云数据库 MySQL

实践教程

产品文档

腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2024 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

文档目录

实践教程
- 云数据库 MySQL 使用规范
- 应用程序配置自动重连功能
- MySQL 主实例参数修改的影响
- MyISAM 自动转换为 InnoDB 引擎限制
- 为云数据库 MySQL 创建 VPC
- 使用云数据库 MySQL 提高业务负载能力
- 两地三中心灾备建设
- 读写分离扩展云数据库 MySQL 性能
- 使用 DTS 将 InnoDB 数据迁移至 RocksDB
- 构建 LAMP 堆栈 Web 应用程序
- 构建 Drupal 网站
- 构建全场景高可用架构
- 通过 Python 语言使用 MySQL API
  - 实例购买
  - 实例管理
  - 备份任务

版权所有：腾讯云计算（北京）有限责任公司

实践教程
云数据库 MySQL 使用规范

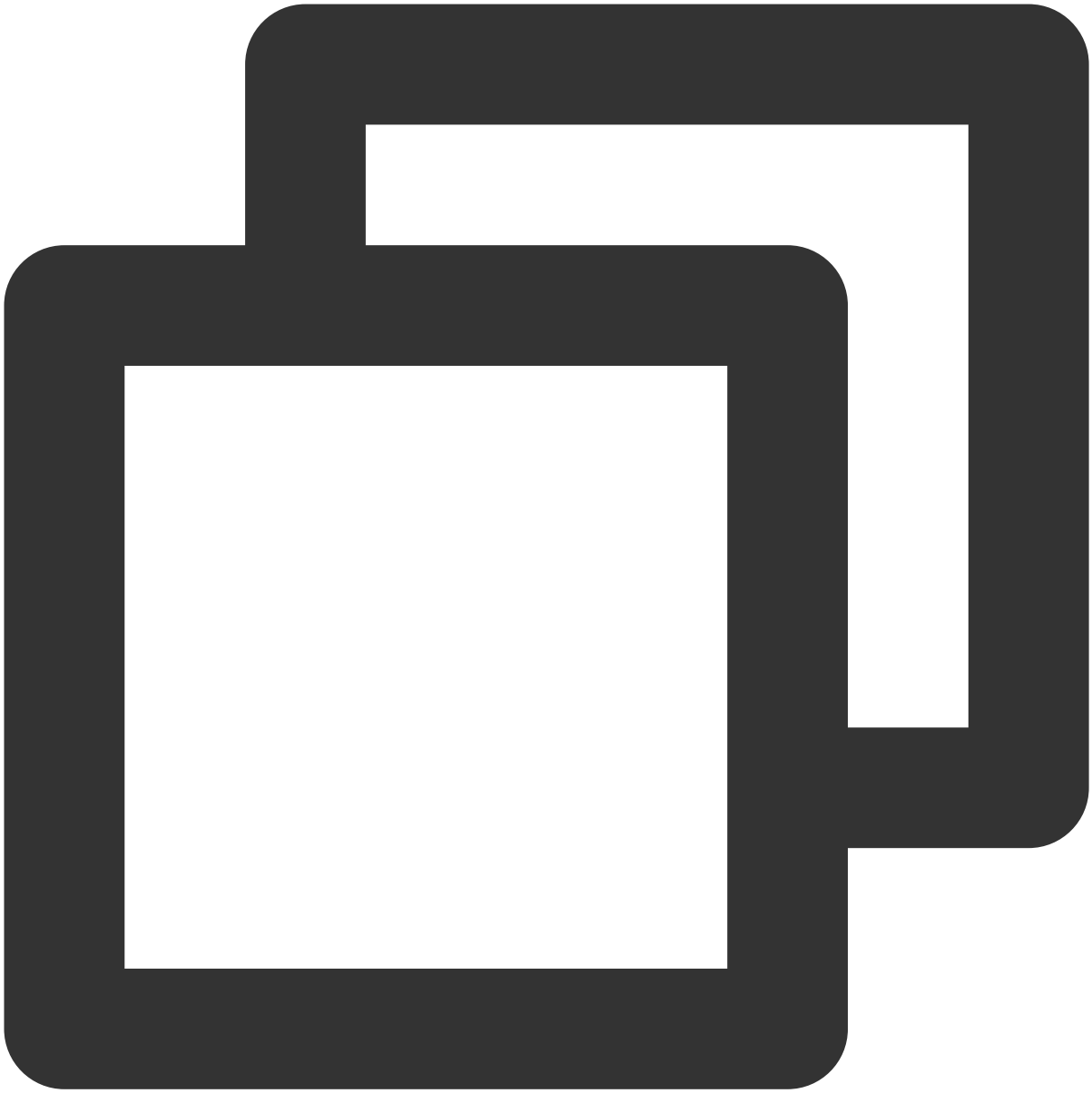
最近更新时间：2024-03-20 15:38:16

目的

规范化对云数据库 MySQL 的管理和维护，避免操作不当对云数据库 MySQL 造成不可用等影响。
指导数据库开发人员合理编写 SQL，发挥云数据库 MySQL 最优性能。

权限管理规范

考虑到云数据库 MySQL 的稳定性和安全性，云数据库 MySQL 限制了 super、shutdown、file 权限，有时在云数据库 MySQL 上执行 set 语句时，会出现如下的报错：



```
#1227-Access denied;you need(at least one of)the SUPER privilege (s) for this opera
```

解决方法：如果需要 set 修改相关参数，可以使用控制台实例管理页的**数据库管理>参数设置**功能完成。

按需授权，一般应用程序只授权 DML（SELECT、UPDATE、INSERT、DELETE）权限即可。

授权对象最小化原则，一般的应用程序访问用户按库级别来授权。

授权用户访问时只允许特定 IP 或 IP 段访问，可以在控制台配置安全组来做限制，安全组的设置请一定按照控制台提示的标准来操作，如果是公网访问设置安全组的场景，请一定放通所有涉及到的出口 IP。

管理账号与开发账号分离。

日常操作规范

注意事项

禁止使用弱密码，提升数据库实例安全性。
内网连接登录须确保 client 端的云服务器 CVM 与云数据库 MySQL 是同一账号同一地域以及同一 VPC 的机器。
控制台下载的 binlog 日志，若需要在本地解析，须确保客户端 MySQL 版本与云数据库 MySQL 实例版本一致，否则会解析出乱码，建议使用3.4或以上版本的 mysqlbinlog。
控制台上通过内网在 CVM 上下载冷备文件时，请用引号将 url 包起来，否则会出现404报错。

建议事项

尽量避免业务高峰期做 online ddl 操作，可以使用的工具请参考：`pt-online-schema-change`。
尽量避免业务高峰期批量操作数据，最好在业务低峰期分批来操作。
尽量避免一个实例跑多个业务，耦合度太高会存在业务之间互相影响的风险。
建议关闭事务自动提交，线上操作养成 `begin;` 先行的习惯，降低误操作导致数据丢失的风险，误操作亦可使用云数据库 MySQL 的回档功能（目前支持5天内任意时间点回档），若相关表不涉及跨库跨表的逻辑，可使用快速回档或者极速回档来更快恢复数据，回档新生成的库表名默认是 `原库表名_bak`。
业务有推广活动等，请提前预估资源并做好实例相关优化，如需求量比较大时请及时与对应的服务经理联系。

库表设计规范

注意事项

云数据库 MySQL 5.6 及以上版本不支持 MyISAM 引擎和 Memory 引擎，若有 Memory 引擎的需求建议使用云数据库 Redis 、Memcached；自建数据库迁移到云数据库 MySQL 时，会自动将 MyISAM 引擎转换成 InnoDB 引擎。
存在自增列的表，自增列上应该至少有一个单独的索引，或者以自增列开头的一个复合索引。
`row_format` 必须保证为非 fixed。
每张表必须要有主键，即使选不出合适的列做主键，亦必须添加一个无意义的列做主键，MySQL 第一范式标准 InnoDB 辅助索引叶子节点会保存一份主键值，推荐用自增短列作为主键，降低索引所占磁盘空间提升效率，`binlog_format` 为 row 的场景下，批量删数据没主键会导致严重的主从延迟。
字段尽量定义为 NOT NULL 并加上默认值，NULL 会给 SQL 开发带来很多问题导致走不了索引，对 NULL 计算时只能用 IS NULL 和 IS NOT NULL 来判断。

建议事项

通过业务场景分析和数据访问（包括数据库读写 QPS、 TPS、存储空间等）的预估，合理规划数据库使用资源，也可以在控制台腾讯云可观测平台界面，配置云数据库 MySQL 实例的各项监控。
建库原则就是同一类业务的表放一个库，不同业务的表尽量避免共用同一个库，尽量避免在程序中执行跨库的关联操作，此操作对后续的快速回档也会产生一定的影响。

字符集统一使用 utf8mb4 降低乱码风险，部分复杂汉字和 emoji 表情必须使用 utf8mb4 方可正常显示，修改字符集只对修改后创建的表生效，故建议新购云数据库 MySQL 初始化实例时即选择 utf8mb4。
小数字段推荐使用 decimal 类型，float 和 double 精度不够，特别是涉及金钱的业务，必须使用 decimal。
尽量避免数据库中使用 text/blob 来存储大段文本、二进制数据、图片、文件等内容，而是将这些数据保存成本地磁盘文件，数据库中只保存其索引信息。
尽量不使用外键，建议在应用层实现外键的逻辑，外键与级联更新不适合高并发场景，降低插入性能，大并发下容易产生死锁。
降低业务逻辑和数据存储的耦合度，数据库存储数据为主，业务逻辑尽量通过应用层实现，尽可能减少对存储过程、触发器、函数、event、视图等高级功能的使用，这些功能移植性、可扩展性较差，若实例中存在此类对象，建议默认不要设置 definer，避免因迁移账号和 definer 不一致导致的迁移失败。
短期内业务达不到一个比较大的量级，建议禁止使用分区表。分区表主要用作归档管理，多用于快递行业和电商行业订单表，分区表没有提升性能的作用， 除非业务中80%以上的查询走分区字段。
对读压力较大，且一致性要求较低（接受数据秒级延时）的业务场景，建议购买只读实例从库来实现读写分离策略。

索引设计规范

注意事项

禁止在更新十分频繁、区分度不高的列上建立索引，记录更新会变更 B+ 树，更新频繁的字段建立索引会极大降低数据库性能。
建复合索引时，区分度最高的列放索引的最左边，例如 `select xxx where a = x and b = x;` ，a 和 b 一起建组合索引，a 的区分度更高，则建 `idx_ab(a,b)` 。存在非等号和等号混合判断条件时，必须把等号条件的列前置，例如， `where a xxx and b = xxx` 那么即使 a 的区分度更高，也必须把 b 放在索引的最前列，因为走不到索引 a。

建议事项

单表的索引数建议不超过5个，单个索引中的字段数建议不超过5个，太多起不到过滤作用，索引也占空间，管理起来也耗资源。
选择业务中 SQL 过滤走的最多的并且 cardinality 值比较高的列建索引，业务 SQL 不走的列建索引是无意义的，字段的唯一性越高即代表 cardinality 值越高，索引过滤效果也越好，一般索引列的 cardinality 记录数小于10%我们可认为这是一个低效索引，例如性别字段。
varchar 字段上建索引时，建议指定索引长度，不要直接将整个列建索引，一般 varchar 列比较长，指定一定长度作索引已经区分度够高，没必要整列建索引，整列建索引会显得比较重，增大了索引维护的代价，可以用 count(distinct left(列名, 索引长度))/count(*) 来看索引区分度。
避免冗余索引，两个索引 (a,b) (a) 同时存在，则 (a) 属于冗余索引 redundant index，若查询过滤条件为 a 列，(a,b) 索引就够了，不用单独建 (a) 索引。

合理利用覆盖索引来降低 IO 开销，在 InnoDB 中二级索引的叶子节点只保存本身的键值和主键值，若一个 SQL 查询的不是索引列或者主键，走这个索引就会先找到对应主键然后根据主键去找需要找的列，这就是回表，这样会带来额外的 IO 开销，此时我们可以利用覆盖索引来解决这个问题，例如 `select a,b from xxx where a = xxx` ，若 a 不是主键，这时候我们可以创建 a,b 两个列的复合索引，这样就不会回表。

SQL 编写规范

注意事项

UPDATE、DELETE 操作不使用 LIMIT，必须走 WHERE 精准匹配，LIMIT 是随机的，此类操作会导致数据出错。
禁止使用 `INSERT INTO t_xxx VALUES(xxx)` ，必须显式指定插入的列属性，避免表结构变动导致数据出错。
SQL 语句中最常见的导致索引失效的情况需注意：
隐式类型转换，如索引 a 的类型是 varchar，SQL 语句写成 where a = 1; varchar 变成了int。
对索引列进行数学计算和函数等操作，例如，使用函数对日期列进行格式化处理。
join 列字符集不统一。
多列排序顺序不一致问题，如索引是 (a,b)，SQL 语句是 order by a ASC b DESC。
模糊查询使用的时候对于字符型 `xxx%` 形式可以走到一些索引，其他情况都走不到索引。
使用了负方向查询（not，!=，not in 等）。

建议事项

按需索取，拒绝 `select *` ，规避以下问题：
无法索引覆盖，回表操作，增加 I/O。
额外的内存负担，大量冷数据灌入 `innodb_buffer_pool_size` ，降低查询命中率。
额外的网络传输开销。
尽量避免使用大事务，建议大事务拆小事务，规避主从延迟。
业务代码中事务及时提交，避免产生没必要的锁等待。
少用多表 join，大表禁止 join，两张表 join 必须让小表做驱动表，join 列必须字符集一致并且都建有索引。
LIMIT 分页优化，LIMIT 80000，10这种操作是取出80010条记录，再返回后10条，数据库压力很大，推荐先确认首记录的位置再分页，例如 `SELECT * FROM test WHERE id >= ( SELECT sql_no_cache id FROM test order by id LIMIT 80000,1 ) LIMIT 10 ;` 。
避免多层子查询嵌套的 SQL 语句，MySQL 5.5 之前的查询优化器会把 in 改成 exists，会导致索引失效，若外表很大则性能会很差。
说明：
上述情况很难完全避免，推荐方案是不要将此类条件作为主要过滤条件，跟在走索引的主要过滤条件之后则问题不大。
监控上发现全表扫描的量比较大，可以在控制台参数设置 `log_queries_not_using_indexes` ，稍后下载慢日志文件分析，但不要开太久以免慢日志暴增。

业务上线之前做有必要的 SQL 审核，日常运维需定期下载慢查询日志做针对性优化。

版权所有：腾讯云计算（北京）有限责任公司

应用程序配置自动重连功能

最近更新时间：2020-12-04 16:05:47

本文为您介绍实例切换时出现秒级别的连接闪断的影响，及配置自动重连功能。

背景

云数据库 MySQL 在进行 调整数据库实例规格、升级数据库引擎版本 等操作或主实例因负载过高 hang 住、硬件故障等时，可能会涉及实例切换，实例切换时会出现秒级别的连接闪断。

若应用程序未配置自动重连功能，当发生主备切换后，会导致应用连接异常，进而影响业务的正常访问。

建议您的应用程序配置自动重连功能，并且选择实例在 维护时间 内做切换。

配置自动重连功能

避免因主备切换而影响应用连接异常，推荐您配置云数据库 MySQL 的应用程序自动重连功能，在连接池配置 connectTimeOut 和 socketTimeOut 参数。
请根据不同的业务场景设置合理的参数值，基于 OLTP（On-Line Transaction Processing）的业务场景建议统一设置为20秒。

说明：

- connectTimeOut：应用程序与数据库服务器建立 TCP 连接的超时时间，建议至少大于应用程序到数据库服务器的响应时间。
- socketTimeOut：通过 TCP 连接发送数据包后，等待响应的超时时间，建议设置为单条 SQL 最长执行时间。

MySQL 主实例参数修改的影响

最近更新时间：2020-07-02 10:10:00

对于云数据库 MySQL 实例，您可以通过 控制台 修改主实例的参数。其中对于某些重要参数而言，使用不恰当的修改方式会导致灾备实例异常或数据不一致，本文将介绍如下重要参数修改后的影响。

lower_case_table_names

默认值：0
作用：创建数据库及表时，存储与查询时是否大小写敏感。该参数可以设置的值为 0、1，默认的参数值为 0，表示创建数据库及表时，存储与查询均区分大小写，反之则不做区分。
影响：主实例修改参数后，无法同步修改灾备实例的参数，当主实例大小写敏感而灾备实例大小写不敏感时，例如主实例创建两张表，表名分别为 Test、TEst 时，当灾备实例在应用对应日志时，会导致数据同步状态异常，而错误原因为TEst 表名已存在。

auto_increment_increment

默认值：1
作用：用于自增列 AUTO_INCREMENT 的增量值，该参数可以设置的范围为 1-65535，默认值为 1。
影响：主实例修改参数后，无法同步修改灾备实例的参数，当 binlog_format 设置为 statement 时，仅记录执行语句，此时主实例修改自增列的增量值，而灾备实例未同步更改，会导致主、备实例的数据不一致。

auto_increment_offset

默认值：1
作用：用于自增列 AUTO_INCREMENT 的起始值（偏移量），该参数可以设置的范围为 1-65535，默认值为 1。
影响：主实例修改参数后，无法同步修改灾备实例的参数，当主实例修改自增列的起始值，而灾备实例未同步更改，会导致主、备实例的数据不一致。

sql_mode

默认值：NO_ENGINE_SUBSTITUTION
作用：MySQL 可以运行在不同 sql mode 模式，sql mode 模式定义了 mysql 应该支持的 sql 语法，数据校验等。该参数 5.6 版本的默认参数值为 NO_ENGINE_SUBSTITUTION，表示使用的存储引擎被禁用或未编译则抛出错误；5.7版本的默认参数值为
`ONLY_FULL_GROUP_BY`，`STRICT_TRANS_TABLES`，`NO_ZERO_IN_DATE`，`NO_ZERO_DATE`，`ERROR_FOR_DIVISION_BY_ZERO`，`NO_AUTO_CREATE_USER`，`NO_ENGINE_SUBSTITUTION`，
其中：

- `ONLY_FULL_GROUP_BY` 表示在 GROUP BY 聚合操作时，如果在 SELECT 中的列、HAVING 或者 ORDER BY 子句的列，必须是 GROUP BY 中出现或者依赖于 GROUP BY 列的函数列。
- `STRICT_TRANS_TABLES` 为启用严格模式。

- `NO_ZERO_IN_DATE` 是否允许日期中的月份和日包含 0，且受是否开启严格模式的影响。
- `NO_ZERO_DATE` 数据库不允许插入零日期，且受是否开启严格模式的影响。
- `ERROR_FOR_DIVISION_BY_ZERO` 在严格模式下，INSERT 或 UPDATE 过程中，如果数据被零除，则产生错误而非警告，而非严格模式下，数据被零除时 MySQL 返回 NULL。
- `NO_AUTO_CREATE_USER` 禁止 GRANT 创建密码为空的用户。
- `NO_ENGINE_SUBSTITUTION` 使用的存储引擎被禁用或者未编译则抛出错误。

影响：主实例修改参数后，无法同步修改灾备实例的参数，当主实例修改了 sql mode 模式，而灾备实例未同步更改，如主实例 sql mode 模式限制小于灾备实例 sql mode 模式的限制，可能会出现在主实例执行成功的 SQL 同步至灾备实例时出现报错，进而导致主、备实例数据不一致。

MyISAM 自动转换为 InnoDB 引擎限制

最近更新时间：2020-06-04 11:37:32

本文为您介绍 MyISAM 引擎自动转换为 InnoDB 引擎后，创建表时报错的解决方案。

背景

腾讯云数据库 MySQL 默认支持 InnoDB 存储引擎，并在 MySQL 5.6 及以上的版本中，不再支持 MyISAM 引擎和 Memory 引擎，详情请参见 数据库存储引擎。
当数据库迁移或升级到云数据库 MySQL 5.6 及以上版本时，系统会自动将 MyISAM 引擎转换为 InnoDB 引擎。

由于 MyISAM 引擎支持复合主键包含自增列，而 InnoDB 引擎不支持，因此 MyISAM 引擎转换为 InnoDB 引擎后，创建表时会报错，报错信息为 `ERROR 1075 (42000):Incorrect table definition;there can be only one auto column and it must be defined as a key` 。

建议您通过为自增列创建索引的方式，实现 InnoDB 引擎的复合主键包含自增列语法。

InnoDB 引擎复合主键包含自增列的修改方案

1. 原创建表报错的 SQL 语句：

```
create table t_complexkey
(
id int(8) AUTO_INCREMENT,
name varchar(19),
value varchar(10),
primary key (name,id)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

如下图创建报错：

```
MySQL [    ]> create table t_complexkey
    -> (
    ->    id int(8) AUTO_INCREMENT,
    ->    name varchar(19),
    ->    value varchar(10),
    ->    primary key (name,id)
    -> ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
ERROR 1075 (42000): Incorrect table definition; there can be only one auto column and it must be defined as a key
```

2. 修改创建索引后的 SQL 语句：

```
create table t_complexkey
(
```

```
  id int(8) AUTO_INCREMENT,
  name varchar(19),
  value varchar(10),
  primary key (name,id),
  key key_id (id) ## 为自增列创建索引
  ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

如下图创建成功：

```
MySQL [     ı]> create table t_complexkey
    -> (
    ->    id int(8) AUTO_INCREMENT,
    ->    name varchar(19),
    ->    value varchar(10),
    ->    primary key (name,id),
    ->    key key_id (id)
    -> ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
Query OK, 0 rows affected, 1 warning (0.01 sec)
```

3. 查看创建好的表结构：

```
show create table t_complexkey;
```

```
ySQL [huawu]> show create table t_complexkey;
--------------+-------------------------------------------------------------
------------------------------+
 Table        | Create Table
                                |
--------------+-------------------------------------------------------------
------------------------------+
 t_complexkey | CREATE TABLE `t_complexkey` (
 `id` int(8) NOT NULL AUTO_INCREMENT,
 `name` varchar(19) NOT NULL,
 `value` varchar(10) DEFAULT NULL,
 PRIMARY KEY (`name`,`id`),
 KEY `key_id` (`id`)
 ENGINE=InnoDB DEFAULT CHARSET=utf8 |
--------------+-------------------------------------------------------------
------------------------------+
 row in set (0.00 sec)
```

为云数据库 MySQL 创建 VPC

最近更新时间：2023-07-05 11:40:23

腾讯云提供托管云数据库的平台：腾讯云 私有网络 VPC。您可以在私有网络中启动腾讯云资源，例如腾讯云云数据库实例。

一种常见的方案是运行在同一私有网络的云数据库实例和 Web 服务器共享数据。在本教程中，针对此方案创建私有网络，并将云数据库添加进私有网络以配合使用。
本文为您介绍在同一私有网络内，添加云服务器和云数据库 MySQL，实现私有网络内云资源内网互通。

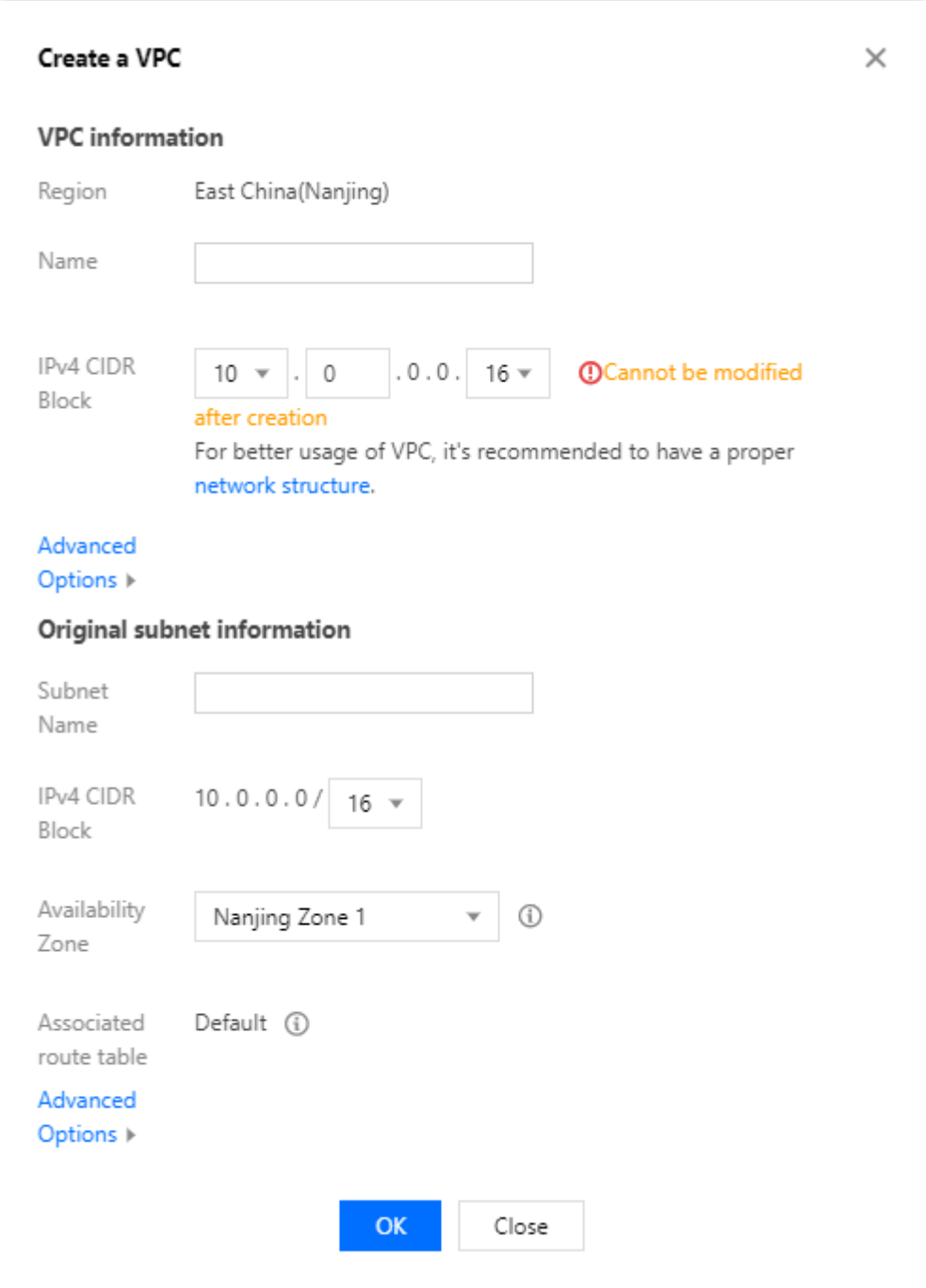
步骤1：创建私有网络

私有网络至少包含一个子网，只有在子网中才可以添加云服务资源。

1. 登录 私有网络控制台。
2. 在列表上方，选择私有网络所属的地域，单击**+新建**。
3. 填写私有网络信息和初始子网信息，单击**确定**。其中，私有网络和子网的 CIDR 创建后不可修改。

- 私有网络 CIDR 支持使用如下网段中的任意一个，如果您有不同私有网络之间内网通信的需要，两端 CIDR 的配置请不要重叠：
  - `10.0.0.0` - `10.255.255.255` （掩码范围需在11 - 28之间）
  - `172.16.0.0` - `172.31.255.255` （掩码范围需在11 - 28之间）
  - `192.168.0.0` - `192.168.255.255` （掩码范围需在16 - 28之间）
- 子网的 CIDR 必须在私有网络的 CIDR 内或相同。
  例如，私有网络的网段是 `192.168.0.0/16` ，那么该私有网络内的子网的网段可以

是 `192.168.0.0/16` 、 `192.168.0.0/17` 等。

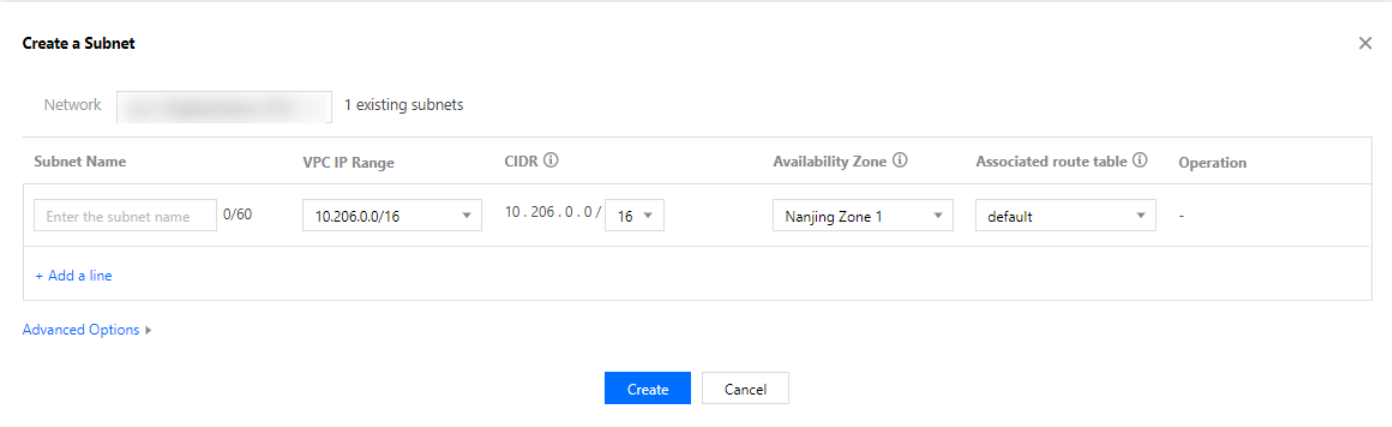


步骤2：创建子网

用户可以同时创建一个或多个子网。

1. 登录 私有网络控制台。
2. 单击左侧目录中的**子网**，进入管理页面。

3. 选择需要创建子网的地域和私有网络，单击**新建**。
4. 填写子网名称、CIDR、可用区和关联路由表。



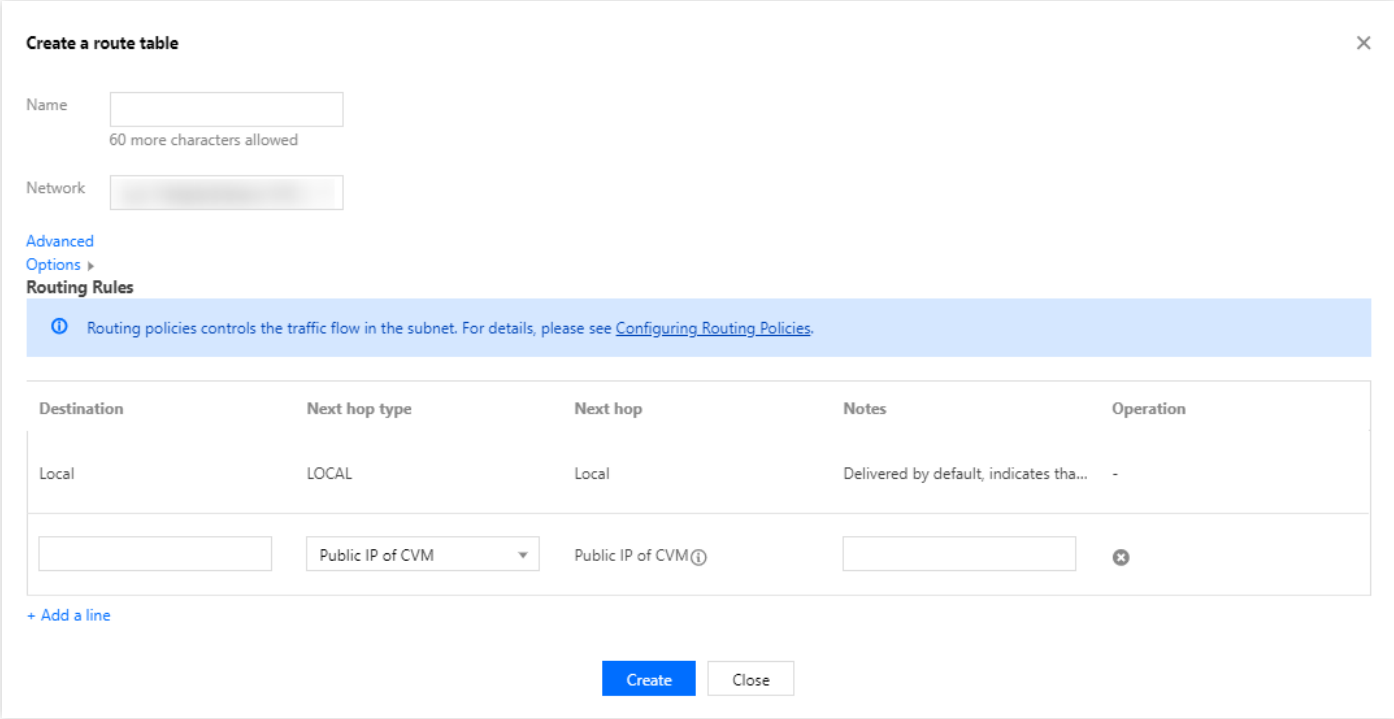
5. （可选）单击**+新增一行**，可以同时创建多个子网。
6. 单击**创建**。

