，单击下一步。
8. 在弹出的**购买配置确认**对话框，选择**链路规格**并选中**数据传输（按量付费）服务条款**。
9. 单击**购买并启动**，迁移任务正式开始。

② 说明 请勿手动结束迁移任务，否则可能会导致数据不完整。您只需等待迁移任务完成即可，迁移任务会自动结束。

## 步骤五、释放集群

等待迁移任务完成后，原集群即已经恢复到选中的时间点。此时，若您还是想继续使用原集群，您可以释放新集群，避免产生不必要的费用。

或者，当验证恢复完成后，若您打算直接使用新集群，在完成业务切换以及新集群配置后，您可以释放原集群。

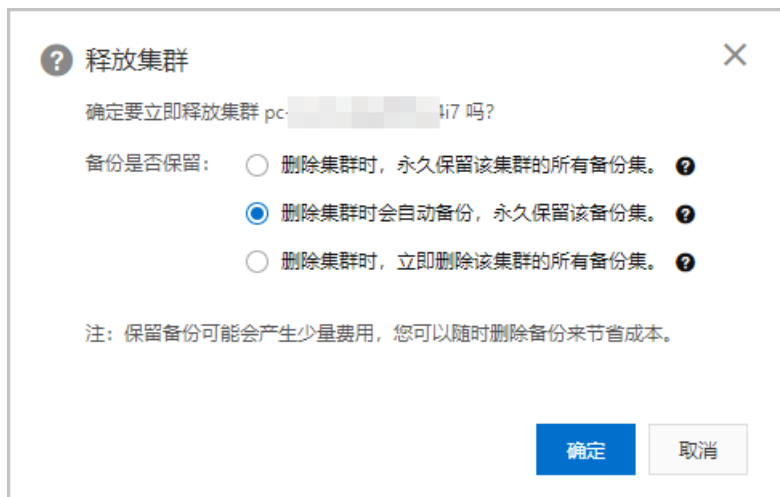
1. 在集群列表页找到目标集群，单击右侧操作栏的更多 > 释放。



2. 在弹出的释放集群对话框中，选择备份保留策略。

### 说明

如果您选择继续使用原集群，且您已确认数据恢复和迁移成功，在释放新集群时，建议您选择删除集群时，立即删除该集群的所有备份集。



3. 单击确定。

## 4.3.3. 集群恢复：按备份集恢复

若您要恢复至的时间点是某个备份集（快照）的时间点，可通过按备份集（快照）恢复的方式恢复全量集群数据。

### 前提条件

- 请确保误操作前已存在备份集（快照）。

- 建议在业务低峰期进行数据恢复。

## 注意事项

恢复后的集群包含原集群的数据和账号信息，不包含原集群的参数设置。

## 步骤一、恢复数据到新集群

- 进入集群详情页，在左侧导航栏中，选择配置与管理 > 备份恢复。
- 找到目标备份集（快照），单击右侧操作栏中的恢复数据到新集群。

备份集ID	备份开始时间	备份结束时间	状态	一致性快照时间点	备份方法	备份类型	大小	存储位置	是否有效	备份策略	操作
12345678	2021年3月16日11:09:45	2021年3月16日11:09:55	备份完成	2021年3月16日11:09:47	快照备份	全量备份	4.26 GB	一级备份	有效	手动备份	<a href="#">恢复数据到新集群</a> <a href="#">删除备份</a>

- 在打开的克隆实例页面，设置要恢复的备份集以及新集群的参数。
  - 选择新集群的付费模式。

### 说明

若您恢复完成后不打算使用新的集群，而是继续使用原集群，那么新集群仅作临时用，付费模式选择按量付费即可。

若您恢复完成后打算直接使用这个新的集群，则付费模式和原集群保持一致，或者根据需要进行调整。

- 操作类型选择从备份集恢复数据，并在备份集的下拉框中选择需要恢复的备份集。

操作类型

从备份集恢复数据

恢复到过去时间点

克隆一个独立集群

基于备份集，恢复一个新集群

备份集

2021年4月11日20:34:15

说明 此处展示的为各备份集的备份开始时间，您可以根据该时间确定是否为需要恢复的备份集。

- 设置其他集群参数。

### 说明

若您恢复完成后不打算使用新的集群，而是继续使用原集群，那么新集群仅作临时用，建议使用默认设置即可，其规格和设置不会影响整个恢复过程。

若您恢复完成后打算直接使用这个新的集群，则这些参数可以和原集群保持一致，或者根据需要进行调整。

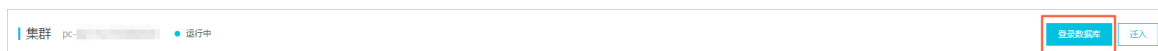
- 阅读并选中服务协议，单击立即购买。
- 在支付页面，确认待支付订单信息，单击订购即可。

## 步骤二、在新集群中进行数据比对

当全量集群数据恢复至新集群中完成后，您需要登录到新集群，并进行数据比对，验证数据。

您可通过DMS登录新集群，也可通过其他客户端或者命令行的方式。本文介绍通过DMS登录集群进行验证，关于其他方式，请参考[连接数据库集群](#)。

1. 在新集群的基本信息页面右上角，单击登录数据库。



2. 在弹出的对话框中，输入原集群中创建的数据库账号和数据库密码，单击登录。

3. 登录DMS后请刷新页面，在左侧导航栏中，单击已登录实例。
4. 在已登录实例列表中，单击目标集群名称，找到并双击目标数据库名称，即可切换到目标数据库进行管理。



5. 查找到误操作的数据，确认其是否已恢复到误操作前，且其他数据也和误操作发生前一致。

### 步骤三、将新集群的数据迁移到原集群

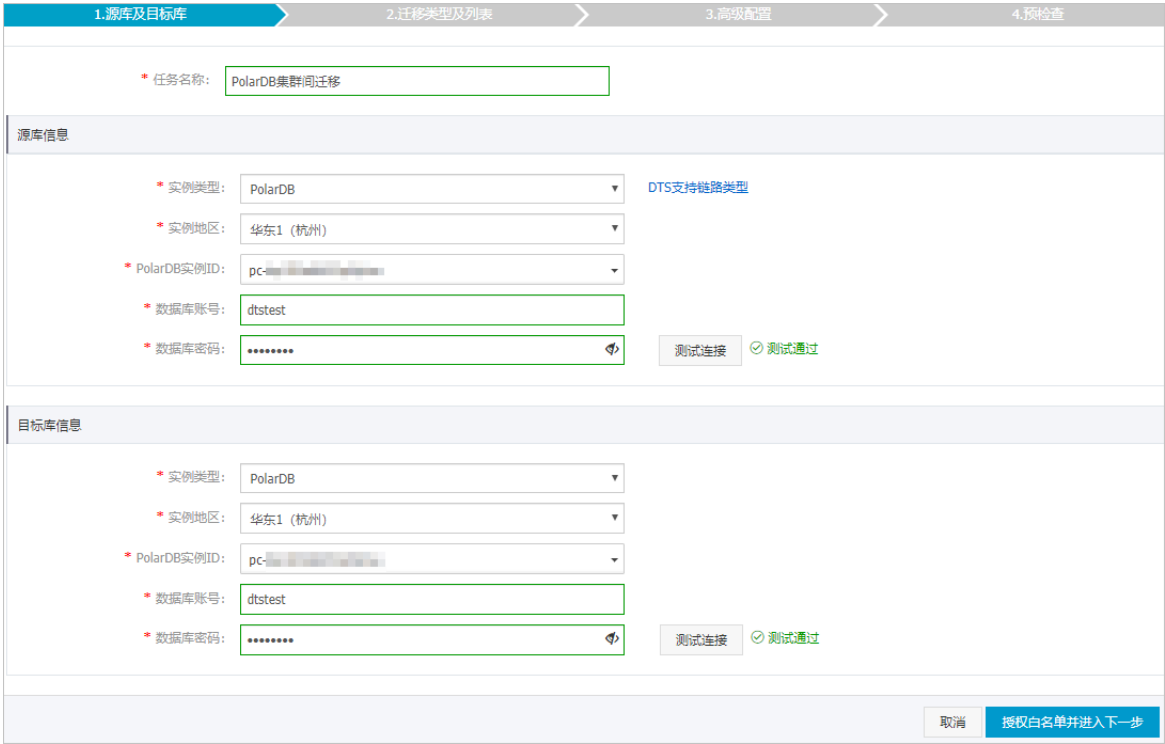
确认新集群的数据之后，若您还是想继续使用原集群，您可以使用DTS将数据从新集群迁移回原集群。

- 1. 登录[数据传输控制台](#)，在左侧导航栏，单击数据迁移。
- 2. 在迁移任务列表页面顶部，选择迁移的目标集群所属地域。




- 3. 单击页面右上角的创建迁移任务。
- 4. 配置迁移任务的源库和目标库连接信息。

源库，即新集群；目标库信息，即误操作发生的原集群，也是此次要恢复的集群。



- 5. 完成上述配置后，单击页面右下角的授权白名单并进入下一步。

 **说明** 此步骤会将DTS服务器的IP地址自动添加到源和目标集群的白名单中，用于保障DTS服务器能够正常连接源和目标集群。

- 6. 选择迁移类型和迁移对象。

1.源库及目标库

2.迁移类型及列表

3.高级配置

4.预检查

\* 迁移类型：☒结构迁移    ☒全量数据迁移    ☒增量数据迁移    注：增量迁移不支持trigger的同步，详情请[参考文档](#)

注：DTS全量任务运行期间，不要清理DTS任务启动后源库产生的增量数据日志。源库如果过早清理日志，可能会导致DTS增量任务失败。

数据迁移适合于短期的数据迁移场景，主要应用于上云迁移、数据库扩容拆分及阿里云数据库之间的数据迁移。  
如果需要进行长期的数据实时同步，请使用数据同步功能。

迁移对象

若全局搜索，请先展开树 | Q

dtstestdata

Tables

Views

全选中

>

<

已选择对象（鼠标移到对象行，点击编辑可修改对象名或过滤条件）[详情点我](#)

| Q

dtstestdata (2个对象)

customer

order

全移除

\* 映射名称更改：  
☒ 不进行库表名称批量更改    ☐ 要进行库表名称批量更改

\* 源库、目标库无法连接后的重试时间  
720 分钟 ?