步骤3：新建路由表关联子网

您可以创建自定义路由表、编辑路由策略，然后关联指定子网，子网关联的路由表用于指定该子网的出站路由。

1. 登录 私有网络控制台，在左侧栏选择**路由表**页。
2. 在列表上方选择地域和私有网络，单击**新建**。

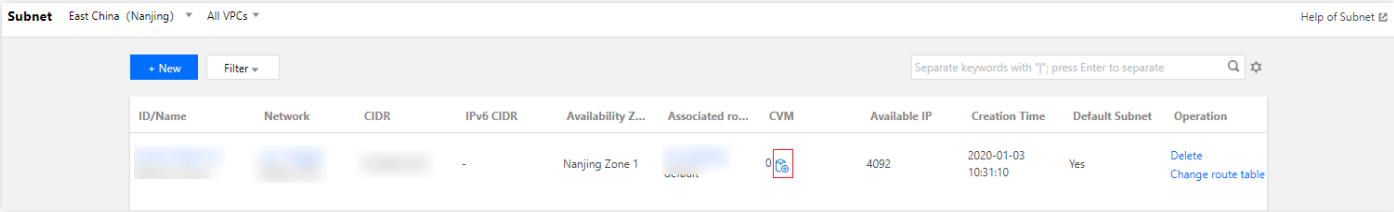
3. 在弹出的对话框中输入名称、所属网络及新建路由策略，单击**创建**。返回路由表列表即可看到您新建的路由表。



4. 在控制台左侧栏选择**子网**页，选择需要关联该路由表的子网，在**操作**列单击**更换路由表**进行关联。

步骤4：添加云服务器

1. 登录 私有网络控制台。
2. 单击左侧目录中的**子网**，进入管理页面。
3. 在需要添加云服务器的子网所在行，单击添加云服务器的图标。



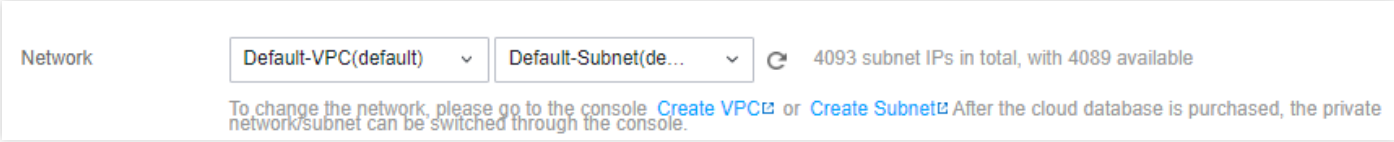
4. 根据页面提示，完成云服务器的购买即可，详情请参见云服务器文档 购买方式。

步骤5：添加云数据库

新建数据库

1. 登录 云数据库 MySQL 控制台，单击**新建**，进入购买页。

2. 在购买页的**网络**选项，选择之前创建的私有网络以及相应子网，将新购的云数据库添加进私有网络。



已有数据库

1. 在 实例列表，单击实例 ID 或**操作**列的**管理**，进入实例详情页面。
2. 在详情页的**所属网络**处，可切换对应的私有网络。

使用云数据库 MySQL 提高业务负载能力

最近更新时间：2023-03-13 11:54:01

具备优异性能和扩展能力的数据库可帮助您迅速提高原有系统的负载能力，在同等数据库规模下合理使用 MySQL 可帮助您提高数据库的并发能力，支撑更高的业务每秒访问次数。

1. 选择适合自己的数据库配置

1.1 选择数据库版本

云数据库 MySQL 目前提供完全兼容原生 MySQL 的5.5、5.6、5.7、8.0版本，建议您选择5.6或更高的版本，它们提供了更稳定的数据库内核，优化改进5.5及更老版本的设计以提升系统的性能，并提供了多项极具吸引力的新特性。

本文以 MySQL 5.7 为例介绍新版本特性。MySQL 5.7 具有被普遍认可的高性能、可靠性和易用性。它部分优化点和新特性如下：

- **原生 JSON 支持**
  MySQL 5.7 新增了一种数据类型，用来在 MySQL 的表中存储 JSON 格式的数据。原生支持 JSON 数据类型主要有如下好处：
- 文档校验：只有符合 JSON 规范数据段才能被写入类型为 JSON 的列中，所以相当于有了自动 JSON 语法校验。
- 高效访问：当您在一个 JSON 类型的列中存储 JSON 文档的时候，数据不会被视为纯文本进行存储。实际上，数据用一种优化后的二进制格式进行存储，以便可以更快速地访问其对象成员和数组元素。
- 性能提升：可以在 JSON 类型列的数据上创建索引以提升 query 性能。这种索引可以由在虚拟列上所建的“函数索引”来实现。
- 便捷：针对 JSON 类型列附加的内联语法可以非常自然地在 SQL 语句中集成文档查询。例如 features，feature 是一个 JSON 字段：

```
SELECT feature->"$.properties.STREET" AS property_street FROM features WHERE id
= 121254;
```

使用 MySQL 5.7 可以在一个工具中无缝地混合最好的关系和文档范例，在不同的应用和使用案例中应用关系型范例或文档型范例当中最适合的范例。这为 MySQL 用户极大扩大了应用范围。

- **SYS Schema**
  MySQL SYS Schema 是一个由一系列对象（视图、存储过程、存储方法、表和触发器）组成的 database schema，使存储在 Performance Schema 和 INFORMATION_SCHEMA 的各类表中的监测数据资源，可以通过方便、可读、对 DBA 和开发者的友好的方式进行访问。
  MySQL SYS Schema 默认包含在 MySQL 5.7 中，并提供摘要视图以回答诸如下面所列的常见问题：
- 谁占了数据库服务的所有资源?

- 哪些主机对数据库服务器的访问量最大?
- 实例上的内存都上哪去了?
- **InnoDB 相关改进**
- InnoDB 在线操作（Online DDL）：您可以在不重启 MySQL 的情况下，动态地调整您的 Buffer Pool size 以适应需求的改变。现在 InnoDB 也可以在线自动清空 InnoDB 的 UNDO 日志和表空间，消除了产生大共享表空间文件（ibdata1）问题的一个常见原因。最后，MySQL 5.7 支持重命名索引和修改 varchar 的大小，这两项操作在之前的版本中，都需要重建索引或表。
- InnoDB 原生分区：在 MySQL 5.7 InnoDB 中包含了对分区的原生支持。InnoDB 原生分区会降低负载，减少多达90%的内存需求。
- InnoDB 缓存预热：当 MySQL 重启时，InnoDB 自动保留您缓存池中最热的25%的数据。您再也不需要任何预加载或预热您数据缓存的工作，也不需要承担 MySQL 重启带来的性能损失。

MySQL 5.7 更多优化和新特性可参见 MySQL 官方资料。

1.2 选择实例规格（数据库内存）

当前 MySQL 并未提供单独的 CPU 选项，CPU 将根据内存规格按比例分配。您可以根据自己的业务特征购买相应的数据库规格，我们为每一种实例都做了详尽的标准化测试以为您提供选型时的性能参考。
但需要注意的是，Sysbench 标准化测试并不能代表所有的业务场景，建议您在将业务正式运行在 MySQL 之前对数据库做一次压力测试，以便于更加了解 MySQL 在您的业务场景下的性能表现。请参见 MySQL 性能说明。

内存是实例的核心指标之一，访问速度远远大于磁盘。通常情况下，内存中缓存的数据越多，数据库的响应就越快；如果内存较小，当数据超过一定量后，就会被刷新到磁盘上，如果新的请求再次访问该数据，就要从磁盘上把它从磁盘中读取进内存，消耗磁盘 IO，这个时候数据库响应就会变慢。

对于读并发较大或读延迟较为敏感的业务，建议您不要选择过小的内存规格，以保障数据库的性能。

1.3 选择硬盘

云数据库 MySQL 实例的硬盘空间包括数据文件、系统文件、binlog 文件、临时文件。在写入的数据量超出实例硬盘空间时，如未及时升级，可能会触发实例锁定。因此在选购硬盘空间时，建议您对未来一段时间内可能的数据量增长保留一定冗余，避免因硬盘容量不足引起的实例锁定或频繁升级。

1.4 选择适合您的数据复制方式

云数据库 MySQL 提供了异步、半同步、强同步三种复制方式，请参见 数据库实例复制方式，如您的业务对写入时延或数据库性能较为敏感，建议您选择异步复制方式。

1.5 云数据库的高可用

云数据库 MySQL 采用主备 M-S（主备模式）的高可用架构，其主备之间的数据同步依靠 binlog 日志的方式。同时支持将实例恢复到任何一个时间点，这个功能需要依靠运用备份和日志。因此，通常情况下您无需再搭建备份恢复系统或付出其他额外支出来保障实例的高可用。

1.6 云数据库的扩展性

云数据库 MySQL 的数据库版本、内存/硬盘规格均支持在线的动态热升级。升级过程不会中断您的业务，您无需担心业务规模增长带来的数据库瓶颈。

1.7 将 CVM 和 MySQL 配合使用

通常情况下，在购买成功后您需要将云服务器 CVM 和 MySQL 配合使用，请参见 使用 CVM 访问 MySQL。

2. 使用只读实例作为读扩展

在常见的互联网业务中，数据库读写比例通常为4:1至10:1之间。在这类业务场景下，数据库的读负载远高于写负载，在遇到性能瓶颈时一个常见的解决方案就是增加读负载。
MySQL 只读实例为您提供了读扩展解决方案，请参见 只读实例。
只读实例也可应用于不同业务的只读访问中，例如，主实例承担在线业务读写访问，只读实例为内部业务或数据分析平台提供只读查询。

3. 云数据库的灾备方案

云数据库 MySQL 提供了 灾备实例 帮助您一键搭建跨城域的异地数据库灾备。

使用灾备实例，可实现多地域之间不同机房互为冗余，当一个机房发生故障或由于不可抗因素导致无法提供服务，快速切换到另外一个机房。灾备实例使用腾讯内网专线做数据同步，且经过 MySQL 的内核级复制优化，尽可能消除灾难情况下同步延迟给业务带来的影响，在异地业务逻辑就绪的情况下，可以达到秒级别的灾备切换。

4. 两地三中心方案

使用云数据库 MySQL，仅需在页面简单配置几步即可实现两地三中心方案：

- 购买 MySQL 同城强一致性集群，选择 多可用区部署（灰度开放），提供一地两中心能力。
- 为该集群添加异地灾备节点，即可实现两地三中心架构。

5. 使用灾备实例为用户提供就近接入

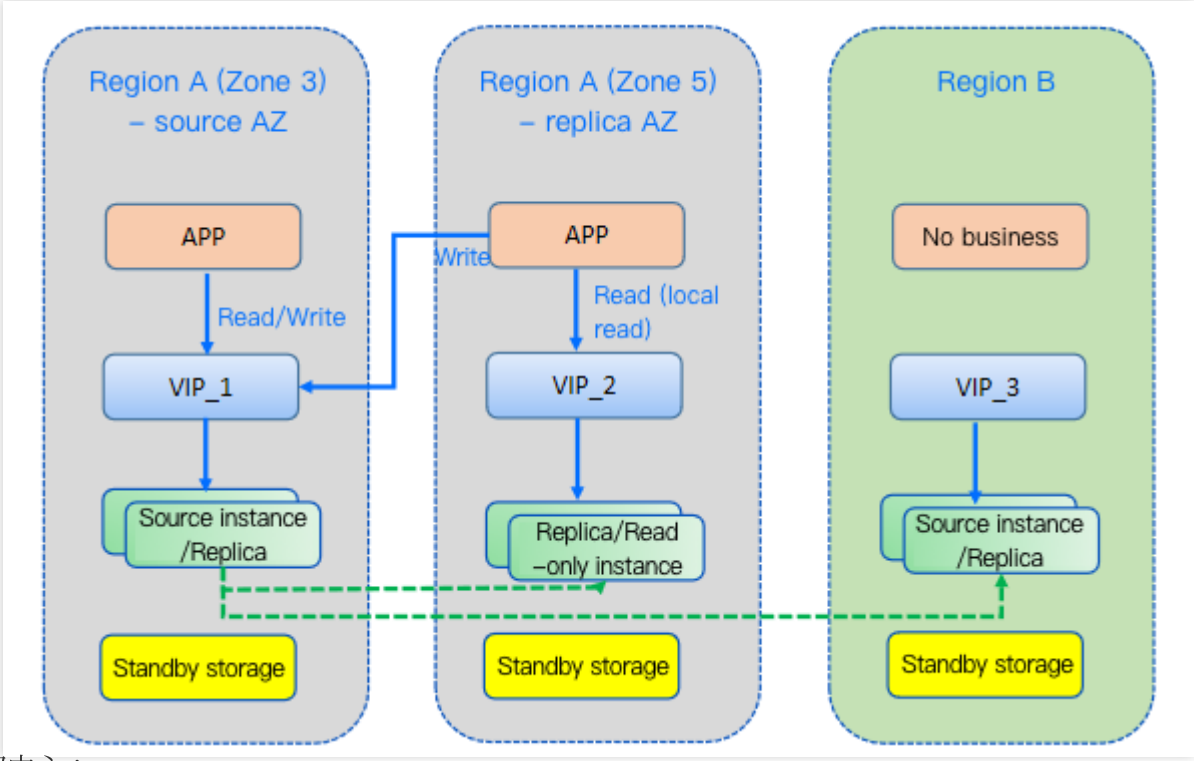
灾备实例采用独立 M-S（主备模式）的高可用架构，同时可对外提供只读的访问能力。因此，如有需要跨地域的用户就近接入业务场景，您可放心使用灾备实例。

两地三中心灾备建设

最近更新时间：2024-07-23 17:31:18

本文为您介绍通过跨可用区部署实例和异地搭建灾备实例实现两地三中心架构建设。

两地三中心部署架构



同城双中心：

地域 A 三区和地域 A 五区组成同城双中心，地域 A 三区宕机后可切换到地域 A 五区，以保护数据库。

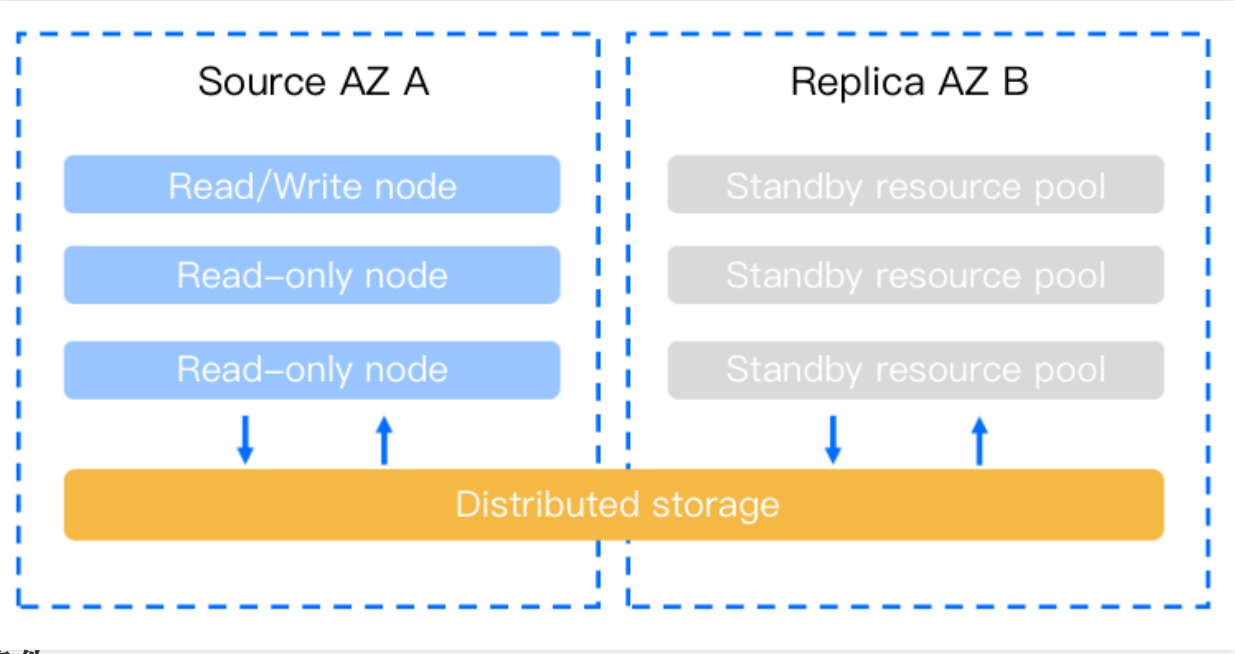
两地：

地域 A 和地域 B 组成两地，即使在地域 A 的三区、五区的机房全部宕机后，业务也可切换到地域 B 的机房。

跨可用区部署

云数据库 MySQL 支持跨可用区部署，相比单可用区，多可用区部署方式具备更高的容灾能力，可以保护数据库，以防数据库实例发生故障或可用区中断，可以抵御机房级别的故障。

云数据库 MySQL 多可用区部署为数据库实例提供高可用性和故障转移支持。多可用区是在单可用区的级别上，将同一地域的多个单可用区组合成的物理区域。



前提条件

实例状态为运行中。
实例所在的地域需要包含两个及以上的可用区。
部署的可用区拥有足够计算资源。

支持的地域和可用区

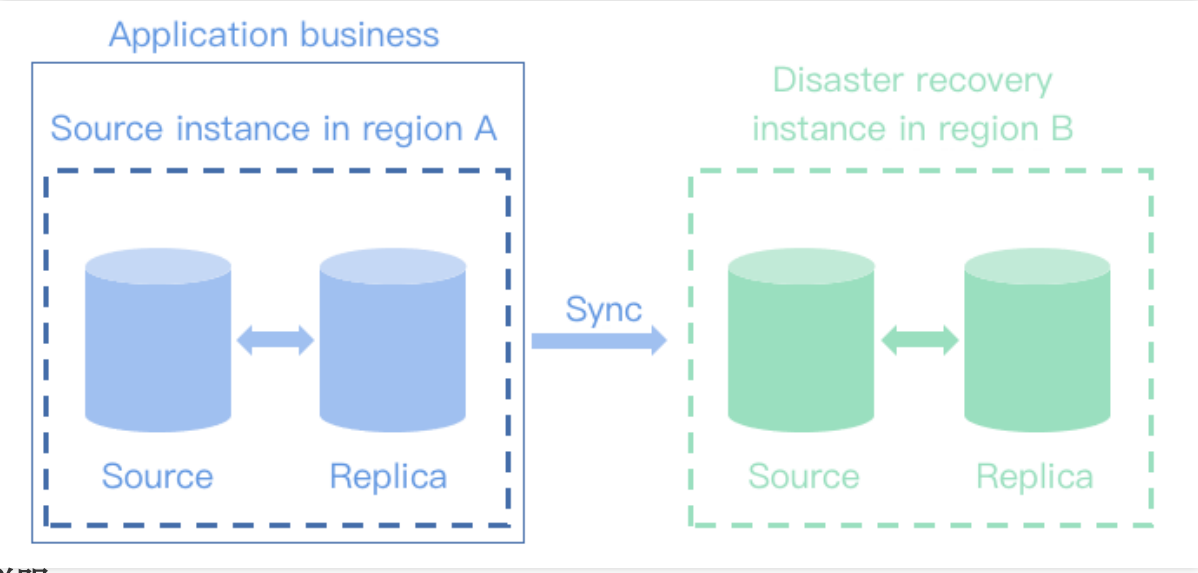
云数据库 MySQL 多可用区部署目前支持广州、上海、南京、北京、成都、中国香港、新加坡、雅加达、曼谷、孟买、首尔、东京、弗吉尼亚、法兰克福地区。对应城市支持的主备可用区可选范围以 云数据库 MySQL 购买页 的主备可用区展示为准。
此功能会逐渐扩充支持地域和可用区。
如因业务需要，您可 提交工单 申请其他地域和可用区部署。

费用说明

本功能免费。实例从单可用区迁移至多个可用区，也不收取费用。

异地灾备实例

针对业务连续服务和数据可靠性有强需求或是监管需要的场景，云数据库 MySQL 提供跨地域灾备实例，帮助用户以较低的成本提升业务连续服务的能力，同时提升数据的可靠性。
异地灾备实例提供独立的数据库连接地址，灾备实例可提供读访问能力，用于就近接入、数据分析等场景，设备冗余成本低，且使用主备高可用架构，避免了数据库的单点风险。



费用说明

灾备实例费用与主实例相同。

为云数据库 MySQL 实例建设两地三中心架构

步骤1：设置跨可用区部署

购买实例时，为实例选择跨可用区部署

1. 登录 MySQL 控制台，在实例列表，单击**新建**，进入购买页。
2. 在购买页，选择对应支持地域后，在**备可用区**选项，选择与主可用区不同区的备可用区。

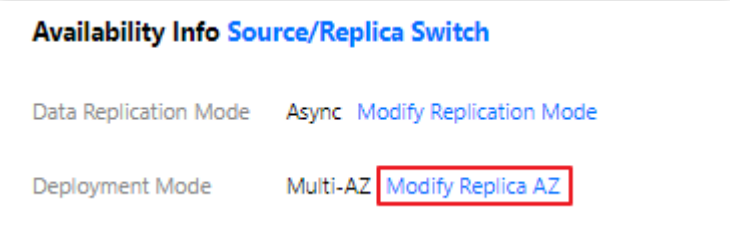
说明：

主备机处于不同可用区，可能会增加2~3ms的同步网络延迟。

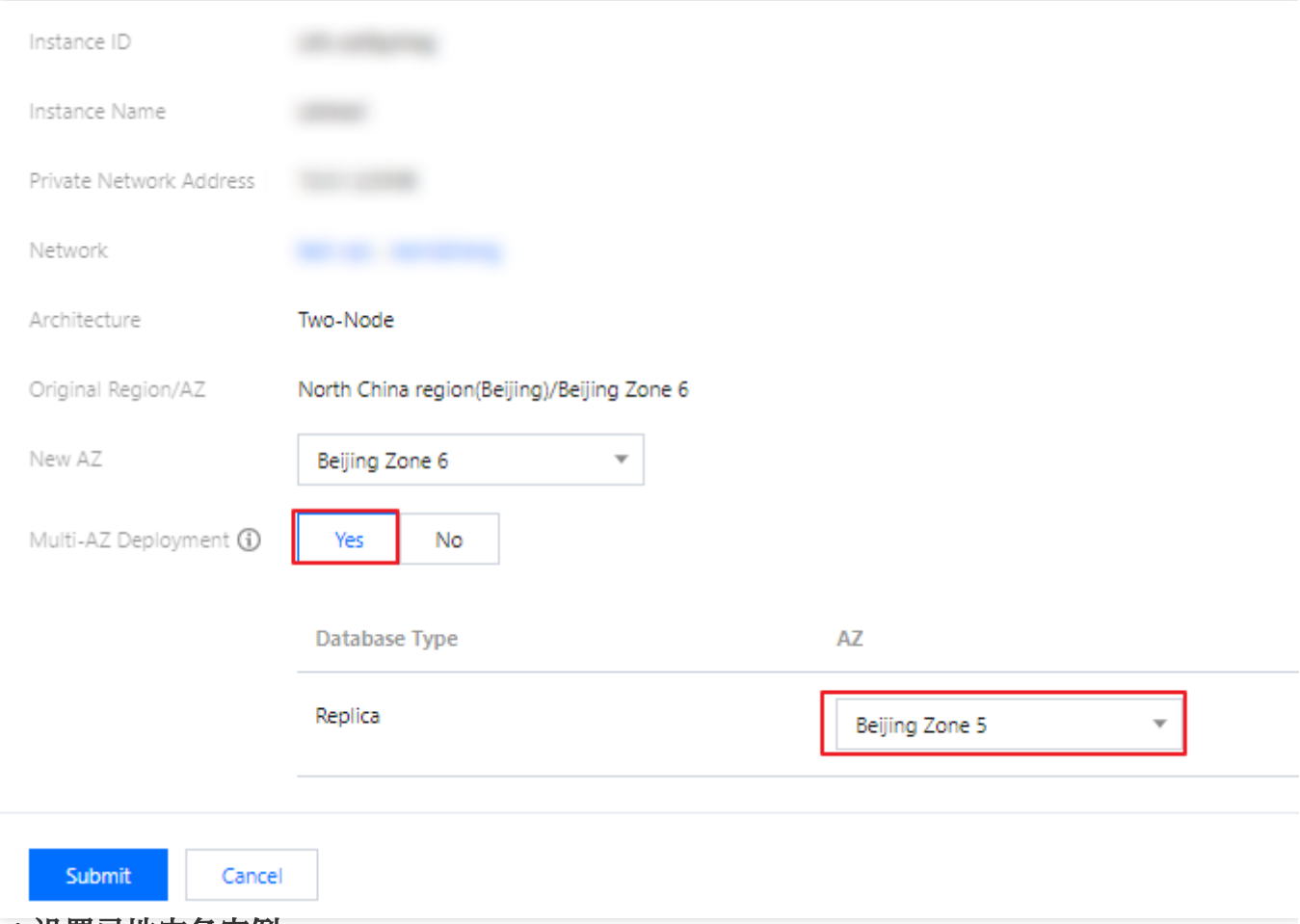
3. 确认无误后，单击**立即购买**，完成付款后，可在**实例详情页 > 可用性信息**查看该实例的主备可用区。

已创建实例，单可用区变更为多可用区

1. 登录 MySQL 控制台。
2. 在实例列表，单击需变更可用区的实例 ID 或**操作**列的**管理**，进入实例详情页。
3. 在实例详情页的**可用性信息 > 部署方式**下，单击**变更可用区**。



4. 在弹出的对话框，选择多可用区部署为**是**，选择备库可用区，单击**提交**。



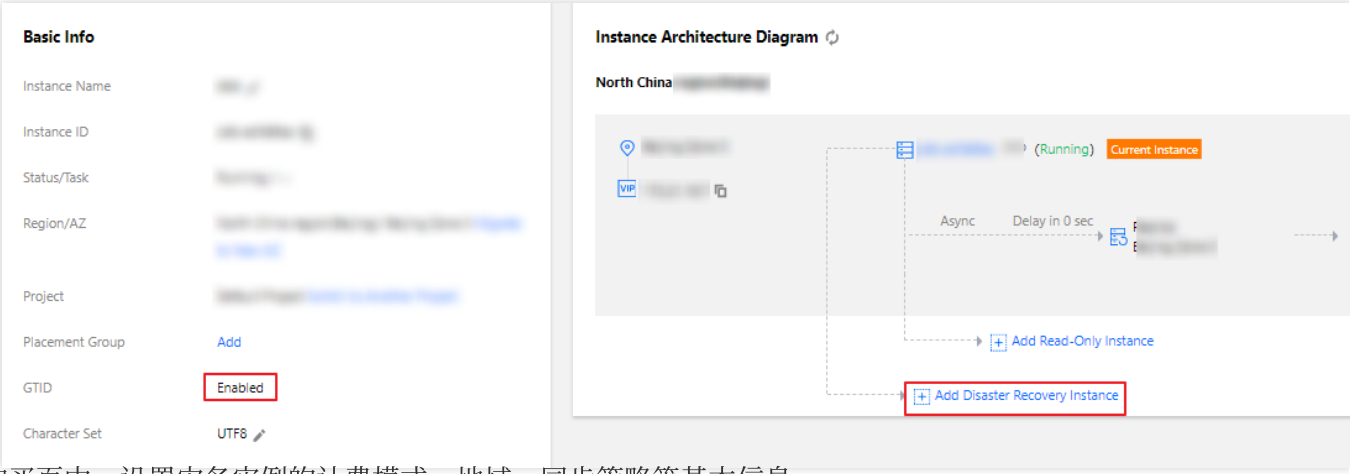
步骤2：设置异地灾备实例

说明：

仅支持1GB内存、50GB硬盘及以上规格，且 MySQL 5.6 及以上版本、InnoDB 引擎的高可用版主实例购买灾备实例，若主实例低于此规格，请先升级主实例规格。且主实例须已开启 GTID 功能。

2.1 创建灾备实例

1. 登录 MySQL 控制台，在实例列表，单击实例 ID 或**操作**列的**管理**，进入详情页。
2. 在实例详情页的基本信息中确认 GTID 功能开启，在实例架构图中单击**添加灾备实例**，进入灾备实例购买页。



3. 在购买页中，设置灾备实例的计费模式、地域、同步策略等基本信息。

同步策略为**立即同步**时，创建完灾备实例会立即同步数据。

同步策略为**创建后同步**时，灾备实例创建成功后，需对灾备同步链接进行配置，详细操作参见下文 创建同步链接。

说明：

创建时长受数据量的影响，期间主实例的控制台操作会被锁定，请妥善安排。

目前只支持整个实例数据同步，请确保磁盘空间充足。

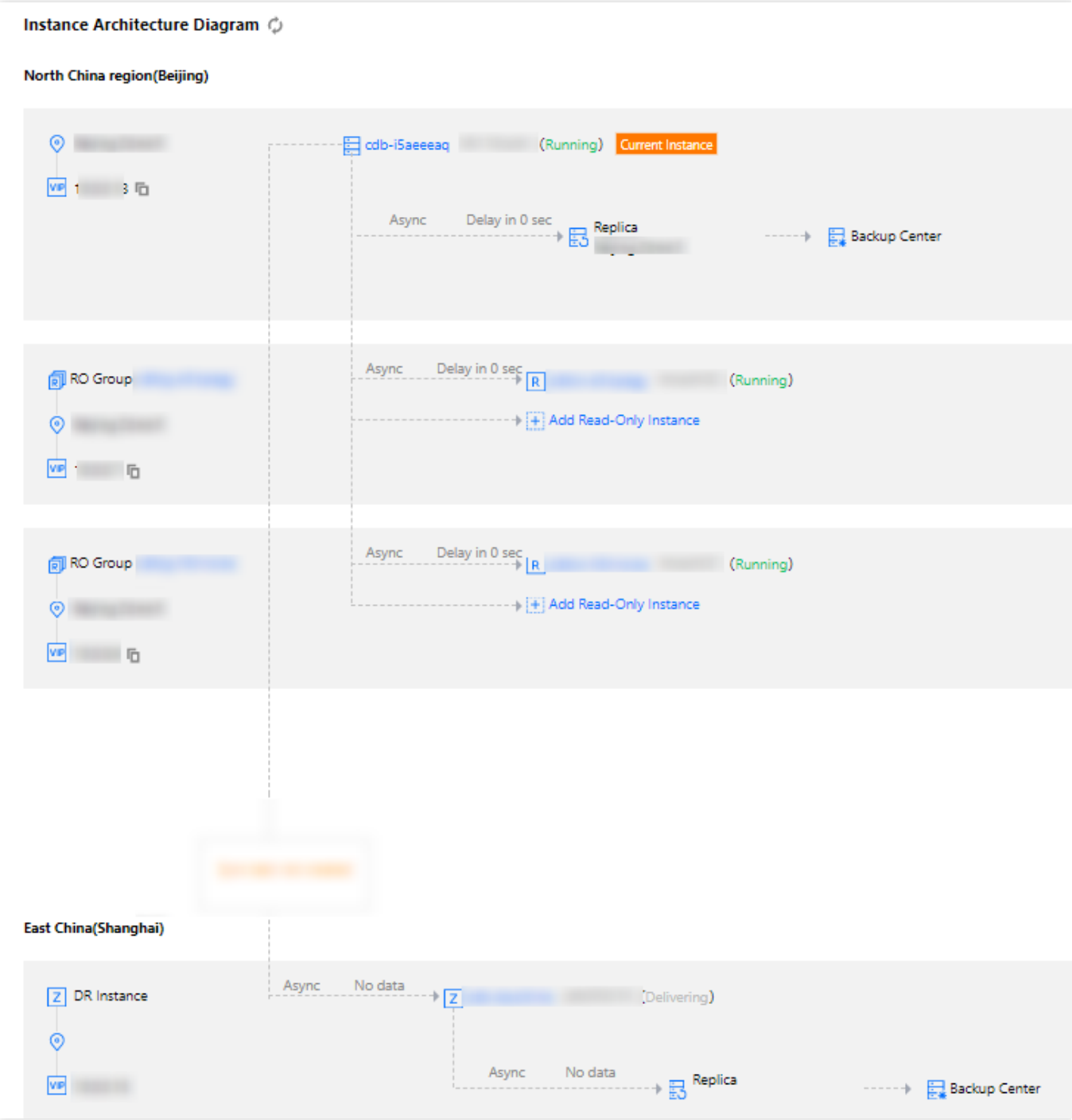
请确保主实例状态为运行中并且没有任何运行相关变更任务执行，如升降配、重启等，否则同步任务有可能失败。

4. 确认无误后，单击**立即购买**，待灾备实例发货。

5. 返回实例列表，待实例状态变为**运行中**，即可进行后续操作。

查看两地三中心架构

配置完成后数据库部署方式呈现如下两地三中心的架构：

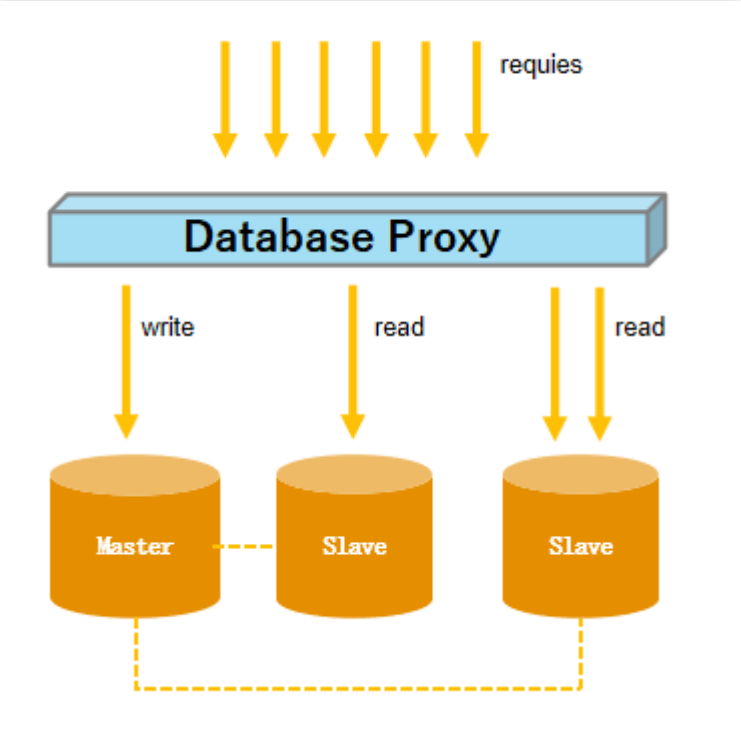

Instance Architecture Diagram
North China region(Beijing)
cdb-i5aeeeaq
(Running)
Current Instance
Async
Delay in 0 sec
Replica
Backup Center
RO Group
Add Read-Only Instance
East China(Shanghai)
DR Instance
No data
(Delivering)

读写分离扩展云数据库 MySQL 性能

最近更新时间：2022-06-13 15:37:16

本文为您介绍通过数据库代理开启读写分离功能，实现横向扩展，提升云数据库 MySQL 的性能。

数据库代理实现读写分离架构



数据库代理

数据库代理是位于云数据库服务和应用服务之间的网络代理服务，用于代理应用服务访问数据库时的所有请求。数据库代理访问地址独立于原有的数据库访问地址，通过数据库代理地址的请求，全部通过代理集群中转访问数据库的主从节点，进行读写分离，将读请求转发至只读实例，降低主库的负载。

自动读写分离

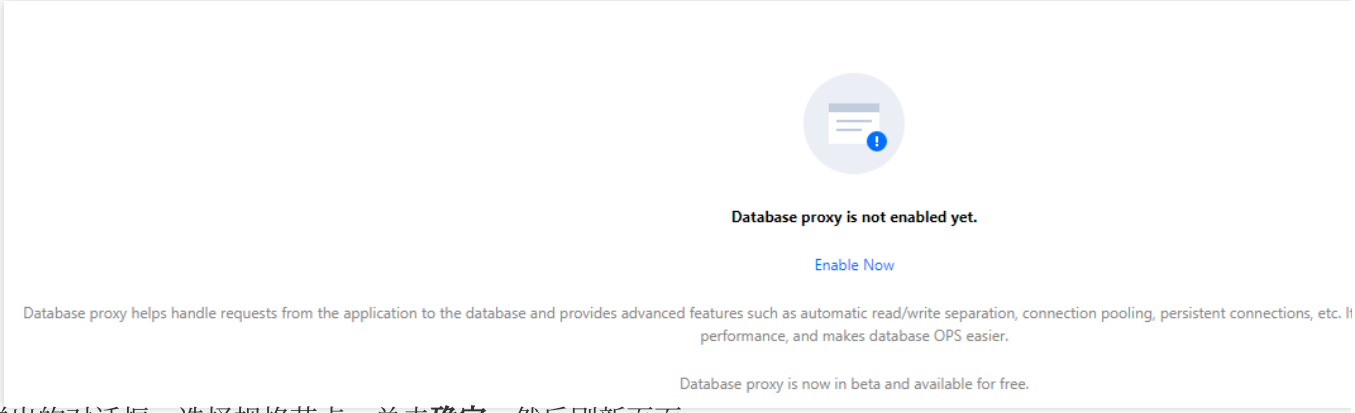
用户业务场景中存在读多写少、业务负载无法预测等场景，有大量读请求的应用场景下，单个实例可能无法承受读取压力，甚至对业务产生影响。
为了实现读取能力的弹性扩展，分担数据库压力，可以创建一个或多个只读实例，利用只读实例满足大量的数据库

读取需求。但此类解决方案需要业务侧支持读写分离改造，其代码的健壮性决定了业务读写分离的质量，对客户的技术要求较高，而且灵活性和可扩展性较差。
故创建只读实例后，可通过购买数据库代理开启读写分离功能，在应用程序中配置数据库代理地址，就可以使写请求自动转发到主实例，读请求自动转发到各个只读实例。

通过数据库代理开启读写分离功能

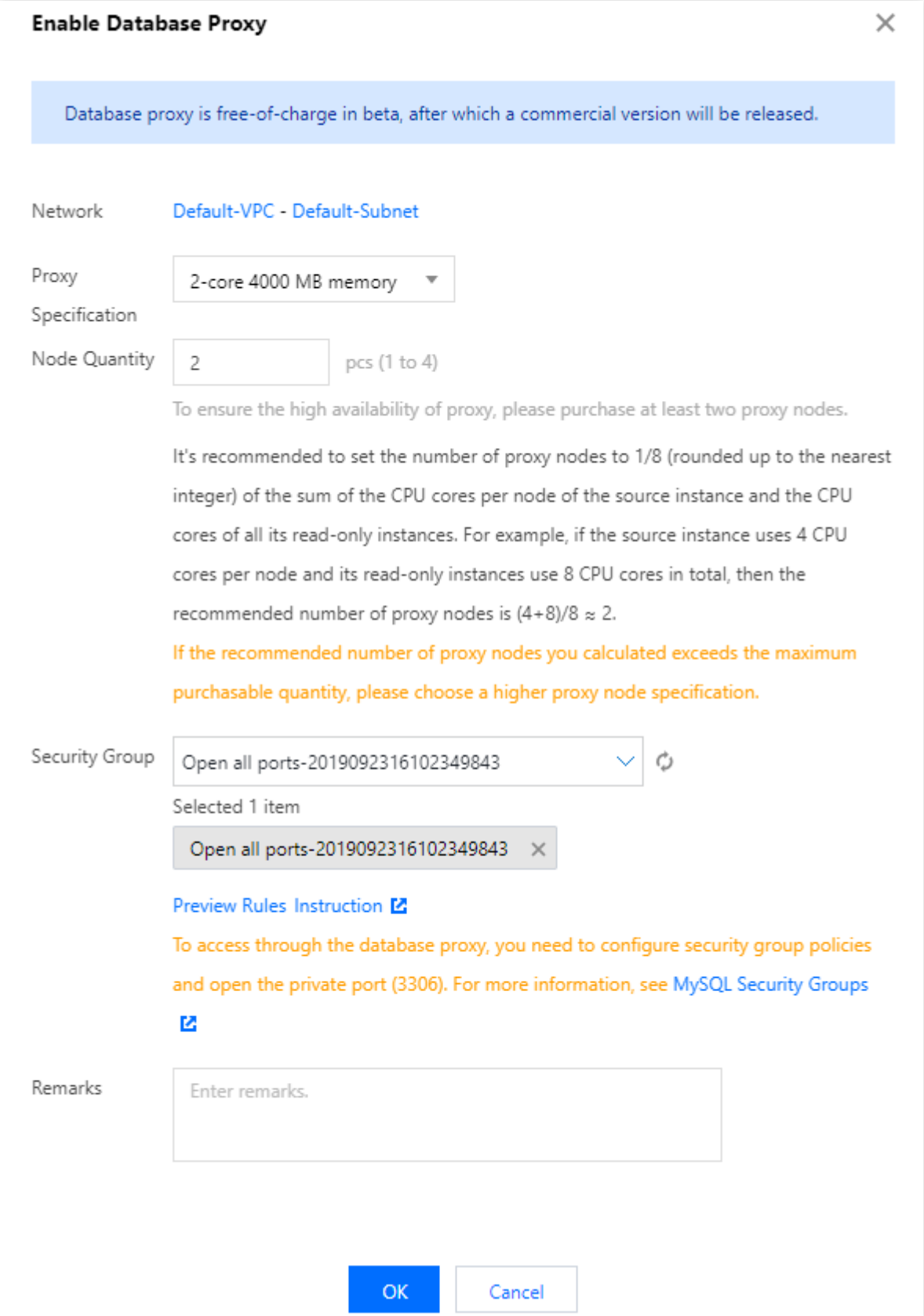
步骤1：开通数据库代理

1. 登录 MySQL 控制台，在实例列表，选择需要开启代理的主实例，单击实例 ID 或**操作**列的**管理**，进入实例管理页面。
2. 在实例管理页面，选择**数据库代理**页，单击**立即开启**。



3. 在弹出的对话框，选择规格节点，单击**确定**，然后刷新页面。

网络类型：仅支持私有网络 VPC，默认与主实例保持一致。
代理规格：2核4000MB内存、4核8000MB内存、8核16000MB内存。
节点个数：代理节点个数，推荐代理个数为：主实例和只读实例 CPU 核数之和的 1/8（向上取整），例如主实例为4核 CPU，只读实例为8核 CPU，则推荐代理数量 = (4 + 8) / 8 ≈ 2
连接池状态：连接池能有效解决短连接业务频繁建立新连接导致实例负载过高的问题。
安全组：是重要的网络安全隔离手段，可根据需要选择已有安全组或新建安全组。

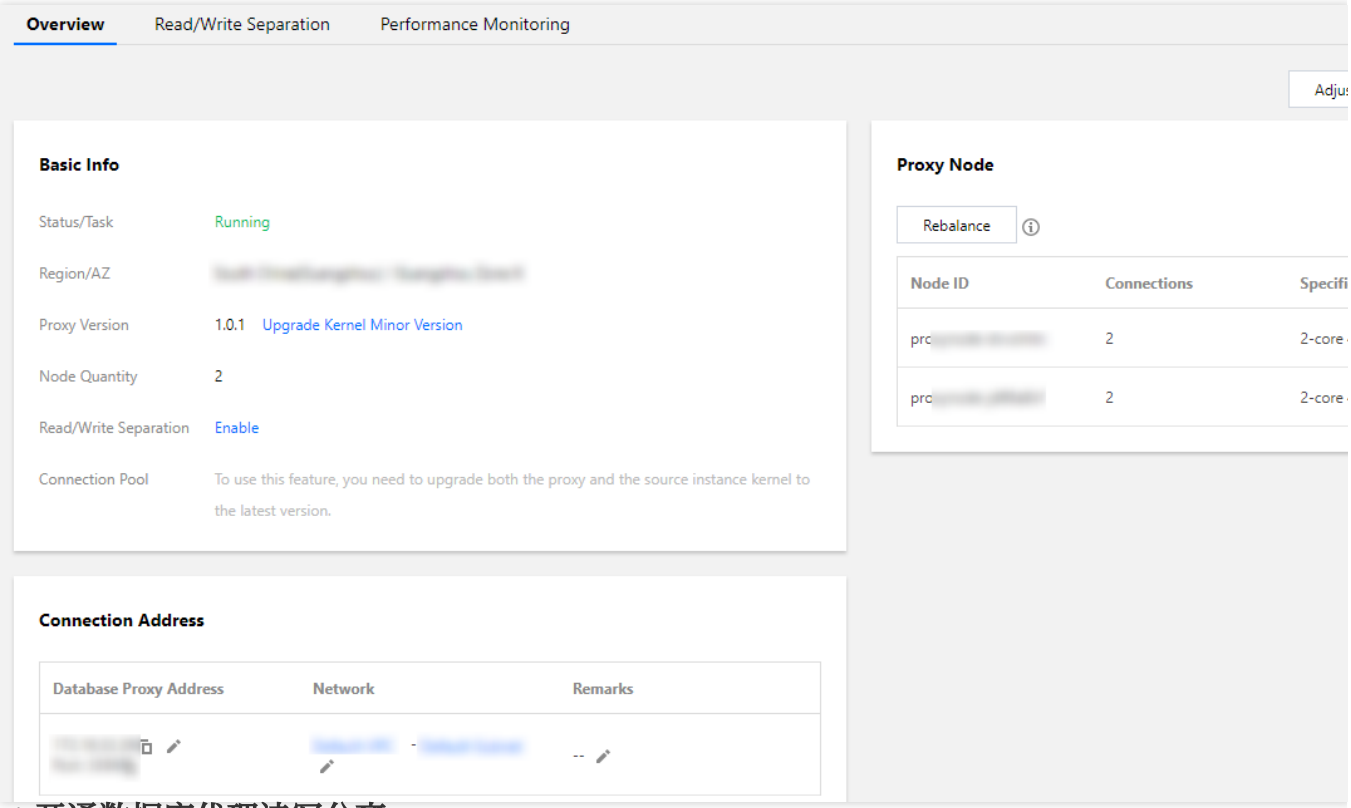


4. 开通成功后，可在数据库代理页查看基本信息、管理代理节点、修改数据库代理地址、调整配置等。

说明：

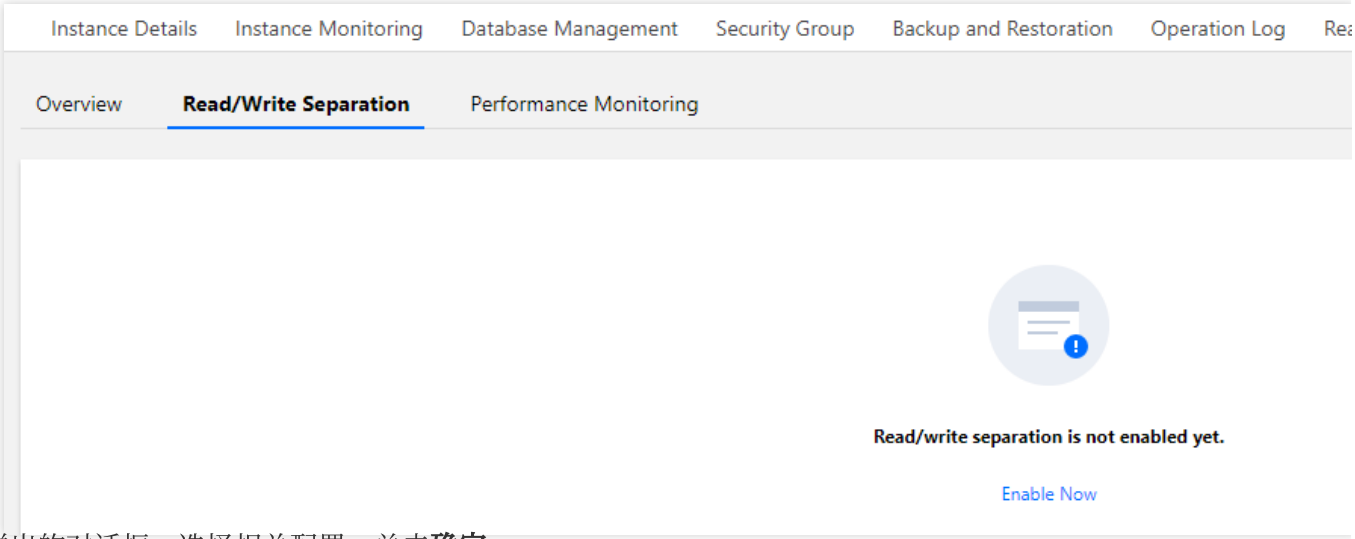
您可通过查看代理节点列表的**连接数**或查看各代理节点的性能监控，来判断各节点是否出现访问不均衡现象。若出现各代理节点连接数不均衡的现象，可单击**重新负载均衡**打散连接。

重新负载均衡会触发代理节点重启，重启过程中会有短暂服务不可用现象，建议在低峰期重启服务，请确保业务具备重连机制。



步骤2：开通数据库代理读写分离

1. 登录 MySQL 控制台。
2. 在上方选择地域，单击实例 ID 或**操作**列的**管理**，进入实例管理页。
3. 在实例管理页的**数据库代理**页，选择**读写分离**页，单击**立即开启**。



4. 在弹出的对话框，选择相关配置，单击**确定**。

注意：

仅状态为运行中的主实例和只读实例，才能加入数据库代理中。

目前异地 RO 和延迟 RO 不允许挂载到数据库代理下。

只读实例延迟剔除：是否启动剔除策略。当只读实例发生复制异常（复制延迟、复制中断）时，数据库代理会将只读实例暂时移出读写分离。延迟剔除阈值默认为10秒、只读实例最小保留数为1个。

说明：

只读实例剔除设置阈值和实例最小保留数后，仅对新连接生效。

延迟剔除阈值：只读实例同步主实例数据时允许的最长延迟时间。若一个只读实例的延迟时间超过该阈值，则不论该只读实例的权重是多少，读请求都不会转发至该只读实例。取值范围为大于或等于1的整数。

若只读实例延迟超过阈值时，则会被剔除，被剔除的实例权重自动设置为0，且系统会向用户发出告警（须先订阅“数据库代理挂载节点剔除”告警，配置请参见 告警功能）。

当只读实例延迟小于阈值时，会重新加入到数据库代理中。同时，不管是否启用延迟剔除功能，只读实例故障被剔除后，待实例修复也会重新加入到数据库代理中。

通过设置只读最小保留数，数据库代理发现当前路由中的只读实例小于设定的值时，将会把异常的只读实例加回读写分离，直至参与读写分离的只读实例个数满足最小保留数。

注意：

当只读库发生致命故障（如宕机）时，最小保留数无法作用于致命故障的实例。

只读实例最小保留数：需要保证的实例下限数，若现有只读实例数小于等于此下限且延迟时间超过阈值，现有只读实例均不被剔除。

读权重分配：为实例分配读权重，可选择系统自动分配或自定义，权重分配范围为0 - 100的整数。读权重分配配置后，会立即对所有连接生效。

数据库代理将按照权重设置分配读请求的流量，如，两个只读库的权重分别为10和20，那么他们的读请求流量将按照1:2的比例分配。

权重仅为读请求权重，写请求会直接路由到主库，不参与权重的计算。如，客户端发送了10条写语句和10条读语句，主库和只读库的权重比为1:1，这时主库会收到10条写语句和5条读语句，只读库收到5条读语句。

选择系统自动分配权重时，系统会根据实例的 CPU 和内存规格来自动分配权重，此时仅能设置主实例的权重。

如果只读实例的权重为0，则数据库代理不会对该只读实例建连，如果只读实例的权重从0改为非0，权重无法立即生效，只能对新连接生效。

故障转移：设置是否开启，推荐开启，当只读实例异常时，数据库代理会将读请求发送至主实例。

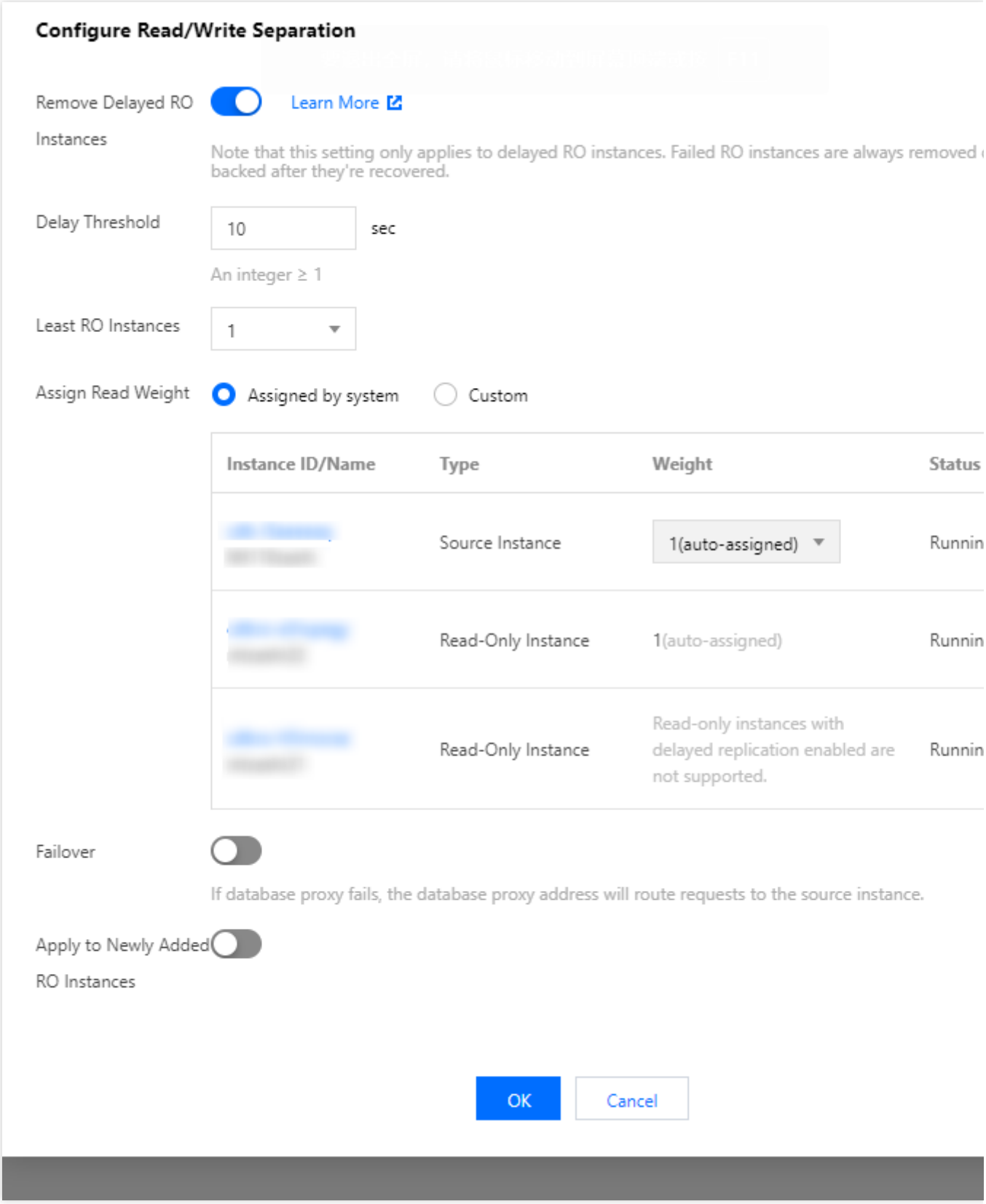
说明：

故障转移配置后，仅对新连接生效。

自动添加只读实例：设置是否开启，开启后，若您购买新的只读实例，会自动添加到数据库代理中。

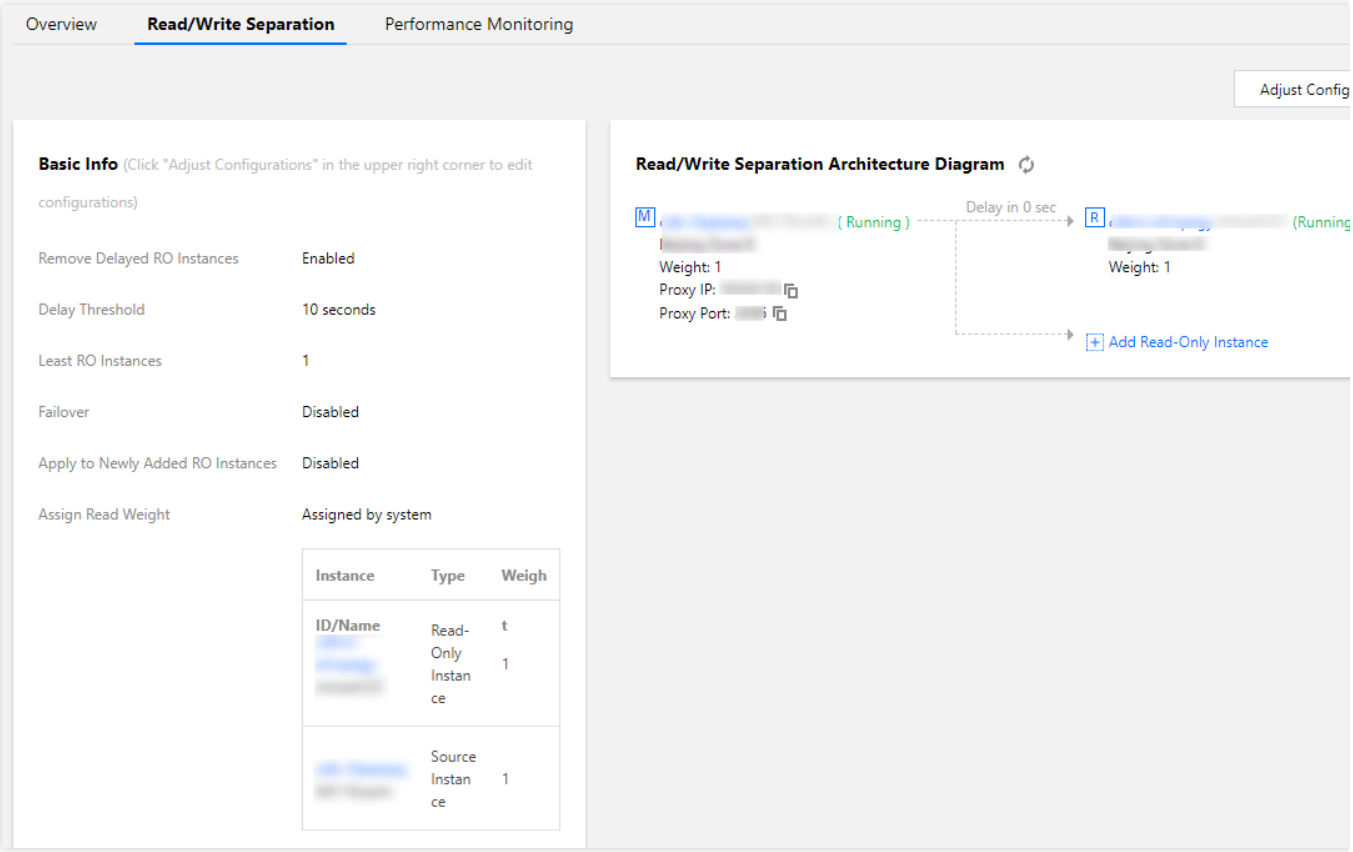
当读权重为系统自动分配时，新购只读实例按照规格大小默认权重分配。

当读权重为自定义时，新购只读实例默认加入时权重为0，可通过数据库代理读写分离的调整配置修改。



成功开启数据库代理读写分离

成功开启数据库代理读写分离后，数据库代理页面如下：



相关文档

设置数据库代理连接地址

切换数据库代理网络

设置会话级连接池

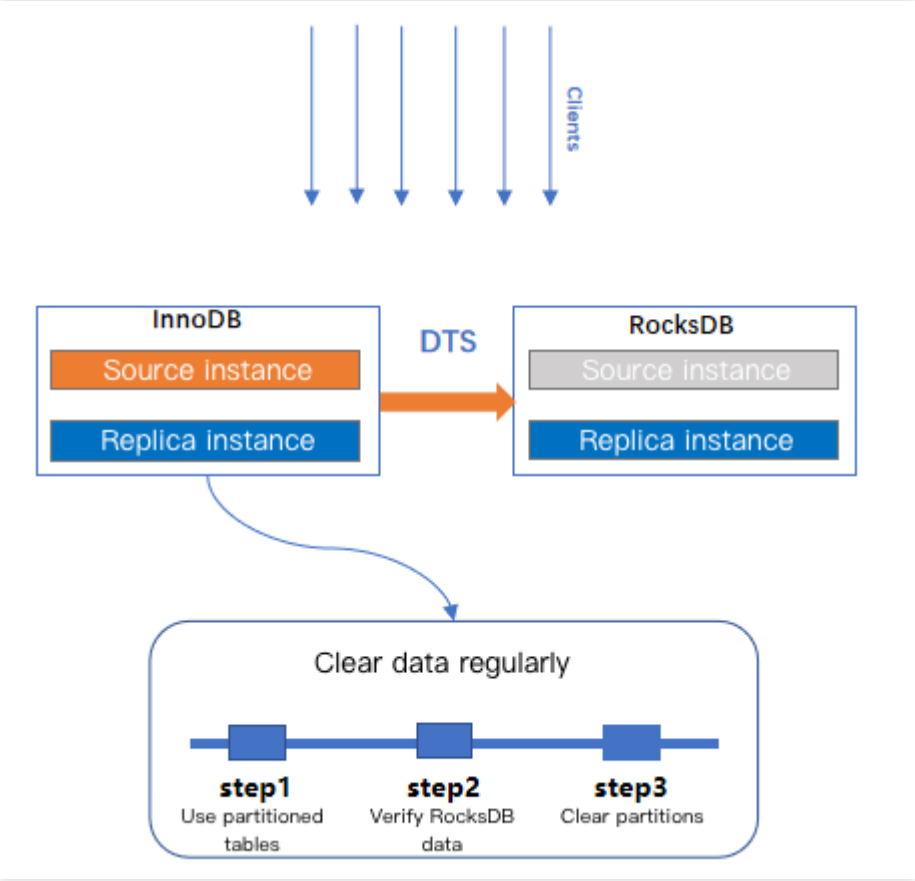
使用 DTS 将 InnoDB 数据迁移至 RocksDB

最近更新时间：2022-06-13 15:37:16

RocksDB 是一个非常流行的高性能持久化 KV（key-value）存储，TXRocks 是腾讯 TXSQL 团队基于此开发的事务型存储引擎。

相比 InnoDB 使用的 B+Tree 索引结构，TXRocks 使用的 LSM Tree 索引结构，可以节省相对比例的存储空间。InnoDB 的 B+Tree 分裂通常会导致页面半满，页面内空闲空间浪费，页面有效利用率相对比较低。TXRocks 的 SST 文件一般设置为 MB 量级或者更大，文件要4K对齐产生的浪费比例很低，SST 内部虽然也划分为 Block，但 Block 是不需要对齐的。另外，TXRocks 的 SST 文件采用前缀压缩，相同的前缀只会记录一份，同时 TXRocks 不同层的 SST 可以采用不同的压缩算法，进一步降低存储空间开销。通常情况下可降低50%存储空间。