\* 源库DMS\_ONLINE\_DDL过程中是否复制临时表到目标库：  
☐ 是    ☒ 否 ?


注意：  
1. 数据迁移只会将源库的数据（结构）复制一份到目标数据库，并不会对源数据库数据（结构）造成影响。  
2. 在做结构和全量迁移期间不要做DDL操作，否则可能导致任务失败。

取消

上一步

保存

预检查并启动

配置	说明
迁移类型	勾选 <b>结构迁移</b> 和 <b>全量数据迁移</b> 。
迁移对象	<p>在<b>迁移对象</b>框中单击待迁移的对象，然后单击  图标将其移动至<b>已选择对象</b>框。</p> <div>  <b>注意</b> 迁移对象选择的粒度为库、表、列。 </div>

7. 上述配置完成后，单击页面右下角的**预检查并启动**，预检查通过后，单击下一步。
8. 在弹出的**购买配置确认**对话框，选择**链路规格**并选中**数据传输（按量付费）服务条款**。
9. 单击**购买并启动**，迁移任务正式开始。

## 步骤四、释放集群

等待迁移任务完成后，原集群即已经恢复到选中的时间点。此时，若您还是想继续使用原集群，您可以释放新集群，避免产生不必要的费用。



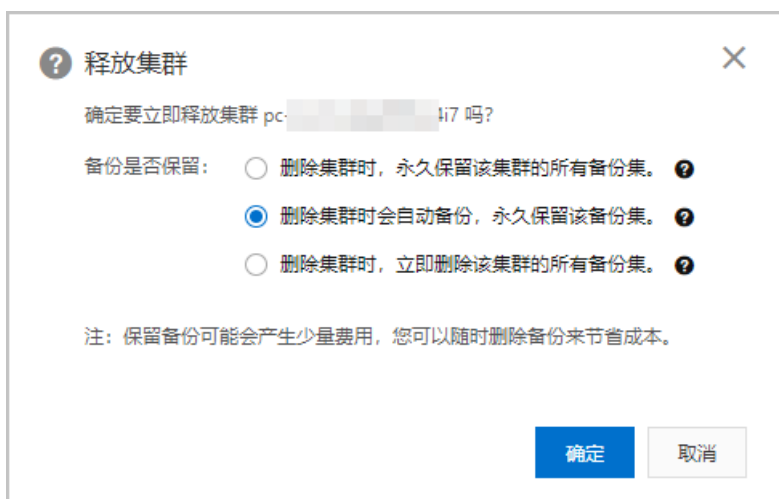
1. 在集群列表页找到目标集群，单击右侧操作栏的更多 > 释放。



2. 在弹出的释放集群对话框中，选择备份保留策略。

#### ? 说明

如果您选择继续使用原集群，且您已确认数据恢复和迁移成功，在释放新集群时，建议您选择删除集群时，立即删除该集群的所有备份集。



3. 单击确定。

## 4.4. 库表恢复的方式恢复对应表

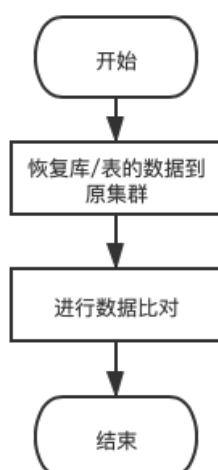
### 4.4.1. 整体流程和预估时间

当发生部分表的误操作时，可进行库表恢复，将误操作的某个或某些表恢复到原集群。

库表恢复分为按时间点恢复和按备份集（快照）恢复，两者的差别在于要恢复至的时间点是否是备份集的时间点。如果是备份集的时间点，则可选择按备份集（快照）恢复，更方便。如果要恢复至的时间不是已有的备份集的时间点，则只能选择按时间点恢复。

#### 整体流程

不管是按时间点恢复还是按备份集（快照）恢复，两者的关键流程是一致的：先将库或者表的数据恢复到原集群，然后在原集群中进行数据比对。整体流程如下：



## 预估时间

库表恢复所需的时间与备份集的大小和Redo日志的大小有关。预估耗时如下，供参考：

步骤	预估耗时
恢复备份集（快照）数据	3 GB/分钟
恢复Redo日志增量数据	1.5 GB/分钟
<div>❓ 说明 仅按时间点恢复的方式需要恢复该类数据。</div>	

例如，若集群备份集（快照）数据大小为10 GB，Redo日志大小为3 GB，则每个步骤耗时预估为：

- 恢复备份集（快照）数据：10/3=3.3分钟
- 恢复Redo日志增量数据：3/1.5=2分钟

总耗时预估为5分钟以上。

❓ 说明 建议在业务低峰期进行数据恢复。


### 4.4.2. 库表恢复：按时间点恢复

通过一个完整的数据全量备份（快照）以及后续一段时间的Redo日志，就可以将部分库或部分表的数据恢复到日志保存期限内任意时间点。在知晓误操作发生的时间点后，可以通过此方式，将误操作的数据库或者表恢复到误操作发生之前。

#### 前提条件

- 已存在备份集。因为按时间点恢复是先将所选时间点前的一个全量数据备份恢复到集群，然后根据Redo日志恢复增量数据到所选时间点。若您希望提升恢复速度，可以开启增强备份，从而缩短备份周期，增加备份密度。
- 仅PolarDB集群版支持库表恢复，且版本需为以下版本之一：

- PolarDB MySQL引擎5.6版本且内核小版本需为5.6.1.0.25及以上。关于如何升级小版本，请参见[版本管理](#)。
- PolarDB MySQL引擎5.7版本且内核小版本需为5.7.1.0.8及以上。
- PolarDB MySQL引擎8.0版本且内核小版本需为8.0.1.1.14及以上。

 **说明** PolarDB MySQL引擎内核小版本8.0.2暂时不支持库表恢复功能。

- 开启透明数据加密TDE的集群暂不支持库表恢复功能。
- GDN中的集群不支持库表恢复功能。

## 注意事项

- 库表恢复只会恢复指定的表，操作时请确认已选中所有需要恢复的表。

 **说明** 若无法确定所有涉及的表，建议您使用集群恢复的方式恢复全量集群。

- 集群内的表低于50000张才可以使用库表恢复功能，超过50000张表时无法使用。库表恢复每次最多支持恢复100张表。若选择恢复库，则支持恢复的表数量为该库下所有的表。

 **说明** 若您需要恢复的表数量较多，建议您使用集群恢复的方式恢复全量集群。


- 库表恢复功能不支持恢复触发器（Trigger），若原表设置了Trigger，该Trigger不会被恢复。
- 库表恢复功能不支持恢复外键（Foreign Key），若原表设置了Foreign Key，该Foreign Key不会被恢复。
- 建议在业务低峰期进行数据恢复。

## 步骤一、获取误操作的时间点

如果您已知误操作的时间点，可以跳过本步骤。如果您不记得误操作的具体时间点，可以通过此步骤查看。

### 方式一：SQL洞察


对于PolarDB MySQL集群版，若集群已开通SQL洞察功能，可通过该功能查看审计日志，获取误操作的时间点。

 **说明** SQL洞察会收费，费用与审计日志的存储容量和存储时长有关。关于更多SQL洞察的说明，请参见[SQL洞察](#)。

SQL洞察功能仅支持查看开通该功能后产生的SQL日志。因此若集群未开通SQL洞察，可通过方式二查看误操作时间点。

### 方式二：远程获取并解析Binlog日志

1. 为集群开启Binlog。详细操作步骤，请参见[开启Binlog](#)。

 **说明** 您需要先为集群开启Binlog才能查看并获取Binlog日志，否则将出现 `You are not using binary logging` 错误提示。

2. 在本地服务器上安装MySQL，并通过MySQL客户端连接集群。详细操作步骤，请参见[使用客户端连接集群](#)。本文以Linux系统为例。

```
root@~:~# mysql -h polardb.rwlb.rds.aliyuncs.com -P3306 -utest_api -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is
Server version: 8.0.13 Source distribution

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

3. 在已连接集群的客户端中执行如下命令查看目标集群的Binlog文件列表：

```
show binary logs;
```

返回结果如下：

```
+-----+-----+
| Log_name          | File_size |
+-----+-----+
| mysql-bin.000005 |      2639 |
+-----+-----+
1 row in set (0.00 sec)
```

4. 在客户端执行如下命令，远程获取Binlog日志并保存至本地。

本文中以Linux系统为例，需先执行exit退出MySQL后，才能远程获取Binlog日志并保存至本地。

```
mysql> exit
Bye
root@~:~# mysqlbinlog -utest_api -p -h polardb.rwlb.rds.aliyuncs.com --read-from-remote-server --raw mysql-bin.000005
Enter password:
root@~:~#
```

执行如下语句：

```
mysqlbinlog -u<用户名> -p<密码> -h<连接地址> --read-from-remote-server --raw mysql-bin.**
****
```

参数	说明	示例值
-u	集群中的账号名称。	test_api
-p	以上账号的密码。如果此处留空，则在执行此命令后会被要求输入。	TestPwd123
-h	集群的公网连接地址。 <div><div>② 说明</div><div><ul style="list-style-type: none"><li>若您的集群连接地址使用的是默认的端口号3306，端口号可省略不写，否则需要在连接地址后加上端口号。</li><li>当前仅支持通过主地址或集群地址（包括默认集群地址和自定义集群地址）的公网连接地址远程获取Binlog。如何申请公网连接地址，请参见<a href="#">申请集群地址和主地址</a>。</li></ul></div></div>	test-polardb.rwlb.rds.aliyuncs.com

参数	说明	示例值
<code>--raw</code>	表示将获取到的Binlog文件按数据原有格式打印，而不会展示解析后的数据。	<code>--raw</code>
<code>mysql-bin.*****</code>	通过 <code>show binary logs;</code> 命令获取的目标Binlog文件的名字，即 <code>Log_name</code> 。	<code>mysql-bin.000005</code>

示例：

```
mysqlbinlog -utest_api -p -h test-polaradb.rwlb.rds.aliyuncs.com --read-from-remote-server --raw mysql-bin.000005
```

5. 执行如下命令，通过mysqlbinlog工具查看Binlog日志文件内容：

```
mysqlbinlog -vv --base64-output=decode-rows mysql-bin.***** | more
```

#### 说明

- `-vv`：查看具体SQL语句及备注。
- `--base64-output=decode-rows`：解析Binlog日志文件。

具体的Binlog日志内容如下图中红框所示部分：