您可以通过数据传输服务 DTS 将 InnoDB 数据同步到 RocksDB，以提高写入性能和节省存储空间。



说明：
数据传输服务（Data Transmission Service，DTS）是提供数据迁移、数据同步、数据订阅于一体的数据库数据传输服务。帮助您在业务不停服的前提下轻松完成数据库迁移，利用实时同步通道轻松构建异地容灾的高

可用数据库架构，利用数据订阅提供的云数据库实时增量更新数据，您可根据自身业务需求自由消费增量数据。

注意事项

- 使用数据传输服务 DTS 进行数据同步，确保增量数据同步至 RocksDB 中，并屏蔽掉 Delete 操作。
- 当数据传输至 RocksDB 后，检查数据有效性，确认有效性后再将源库中的数据进行清理，降低存储使用空间。
- 源表可以使用分区表，提升数据清理效率（必须注意分区表的使用限制且目标表不支持分区）。
- RocksDB 定期执行 compaction 操作，可以有效降低数据空间占用。
- 有关 RocksDB 引擎的注意事项，可参见 TXRocks 引擎使用须知。

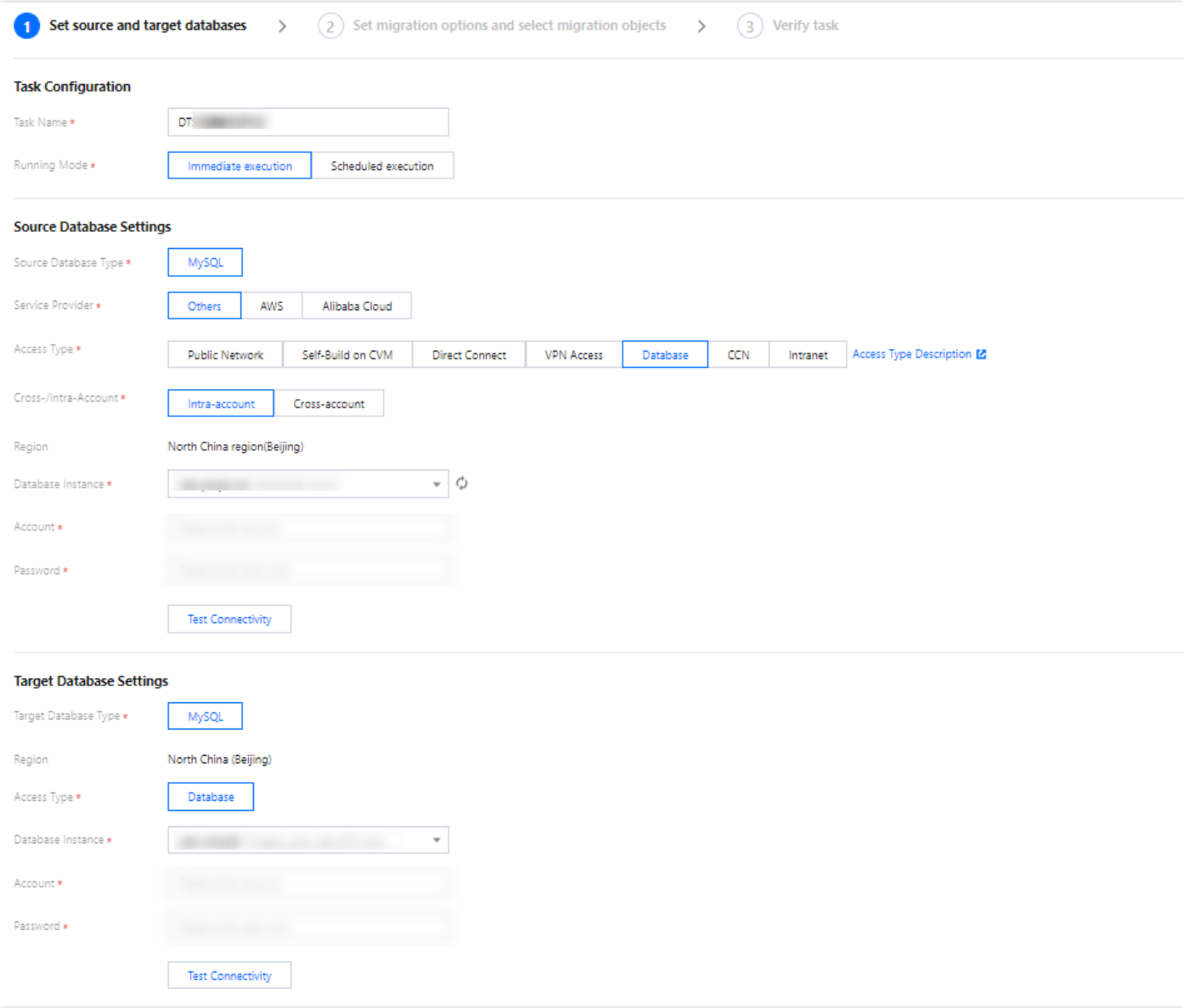
操作步骤

1. 登录 DTS 控制台，在左侧导航选择**数据迁移**页，单击**新建迁移任务**，进入新建迁移任务页面。
2. 在新建迁移任务页面，选择迁移的源实例类型和所属地域，目标实例类型和所属地域，规格等，然后单击**立即购买**。

配置项	说明
源实例类型	请根据您的源数据库类型选择，购买后不可修改。此处选择“MySQL”。
源实例地域	选择源数据库所属地域。如果源库为自建数据库，选择离自建数据库最近的一个地域即可。
目标实例类型	请根据您的目标数据库类型选择，购买后不可修改。此处选择“MySQL”。
目标实例地域	选择目标数据库所属地域。
规格	根据业务情况选择迁移链路的规格，不同规格的性能和计费详情请参考 计费概述。

3. 在迁移任务的**操作**列选择**更多** > **配置**，在设置源和目标数据库页面，完成任务设置、源库设置和目标库设置，测试源库和目标库连通性通过后，单击**保存**。

> 说明：
> 如果连通性测试失败，请根据提示和 修复指导 进行排查和解决，然后再次重试。



4. 在设置迁移选项及选择迁移对象页面，设置迁移类型、对象，单击**保存**。

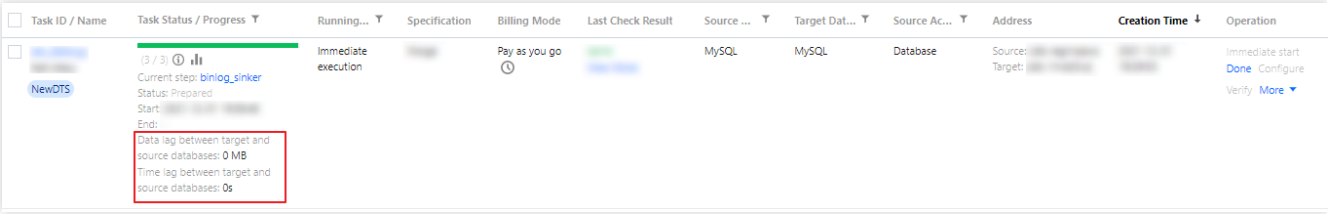
说明：

- 如果用户在迁移过程中确定会使用 gh-ost、pt-osc 等工具对某张表做 Online DDL，则**迁移对象**需要选择这个表所在的整个库（或者整个实例），不能仅选择这个表，否则无法迁移 Online DDL 变更产生的临时表数据到目标数据库。
- 如果用户在迁移过程中确定会对某张表使用 rename 操作（例如将 table A rename 为 table B），则**迁移对象**需要选择 table A 所在的整个库（或者整个实例），不能仅选择 table A，否则系统会报错。

配置项	说明

配置项	说明
迁移类型	请根据您的场景选择。 ○ 结构迁移：迁移数据库中的库、表等结构化的数据。 ○ 全量迁移：迁移整个数据库，迁移数据仅针对任务发起时，源数据库已有的内容，不包括任务发起后源库实时新增的数据写入。 ○ 全量 + 增量迁移：迁移数据包括任务发起时源库的已有内容，也包括任务发起后源库实时新增的数据写入。如果迁移过程中源库有数据写入，需要不停机平滑迁移，请选择此场景。
迁移对象	○ 整个实例：迁移整个实例，但不包括系统库，如 information_schema、mysql、performance_schema、sys。 ○ 指定对象：迁移指定对象。
指定对象	在源库对象中选择待迁移的对象，然后将其移到已选对象框中。
是否迁移账号	如果需要迁移源库的账号信息，则勾选此功能。

5. 在校验任务页面，进行校验，校验任务通过后，单击**启动任务**。
6. 返回数据迁移任务列表，任务进入准备运行状态，运行1分钟 - 2分钟后，数据迁移任务开始正式启动。
   - 选择**结构迁移**或者**全量迁移**：任务完成后会自动结束，不需要手动结束。
   - 选择**全量 + 增量迁移**：全量迁移完成后会自动进入增量数据同步阶段，增量数据同步不会自动结束，需要您手动单击**完成**结束增量数据同步。
     - 请选择合适时间手动完成增量数据同步，并完成业务切换。
     - 观察迁移阶段为增量同步，并显示无延迟状态，将源库停写几分钟。
     - 目标与源库数据差距为0MB及目标与源库时间延迟为0秒时，手动完成增量同步。



7. 当迁移任务状态变为**任务成功**时，即可实现将 InnoDB 数据同步到 RocksDB。

构建 LAMP 堆栈 Web 应用程序

最近更新时间：2024-04-19 18:15:24

LAMP 指 Linux+Apache+Mysql/MariaDB+Perl/PHP/Python，是一组常用来搭建动态网站或者服务器的开源软件。程序都是独立的，但是因为常被一起使用，相互间的兼容性越来越高，共同组成了一个强大的 Web 应用程序平台。
本教程将指导您完成以下过程：启动一个腾讯云数据库实例，通过腾讯云服务器配置一个 LAMP 应用程序，用以连接腾讯云数据库实例的高可用性环境。
运行腾讯云数据库实例会将数据库与环境的生命周期分离，这让您可以从多个服务器连接到同一个数据库，并且简化数据库的运维，让您无需再关心数据库的安装、部署、版本更新及故障处理等问题。
说明：
本教程中使用到的云数据库实例和云服务器实例处于同一地域下，如果您的云数据库实例和云服务器实例不处于同一地域下，请参见 外网访问。

初始化云数据库实例

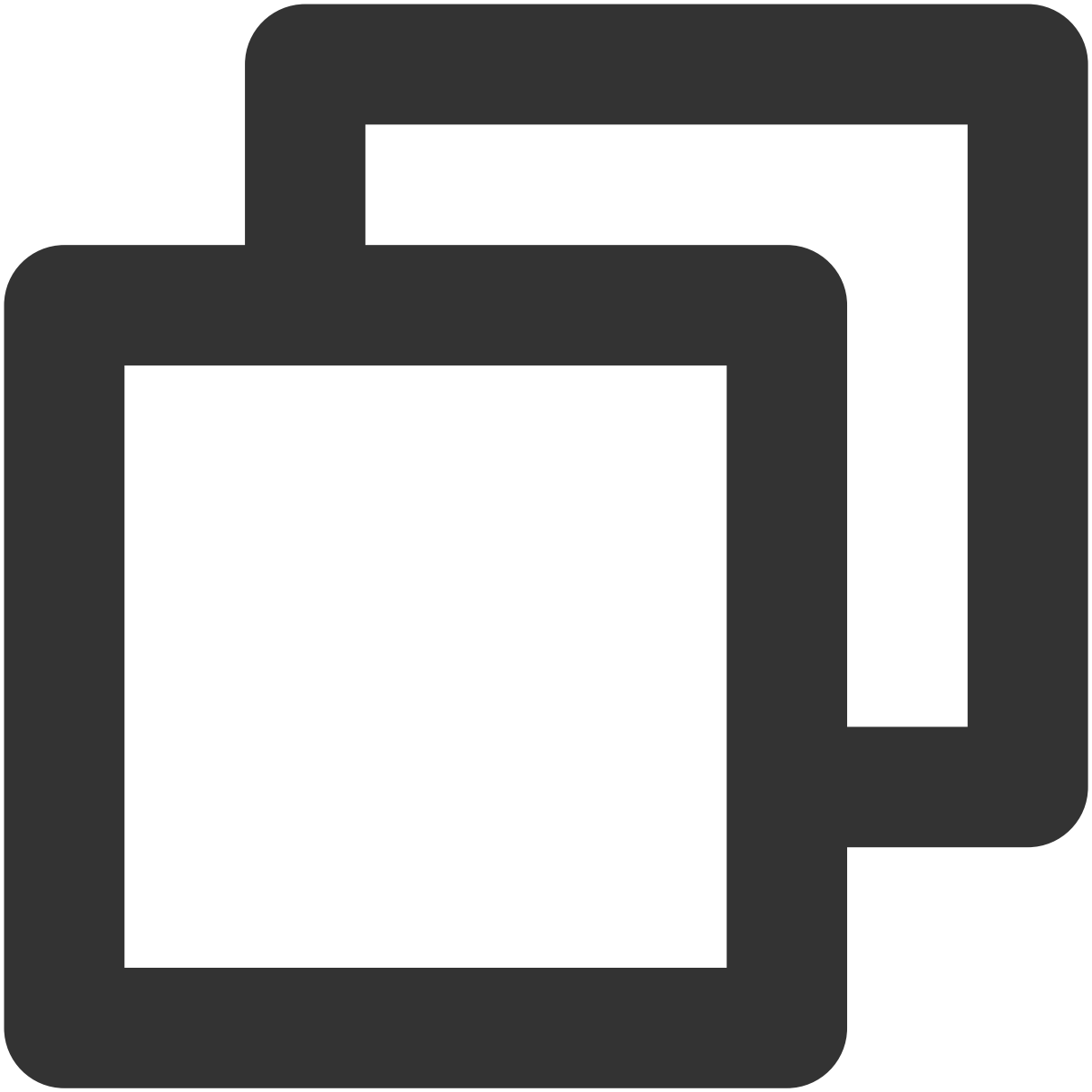
云数据库的购买和初始化请参见 购买方式 和 初始化 MySQL 数据库。

登录云服务器实例

云服务器的购买和访问请参见 快速入门 Linux 云服务器，本教程中使用的是 CentOS 系统。

安装 MySQL 客户端

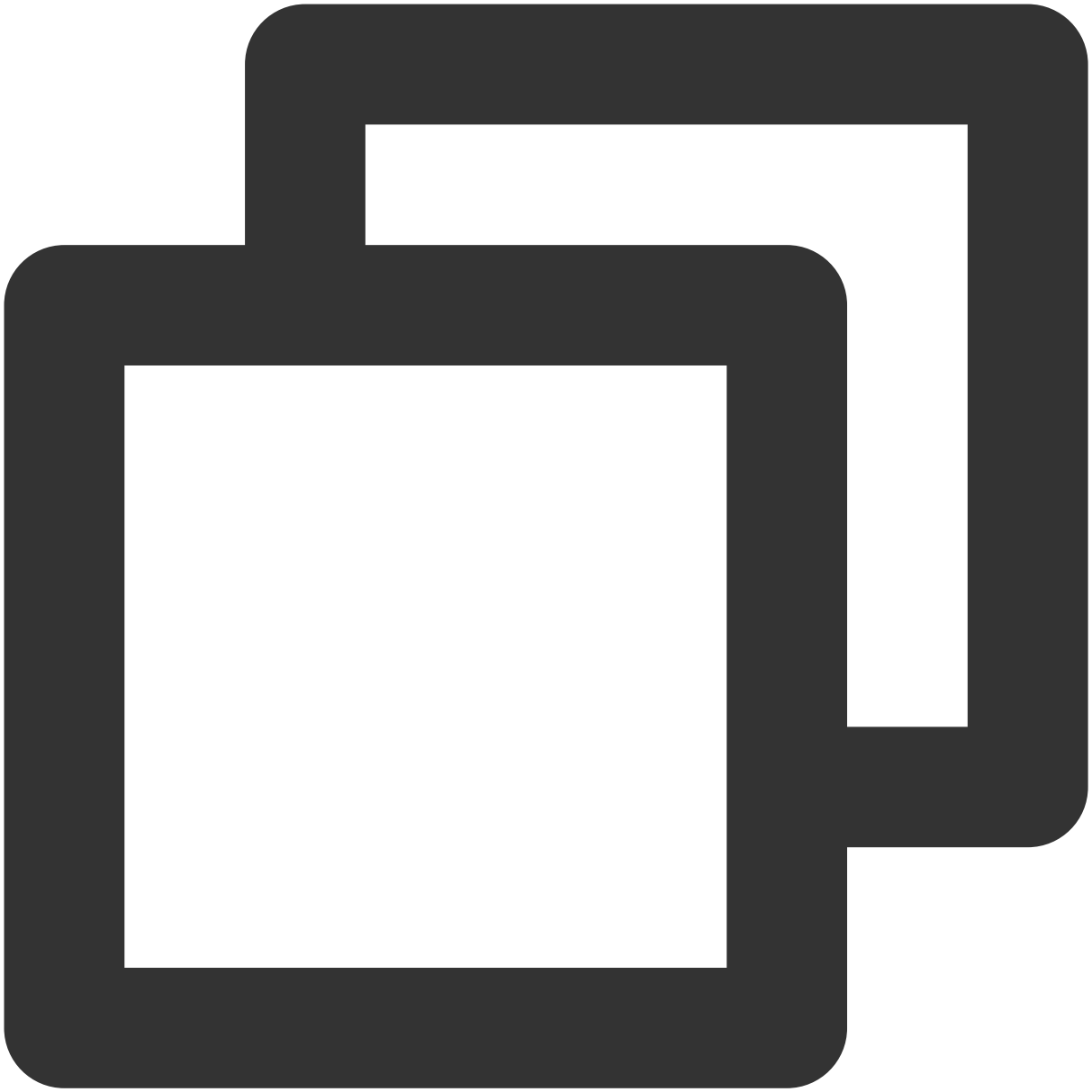
1. 在云服务器实例中使用 `yum` 安装 MySQL 客户端。



```
yum install mysql -y
```

```
[root@VM_165_193_centos html]# yum install mysql -y
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package mariadb.x86_64 1:5.5.52-1.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                      Arch                       Version
================================================================================
Installing:
 mariadb                      x86_64                     1:5.5.52-1.el7

Transaction Summary
================================================================================
Install  1 Package

Total download size: 8.7 M
Installed size: 48 M
Downloading packages:
mariadb-5.5.52-1.el7.x86_64.rpm
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 1:mariadb-5.5.52-1.el7.x86_64
  Verifying  : 1:mariadb-5.5.52-1.el7.x86_64

Installed:
  mariadb.x86_64 1:5.5.52-1.el7

Complete!
```

2. 安装完成后，连接到腾讯云数据库实例。

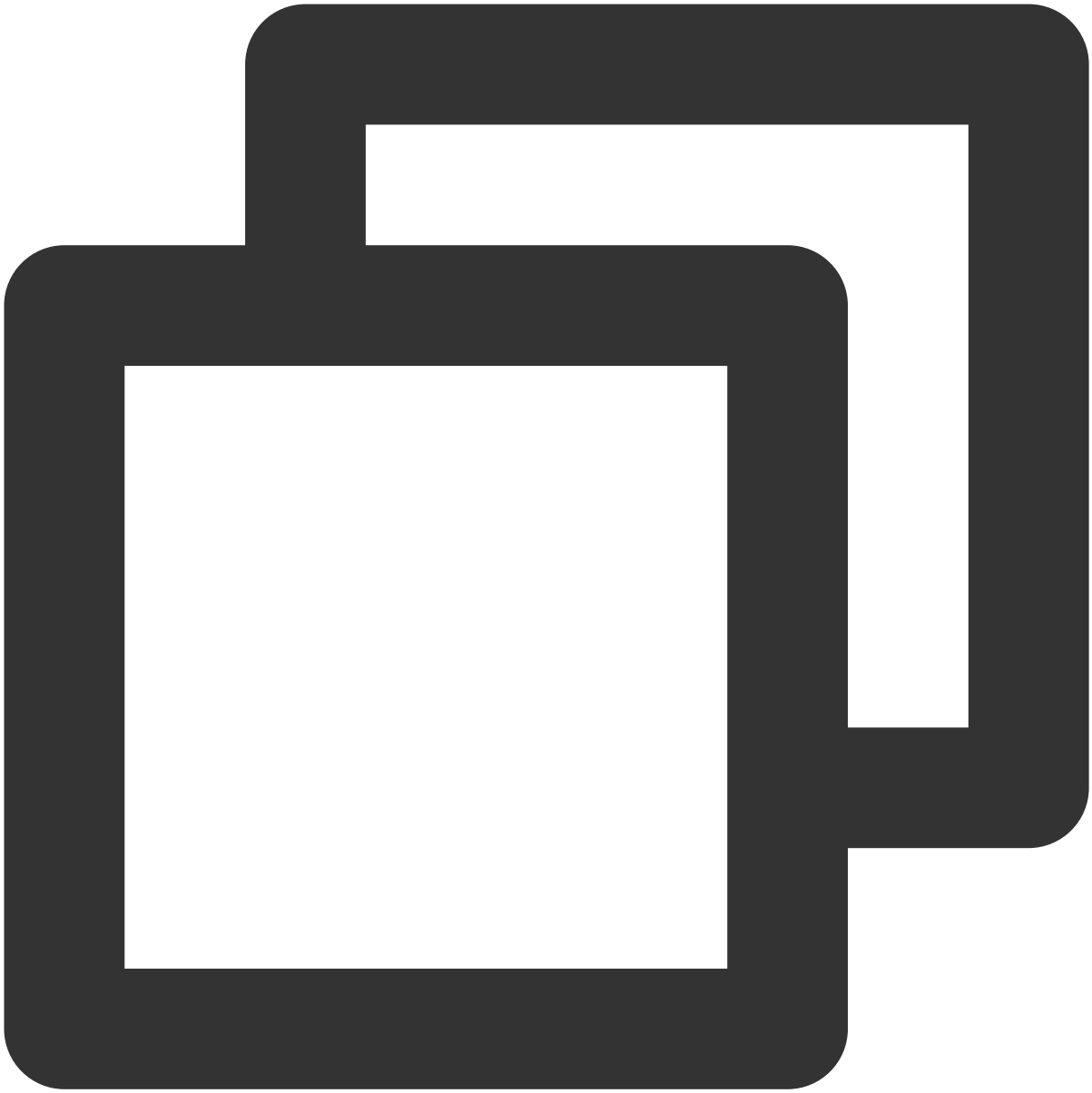


```
mysql -h hostname -u username -p
```

```
[root@VM_165_193_centos html]# mysql -h [redacted] -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 19768
Server version: 5.6.28-cdb2016-log 20170228

Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]>
```

其中，hostname 为数据库实例的内网 IP 地址，username 为您的数据库用户名。

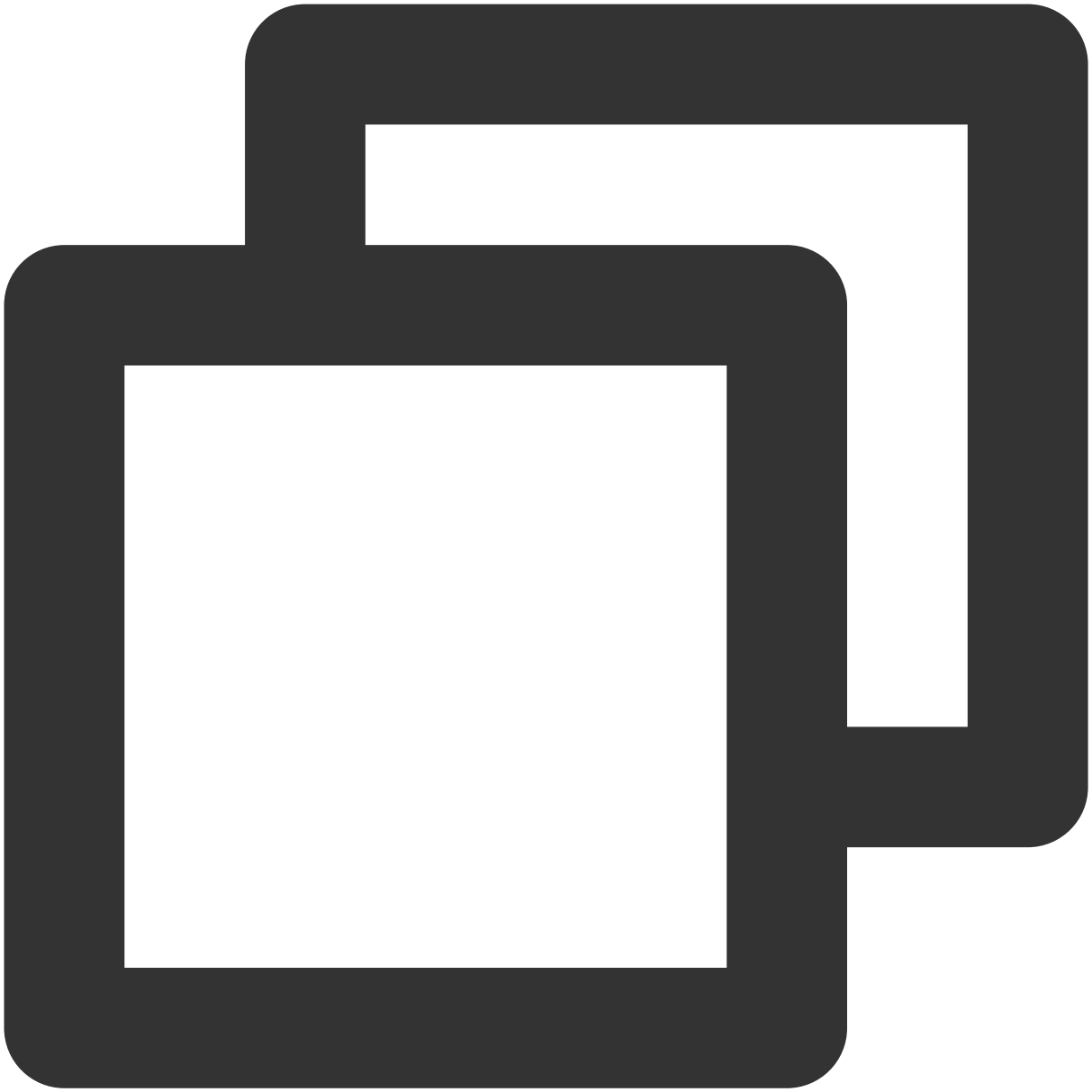
3. 连接成功后，即可退出数据库，进行下一步操作。



```
quit;
```

安装 Apache 服务

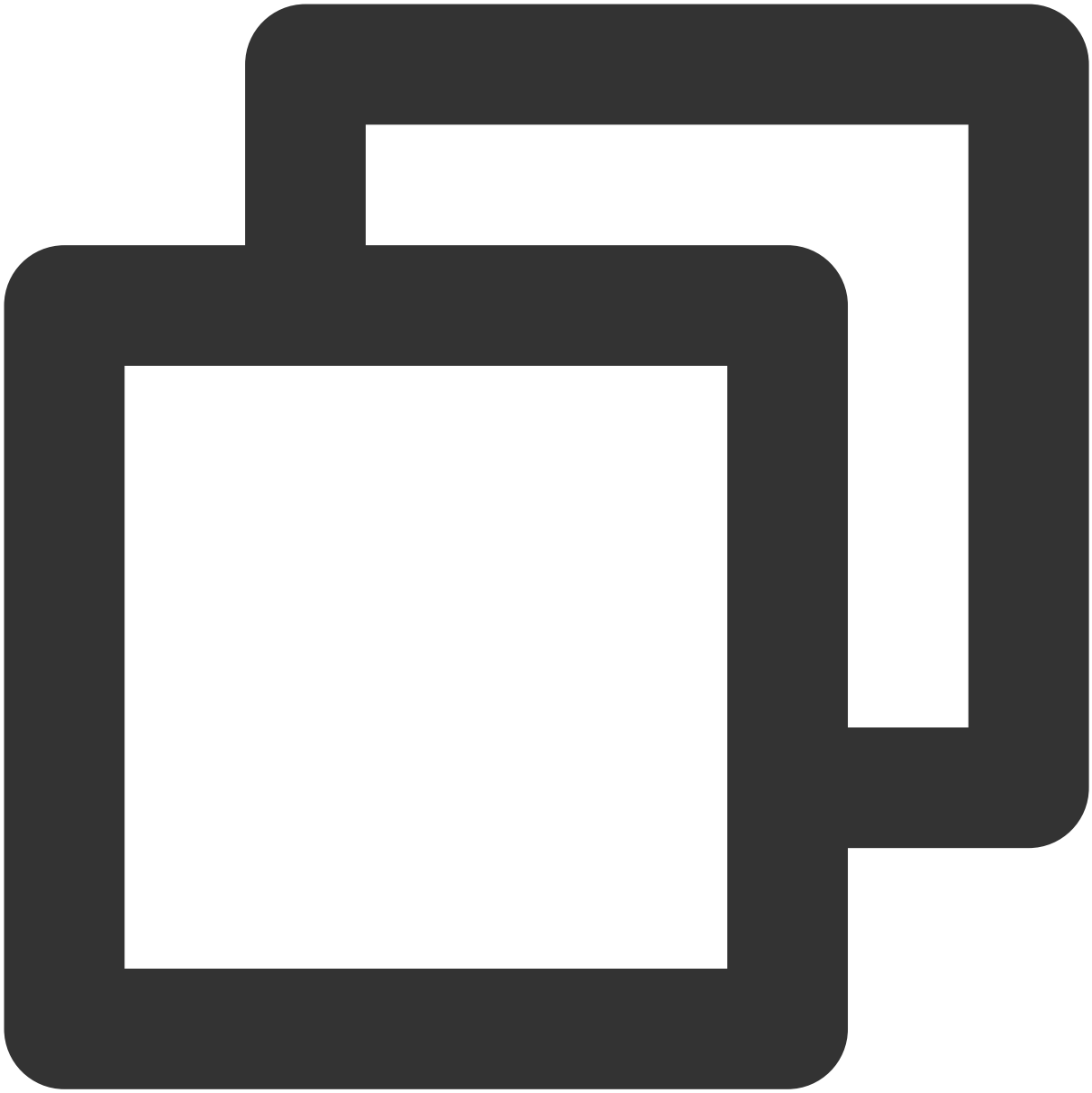
1. 在云服务器实例中使用 `yum` 安装 Apache。



```
yum install httpd -y
```

```
[root@VM_165_193_centos html]# yum install httpd -y
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.6-45.el7.centos.4 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                  Arch                     Version
================================================================================
Installing:
 httpd                    x86_64                   2.4.6-45.el7.centos.4

Transaction Summary
================================================================================
Install  1 Package

Total download size: 2.7 M
Installed size: 9.4 M
Downloading packages:
httpd-2.4.6-45.el7.centos.4.x86_64.rpm
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : httpd-2.4.6-45.el7.centos.4.x86_64
  Verifying  : httpd-2.4.6-45.el7.centos.4.x86_64

Installed:
  httpd.x86_64 0:2.4.6-45.el7.centos.4

Complete!
```

2. 启动 Apache 服务。



```
service httpd start
```

3. 测试 Apache。

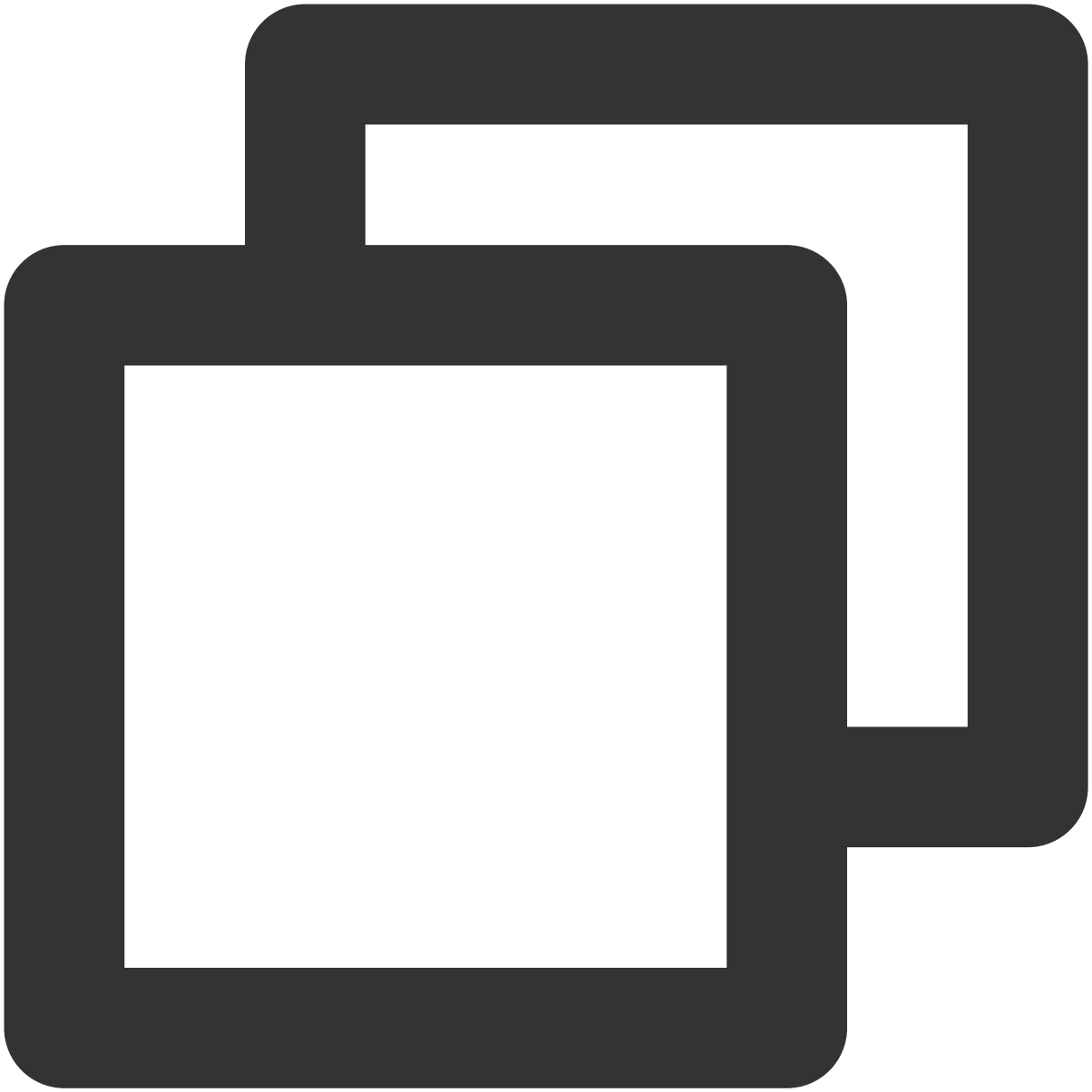
注意：

此步骤需要您的云服务器在安全组中配置来源为 **all**，端口协议为 **TCP:80** 的入站规则。安全组配置请参见 安全组。

在您本地的浏览器中输入 `https://xxx.xxx.xxx.xxx/` （其中 `xxx.xxx.xxx.xxx` 为您的云服务器公网 IP 地址），出现下列画面表示 Apache 启动成功。

安装 PHP

1. 在云服务器实例中使用 `yum` 安装 PHP。

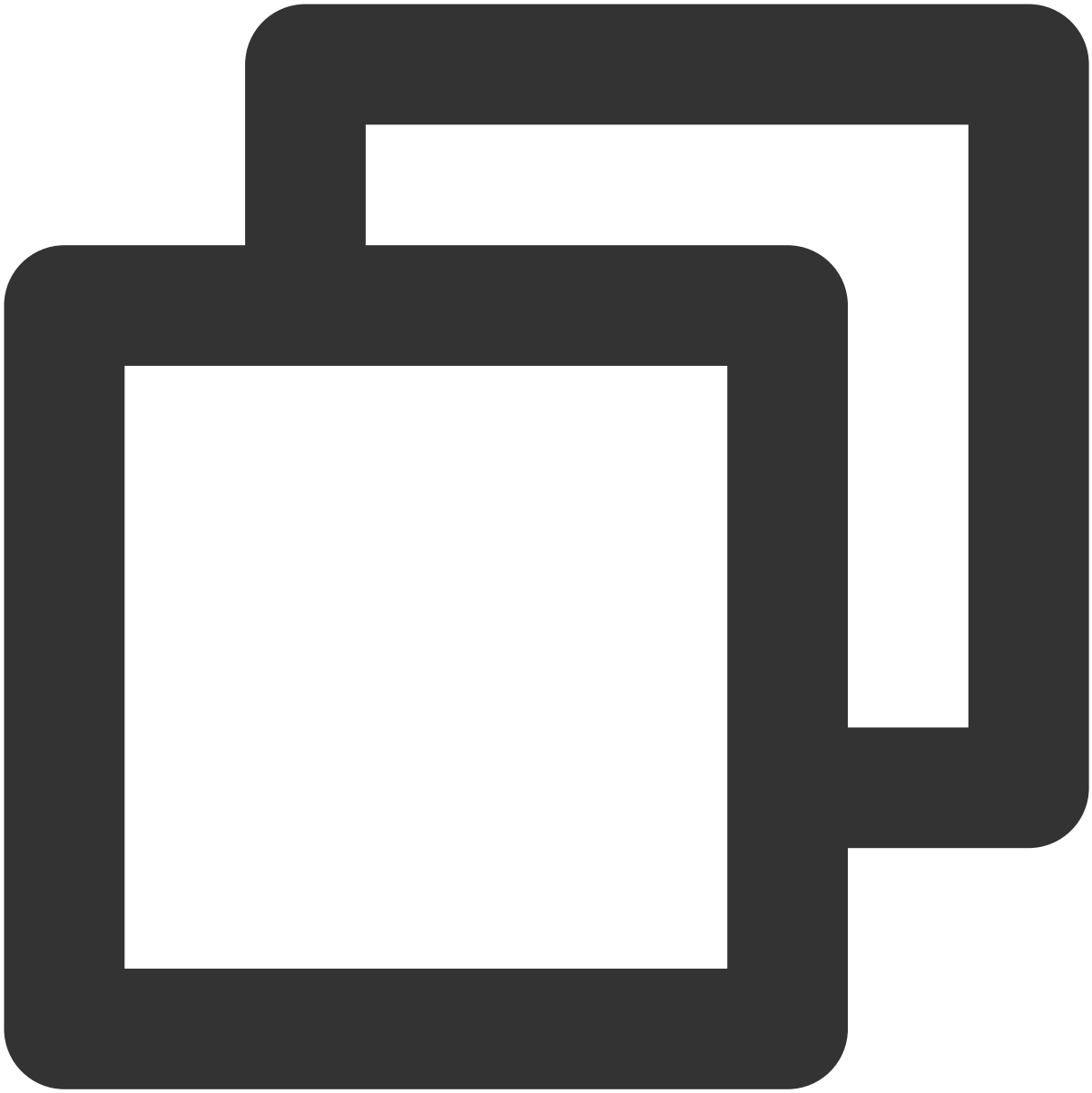


```
yum install php -y
```

```
[root@VM_165_193_centos html]# yum install php -y
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package php.x86_64 0:5.4.16-42.el7 will be installed
--> Processing Dependency: php-common(x86-64) = 5.4.16-42.el7 for p
--> Processing Dependency: php-cli(x86-64) = 5.4.16-42.el7 for pack
--> Running transaction check
---> Package php-cli.x86_64 0:5.4.16-42.el7 will be installed
---> Package php-common.x86_64 0:5.4.16-42.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=================================================================
 Package                        Arch                       Versio
=================================================================
Installing:
 php                            x86_64                     5.4.16
Installing for dependencies:
 php-cli                        x86_64                     5.4.16
 php-common                     x86_64                     5.4.16

Transaction Summary
=================================================================
Install  1 Package (+2 Dependent packages)

Total download size: 4.6 M
Installed size: 17 M
Downloading packages:
(1/3): php-5.4.16-42.el7.x86_64.rpm
(2/3): php-common-5.4.16-42.el7.x86_64.rpm
(3/3): php-cli-5.4.16-42.el7.x86_64.rpm
-----------------------------------------------------------------
Total
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
```

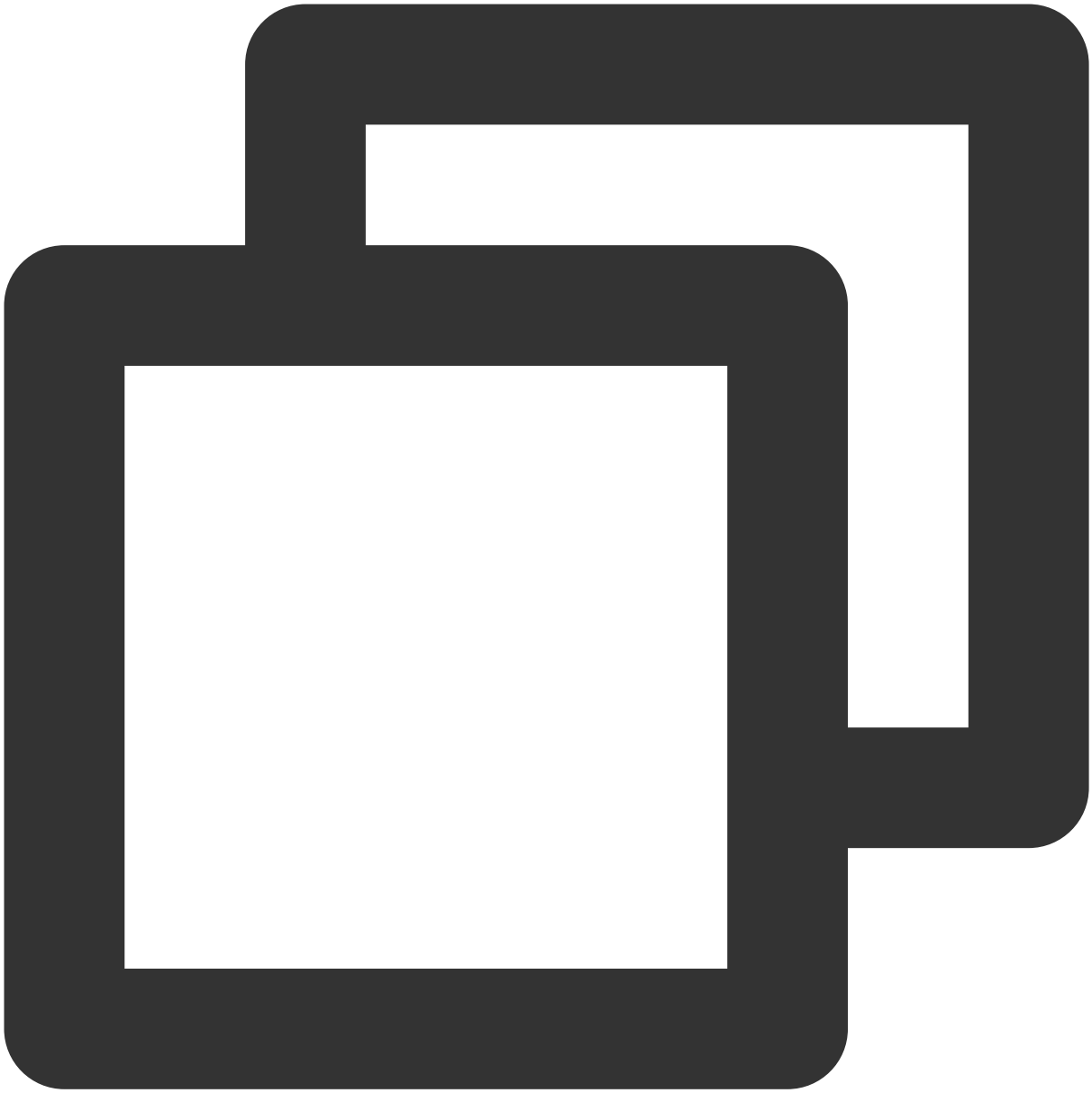
创建项目测试 LAMP 环境

1. 在云服务器 `/var/www/html` 目录下创建一个 info.php 文件，示例代码如下：



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

2. 重启 Apache 服务。



```
service httpd restart
```

3. 在您本地的浏览器中输入 `https://xxx.xxx.xxx.xxx/info.php` ，其中 `xxx.xxx.xxx.xxx` 为您的云服务器公网 IP 地址，出现下列画面表示 LAMP 服务部署成功。

PHP Version 5.4.16

System	Linux VM_165_193_centos 3.10.0-327.36.3.el7.x86_64 #1 SMP Mon Oct 24 16:09:20 UTC 2016 x86_64
Build Date	Nov 6 2016 00:30:05
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/curl.ini, /etc/php.d/fileinfo.ini, /etc/php.d/json.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/sqlite3.ini, /etc/php.d/zip.ini
PHP API	20100412
PHP Extension	20100525
Zend Extension	220100525

构建 Drupal 网站

最近更新时间：2024-04-19 18:11:31

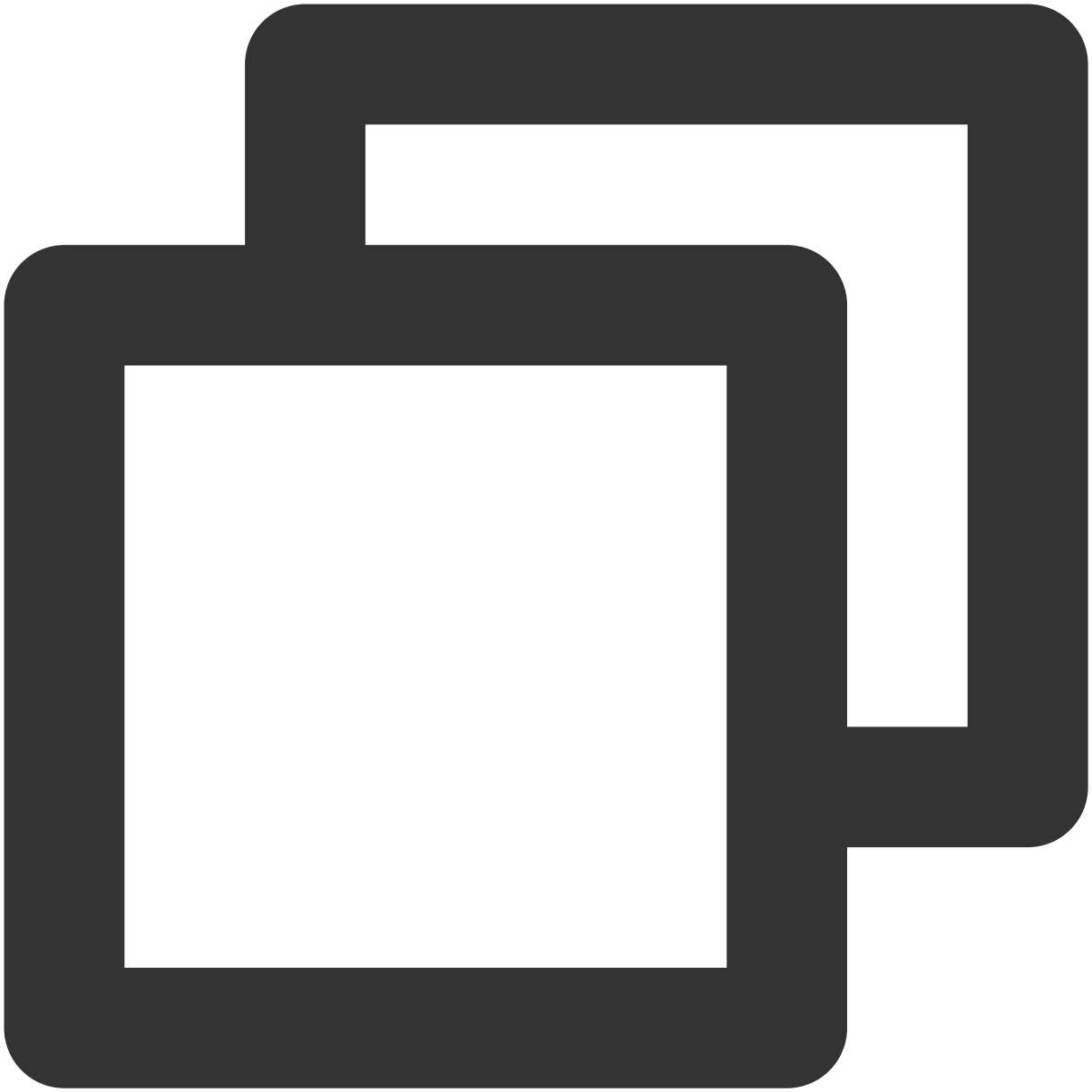
Drupal 是使用 PHP 语言编写的开源内容管理框架（Content Management Framework），它由内容管理系统（Content Management System）和 PHP 开发框架（Framework）共同构成。Drupal 可用于构造提供多种功能和服务的动态网站，能支持从个人博客到大型社区等各种不同应用的网站项目。
您可以通过本教程了解如何在腾讯云服务器上搭建 Drupal 电子商务网站。
使用的软件环境为：CentOS7.2、Drupal7.56、PHP5.4.16。

登录到云服务器实例

云服务器的购买和访问请参见 快速入门 Linux 云服务器。

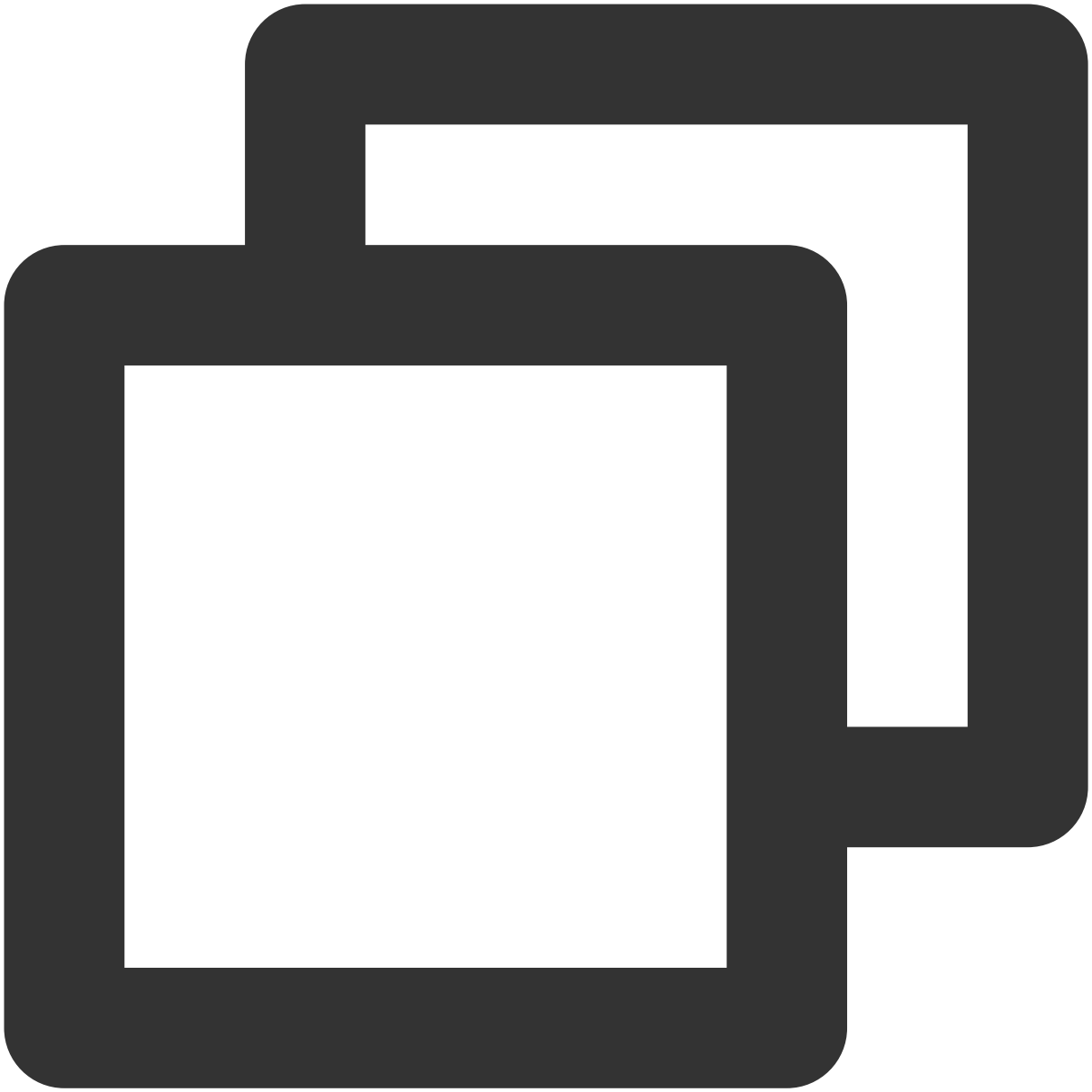
安装 MariaDB 服务

1. CentOS7 以上版本默认支持 MariaDB 数据库，因此我们将使用 MariaDB 数据库。在云服务器实例中使用 `yum` 安装 MariaDB 服务。



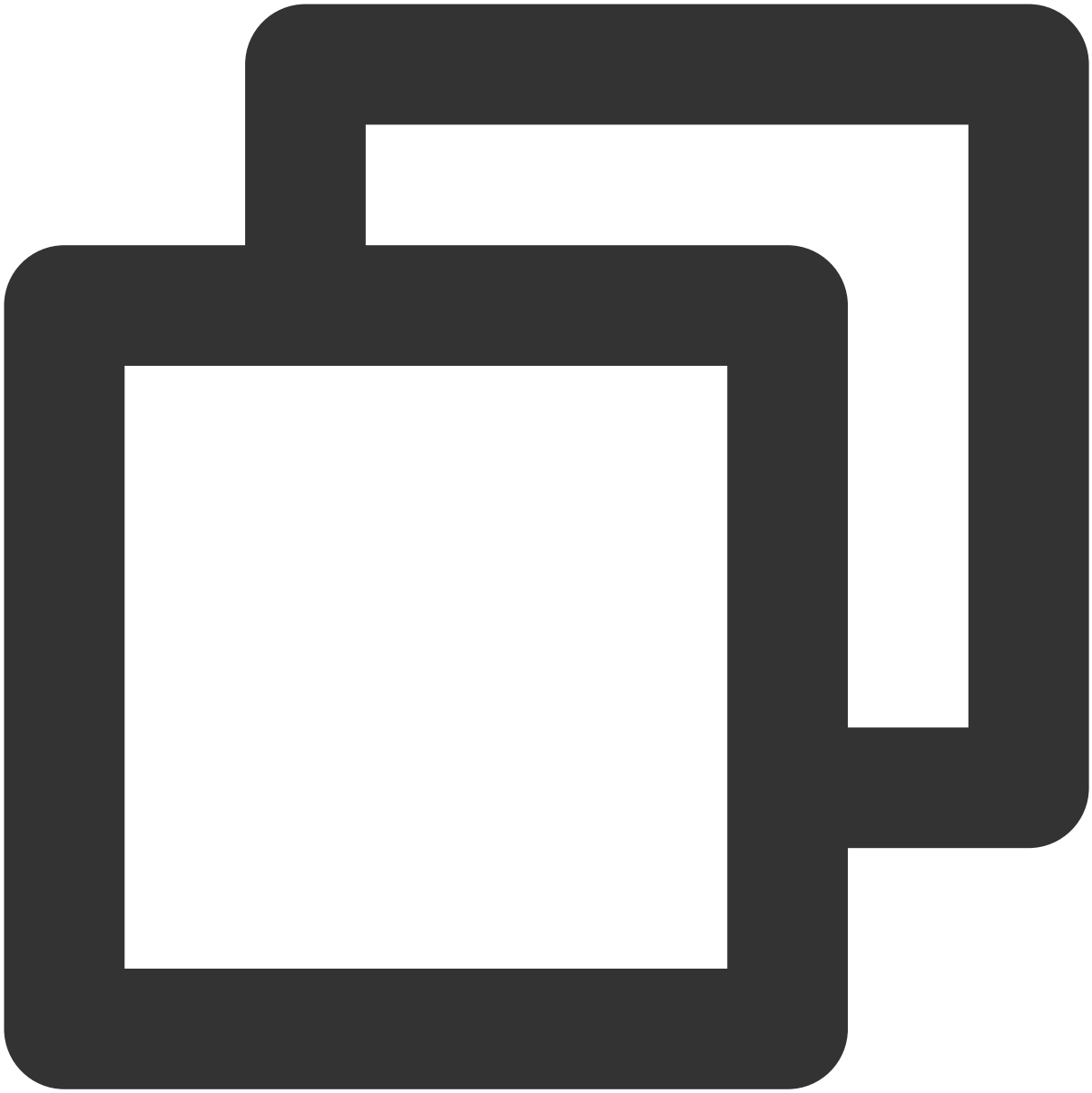
```
yum install mariadb-server mariadb -y
```

2. 启动 MariaDB 服务。



```
systemctl start mariadb
```

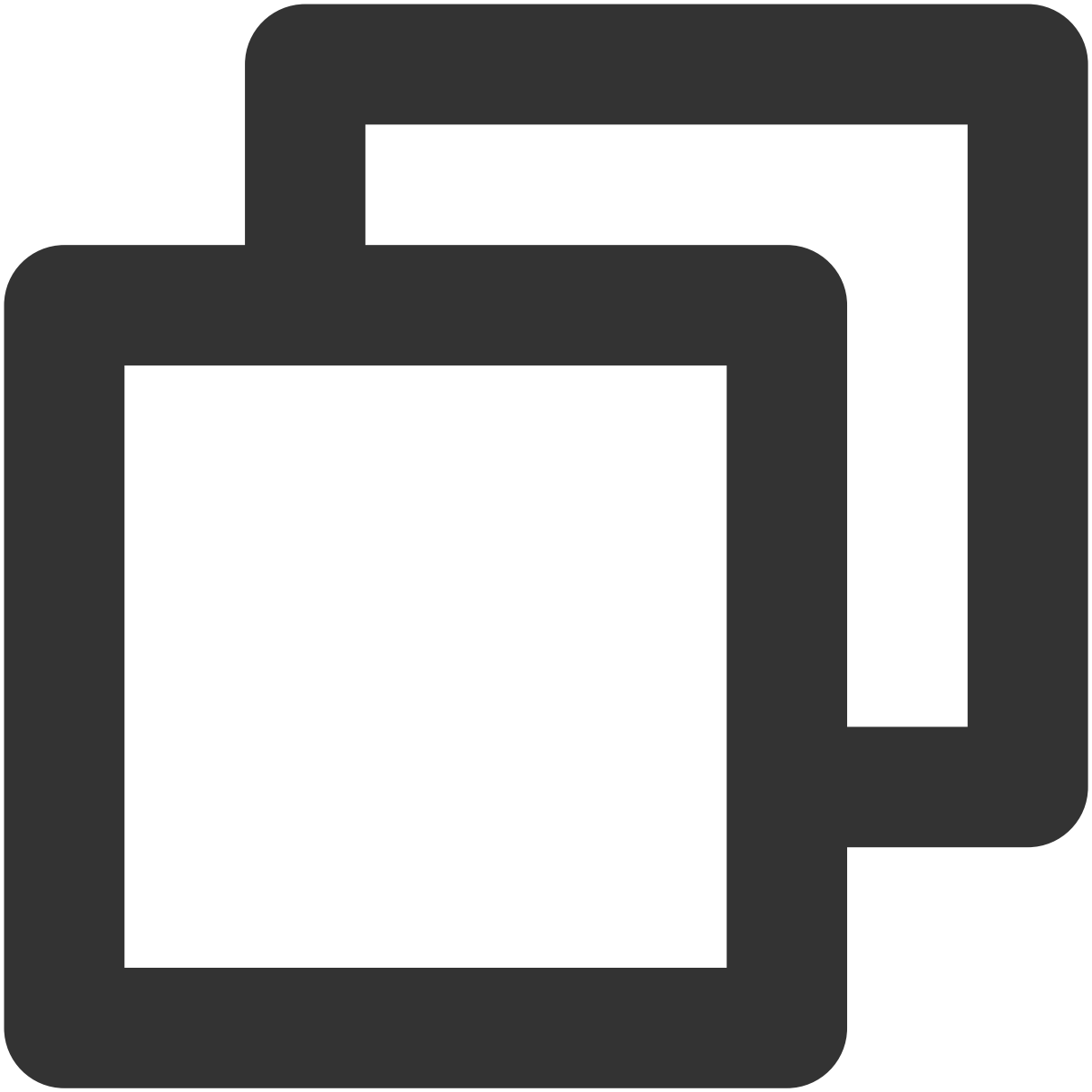
3. 为 Drupal 创建数据库。（本项目中使用 drupal 作为数据库名）



```
mysqladmin -u root -p create drupal
```