```
[root@iz... ~]# mysqlbinlog -vv --base64-output=decode-rows mysql-bin.000110 | more
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!40019 SET @@session.max_insert_delayed_threads=0*/;
/*!50003 SET @@OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#160217 23:04:37 server id 2802943055 end_log_pos 107 Start: binlog v 4, server v 5.5.18.1-log created 160217 23:04:37
# at 107
#160217 23:04:38 server id 2802943055 end_log_pos 171 Query thread id=584632 exec time=0 error code=0
SET TIMESTAMP=1455721478/*!*/;
SET @@session.pseudo_thread_id=584632/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=0, @@session.unique_checks=1, @@session.autocommit=1/*!*/;
SET @@session.sql_mode=2097152/*!*/;
SET @@session.auto_increment_increment=1, @@session.auto_increment_offset=1/*!*/;
/*!\\C utf8 *//*!*/;
SET @@session.character_set_client=33,@@session.collation_connection=33,@@session.collation_server=33/*!*/;
SET @@session.lc_time_names=0/*!*/;
SET @@session.collation_database=DEFAULT/*!*/;
BEGIN
/*!*/;
# at 171
```

6. 在获取Binlog日志文件内容后，可参见[解析Binlog日志](#)进行Binlog日志解析。

## 步骤二、开始库表恢复

- 进入集群详情页，在左侧导航栏中，选择配置与管理 > 备份恢复。
- 在备份恢复页面，单击库（表）恢复。
- 在弹出的对话框中，您可以根据业务需要选择恢复方式为按时间点，并在恢复至中选择要恢复到的过去的时间点，即误操作发生前的时间。

恢复位置	<input checked="" type="radio"/> 恢复到当前集群
恢复方式	<input type="radio"/> 按备份集 <input checked="" type="radio"/> 按时间点
可恢复时间点	2021-03-30 14:37:43 至 2021-04-06 14:37:43
恢复至	2021-03-30 14:37:43

说明 恢复至的时间点需符合可恢复时间点的限制，即仅支持恢复到7天内的任意时间点。

4. 在需要恢复的库和表区域左侧，选中需要恢复的目标库，并在右侧选中目标表。

需要恢复的库和表

请输入

库名

☐

☐

☒

☐

库名

表名

☐

☒

<< < 1 > >>

已选择的库和表

库名	恢复后库名	表名	恢复后表名	操作
				删除

说明

- 若选中目标库后但不指定具体的数据表，默认恢复该库下的所有数据。
- 选中目标库或目标表后，系统会自动在原库或原表名称（如 `test`）后加 `_backup` 作为新库或新表的名称（如 `test_backup`），您也可以自定义恢复后的库名或表名。

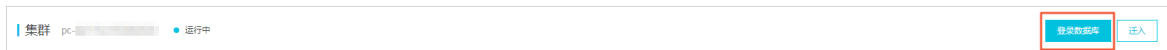
5. 单击确定即可。

### 步骤三、进行数据比对

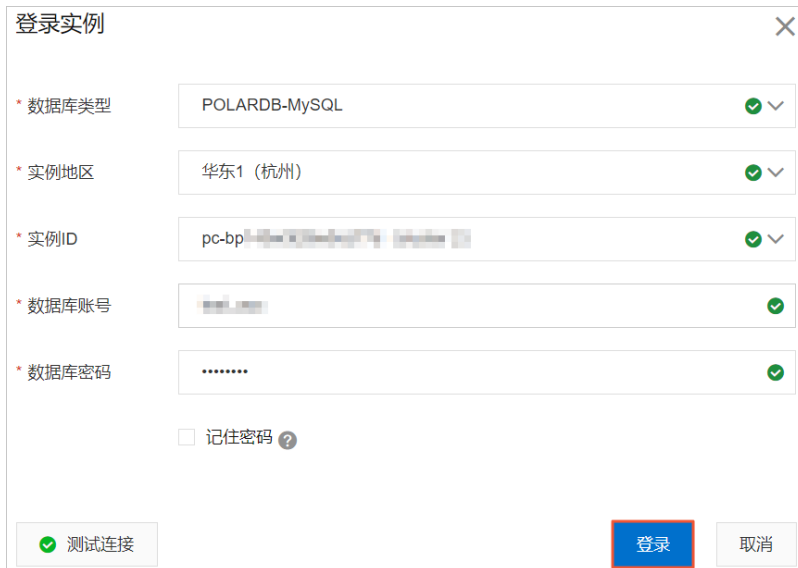
当库表恢复完成后，您可以登录集群，进行数据比对，验证数据。

您可通过DMS登录集群，也可通过其他客户端或者命令行的方式。本文介绍通过DMS登录集群进行验证，关于其他方式，请参考[连接数据库集群](#)。

1. 在集群的基本信息页面右上角，单击登录数据库。



2. 在弹出的对话框中，输入集群的数据库账号和数据库密码，单击登录。



3. 登录DMS后请刷新页面，在左侧导航栏中，单击已登录实例。
4. 在已登录实例列表中，单击目标集群名称，找到并双击目标数据库名称，即可切换到目标数据库进行管理。




5. 查找到误操作的数据，确认其是否已恢复到误操作前，且其他数据也和误操作发生前一致。

### 4.4.3. 库表恢复：按备份集恢复

若您要恢复至的时间点是某个备份集（快照）的时间点，可通过按备份集（快照）恢复的方式恢复误操作的数据库或者表的数据。

## 前提条件

- 请确保误操作前已存在备份集（快照）。
- 仅PolarDB集群版支持库表恢复，且版本需为以下版本之一：
  - PolarDB MySQL引擎5.6版本且内核小版本需为5.6.1.0.25及以上。关于如何升级小版本，请参见[版本管理](#)。
  - PolarDB MySQL引擎5.7版本且内核小版本需为5.7.1.0.8及以上。
  - PolarDB MySQL引擎8.0版本且内核小版本需为8.0.1.1.14及以上。

 **说明** PolarDB MySQL引擎内核小版本8.0.2暂时不支持库表恢复功能。

- 开启透明数据加密TDE的集群暂不支持库表恢复功能。
- GDN中的集群不支持库表恢复功能。

## 注意事项

- 库表恢复只会恢复指定的表，操作时请确认已选中所有需要恢复的表。

 **说明** 若无法确定所有涉及的表，建议您使用集群恢复的方式恢复全量集群。

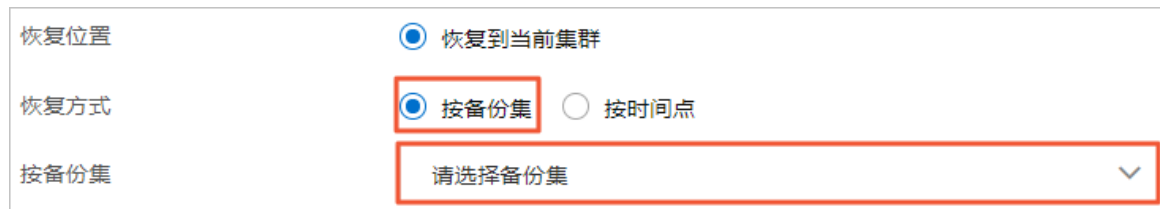
- 集群内的表低于50000张才可以使用库表恢复功能，超过50000张表时无法使用。库表恢复每次最多支持恢复100张表。若选择恢复库，则支持恢复的表数量为该库下所有的表。

 **说明** 若您需要恢复的表数量较多，建议您使用集群恢复的方式恢复全量集群。

- 库表恢复功能不支持恢复触发器（Trigger），若原表设置了Trigger，该Trigger不会被恢复。
- 库表恢复功能不支持恢复外键（Foreign Key），若原表设置了Foreign Key，该Foreign Key不会被恢复。

## 步骤一、开始库表恢复

1. 进入集群详情页，在左侧导航栏中，选择**配置与管理 > 备份恢复**。
2. 在**备份恢复**页面，单击**库（表）恢复**。
3. 在弹出的对话框中，您可以根据业务需要选择**恢复方式**为**按备份集**，并在备份集列表中选择目标备份集。



对话框显示了恢复配置选项：

- 恢复位置**：包含一个单选按钮 ☒ **恢复到当前集群**。
- 恢复方式**：包含两个单选按钮，☒ **按备份集** 和 ☐ **按时间点**。
- 按备份集**：下方是一个下拉菜单，显示 **请选择备份集**，并带有向下箭头。

4. 在**需要恢复的库和表**区域左侧，选中需要恢复的目标库，并在右侧选中目标表。



需要恢复的库和表

请输入

Q

库名

☐

☐

☒

☐

<<

<

1

>

>>

请输入

Q

库名

表名

☐

☒

<<

<

1

>

>>

已选择的库和表

库名	恢复后库名	表名	恢复后表名	操作
				删除

#### 说明

- 若选中目标库后但不指定具体的数据表，默认恢复该库下的所有数据。
- 选中目标库或目标表后，系统会自动在原库或原表名称（如 `test`）后加 `_backup` 作为新库或新表的名称（如 `test_backup`），您也可以自定义恢复后的库名或表名。

5. 单击**确定**即可。

## 步骤二、进行数据比对

当库表恢复完成后，您可以登录集群，进行数据比对，验证数据。

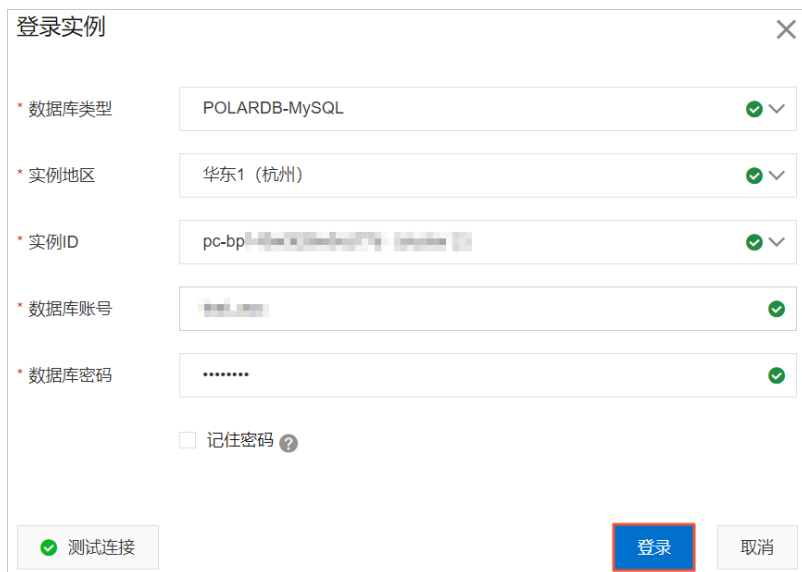
您可通过DMS登录集群，也可通过其他客户端或者命令行的方式。本文介绍通过DMS登录集群进行验证，关于其他方式，请参考[连接数据库集群](#)。

1. 在集群的基本信息页面右上角，单击登录数据库。

集群 pc- 运行中

登录数据库 迁入

2. 在弹出的对话框中，输入集群的数据库账号和数据库密码，单击登录。



3. 登录DMS后请刷新页面，在左侧导航栏中，单击已登录实例。
4. 在已登录实例列表中，单击目标集群名称，找到并双击目标数据库名称，即可切换到目标数据库进行管理。



5. 查找到误操作的数据，确认其是否已恢复到误操作前，且其他数据也和误操作发生前一致。

## 4.5. 从集群回收站恢复误删的集群

集群回收站用于保存已释放的PolarDB MySQL引擎集群，您可以在回收站中找到误删集群，并将其恢复到新集群。

### 操作步骤

1. 在PolarDB控制台左侧导航栏中单击集群回收站。
2. 找到目标集群，单击右侧操作栏中的恢复数据到新集群。

集群回收站							
集群ID	请输入						
集群ID/名称	地域	主节点配置	兼容性	创建时间	删除时间	状态	操作
+ [集群名称]	华南1 (深圳)	4核 16GB	100% 兼容 MySQL 5.6	2020年4月28日 17:01:58	2020年5月8日 15:20:18	● 已释放	<a href="#">恢复数据到新集群</a>
+ [集群名称]	华南1 (深圳)	4核 16GB	兼容ORACLE语法	2020年4月26日 13:52:00	2020年4月28日 16:10:17	● 已释放	<a href="#">恢复数据到新集群</a>

### 3. 选择商品类型为包年包月或按量付费。

- **包年包月**：在创建集群时支付计算节点的费用，而存储空间会根据实际数据量按小时计费，并从账户中按小时扣除。
- **按量付费**：无需预先支付费用，计算节点和存储空间（根据实际数据量）均按小时计费，并从账户中按小时扣除。


### 4. 设置如下参数。

参数	说明
地域	<p>集群所在的地理位置。购买后无法更换地域。</p> <p>② 说明 请确保PolarDB与需要连接的ECS创建于同一个地域，否则它们无法通过内网互通，只能通过外网互通，无法发挥最佳性能。</p>
创建方式	选择从回收站恢复，即从回收站恢复已经删除的数据库集群。
原版本	选择已删除集群的版本。
已删除集群	选择已删除的集群ID。
历史备份	<p>选择需要恢复的备份。</p> <p>② 说明 历史备份显示的备份时间为UTC时间，数据备份列表页签中显示的备份时间为当前系统时间，请确保所选择的历史备份无误。</p> <p>例如，当前备份列表显示的备份时间为 2020年05月08日 10:00:00（北京时间），对应的历史备份为 2020-05-08T02:00:00Z。</p>
主可用区	<p>集群的主可用区。</p> <ul style="list-style-type: none"> <li>○ 可用区是地域中的一个独立物理区域，不同可用区之间没有实质性区别。</li> <li>○ 您可以选择将PolarDB与ECS创建在同一可用区或不同的可用区。</li> <li>○ 您只需要选择主可用区，系统会自动选择备可用区。</li> </ul>
网络类型	固定为专有网络，无需选择。

参数	说明
VPC网络 VPC交换机	<p>请确保PolarDB与需要连接的ECS创建于同一个VPC，否则它们无法通过内网互通，无法发挥最佳性能。</p> <ul style="list-style-type: none"> <li>如果您已创建符合您网络规划的VPC，直接选择该VPC。例如，如果您已创建ECS，且该ECS所在的VPC符合您的规划，那么选择该VPC。</li> <li>如果您未创建符合您网络规划的VPC，您可以使用默认VPC和交换机： <ul style="list-style-type: none"> <li>默认VPC： <ul style="list-style-type: none"> <li>在您选择的地域中是唯一的。</li> <li>网段掩码是16位，如192.168.0.0/16，最多可提供65536个私网IP地址。</li> <li>不占用阿里云为您分配的VPC配额。</li> </ul> </li> <li>默认交换机： <ul style="list-style-type: none"> <li>在您选择的可用区中是唯一的。</li> <li>网段掩码是20位，如192.168.0.0/20，最多可提供4096个私网IP地址。</li> <li>不占用VPC中可创建交换机的配额。</li> </ul> </li> </ul> </li> <li>如果以上默认VPC和交换机无法满足您的要求，您可以自行创建VPC和交换机，详情请参见<a href="#">创建和管理专有网络</a>。</li> </ul>
兼容性	PolarDB集群的数据库引擎版本，默认与已删除集群的版本保持一致，不可变更。
系列	固定为 <b>集群版（2-16个节点）【推荐】</b> ，无需选择。
小版本号	<p>您可以选择小版本号为<b>8.0.1（与MySQL 8.0.13完全兼容）【推荐】</b>或<b>8.0.2（与MySQL 8.0.18完全兼容）</b>。关于各小版本的具体信息，请参见<a href="#">内核发布记录</a>。</p> <div> <p> 注意</p> <ul style="list-style-type: none"> <li>仅当PolarDB集群的兼容性选择为<b>MySQL 8.0</b>时，支持该配置。</li> <li>8.0.2版本目前正在公测中，可能存在若干稳定性风险，不承诺<a href="#">服务等级协议SLA</a>。</li> </ul> </div>
子系列	<p>PolarDB集群版支持<b>通用规格</b>和<b>独享规格</b>两种子系列，其中：</p> <p>关于两种类型的详细对比，请参见<a href="#">如何选择通用规格和独享规格</a>。</p>
节点规格	按需选择，建议不低于已删除集群的规格。关于PolarDB节点规格，详情请参见 <a href="#">计算节点规格</a> 。
节点个数	<p>无需选择。系统将默认创建规格相同的两个节点（一主一只读）。</p> <div> <p> 说明</p> <p>如果主节点故障，系统会自动将只读节点切换为新的主节点，并重新生成一个只读节点，关于只读节点的更多信息，请参见<a href="#">产品架构</a>。</p> </div>

参数	说明
存储费用	<p>无需选择。系统会根据实际数据使用量按小时计费，详情请参见<a href="#">存储空间计费规则</a>。</p> <p> <b>说明</b> 创建集群时无需选择存储容量，存储容量随数据量的增减而自动弹性伸缩。</p>
时区	设置集群时区，默认时区为UTC+08:00。
表名大小写	<p>设置集群表名是否区分大小写，您可以选择不区分大小写（默认）或当本地数据库区分大小时，您可以选择区分大小写，便于您迁移数据。</p> <p> <b>说明</b> 集群创建后该参数无法修改，请谨慎选择。</p>
删除（释放）集群时	<p>设置删除（释放）集群时的备份保留策略，支持如下3种策略：</p> <ul style="list-style-type: none"> <li>保留最后一个备份（释放前自动备份）（默认）：删除集群时保留最后一个备份。</li> <li>保留全部备份：删除集群时保留所有备份。</li> <li>不保留备份（释放后无法恢复）：删除集群时不保留任何备份。</li> </ul> <p> <b>说明</b> 删除（释放）集群时保留备份可能会产生少量费用，您可以随时删除备份来节省成本，详情请参见<a href="#">释放集群</a>。</p>
集群名称	<p>输入集群名称，集群名称需满足如下要求：</p> <ul style="list-style-type: none"> <li>不能以 <code>http://</code> 或 <code>https://</code> 开头。</li> <li>长度为2~256个字符。</li> </ul> <p>如果留空，系统将为自动生成一个集群名称，创建集群后还可以修改。</p>
资源组	<p>从已创建的资源组中选择一个目标资源组。创建资源组的操作步骤，请参见<a href="#">创建资源组</a>。</p> <p> <b>说明</b> 资源组是在单个云账号下将一组相关资源进行统一管理的容器，一个资源只能归属于一个资源组，详情请参见<a href="#">RAM资源分组与授权</a>。</p>

5. 设置购买时长（仅针对包年包月集群）和集群数量后，单击右侧的立即购买。
6. 在确认订单页面确认订单信息，阅读并选中服务协议，单击立即开通即可。
7. 在支付页面，确认未支付订单信息和支付方式，单击订购。

 **说明** 恢复到新集群所需的时间和您的备份大小有关，备份越大恢复到新集群所需的时间越久。

## 4.6. 从表回收站中恢复误删的表

由于DDL语句无法回滚，如果误删除了表（例如DROP TABLE），可能会导致数据丢失。PolarDB提供了表回收站的功能，删除的表会被临时转移到表回收站，因此您可以从表回收站恢复误删的表。

## 背景信息

关于更多表回收站的信息，请参见[表回收站](#)

## 前提条件

- 仅当集群版本为PolarDB MySQL引擎8.0版本，且内核小版本为8.0.1.1.2及以上时，才支持表回收站功能。
- 请确保误删前已开启表回收站功能：设置集群参数recycle\_bin为ON。
- 请确保误删表的时间点在表回收站内数据的最长保留周期内。可设置集群参数recycle\_bin\_retention（单位：秒），定义该最长保留周期。

## 操作步骤

1. 执行如下语句，查看回收站中所有临时保存的表。

```
CALL DBMS_RECYCLE.show_tables();
```

返回结果如下：


```
+-----+-----+-----+-----+-----+
+-----+
| SCHEMA          | TABLE          | ORIGIN_SCHEMA | ORIGIN_TABLE | RECYCLED_TIME
| PURGE_TIME      |
+-----+-----+-----+-----+-----+
+-----+
| __recycle_bin__ | __innodb_1063 | product_db    | t1           | 2019-08-08 11:01:46
| 2019-08-15 11:01:46 |
| __recycle_bin__ | __innodb_1064 | product_db    | t2           | 2019-08-08 11:01:46
| 2019-08-15 11:01:46 |
| __recycle_bin__ | __innodb_1065 | product_db    | parent       | 2019-08-08 11:01:46
| 2019-08-15 11:01:46 |
| __recycle_bin__ | __innodb_1066 | product_db    | child        | 2019-08-08 11:01:46
| 2019-08-15 11:01:46 |
+-----+-----+-----+-----+-----+
+-----+
4 rows in set (0.00 sec)
```

返回结果中的参数说明如下：

参数	说明
SCHEMA	回收站的Schema。
TABLE	进入回收站后的表名。
ORIGIN_SCHEMA	原始表的Schema。
ORIGIN_TABLE	原始表的表名。
RECYCLED_TIME	回收时间。
PURGE_TIME	预计在回收站中被清理的时间。

2. 执行如下语句，快速恢复回收站内的某张表。