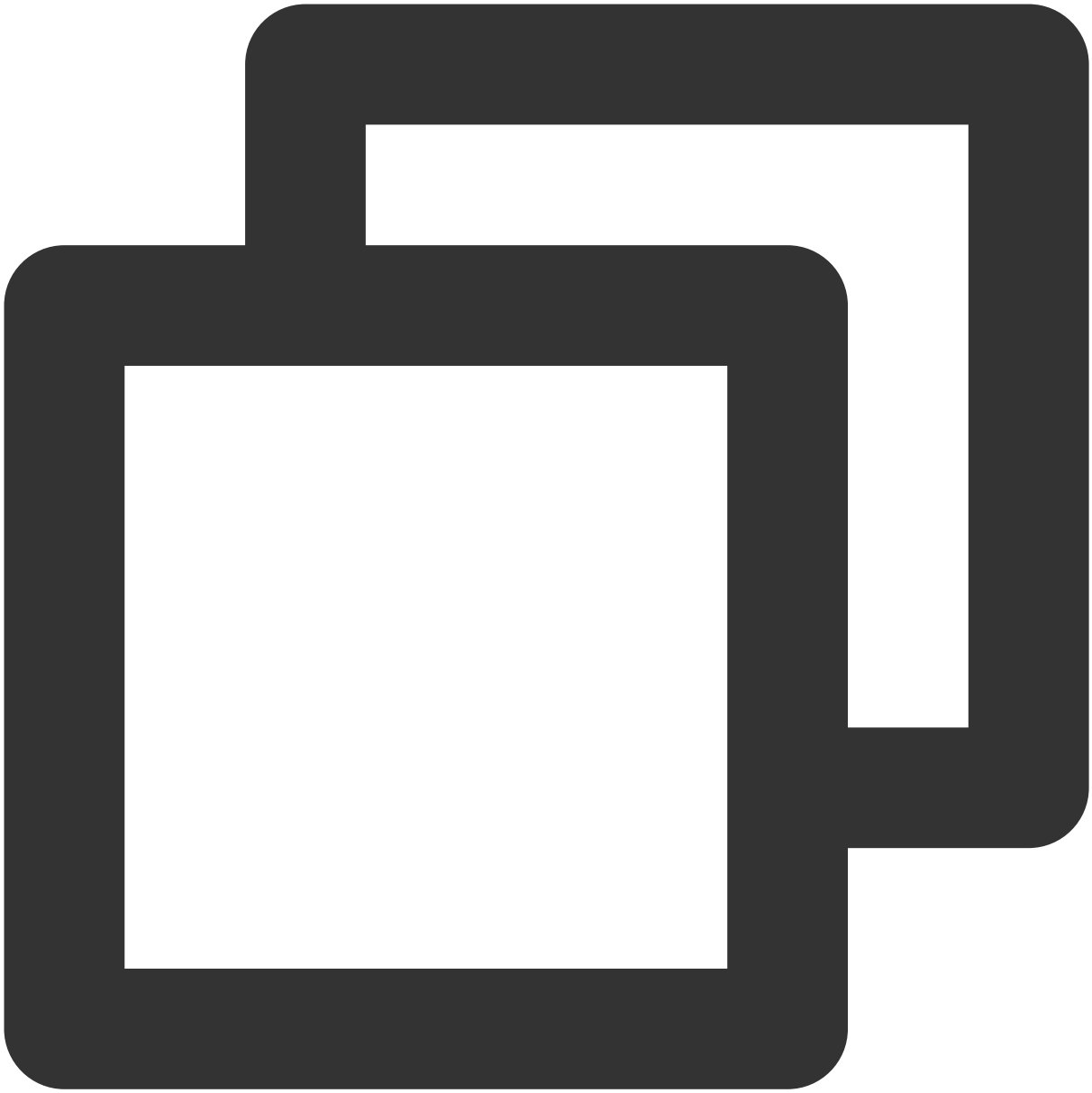
其中，drupal 为 Drupal 服务使用的数据库名。

4. 为数据库创建用户。



```
mysql -u root -p
```

为用户授权，授权成功后退出数据库。

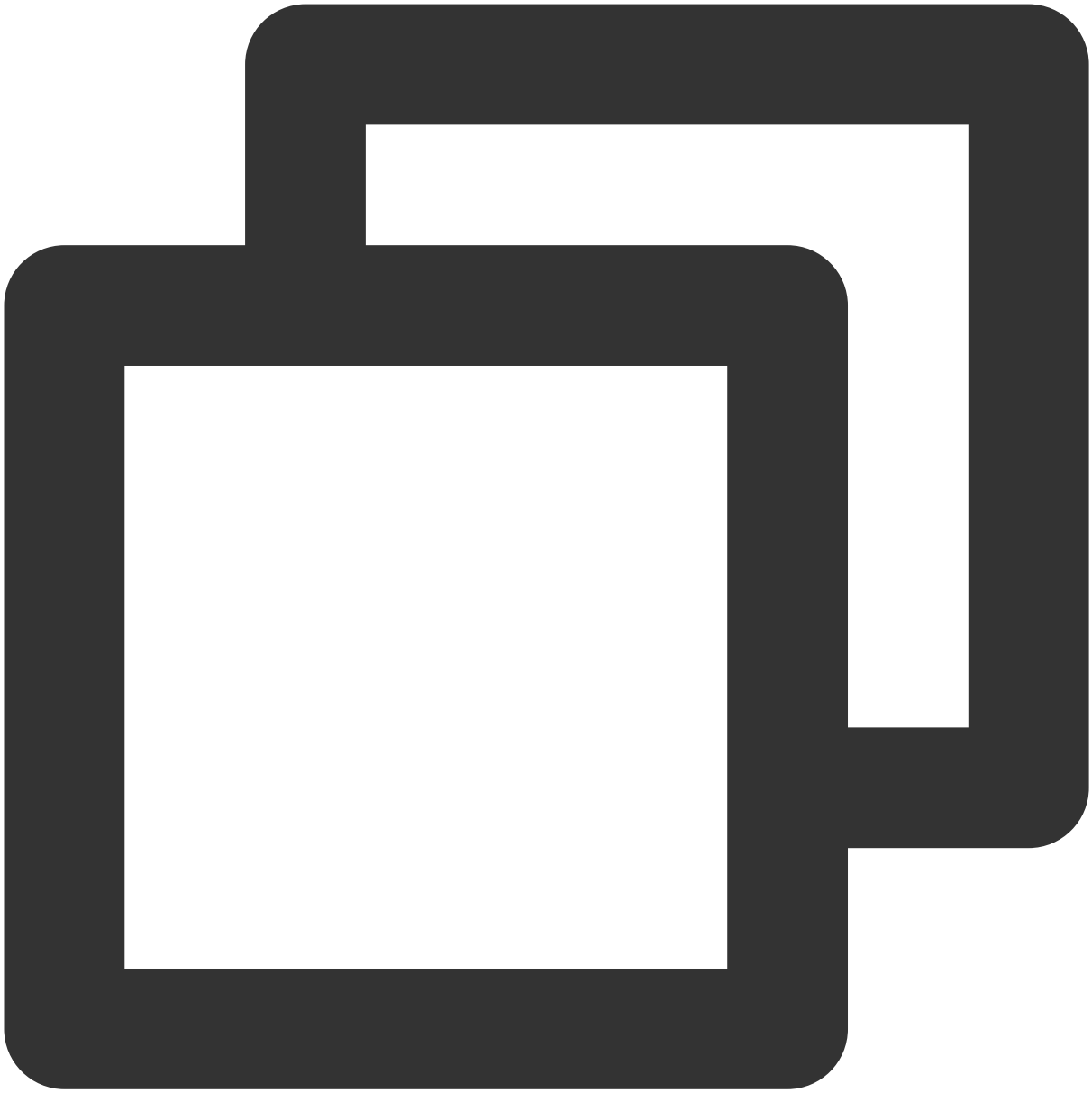


```
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER, CREATE TEMPORARY
FLUSH PRIVILEGES;
exit
```

其中，username 为 Drupal 服务使用的数据库用户名，password 为 Drupal 服务使用的数据库密码。

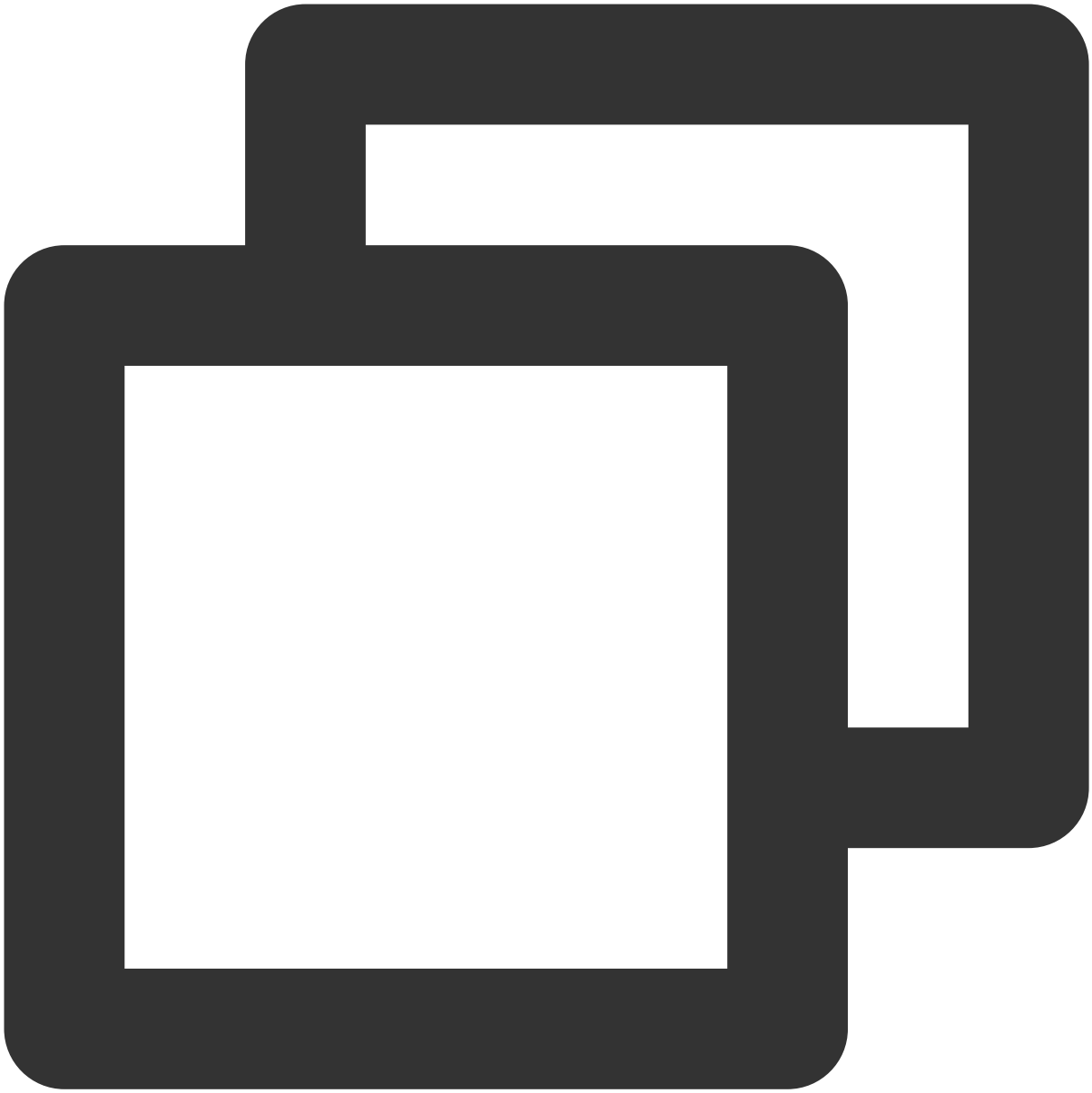
安装 Apache 服务

1. 在云服务器实例中使用 `yum` 安装 Apache。



```
yum install httpd -y
```

2. 启动 Apache 服务。



```
service httpd start
```

3. 测试 Apache。

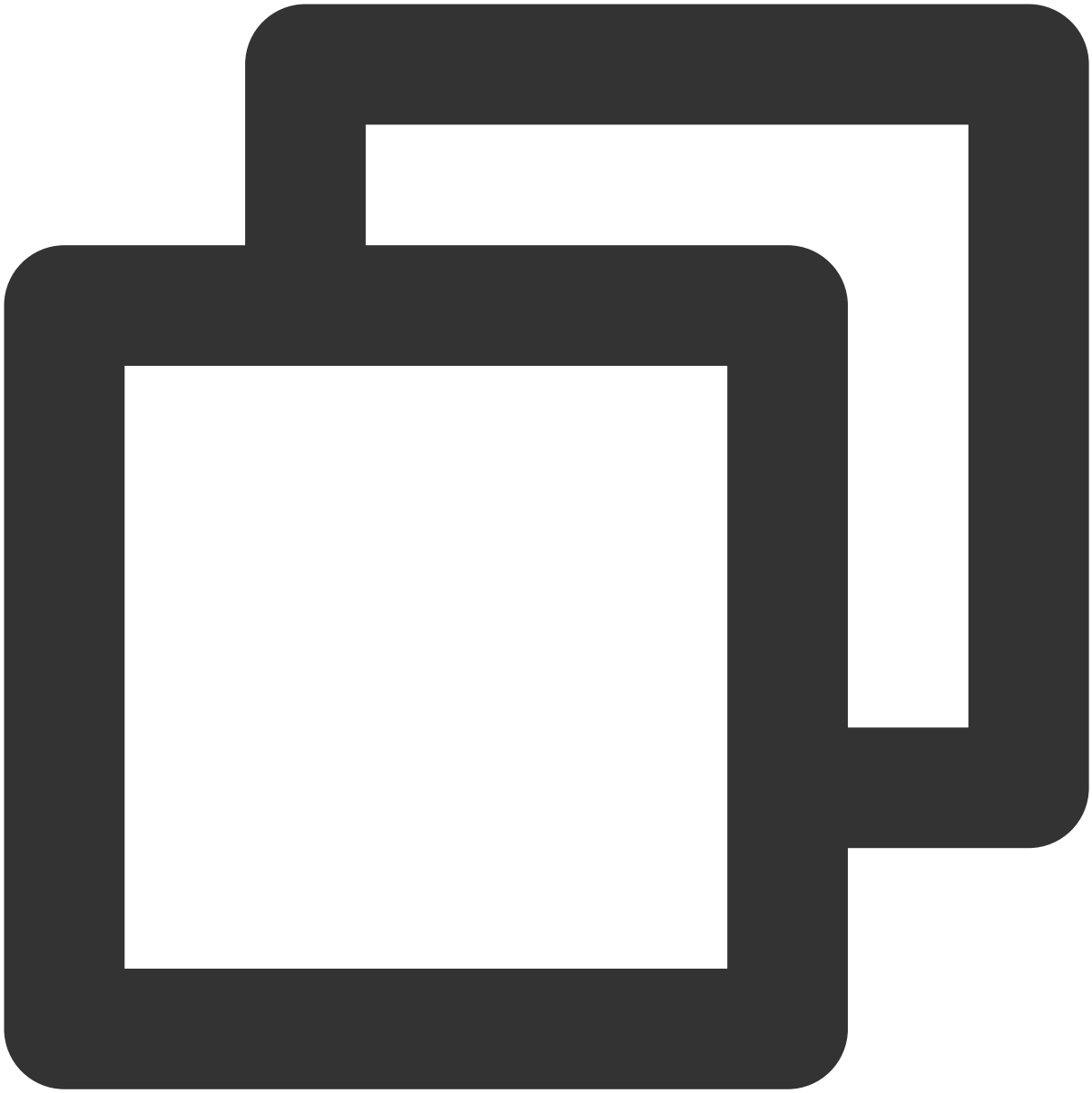
注意：

此步骤需要您的云服务器在安全组中配置来源为 **all**，端口协议为 **TCP:80** 的入站规则。安全组配置请参见 安全组。

在您本地的浏览器中输入 `https://115.xxx.xxx.xxx/` （其中 `115.xxx.xxx.xxx` 为您的云服务器公网 IP 地址），出现下列画面表示 Apache 启动成功。

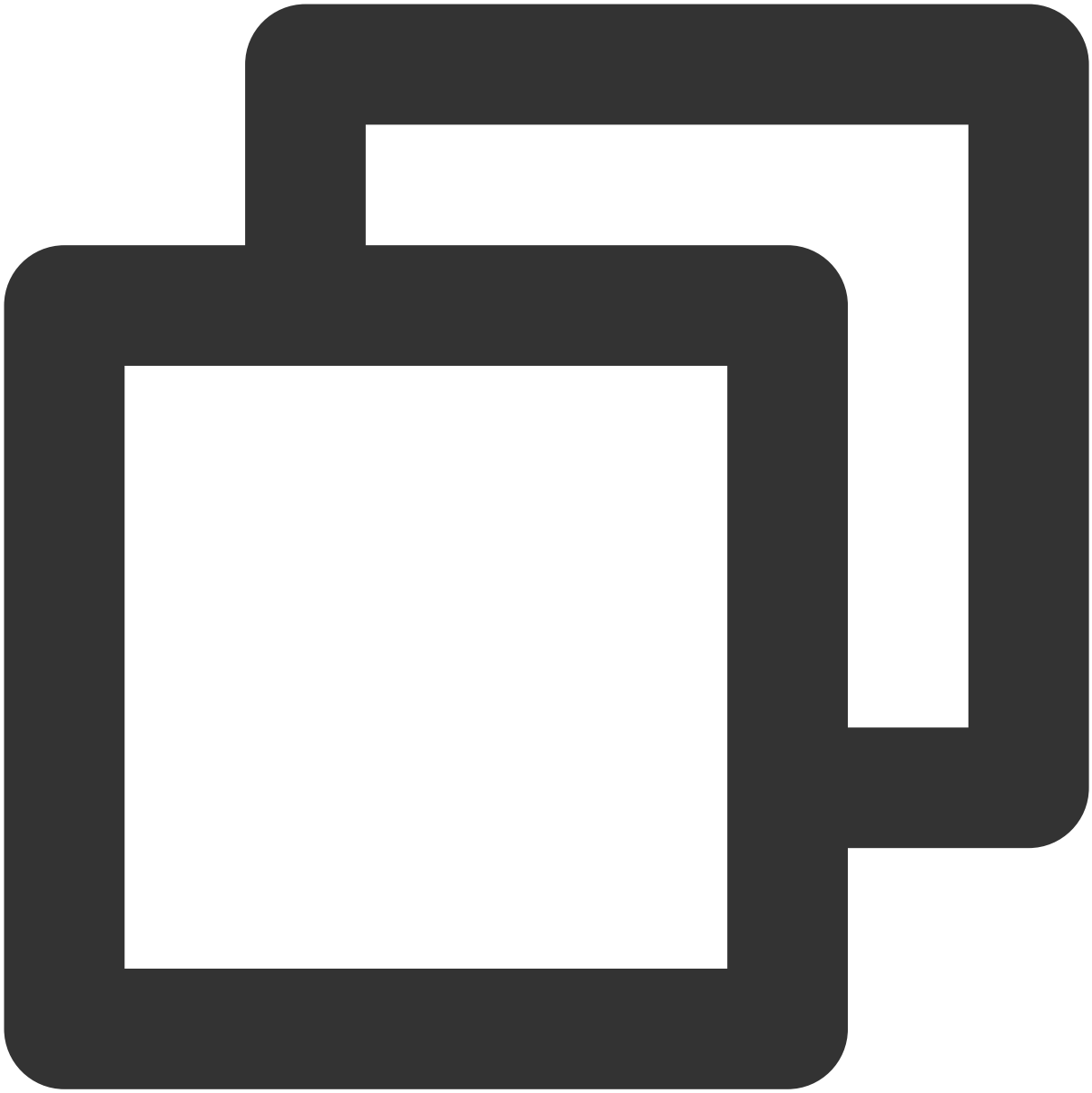
安装 PHP

1. 在云服务器实例中使用 `yum` 安装 PHP 及其扩展。



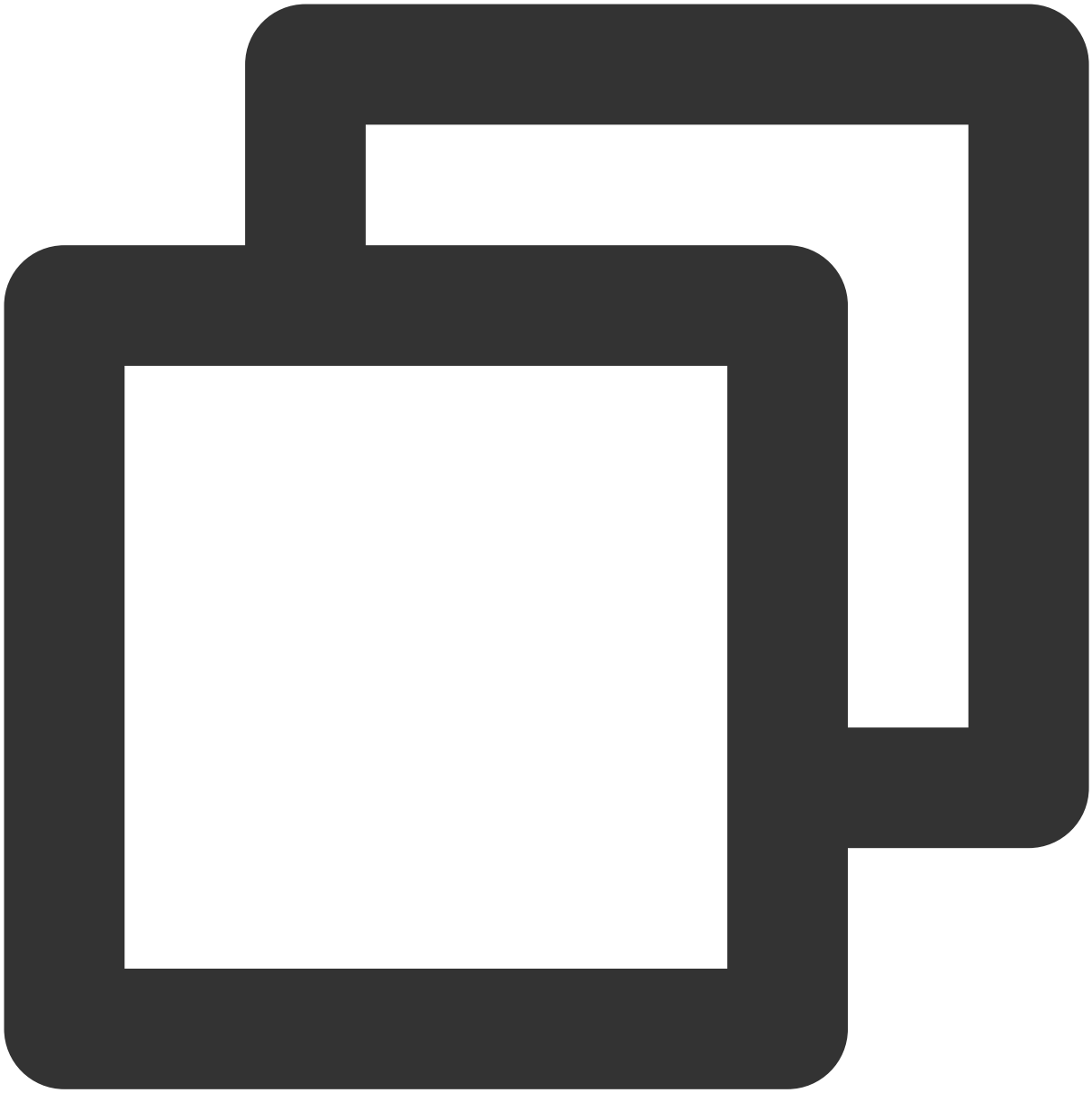
```
yum install php php-dom php-gd php-mysql php-pdo -y
```

2. 在云服务器 `/var/www/html` 目录下创建一个 info.php 文件来检查 PHP 是否安装成功，示例代码如下：



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

3. 重启 Apache 服务。



```
service httpd restart
```

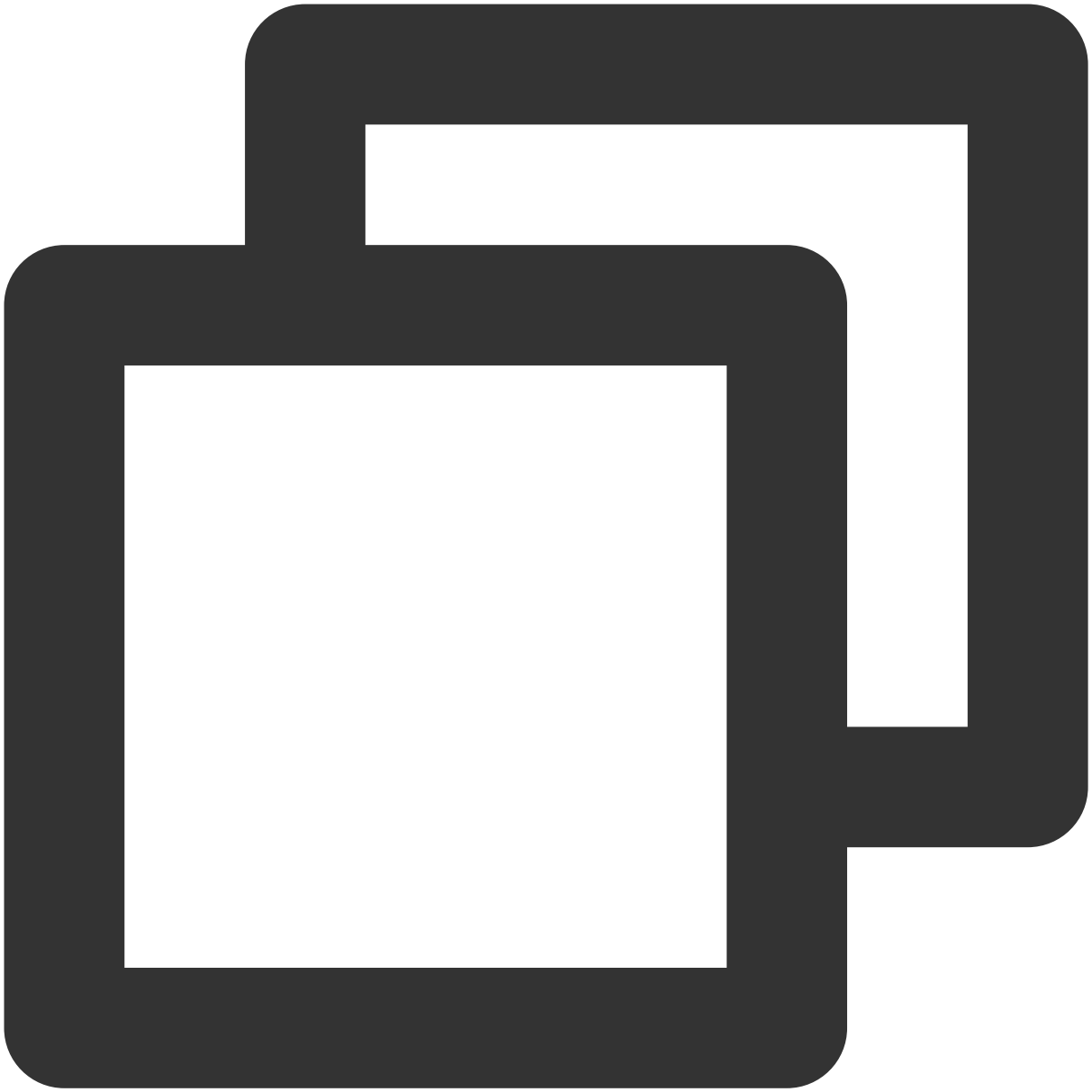
4. 在您本地的浏览器中输入 `https://115.xxx.xxx.xxx/info.php` （其中 `115.xxx.xxx.xxx` 为您的云服务器公网 IP 地址），出现下列画面表示 PHP 安装成功。

PHP Version 5.4.16

System	Linux VM_165_193_centos 3.10.0-327.36.3.el7.x86_64 #1 SMP Mon Oct 24 16:09:20 UTC 2016 x86_64
Build Date	Nov 6 2016 00:30:05
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/curl.ini, /etc/php.d/fileinfo.ini, /etc/php.d/json.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/sqlite3.ini, /etc/php.d/zip.ini
PHP API	20100412
PHP Extension	20100525
Zend Extension	220100525
Zend	API220100525,NTS

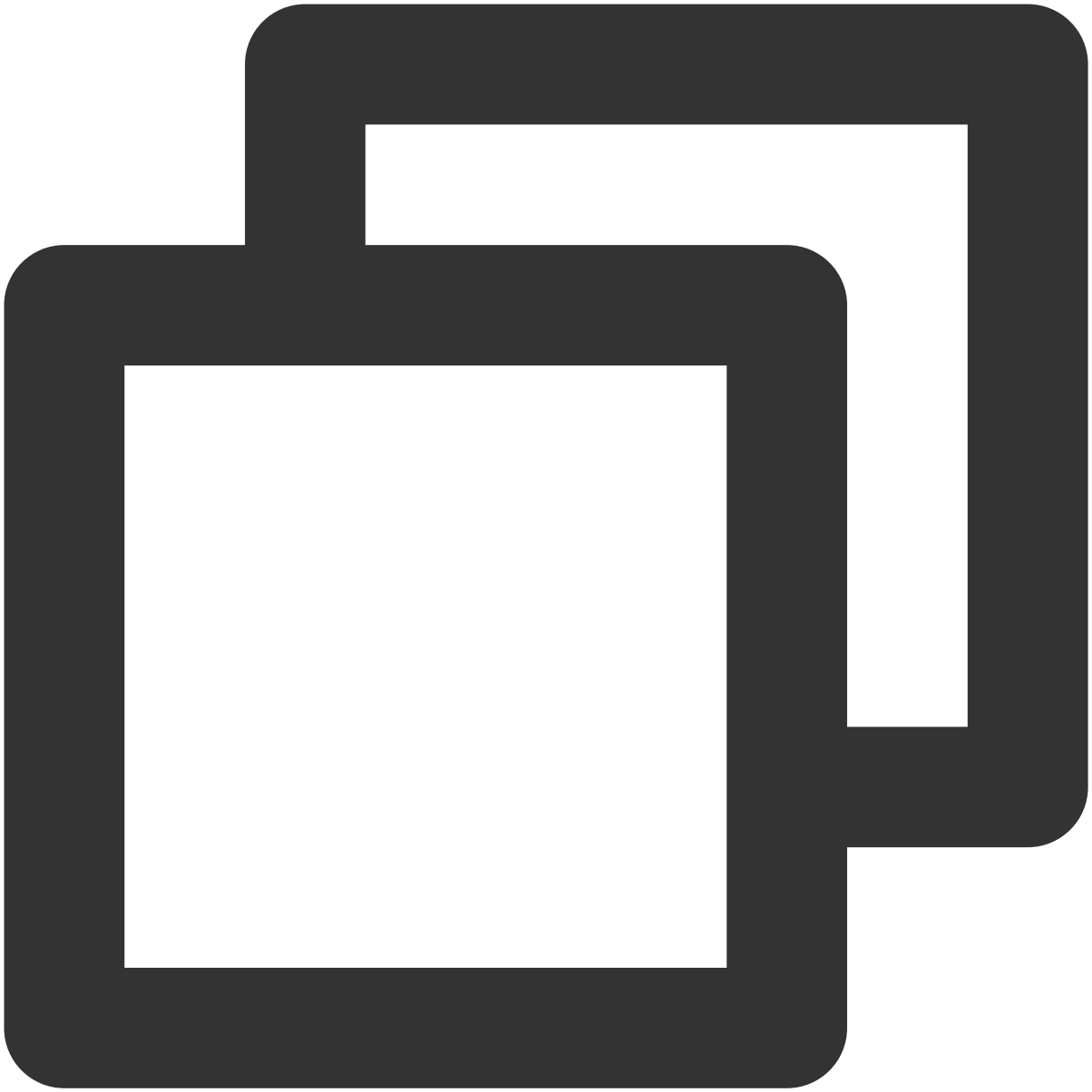
安装 Drupal 服务

1. 下载 Drupal 安装包。



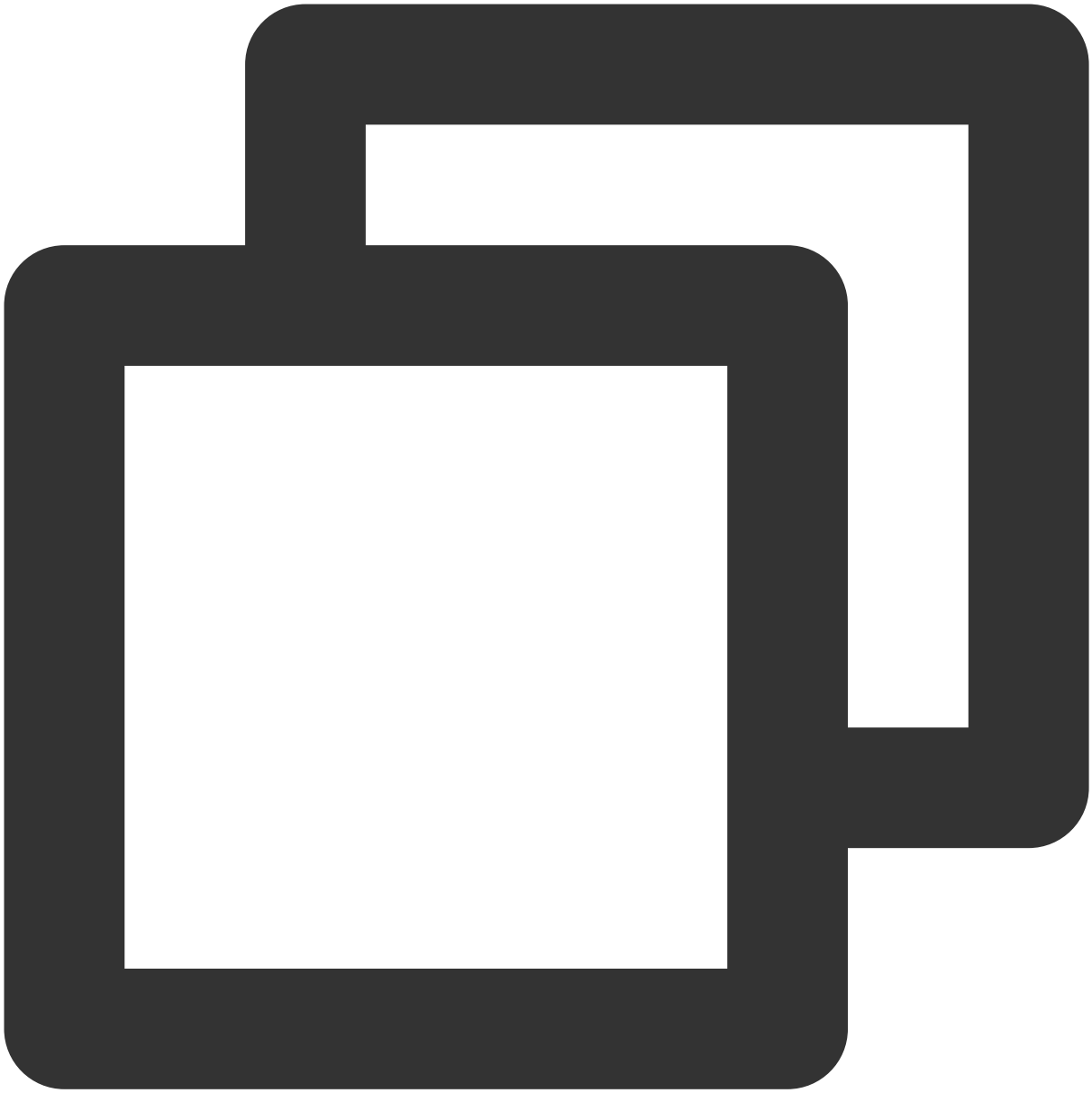
```
wget http://ftp.drupal.org/files/projects/drupal-7.56.zip
```