```
CALL DBMS_RECYCLE.restore_table('RECYCLE_TABLE','DEST_DB','DEST_TABLE');
```

 **说明** PolarDB MySQL引擎8.0集群版的Revision version为8.0.1.1.12或以上才支持通过 `restore_table` 命令，快速恢复回收站内的表。您可以参见[查询版本号](#)确认集群版本。

该语句中的参数说明如下：

参数	说明
RECYCLE_TABLE	需要恢复的表在回收站内的表名。 <div> <b>说明</b> 如果仅传入此参数，会恢复到原始表。</div>
DEST_DB	为恢复后的表指定目标数据库。
DEST_TABLE	为恢复后的表指定新的表名。

示例：

```
call dbms_recycle.restore_table('__innodb_1063','testDB','testTable');
```

## 5. 巡检问题分析最佳实践

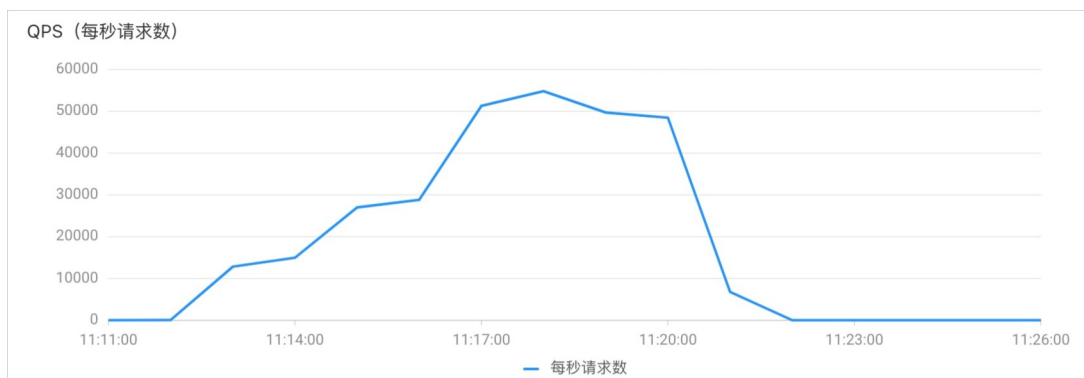
### 5.1. PolarDB MySQL引擎CPU使用率高

CPU作为数据库最核心的资源，是日常运维中需要重点关注的对象。CPU用满，会导致应用RT增高、业务卡顿，更严重会导致数据库实例hang死发生HA等问题，严重影响现网业务。正常情况下，对于CPU的监控需要设定安全水位，超出安全水位时要及时进行处理，否则会引发不可预期的严重后果。

#### 现网业务中的CPU使用率

随着业务的增长，数据库集群的规格可能已经不能满足业务流量的上涨需求。此时由于流量的不断增长，数据库集群的使用率逐渐提升，CPU使用率也逐渐升高。如果从性能曲线进行观察，必然存在某个指标（如QPS/IOPS）呈上涨趋势，与CPU使用率上涨趋势相似。如下图所示：

QPS（每秒请求数）



CPU使用率



此时，如果CPU出现瓶颈，基本可以断定数据库集群规格已不足以支撑当前的业务流量，此时最好对数据库集群增加只读节点或者扩容集群规格。

- 如果大部分业务场景都是读请求，可以通过增加只读节点，进行集群横向扩容，以分流读请求。具体操作请参见[增加或删除节点](#)。
- 如果大部分业务场景都是写请求，此时增加只读节点不会对性能起到提升作用，需要对集群规格进行扩容变更，例如：由4C规格变更为8C规格。具体操作请参见[手动变配](#)。

#### 非预期内CPU使用率增长

导致CPU使用率非预期增长的情况比较复杂，本文主要针对比较常见的几个现象：慢查询、活跃线程高、内核配置不合理、系统BUG进行说明。



## ● 慢查询

正常情况下，CPU使用率的增长，一般是由于SQL语句不合理，产生了慢查询，同时活跃线程堆积导致CPU使用率过高。但是一定要区分清楚，是由于慢查询导致的CPU使用率高，还是由于其他资源打满查询变慢导致的CPU使用率高。

您可以在PolarDB控制台的慢SQL菜单中，查看慢查询情况，具体请参见慢SQL。

如果慢查询中有数据，就需要对慢查询进行分析。如果在慢日志明细页签中，扫描行远远大于返回行数，则说明是慢查询导致的CPU使用率过高。



❓ 说明 此处主要针对TP类查询进行分析，需要排除掉 `count` 类查询，一些AP类查询的扫描行数确实很大。

TP类查询业务的读写数据量都非常小，如果某个查询的扫描数据量非常大，那么大概率是由于索引缺失导致。例如下述查询语句在慢查询列表中显示扫描数据量为1万+，但返回数据为1条，那么很明显在 `name` 列上有索引缺失的情况。

```
select * from tbl where name='testname';
```

那么可以通过下述语句确认在 `name` 列上是否有索引。

```
show index from tbl;
```

- 如果 `name` 列上没有索引，可以通过下述语句添加索引列，消除此类大数据量扫描导致的慢查询。

```
alter table tbl add key ix_name (name);
```

- 如果 `name` 列上有索引，可以通过下述语句查看SQL语句的执行计划，确认是否使用了正确的索引。

```
explain select * from tbl where name='testname';
```

如果发现 `name` 列有索引，但没有被使用，有可能是出现了统计信息不准确导致生成了错误的执行计划。可以通过下述语句重新生成表上的统计信息用以纠正错误计划。

```
analyze table tbl;
```

执行完成后，再通过下述语句确认是否使用了正确的索引。

```
explain select * from tbl where name='testname';
```

## ● 活跃线程高

活跃线程高一定会带来CPU使用率的增长。抽象来说，MySQL实现中，每一个CPU只能在同一时间内处理一个请求。假设是16C规格的集群，最多只能同时处理16个请求。但是要注意，这里的请求指的是内核层面，而非应用的并发层面。

如果排除掉慢查询导致的请求无法正常处理，活跃线程堆积一般都是由于现网业务流量增长造成的。通过查看性能曲线，如果整体流量以及请求趋势和活跃线程的堆积趋势一致，那么说明集群资源已经达到上限，此时需要通过对数据库集群增加只读节点或者扩容集群规格来解决。

需要注意的是，在活跃线程达到临界点时，可能在CPU层面开始产生争抢，内核中会产生大量的mutex排他锁，此时性能曲线表现特征为高CPU使用率、高活跃线程、低IO或低QPS。另外一种情况是突然的业务洪峰，建立连接速度非常快，也可能在CPU层面产生大量争抢，从而导致请求堆积。此类问题一般可以通过开启集群的thread\_pool特性进行流控缓解，具体请参见[Thread Pool](#)。如果活跃线程有所缓解，同时还要注意应用侧是否已经产生了业务堆积，如果CPU负载较高同时活跃线程依然高居不下，此时则同样要考虑是不是对集群进行扩容操作。

另有一种情况是前端连接风暴导致集群流量瞬间堆积，此时流量属于异常流量，一般出现在流量数据被爬虫拉取数据的场景下。此时可以通过SQL限流的方式进行请求拒绝，具体操作请参见[会话管理](#)。

#### ● 内核配置不合理

云上RDS参数是通用场景下的标准配置，对于个别业务可能不太适用，需要进行微调。有些问题在业务上线初期数据量较少的情况下不会触发，但是随着时间的推移，业务数据量增大，在特定条件下有些问题就会暴露出来。

比较常见的问题会出现内存使用争抢。在MySQL体系中，内存主要作为数据缓存使用，也就意味着数据需要不断的迭代，最常用是 `buffer pool` 和 `innodb_adaptive_hash_index` 内存区域。整个数据库系统的缓存区域，是数据交换最为频繁的位置，如果内存不足和内存页争抢，则会出现各种异常的堆积和慢查询。最典型的表现是数据库突然CPU上涨打满，并且出现慢查询。经过排查后发现该问题并非索引缺失，这个时候就有可能是内存系统发生了问题。

例如在进行 `truncate table` 操作时，MySQL要遍历 `buffer pool` 将 `truncate` 表的数据页全部驱逐。此时大规格的集群中，如果 `innodb_buffer_pool_instances` 配置为1并且并发相对较高的情况下，就有可能出现争抢问题。这个问题在业务上线初期就可以发现，一般来说可以将 `innodb_buffer_pool_instances` 的取值与CPU核数对齐，将 `buffer pool` 进行分桶，就可以规避此类问题。

另外一种场景是 `innodb_adaptive_hash_index` 出现争抢，比较明显的表现是执行下述命令时会出现大量的 `hash0hash.cc` 等待。

```
show engine innodb status;
```

在AHI显示段中，会出现明显的数据倾斜。如下图所示：

```
insert 0, delete mark 0, delete 0
Hash table size 25499819, node heap has 20720 buffer(s)
Hash table size 25499819, node heap has 25111 buffer(s)
Hash table size 25499819, node heap has 23884 buffer(s)
Hash table size 25499819, node heap has 16835 buffer(s)
Hash table size 25499819, node heap has 23132 buffer(s)
Hash table size 25499819, node heap has 189284 buffer(s)
Hash table size 25499819, node heap has 38864 buffer(s)
Hash table size 25499819, node heap has 49094 buffer(s)
5469.32 hash searches/s, 5282.36 non-hash searches/s
```

此类问题可以将 `innodb_adaptive_hash_index` 参数关闭，也就是直接弃用AH特性，已有数据表明在混合读写的场景下AH也有可能带来负面的性能影响，关闭后对整体业务的影响不是很大。

#### ● 系统BUG

系统BUG是相对少见的问题，例如比较早的进程死锁、表上统计信息置0导致全表扫描等。随着产品的快速迭代，系统BUG导致的CPU问题相对不多。但是由于排查问题时，涉及更多内核层面的信息，用户自行处理可能有一定的难度，建议[提交工单](#)联系阿里云技术支持进行排查。

## 5.2. PolarDB代理配置与流量异常

PolarDB集群支持读写分离方式接入业务，但在实际业务场景中，经常出现节点上流量负载不均，可能导致单节点承担大量的流量从而被拖垮，最终造成整个集群雪崩。本文主要描述PolarDB代理的配置方法以及流量不均时如何定位处理。

### 数据库代理配置说明

在RDS MySQL产品中，数据库代理需要单独购买并配置，而在PolarDB MySQL引擎中，默认即开启数据库代理功能，同时功能也比MySQL的数据库代理更为强大。具体配置请参见[配置数据库代理](#)。数据库代理的参数说明如下：

- 主地址：读写方式不可配置，直接指向主节点，只能配置连接地址，无法配置代理功能。
- 集群地址：读写方式可配置，支持标准代理配置，不可删除。
- 自定义地址：读写方式可配置，支持标准代理配置，可删除。
- 读写模式：读写模式包括只读和可读可写。只读类型的连接地址如果只使用单节点，尽量不要应用在生产环境中。

#### ② 说明

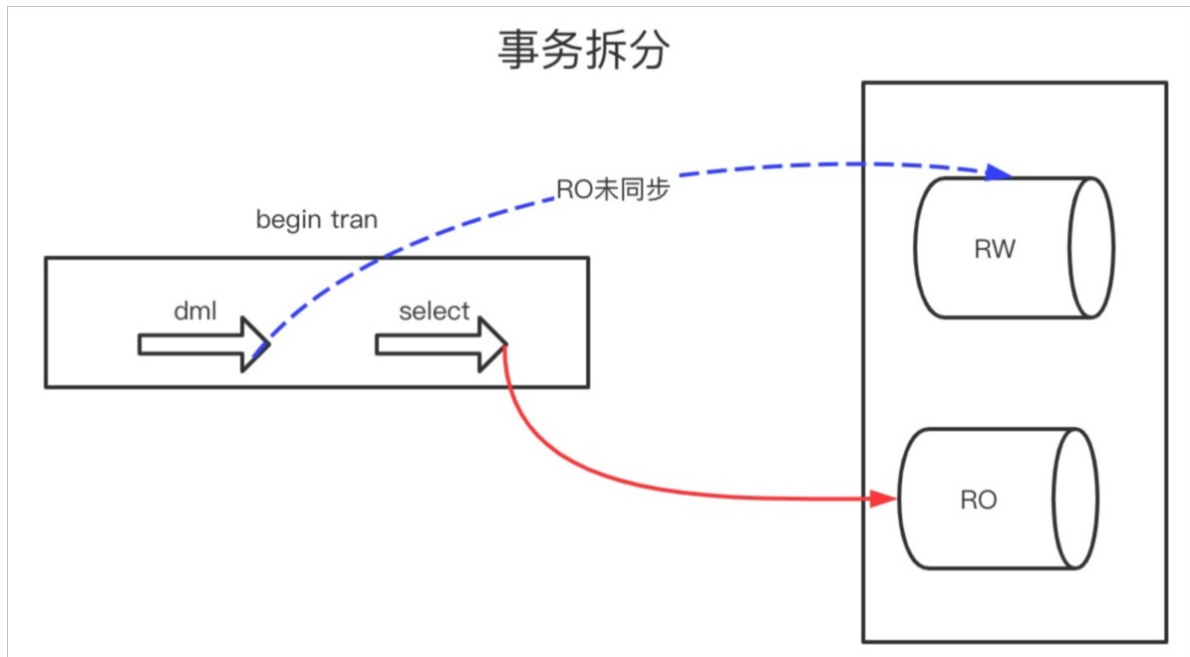
- 配置读写模式时，系统表查询会被路由到主节点，即使节点没有配置在当前连接地址上，可能对一些业务场景有影响。例如：