2. 解压到网站根目录。



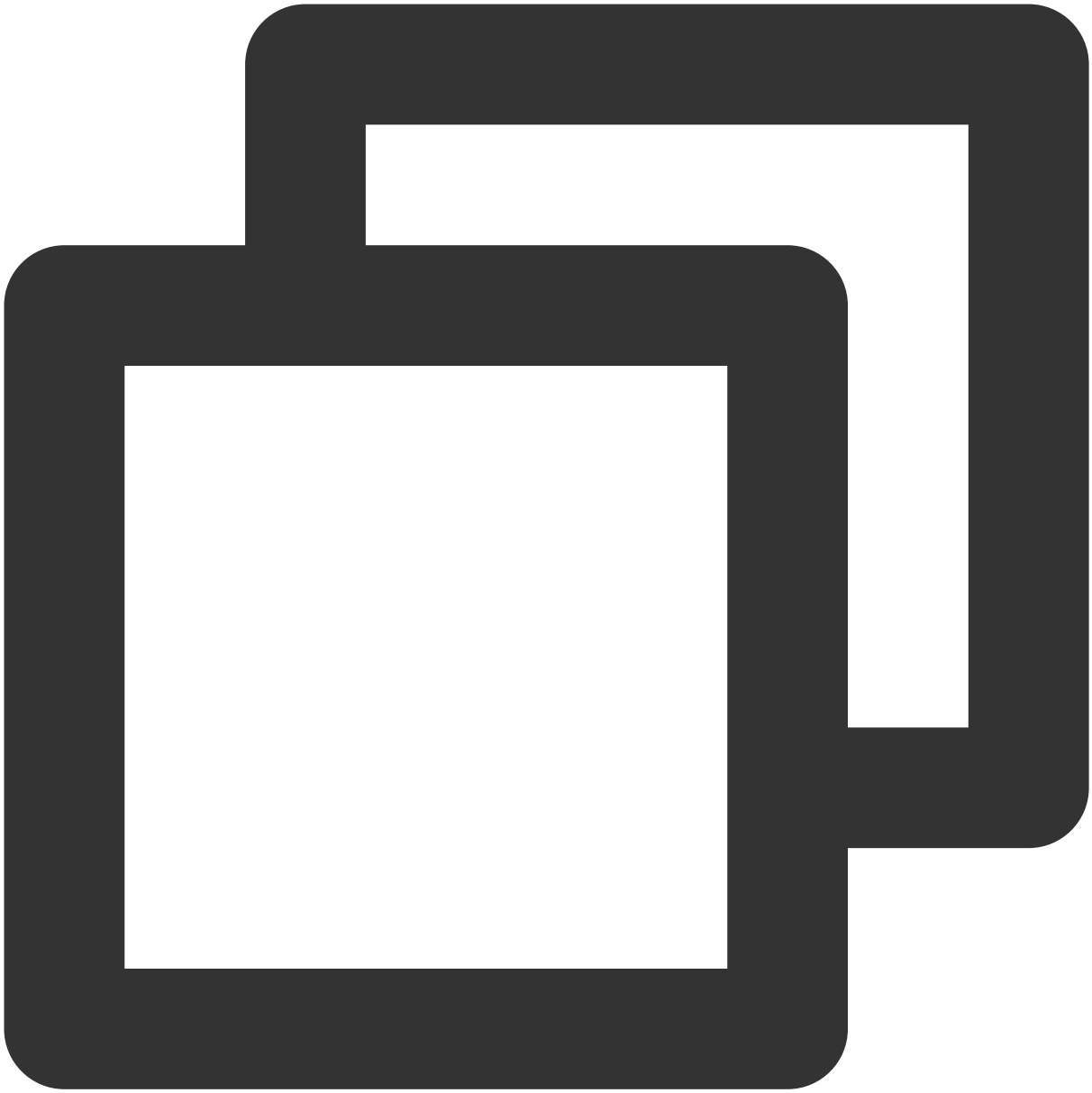
```
unzip drupal-7.56.zip
mv drupal-7.56/* /var/www/html/
```

3. 下载中文翻译包。



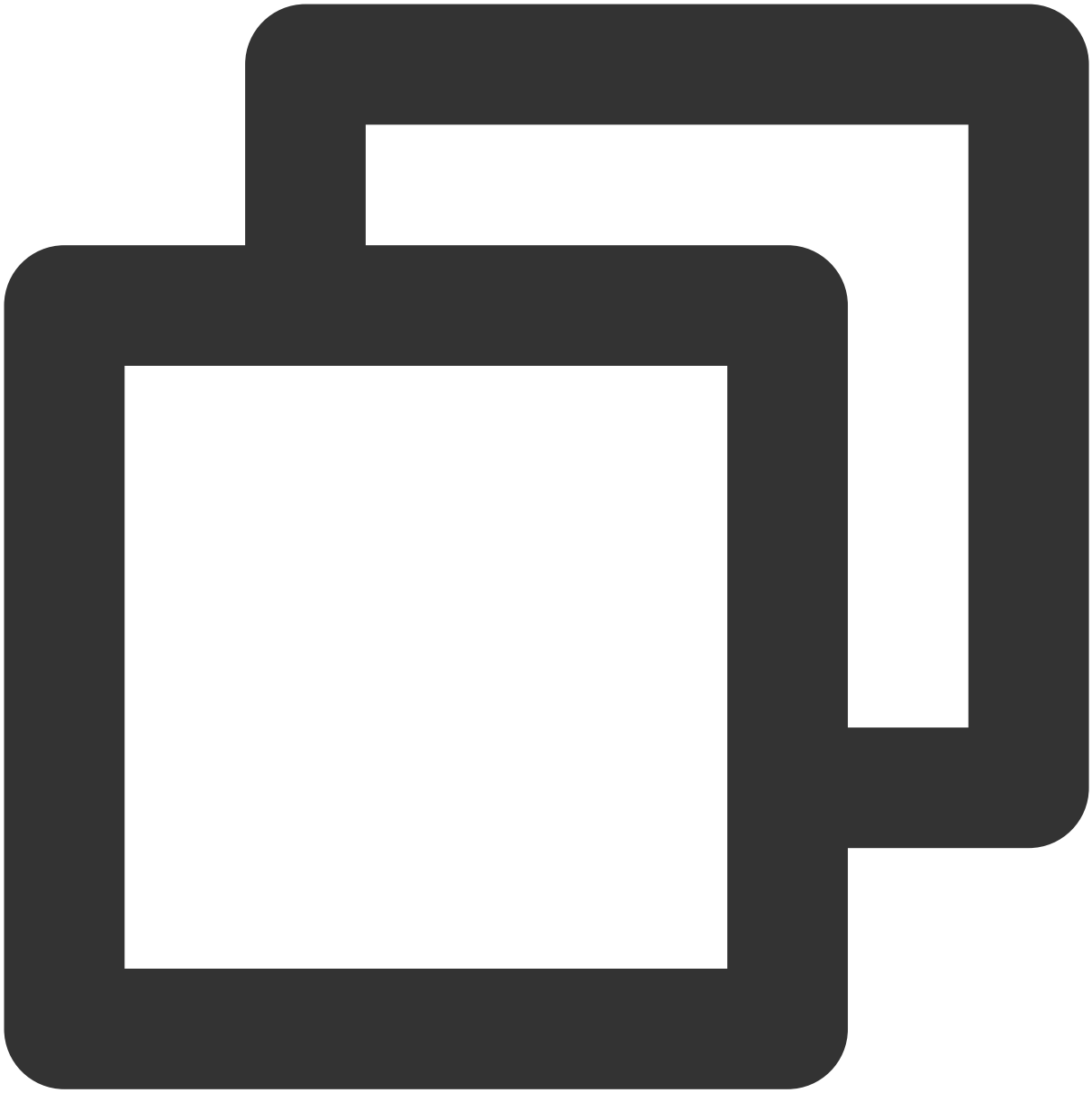
```
cd /var/www/html/
wget -P profiles/standard/translations http://ftp.drupal.org/files/translations/7.x
```

4. 修改 `sites` 目录属主属组。



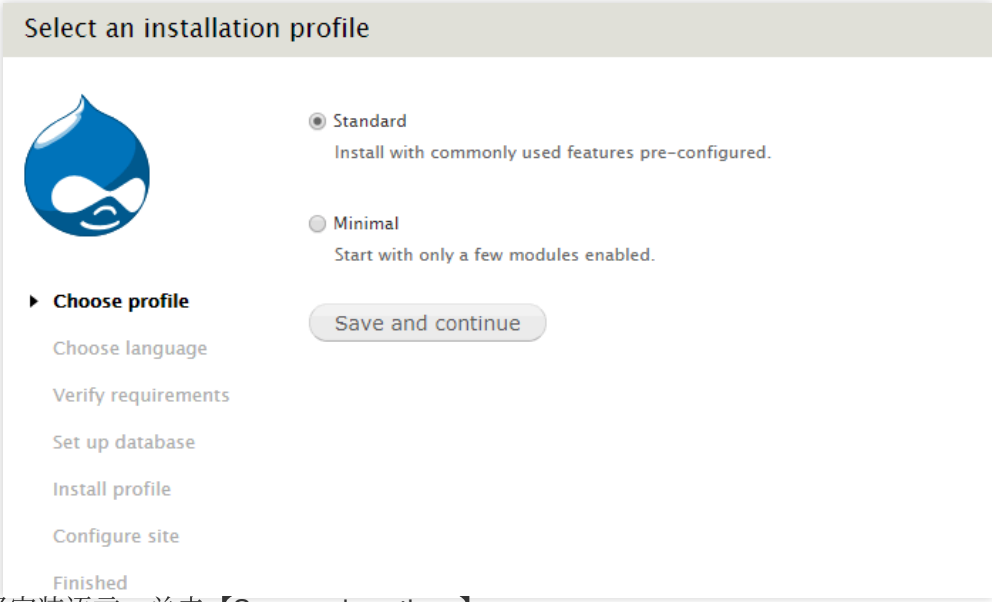
```
chown -R apache:apache /var/www/html/sites
```

5. 重启 Apache 服务。



```
service httpd restart
```

6. 在您本地的浏览器中输入 `https://115.xxx.xxx.xxx/` （其中 `115.xxx.xxx.xxx` 为您的云服务器公网 IP 地址），进入 Drupal 安装界面。选择安装版本，单击【Save and continue】。



7. 选择安装语言，单击【Save and continue】。
8. 设置数据库，输入您在**安装 mariadb 服务**中配置的数据库信息。
9. 输入站点信息。
10. 完成 Drupal 的安装。
11. 后续可以访问 `https://115.xxx.xxx.xxx/` （其中 `115.xxx.xxx.xxx` 为您的云服务器公网 IP 地址）对网站进行个性化设置。

构建全场景高可用架构

最近更新时间：2023-07-26 15:45:24

云数据库 MySQL 基于跨可用区部署、异地备份保障与配套生态工具，搭配数据库代理和弹性 CPU 等多项功能，全方位保障您的业务安全稳定运行。本文为您介绍通过云数据库 MySQL 构建全场景高可用架构。

背景

数据库在企业的核心业务中扮演着至关重要的角色，数据是企业的基础资源和生命线。因此，为保证企业的稳定生产运营，一个高可用的数据库架构是不可或缺的。对于企业来说，如果数据库出现宕机、数据丢失或不可用等问题，将会产生重大的影响和经济损失。因此，云数据库 MySQL 推出全场景高可用性架构（All-Scenario High Availability Architecture，AS-HAA），全方位全流程保障业务稳定运行。

前提条件

已注册腾讯云账号并完成实名认证。

- 注册腾讯云账号
- 完成实名认证

优势

- 高稳定性：快速扩容、均衡负载、自动弹性扩缩容、就近访问能力提供扩展性强、平稳运行、低网络延时的环境。
- 高可用性：全组件实现跨可用区/跨地域部署，降低单点故障风险，通过备份、数据同步等机制，减少业务停机时间和数据丢失率。
- 高数据处理能力和响应速度：数据库代理将负载分布到多个数据库节点上，从而提高数据处理能力和响应速度。
- 高故障诊断和恢复效率：数据库智能管家 DBbrain 7*24h监控，帮助 DBA 快速定位问题解决问题；同时，备份、数据同步等机制也有助于故障恢复和数据恢复，缩短故障修复时间。

构建全场景高可用架构指引

步骤一：高可用架构部署

一、创建三节点架构实例

1. 登录 MySQL 购买页，完成**基础配置**和**实例配置**，单击**下一步**，设置网络和数据库。

基础配置

- **计费模式**：支持包年包月和按量计费。
- **地域**：选择您业务需要部署 MySQL 的地域。建议您选择与云服务器同一个地域，不同地域的云产品内网不通，购买后不能更换。
- **数据库版本**：云数据库 MySQL 目前支持以下版本：MySQL 8.0、MySQL 5.7、MySQL 5.6、MySQL 5.5（需白名单）。
- **引擎**：支持选择 InnoDB 和 RocksDB 引擎。
- **架构**：选择三节点。
- **硬盘类型**：云数据库 MySQL 支持本地盘和云盘两种硬盘类型。
- **可用区**：主可用区和备可用区选择不同的可用区。

实例配置

- **筛选**：快捷筛选所需实例的 CPU 和内存，默认展示全部 CPU、全部内存。
- **类型**：提供通用型与独享型两种实例类型，详情请参见 隔离策略。
- **实例规格**：根据业务需要选择对应规格。
- **硬盘**：用于存放 MySQL 运行时所必须的文件，选择硬盘空间大小。

2. 完成**网络和其他**、**数据库设置**配置，单击**下一步**，确认配置信息。

网络和其他

- **网络**：支持私有网络环境，可选择实例的所属网络和子网。
- **自定义端口**：数据库的访问端口，默认为3306。
- **安全组**：安全组创建与管理请参见 云数据库安全组。

> 说明：
> 安全组入站规则需要放通 MySQL 实例的3306端口。MySQL 内网默认端口为3306，同时支持自定义端口，若修改过默认端口号，安全组中需放通 MySQL 新端口信息。

- **指定项目**：选择数据库实例所属的项目，缺省设置为默认项目。
- **标签**：便于分类管理实例资源。
- **告警策略**：创建告警用于在云产品状态改变时触发警报并发送相关消息。

数据库设置

- **实例名**：可选择创建后命名或立即命名。
- **数据复制方式**：提供异步复制、半同步复制、强同步复制三种方式。
- **参数模板**：除提供的系统参数模板外，您也可以创建自定义参数模板，请参见 使用参数模板。
- **字符集**：支持 LATIN1 、GBK、UTF8 、UTF8MB4 字符集，默认字符集编码格式是 UTF8。
- **排序规则**：实例字符集为系统数据提供的排序规则，即区分大小写属性和重音属性。
- **表名大小写敏感**：表名是否大小写敏感。

> 注意：
> MySQL 8.0 版本仅支持在购买页设置表名大小写敏感，其他版本可通过修改 lower_case_table_names 参数来设置，操作方法参见 设置实例参数。

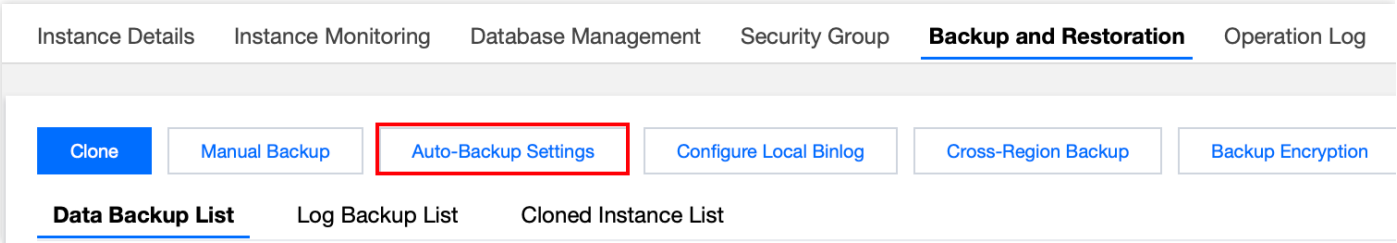
- **密码复杂度**：支持设置密码复杂度以提升数据库安全性，默认为关闭。
- **root 密码**：新创建的 MySQL 数据库的用户名默认为 root，此处用来设置该 root 账号的密码。选择**创建后设置**时，可在创建完实例后再 重置密码。

3. 确认所选配置（如需修改，可单击**编辑**回到对应步骤进行调整），阅读并勾选服务条款，确认购买时长和数量后单击**立即购买**。
4. 支付完成后，返回实例列表，会看到实例显示**发货中**（大概需要3min - 5min中，请耐心等待），待实例状态变为**运行中**，即可进行正常操作。

二、设置本地备份和跨地域备份

设置本地备份

1. 登录 MySQL 控制台，在实例列表，单击**实例 ID** 进入管理页面，选择**备份恢复** > **自动备份设置**。

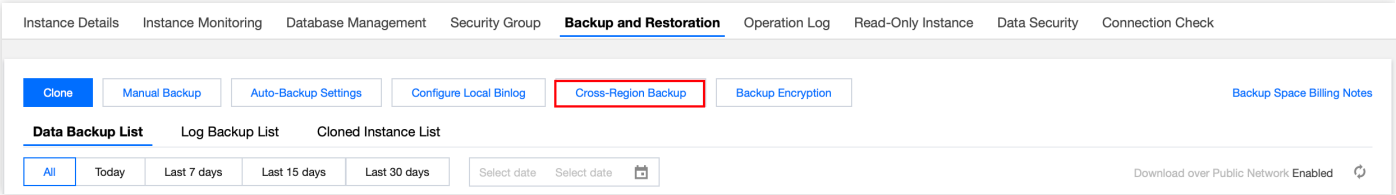


2. 在弹出的备份设置对话框，选择各备份参数，单击**确定**。

开启跨地域备份

1. 登录 MySQL 控制台，在实例列表，单击**实例 ID** 或**操作**列的**管理**，进入实例管理页面。

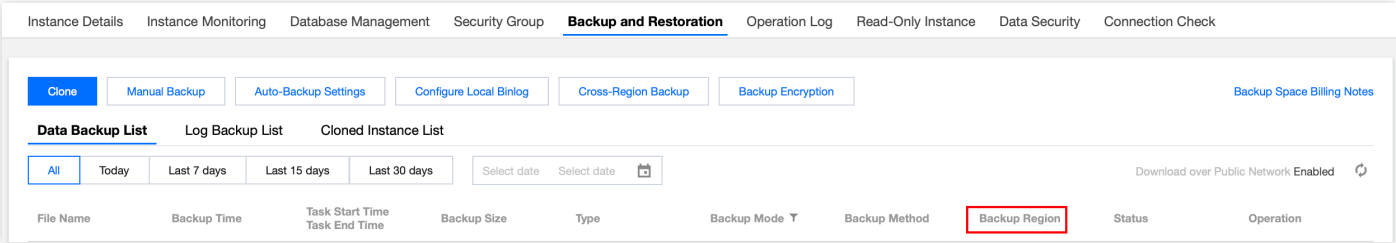
2. 在实例管理页面，选择**备份恢复** > **跨地域备份**。



3. 在跨地域备份设置窗口完成配置后，单击**确定**开启跨地域备份， 阅读并**勾选**跨地域备份空间收费说明，单击**确定**。

- 跨地域备份开关：默认为关闭，这里选择开启。
- 备份 binlog：当开启跨地域备份时自动打开，开启后可单独关闭备份 binlog。
- 备份地域：备份地域最少选择一个与主实例不同的地域，最多可选择两个。
- 备份地域保留时长：默认为7天，支持设置为3天 - 1830天，到期后备份集会自动删除。

4. 跨地域备份完成后，备份会同步到目标地域，可在源实例备份列表查询。

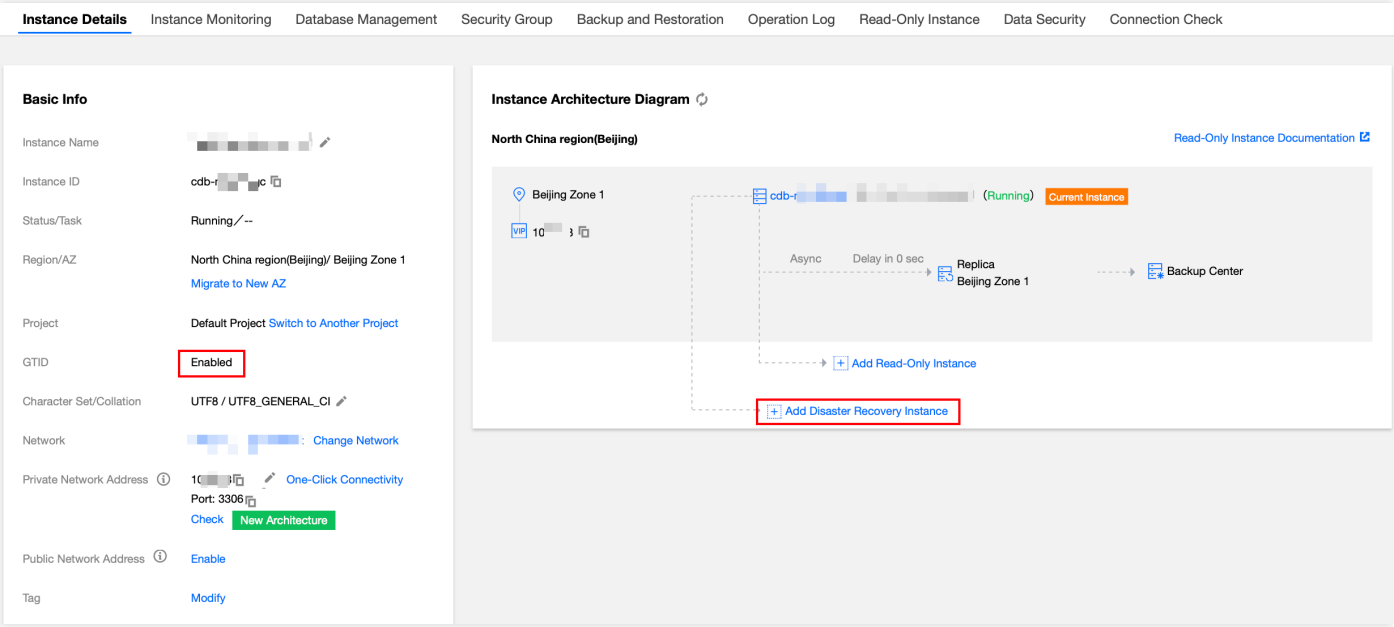


跨地域备份的文件在**备份地域**项下面会展示所有您选择备份的地域。

三、创建异地灾备实例

1. 登录 MySQL 控制台，在实例列表，单击**实例 ID** 或**操作**列的**管理**，进入详情页。

2. 在实例详情页的基本信息中确认 GTID 功能开启，在实例架构图中单击**添加灾备实例**，进入灾备实例购买页。



3. 在购买页中，设置灾备实例的**计费模式**、**地域**等基本信息，地域请选择与步骤一中 创建的三节点架构实例 不同的地域。

说明：

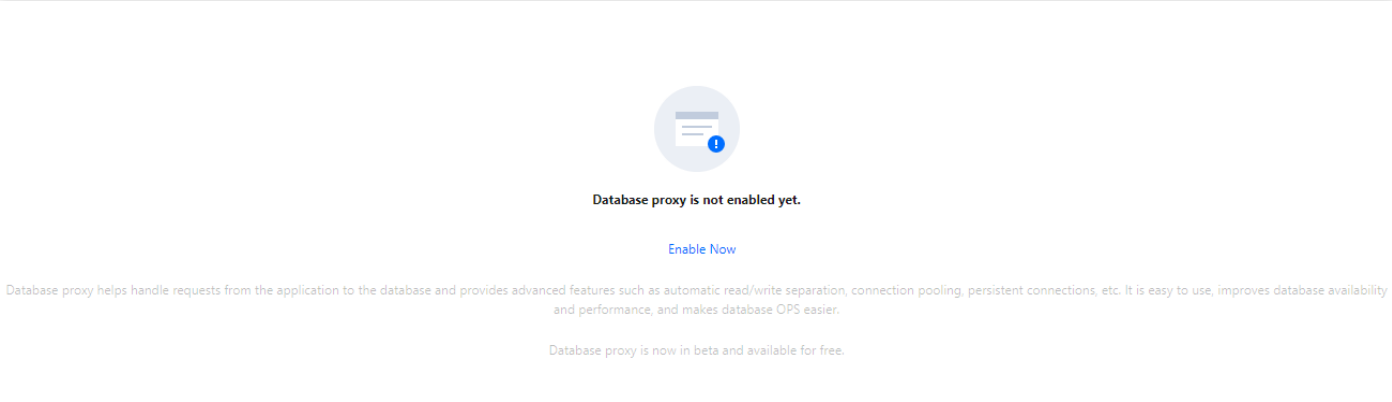
- 创建时长受数据量的影响，期间主实例的控制台操作会被锁定，请妥善安排。
- 同步策略为立即同步，即创建完灾备实例后会立即同步数据。
- 暂只支持整个实例的数据同步，请确保磁盘空间充足。
- 请确保主实例状态为运行中并且没有任何运行相关变更任务执行，如升降配、重启等，否则同步任务有可能失败。

4. 确认无误后，单击**立即购买**，待灾备实例发货。
5. 返回实例列表，待实例状态变为**运行中**，即可进行后续操作。

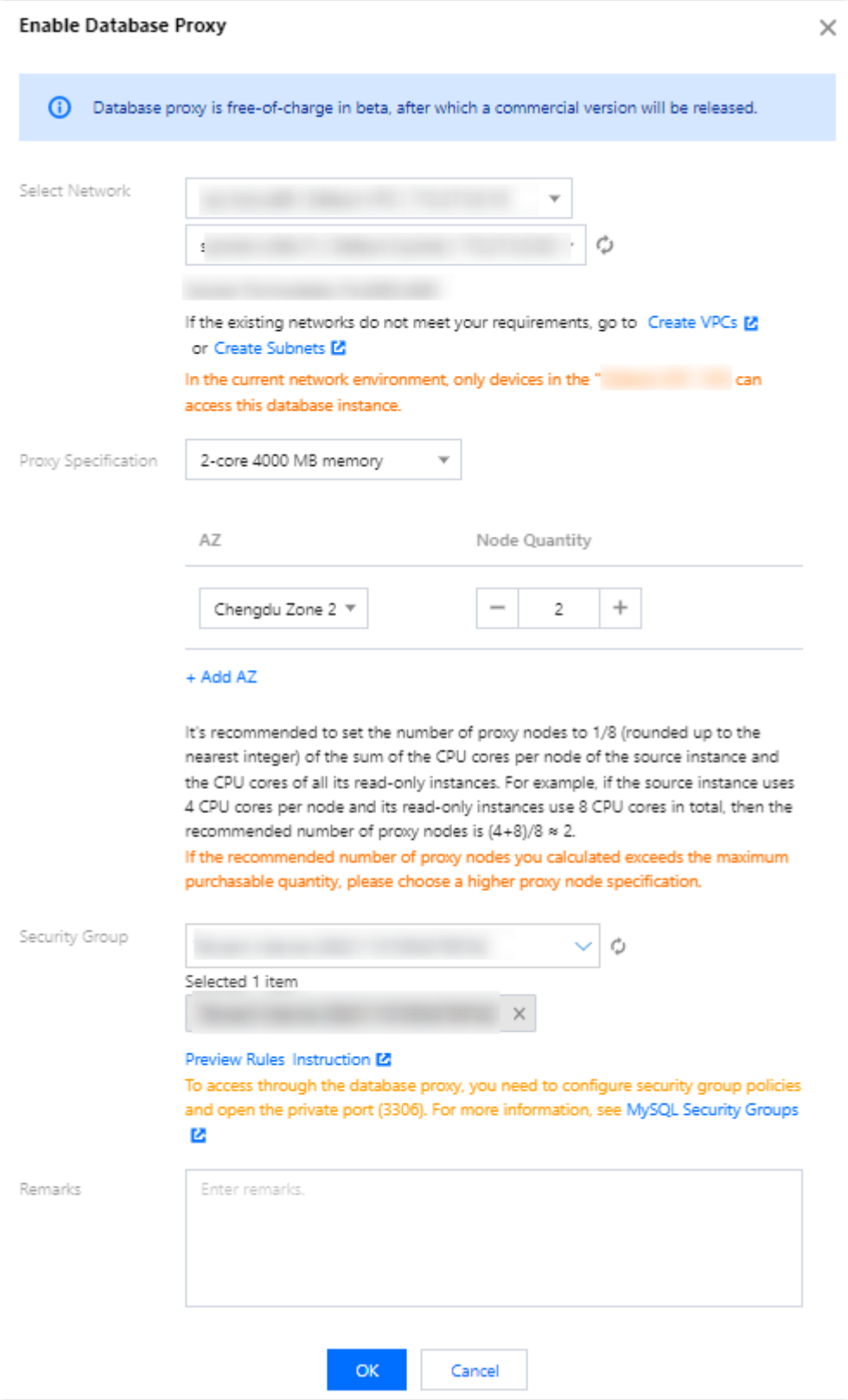
步骤二：开通数据库代理

1. 登录 MySQL 控制台，在实例列表，选择需要开启代理的主实例，单击实例 ID 或**操作**列的**管理**，进入实例管理页面。

2. 在实例管理页面，选择**数据库代理**页，单击**立即开启**。



3. 在弹出的对话框，完成如下配置，单击**确定**。



参数	说明
网络	选择数据库代理的网络，仅支持私有网络 VPC。

参数	说明
代理规格	支持选择规格为2核4000MB内存、4核8000MB内存、8核16000MB内存。
可用区及节点个数	1. 选择数据库代理可用区，支持单击**新增可用区**来多选，可选择的可用区数量与当前地域可选可用区数量相关，最多支持选择三个可用区。 2. 选择节点个数，推荐的代理节点个数为主实例和只读实例 CPU 核数的之和的1/8（向上取整），例如主实例为4核 CPU，只读实例为8核 CPU，则推荐代理数量 = (4 + 8) / 8 ≈ 2。 注意 i. 如果所选数据库代理与主实例不在同一可用区，通过数据库代理连接时，写入性能可能会下降。 ii. 若计算推荐节点个数后所需代理节点数量超过购买限制，建议选择更高规格代理。
安全组	默认选择的安全组与主实例保持一致，也可根据需要选择已有安全组或新建安全组。 注意 访问数据库代理需要开通配置安全策略，放通内网访问端口（当前内网端口为：3306），具体详见 MySQL 安全组配置。
备注	非必填项，可为要开通的数据库代理服务进行备注。