```
select * from information_schema.processlist;
```

- 在可读可写模式下，即使没有配置主节点，写请求会自动路由到主节点。
- 如果对一致性有要求，推荐使用会话一致性，可以满足大部分的业务场景。如果一少部分业务的一致性要求不能在同一会话中完成，可以考虑使用hint方式强制读取主节点，例如：

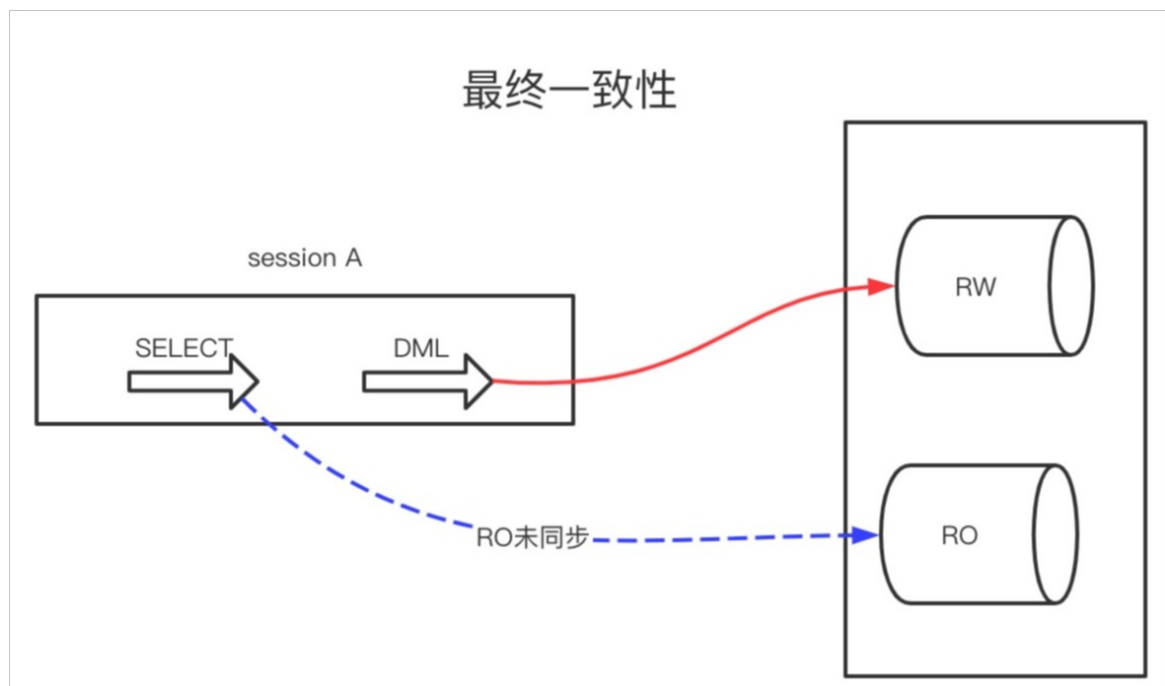
```
/*FORCE_MASTER*/ select * from information_schema.processlist;
```

- 主库是否接受读：满足一致性需求的前提下，将读请求全部分流到只读节点执行，如果不满足一致性需求（只读同步完成），流量还是会路由到主节点。
- 事务拆分：将事务头部的读取语句拆分到只读节点上以承担主节点压力。

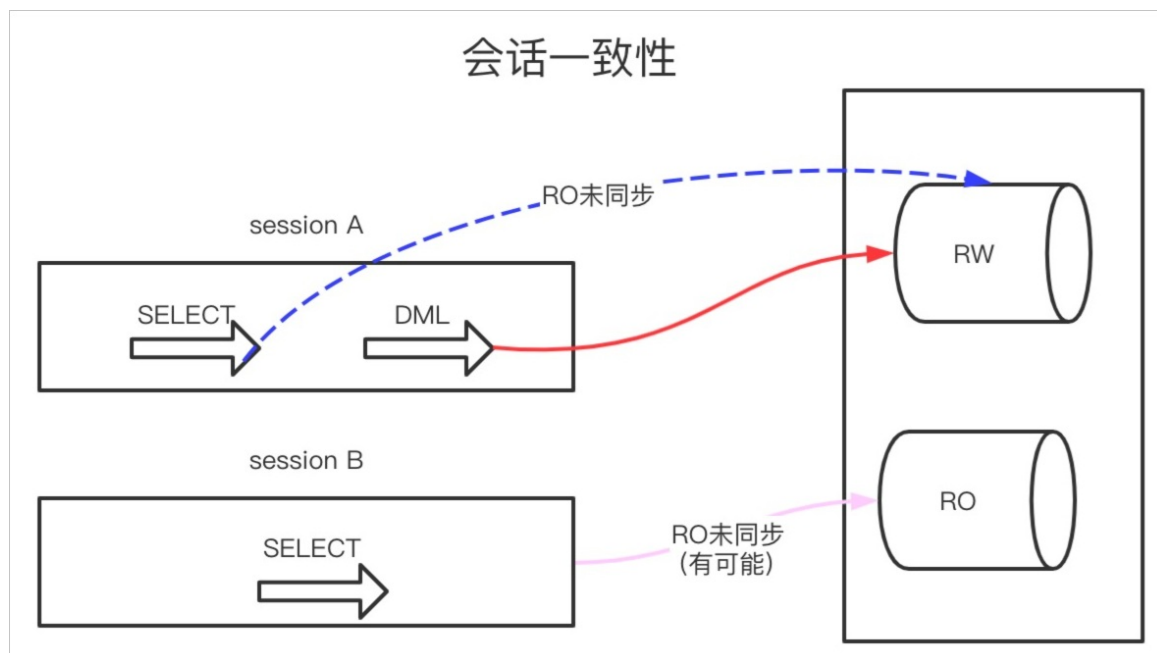


- 一致性级别

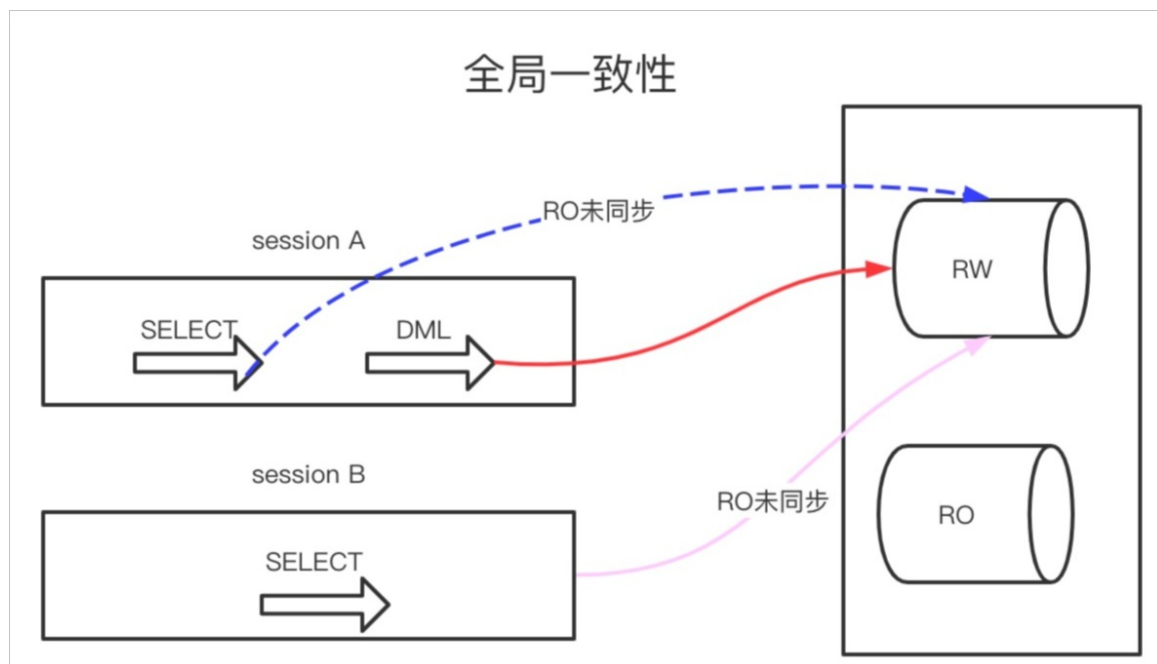
- 最终一致性：不考虑数据的同步情况，按负载进行节点请求的调度，会出现写入的数据未同步完成，只读节点上读取不到的情况。



- 会话一致性：简单理解就是指在同一个连接里的前后请求，一般在写入后立即请求数据时使用，也是PolarDB推荐的一致性配置。



- 全局一致性：每一个会话都要判断只读节点是否已经同步到最新数据，对一致性要求最高的场景下启用。



- 连接池
  - 会话级连接池：用以缓存连接信息，主要在连接风暴的场景下使用。例如PHP大量短连接的场景下，无法控制连接到集群的总连接数。
  - 事务级连接池：标准连接池方案，对长连接进行复用，类似Druid等连接池中间件，控制连接集群的总连接数。
- 并行查询：该选项仅在PolarDB MySQL引擎8.0版本集群只读类型的连接地址中可配置。如果只读类型的连接地址承接一些AP类业务流量，可以单独提供只读节点给此连接地址，并且尽量和TP业务节点不要重

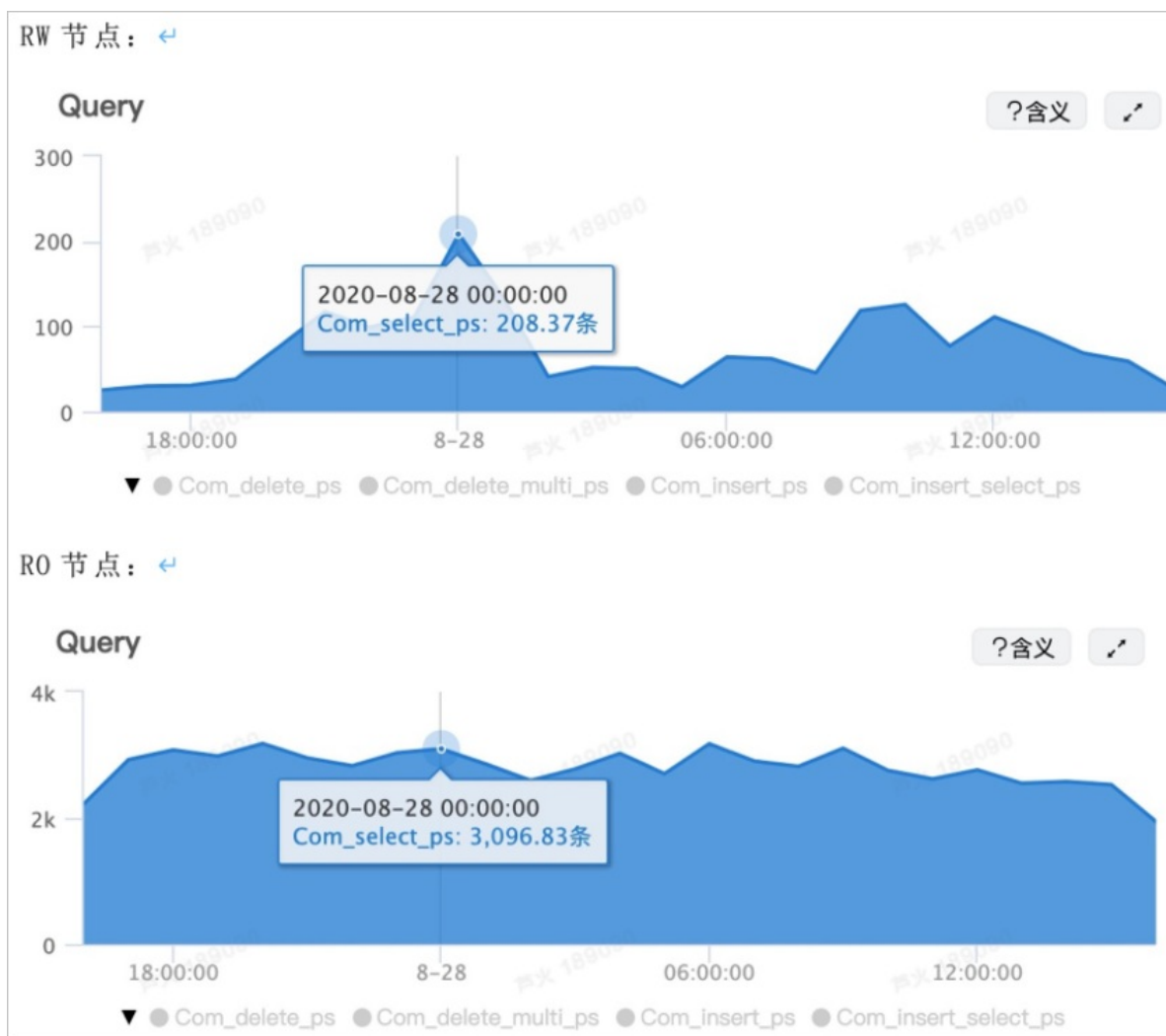
合。同时对并行查询的并行度单独调整，以充分利用CPU资源执行并行查询。

## 流量不均时的定位处理

如果在观察集群的性能时，发现集群的流量不均衡，则需要定位原因，防止流量倾斜导致的集群性能问题。具体场景如下：

- 主节点负载高，只读节点负载低
  - 场景一：业务侧直接连接了主地址导致读请求没有被分流。特别是在RDS一键迁移到PolarDB场景的最终切换过程中，默认原RDS地址会切换到PolarDB集群的主地址上，这种场景下会导致流量全部流转到主节点。此类问题需要业务侧调整连接地址为集群地址或自定义地址。
  - 场景二：使用的是读写分离地址，但业务上有大量的写请求，在代理层会把所有的写请求路由到主节点。此类场景下可以尝试开启主库是否接受读和事务拆分功能，尽可能让只读节点承担一些读流量。
- 只读节点负载高，主节点负载低

此场景一般是预期内的只读节点承载流量，例如开启了主库是否接受读和事务拆分功能，同时大量的请求都是读请求时，此场景是比较理想的PolarDB流量分配方案。即使只读节点资源不够，也可以通过快速添加新的只读节点来分担负载。监控曲线如下图所示：



- 连接地址的复用



由于PolarDB可以自定义多个连接地址，不同的连接地址数据库节点也可以复用，就有可能造成流量交叉导致集群负载不均衡。例如TP业务和AP业务有不同的连接地址，但配置了相同的只读节点，如果出现交叉使用的节点负载异常，很有可能就是由于不同业务之间的资源争抢造成的。可以尝试将问题节点从某个连接地址中去掉，再对比负载情况。

- 集群性能异常

在某些场景下，由于集群自身问题导致资源负载高，例如不同节点分配的流量会导致AH的分配也不相同，从而导致某个节点上出现资源争抢问题，此时就要定位具体原因以优化节点性能。另外数据库代理层是按负载情况分配的，如果单节点负载高，流量也会自动向负载低的节点倾斜，可能导致集群性能整体出现问题。

- 异常流量来源

目前PolarDB还未完全接入DAS的历史数据诊断功能，如果需要定位异常流量来源，只能观察当前连接请求的情况。您可以在控制台的一键诊断 > 会话管理菜单中，查看用Hosts维度统计的会话，以确认是否有非预期的业务来源访问。同时可将正常节点与异常节点做对比以定位异常流量来源。

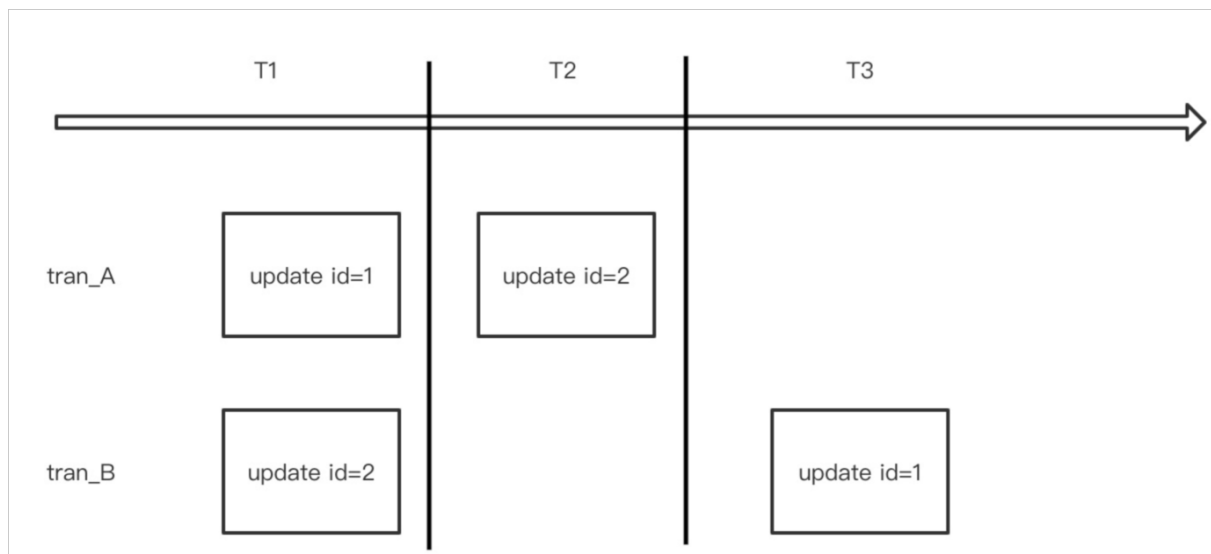


## 5.3. PolarDB死锁

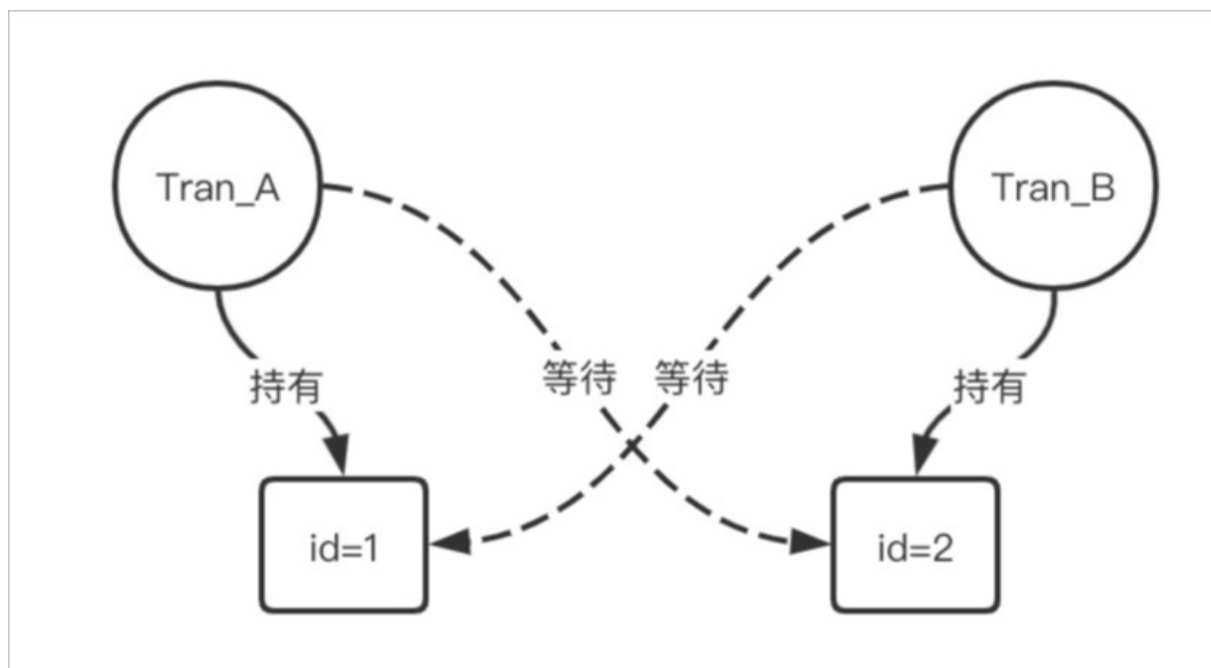
本文主要介绍了利用DAS的锁分析功能与SQL洞察功能进行死锁定位的方法。

### 背景信息

死锁是关系型数据库系统中最为常见的错误，出现在不同事务中同时对某些数据访问加锁时，都要等待对方请求中的数据而无法获取锁。数据库系统会自动牺牲回滚代价最小的事务，从而导致对应的写请求失败。更严重的情况是在大量死锁发生时，会导致数据库系统效率低下，大量进程堆积进而引发性能问题。正常情况下，死锁都是由于逻辑加锁的顺序导致的，也就是我们常说的ABA死锁。如下图所示：



上图中tran\_A与tran\_B两个请求分别持有对方所需要的第二次update的行锁，就形成了死锁。如下图所示：



此时您的业务系统会收到如下报错信息：

```
Error : Deadlock found when trying to get lock; try restarting transaction
```

说明 本文所说的死锁是指deadlock，而非事务锁造成的阻塞（block）。

## 死锁定位



您可以登录PolarDB控制台，在诊断与优化 > 一键诊断菜单中选择锁分析，单击立即诊断。如果集群存在死锁，会在发现死锁列出现是。



目前诊断功能只能拉取最后一次死锁，同样是从innodb status中获取的。如果集群不重启，死锁信息会一直保留最后一组日志，所以需要确认诊断后的日志是不是存量死锁问题，也就是说发现的死锁不一定是新出现的死锁。

发现死锁后，单击查看详情，会显示格式化后的死锁信息。如下：

发生时间: 2020年12月1日 18:57:14		
	事务 1(已回滚)	事务 2
Session ID	136738	136837
Thread id	1622	1746
请求类型	updating	updating
事务ID	85480808	85480807
涉及表	`sbtest`.`sbtest1`	`sbtest`.`sbtest1`
等待锁	index PRIMARY of table `sbtest`.`sbtest1` trx id 385480808 lock_mode X locks rec but not gap waiting	index PRIMARY of table `sbtest`.`sbtest1` trx id 385480807 lock_mode X locks rec but not gap waiting
等待锁索引名	PRIMARY	PRIMARY
等待锁类型	X locks rec but not gap waiting	X locks rec but not gap waiting
持有锁		index PRIMARY of table `sbtest`.`sbtest1` trx id 385480807 lock_mode X locks rec but not gap
持有锁索引名		PRIMARY
持有锁类型	X locks rec but not gap waiting	X locks rec but not gap waiting
事务SQL	update sbtest1 set c='tran1_tran2' where id=2	update sbtest1 set c='tran2_tran1' where id=1

- Thread id：线程ID，和SQL洞察中的线程ID对应。
- 涉及表：死锁出现的表。有时可能左右表不一致，是因为事务中请求的表不一致。
- 等待锁索引名：DML语句会将锁加在索引行上，所以获取不到的锁一定是在某个索引上。
- 事务SQL：引发死锁的SQL语句。

事务流定位

事务流定位的前提条件是在死锁发生前，PolarDB控制台已经开启了SQL洞察功能，才能对执行过的语句进行定位。

通过死锁定位，可以获取到回滚的事务、发生死锁的语句、thread\_id。同时也可以发现死锁的牺牲事务 thread\_id为1622，成功事务thread id为1746。登录PolarDB控制台，在日志与审计 > SQL洞察菜单中的搜索页面先对牺牲事务进行查询定位：

时间范围

2020年12月1日 18:56:26 - 2020年12月1日 18:58:26

自定义

关键字

可多字段组合查询，字段间以空格分隔

用户

可组合查询，如：user1 user2 user3

数据库

可组合查询，如：DB1 DB2 DB3

操作类型

☐ SELECT ☐ INSERT ☐ UPDATE ☐ DELETE ☐ LOGIN ☐ LOGOUT ☐ MERGE ☐ ALTER ☐ CREATEINDEX ☐ DROPINDEX ☐ CREATE ☐ DROP ☐ SET ☐ DESC ☐ REPLACE ☐ CALL ☐ BEGIN ☐ DESCRIBE ☐ ROLLBACK ☐ FLUSH ☐ USE ☐ SHOW ☐ START ☐ COMMIT

客户端IP

可组合查询，如：IP1 IP2 IP3

线程ID

1622

执行状态

☐ 成功 ☐ 失败

执行耗时

扫描记录数

在日志列表中状态列显示为 失败 (1213) ， error 1213 就是死锁回滚的编码，可以用来定位发生回滚的事务。

SQL语句	数据库	线程ID	用户	客户端IP	操作	状态	VIP	节点名	耗时(ms)	执行时间	执行时间 (毫秒)	更新行数	扫描行数
update sbtest1 set c='tran2' where id=2	sbtest	1622	h...	10...	UPDATE	失败 (1213)	null	pi-2...	4,256.70	2020-12-01 18:57:10	2020-12-01 18:57:10.499	0	0
update sbtest1 set c='tran1' where id=1	sbtest	1622	h...	10...	UPDATE	成功	null	pi-2...	0.25	2020-12-01 18:57:05	2020-12-01 18:57:05.400	0	1
begin	sbtest	1622	h...	106...	BEGIN	成功	null	pi-2...	0.06	2020-12-01 18:57:05	2020-12-01 18:57:05.396	0	0

日志列表默认按照秒级排序，如果要获取时序的事务流，需要通过执行时间(毫秒)进行排序。

说明 日志列表中的SQL语句太多，将无法通过执行时间(毫秒)进行排序。需要继续缩小搜索的时间范围，减小返回的SQL数据。

然后再对执行成功的事务thread id为1746进行查询定位：

SQL语句	数据库	线程ID	用户	客户端IP	操作	状态	VIP	节点名	耗时(ms)	执行时间	执行时间 (毫秒)	更新行数	扫描行数
update sbtest1 set c='tran2' where id=1	sbtest	1746	h...	10...	UPDATE	成功	null	pi-2...	0.46	2020-12-01 18:57:14	2020-12-01 18:57:14.755	1	1
update sbtest1 set c='tran2' where id=2	sbtest	1746	h...	106...	UPDATE	成功	null	pi-2...	0.39	2020-12-01 18:57:00	2020-12-01 18:57:00.519	1	1
begin	sbtest	1746	h...	10...	BEGIN	成功	null	pi-2...	0.11	2020-12-01 18:57:00	2020-12-01 18:57:00.515	0	0

通过分析日志可获取事务时间线：

2020-12-01 18:57:00.515 begin

2020-12-01 18:57:00.519 update id=2

2020-12-01 18:57:05.396 begin

2020-12-01 18:57:05.400 update id=1

2020-12-01 18:57:10.499 update id=2

2020-12-01 18:57:14.755 update id=1

说明 如果业务系统没有开启事务，则可能是在框架中配置的。正常情况下开始语句都是 set autocommit=0 ，有begin开始事务的场景比较少。

至此死锁的事务流已经分析出来了，可以交由产品研发人员进行定位了。