4. 开通成功后，支持通过**数据库代理**页 > **概览** > **连接地址**下的内网访问地址直接访问数据库代理，支持对数据库代理地址的访问策略进行配置，在**数据库代理** > **概览** > **连接地址**下找到目标访问地址，单击其**操作**列的**调整配置**。
5. 在跳转的窗口下，设置具体策略的配置，单击**确定**。

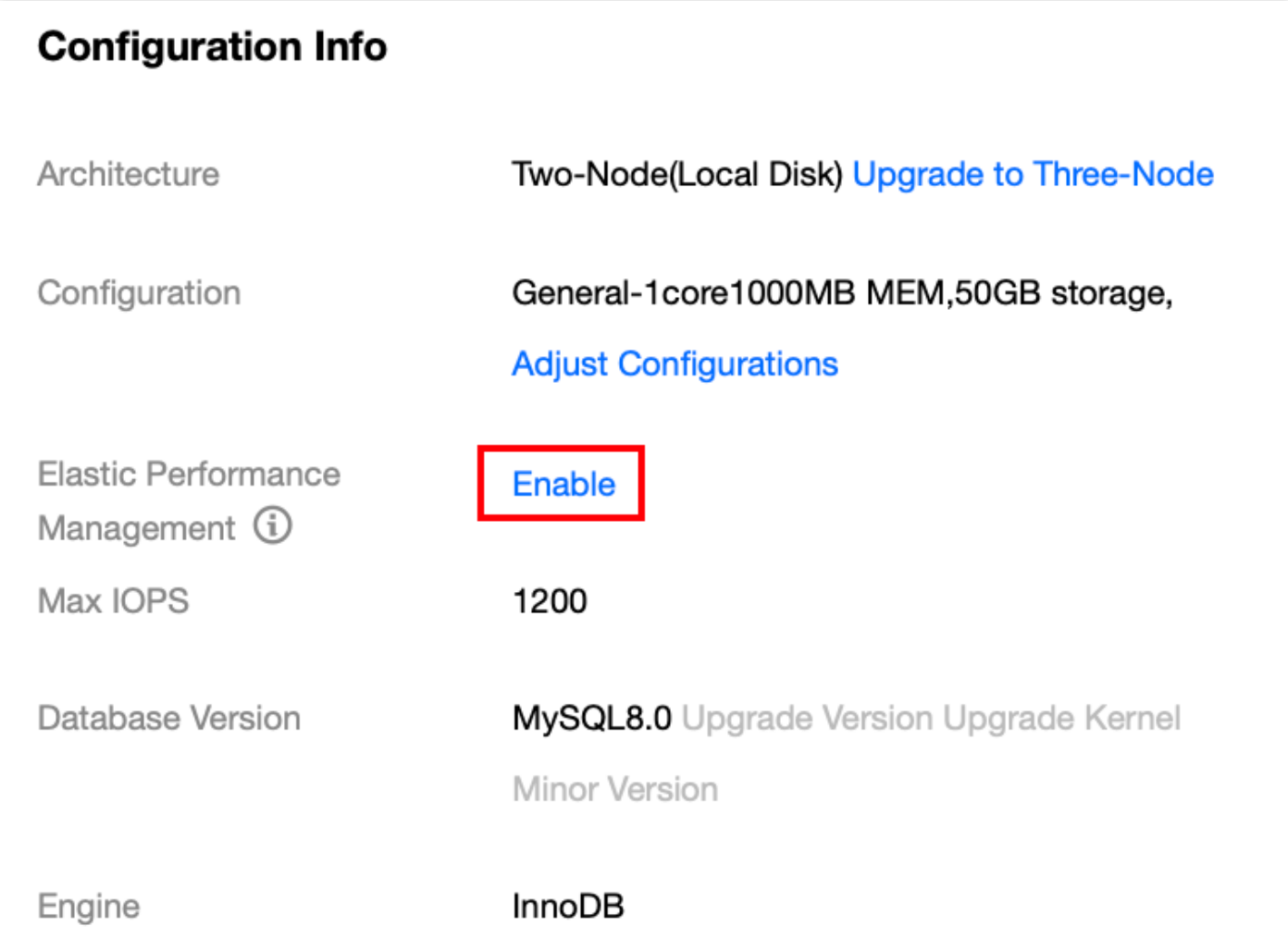
参数	说明
读写属性	修改此代理访问地址的读写属性，支持选择读写分离或只读。
只读实例延迟剔除	设置只读实例延迟剔除策略，此项开启，可设置延迟剔除阈值和只读实例最小保留数。无论此项是否启用，只读实例故障时均会尝试剔除和恢复。 ◦ 延迟剔除阈值：输入大于1或等于1的整数，单位为秒。 ◦ 只读实例最小保留数：数量与主实例拥有只读实例数相关，设置为0时，当只读节点全部被剔除后，所有访问会转发到主实例上，直至只读实例重新加入。
连接池状态	连接池功能主要用于减少短连接业务频繁建立新连接带来的实例负载。此项开启后，可选择支持的连接池类型，目前默认仅支持会话级连接池。

参数	说明
事务拆分	设置是否开启，开启后，在一个事务中拆分读和写到不同的实例上去执行，读请求转发到只读实例，降低主实例负载。
读权重分配	支持选择系统自动分配或自定义，如开通数据库代理时配置了多个可用区，则支持对不同可用区下的代理节点访问数据库的权重进行分别配置。
故障转移（读写属性为读写分离）	设置是否开启，开启后，数据库代理出现故障时，连接地址将会路由到主实例。
自动添加只读实例	设置是否开启，开启后，若您购买新的只读实例，会自动添加到数据库代理中。 ◦ 当读权重为系统自动分配时，新购只读实例按照规格大小默认权重分配。 ◦ 当读权重为自定义时，新购只读实例默认加入时权重为0，可通过数据库代理页，连接地址下的调整配置来修改。

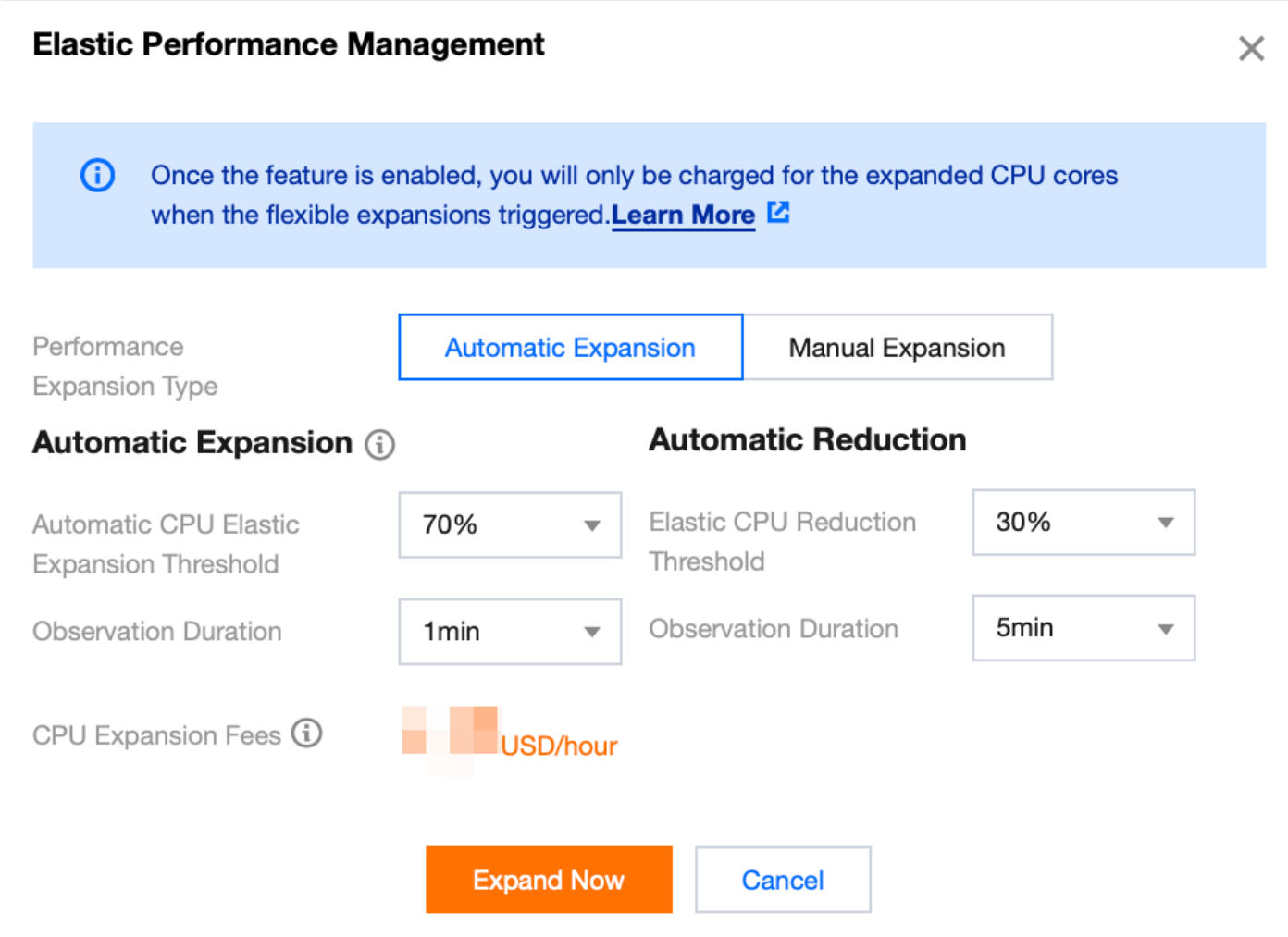
步骤三：开启性能弹性管理-自动扩缩容

1. 登录 MySQL 控制台，在实例列表，单击实例 ID 或操作列的**管理**，进入实例详情页。

2. 在**实例详情** > **配置信息** > **性能弹性管理**后，单击**配置**。



3. 在性能弹性管理窗口下，完成如下配置，确认扩容费用，单击**立即扩容**。



参数	说明
性能扩容类型	选择自动扩容。
CPU 自动弹性扩容阈值	设置 CPU 平均利用率触发自动弹性扩容的阈值，系统支持选项为70%、80%、90%。
观测周期	设置观测周期，系统支持选项为1分钟、3分钟、5分钟、10分钟、15分钟、30分钟，表示指定时间周期内，系统会观测实例的 CPU 平均利用率是否达到设置的扩容阈值，如达到，则会触发系统自动弹性扩容。
CPU 自动弹性回缩阈值	设置 CPU 平均利用率触发自动弹性回缩的阈值，系统支持选项为30%、20%、10%。

参数	说明
观测周期	设置观测周期，系统支持选项为5分钟、10分钟、15分钟、30分钟，表示指定时间周期内，系统会观测实例的 CPU 平均利用率是否达到设置的回缩阈值，如达到，则会触发系统自动弹性缩容。

4. 待实例状态/任务从**配置弹性扩容策略中**变为**运行中**，即成功开启自动扩容。

> 说明：
> 开启自动扩容后，如需修改性能弹性扩容策略，您可在**实例详情** > **配置信息** > **性能弹性管理**后单击**修改**，进行重新配置。

生态工具使用指引

DBbrain 智能管家监测定位问题

运用数据库智能管家，实现数据库性能监控、安全检测和优化诊断，通过智能分析和建议，协助您快速解决数据库性能和安全问题，提升数据库效率。

分类	诊断优化功能	描述
诊断分析	异常诊断	为用户的数据库实例提供实时的性能监控、健康巡检、故障诊断和优化，让用户既可以直观地感知数据库实例实时的运行状况，也可以定位实时出现的性能异常，并根据优化建议进行系统优化。
	慢 SQL 分析	对慢 SQL 的性能进行分析，并给出优化建议。
	空间分析	可查看实例空间的使用率，包括数据空间和日志空间的大小、空间使用率的日均增长量、预估的可用天数，以及查看实例下表空间、库空间所占用的空间详情及变化趋势。
	SQL 优化	为用户提供一键优化 SQL 语句功能，并给出对应执行计划解析和优化建议。
	审计日志分析	对 SQL 的性能进行分析，并针对低质量 SQL 结合索引情况、库表设计，给出优化建议。
定位和处理数据库会话连接问题	KILL 会话	KILL 当前会话和持续 KILL 会话。
	SQL 限流	通过创建 SQL 限流任务，自主设置 SQL 类型、最大并发数、限流时间、SQL 关键词，来限制数据库的请求访问量和 SQL 并发量，进而达到服务的可用性，不同的任务之间不会发生冲突。

分类	诊断优化功能	描述
	热点更新保护	针对于频繁更新或秒杀类业务场景，大幅度优化对于热点行数据的 UPDATE 操作的性能。
数据库自治	自治服务	支持自动 SQL 限流与异常 SQL KILL 功能，当满足触发条件时，自动触发 SQL 限流和异常 SQL KILL 自治任务，解决数据库异常问题，全程无需人工干预。

混沌演练平台模拟异常场景

混沌演练平台 可以帮助您模拟各种突发事件和异常场景，提高系统的容错能力和可靠性。搭建好真实的数据库架构后，平台会对业务进行全方位的压力测试，对传统的异常场景进行模拟并进行自动化部署管理，从而能够更好地发现和诊断系统中的潜在问题，以提升业务的质量和可靠性。
详细演练创建及过程可参见混沌演练快速入门。

相关文档

- 创建 MySQL 实例
- 备份数据库
- 跨地域备份
- 创建灾备实例
- 开通数据库代理
- 事物拆分
- 防闪断
- 设置数据库代理读写属性
- 性能弹性管理

通过 Python 语言使用 MySQL API
实例购买

最近更新时间：2023-09-13 16:04:36

API	描述
CreateDBInstance	创建云数据库实例（包年包月）
CreateDBInstanceHour	创建云数据库实例（按量计费）
DescribeDBInstances	查询实例列表
DescribeDBPrice	查询数据库价格

CreateDBInstance 创建云数据库实例（包年包月）

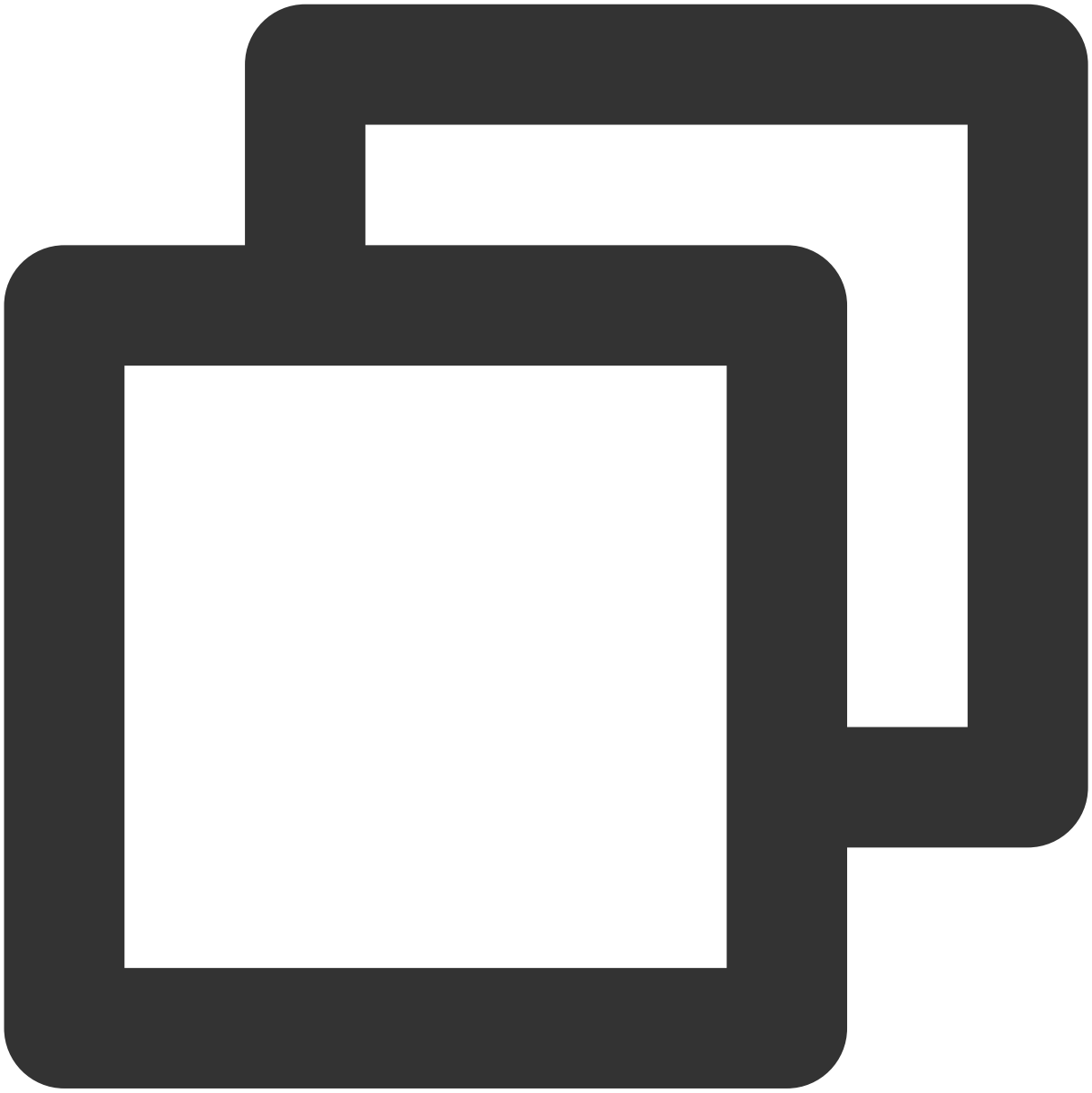
```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
import logging
import traceback
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models
```

```
'''购买主实例'''
def CreateDBInstancedemomaster():
    try:
        # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
        cred = credential.Credential("secretId", "secretKey")

        #实例化要请求产品(以cdb为例)的client对象
        client = cdb_client.CdbClient(cred, "ap-beijing")

        #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
        req = models.CreateDBInstanceRequest()
        req.Memory = 2000
        req.Volume = 120
        req.Period = 1
        req.GoodsNum =1
        req.Zone = "ap-beijing-1"
        req.Port = 3306
        #req.MasterInstanceId = "cdb-7ghaiocc"
        req.InstanceRole = "master"
        req.EngineVersion = "5.6"
        req.Password = "CDB@Qcloud"
        req.ProtectMode = 0
        req.InstanceName = "tencentcdb"
        req.SecurityGroup = ["sg-eq0hvlzp"]


        # 通过client对象调用想要访问的接口，需要传入请求对象
        resp = client.CreateDBInstance(req)
        # 输出json格式的字符串回包
        print(resp.to_json_string())


    except TencentCloudSDKException as err:
        msg = traceback.format_exc() # 方式1
        print (msg)




'''购买只读实例'''
def CreateDBInstancedemoro():
    try:
        # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
        cred = credential.Credential("secretId", "secretId")

        #实例化要请求产品(以cdb为例)的client对象
        client = cdb_client.CdbClient(cred, "ap-beijing")
```

```
        #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
        req = models.CreateDBInstanceRequest()
        req.Memory = 2000
        req.Volume = 200
        req.Period = 1
        req.GoodsNum =1
        req.Zone = "ap-beijing-1"
        req.Port = 3306
        req.InstanceRole = "ro"
        req.EngineVersion = "5.6"
        req.Password = "CDB@Qcloud"
        req.ProtectMode = 0
        req.DeployMode =1
        req.GoodsNum =2
        req.SlaveZone = "ap-beijing-1"
        req.ParamList = [{"name":"max_connections","value":"1000"},{"name":"lower_c
        req.BackupZone = "0"
        req.AutoRenewFlag =0
        req.MasterInstanceId ="cdb-bgr97hu0"
        req.RoGroup = {"RoGroupMode":"allinone","RoGroupName":"roweek"}
        req.InstanceName = "tencentcdbRO"


        # 通过client对象调用想要访问的接口，需要传入请求对象
        resp = client.CreateDBInstance(req)
        # 输出json格式的字符串回包
        print(resp.to_json_string())


    except TencentCloudSDKException as err:
        msg = traceback.format_exc() # 方式1
        print (msg)


'''购买灾备实例'''
def CreateDBInstancedemodr():
    try:
        # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
        cred = credential.Credential("secretId", "secretKey")

        #实例化要请求产品(以cdb为例)的client对象
        client = cdb_client.CdbClient(cred, "ap-shanghai")

        #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
        req = models.CreateDBInstanceRequest()
```

```
        req.Memory = 4000
        req.Volume = 200
        req.Period = 1
        req.GoodsNum =1
        #req.Zone = "ap-shanghai-2"
        req.Port = 3306
        req.InstanceRole = "dr"
        #req.MasterInstanceId
        req.EngineVersion = "5.6"
        req.Password = "CDB@Qcloud"
        req.ProtectMode = 0
        req.DeployMode =0
        #req.SlaveZone = "ap-guangzhou-3"
        req.ParamList = [{"name":"max_connections","value":"1000"},{"name":"lower_c
        req.BackupZone = "0"
        req.AutoRenewFlag =0
        #req.RoGroup = {"RoGroupMode":"alone","RoGroupName":"roweek"}
        #req.RoGroup = {"RoGroupName":"roweek"}
        #param = models.RoGroup()
        #param.RoGroupMode = "alone"
        #param.RoGroupName = "roweek"
        #param.MinRoInGroup = 1
        #req.RoGroup = [param]


        #ro = [{"roGroupMode":"allinone"},{"RoGroupName":"ro_www"}]
        #req.RoGroup = [ro]
        req.MasterInstanceId ="cdb-bgr97hu0"
        req.MasterRegion = "ap-beijing"
        #roGroup = [RoGroupMode="allinone", RoGroupName="weekro",RoOfflineDelay=1,M
        #req.RoGroup = [roGroup]
        req.InstanceName = "tencentcdbDR"


        # 通过client对象调用想要访问的接口，需要传入请求对象
        resp = client.CreateDBInstance(req)
        # 输出json格式的字符串回包
        print(resp.to_json_string())


    except TencentCloudSDKException as err:
        msg = traceback.format_exc() # 方式1
        print (msg)

#CreateDBInstancedemodr()
#CreateDBInstancedemoro()
#CreateDBInstancedemomaster()
```

CreateDBInstanceHour 创建云数据库实例（按量计费）



```
'''小时计费要冻结金额，需要账号有钱，如果账号余额为0，则不能购买'''
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
import logging
```

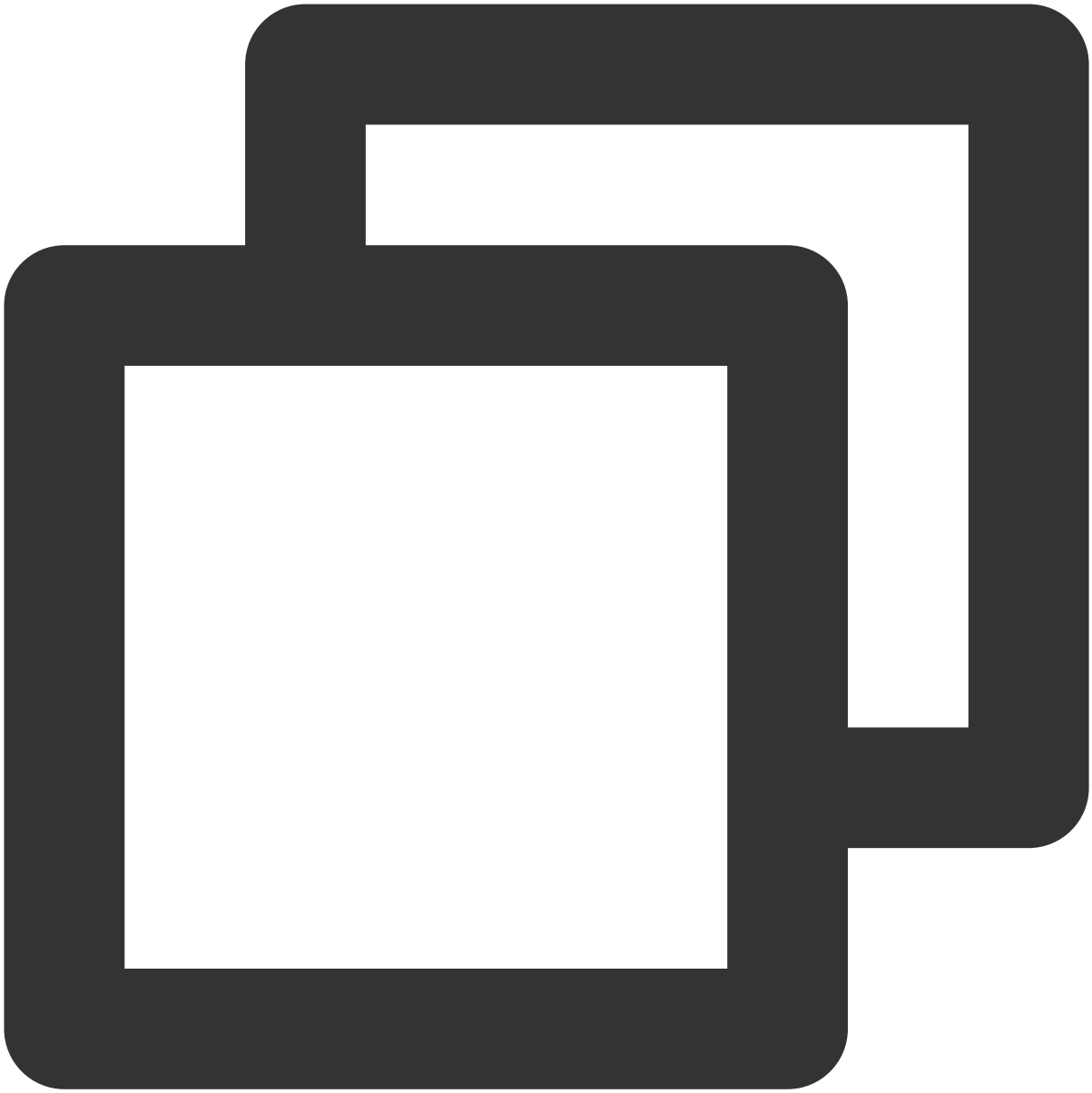
```
import traceback
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-beijing")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.CreateDBInstanceHourRequest()
    req.EngineVersion = "5.6"
    req.Zone = "ap-beijing-3"
    req.ProjectId = 0
    req.GoodsNum = 1
    req.Memory = 1000
    req.Volume = 50
    req.InstanceRole = "master"
    req.Port =3311
    req.Password = "CDB@Qcloud"
    req.ParamList = [{"name":"max_connections","value":"1000"},{"name":"lower_case_
    req.ProtectMode = 1
    req.SlaveZone = "ap-beijing-3"
    req.InstanceName = "oneday1"
    req.AutoRenewFlag = 0


    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.CreateDBInstanceHour(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    msg = traceback.format_exc() # 方式1
    print (msg)
```

DescribeDBInstances 查询实例列表

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
import logging
import traceback
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
```

```
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.DescribeDBInstancesRequest()
    req.EngineVersions = ["5.6"]
    req.OrderBy = "instanceId"
    req.InstanceIds = ["cdb-1j8lumf6"]


    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.DescribeDBInstances(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
        msg = traceback.format_exc() # 方式1
        print (msg)
```

DescribeDBPrice 查询数据库价格



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
    cred = credential.Credential("secretId", "secretKey")
```

```
    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-guangzhou")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.DescribeDBPriceRequest()
    req.Zone = "ap-guangzhou-3"
    req.GoodsNum = 1
    req.Memory =2000
    req.Volume =1000
    req.PayType = 'PRE_PAID'
    req.Period=1


    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.DescribeDBPrice(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```

实例管理

最近更新时间：2019-11-20 15:58:52

API	描述
ModifyInstanceParam	修改实例参数
CloseWanService	关闭实例外网访问
OpenWanService	开通实例外网访问
RestartDBInstances	重启实例
OpenDBInstanceGTID	开启实例的 GTID
ModifyDBInstanceName	修改云数据库实例名
ModifyDBInstanceProject	修改云数据库实例的所属项目
ModifyDBInstanceVipVport	修改云数据库实例的 IP 和端口号
DescribeDBInstanceCharset	查询云数据库实例的字符集
DescribeDBInstanceConfig	查询云数据库实例的配置信息
DescribeDBInstanceGTID	查询云数据实例的 GTID 是否开通
DescribeDBInstanceRebootTime	查询云数据库实例的预期重启时间

ModifyInstanceParam 修改实例参数

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
import logging
import traceback
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")
```

```
#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象
req = models.ModifyInstanceParamRequest()
req.InstanceIds = ["cdb-1y6g3zj8","cdb-7ghaiocc"]
req.ParamList = [{"name":"max_connections","currentValue":"100"},{"name":"charact
er_set_server","currentValue":"utf8"},{"name":"lower_case_table_names","currentVa
lue":"1"}]
#req.ParamList = [{"name":"max_connections","currentValue":"100"}]
#param = models.Parameter()
#param.Name = "max_connections"
#param.CurrentValue = "1000"
#req.ParamList = [param]


print req
# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.ModifyInstanceParam(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
msg = traceback.format_exc() # 方式1
print (msg)
```

CloseWanService 关闭实例外网访问

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")
```

```
#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.CloseWanServiceRequest()
req.InstanceId = "cdb-1y6g3zj8"

# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.CloseWanService(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

OpenWanService 开通实例外网访问

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.OpenWanServiceRequest()
req.InstanceId = "cdb-1y6g3zj8"

# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.OpenWanService(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

RestartDBInstances 重启实例

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.RestartDBInstancesRequest()
req.InstanceIds = ["cdb-7ghaiocc"]

# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.RestartDBInstances(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

OpenDBInstanceGTID 开启实例的 GTID

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")
```

```
#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.OpenDBInstanceGTIDRequest()
req.InstanceId = "cdb-7ghaiocc"


# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.OpenDBInstanceGTID(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

ModifyDBInstanceName 修改云数据库实例名

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-beijing")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.ModifyDBInstanceNameRequest()
req.InstanceId = "cdb-cukm86n2"
req.InstanceName = "1s中文"

# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.ModifyDBInstanceName(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

ModifyDBInstanceProject 修改云数据库实例的所属项目

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
import logging
import traceback
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models


def DescribeDBInstancesList():
try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.ModifyDBInstanceProjectRequest()
req.InstanceIds = ["cdb-7ghaiocc"]
req.NewProjectId =1


# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.ModifyDBInstanceProject(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
msg = traceback.format_exc() # 方式1
print (msg)



DescribeDBInstancesList()
```

ModifyDBInstanceVipVport 修改云数据库实例的 IP 和端口号

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块
import logging
import traceback
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.ModifyDBInstanceVipVportRequest()
req.InstanceId = "cdb-7ghaiocc"
req.DstIp = "10.0.0.13"
req.DstPort =1025
req.UniqVpcId = 1111

# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.ModifyDBInstanceVipVport(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
msg = traceback.format_exc() # 方式1
print (msg)
```

DescribeDBInstanceCharset 查询云数据库实例的字符集

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
```

```
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.DescribeDBInstanceCharsetRequest()
req.InstanceId = "cdb-1y6g3zj8"


# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.DescribeDBInstanceCharset(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

DescribeDBInstanceConfig 查询云数据库实例的配置信息

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.DescribeDBInstanceConfigRequest()
req.InstanceId = "cdb-1y6g3zj8"
```

```
# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.DescribeDBInstanceConfig(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

DescribeDBInstanceGTID 查询云数据实例的 GTID 是否开通

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.DescribeDBInstanceGTIDRequest()
req.InstanceId = "cdb-1y6g3zj8"


# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.DescribeDBInstanceGTID(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

DescribeDBInstanceRebootTime 查询云数据库实例的预期重启时间

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
```

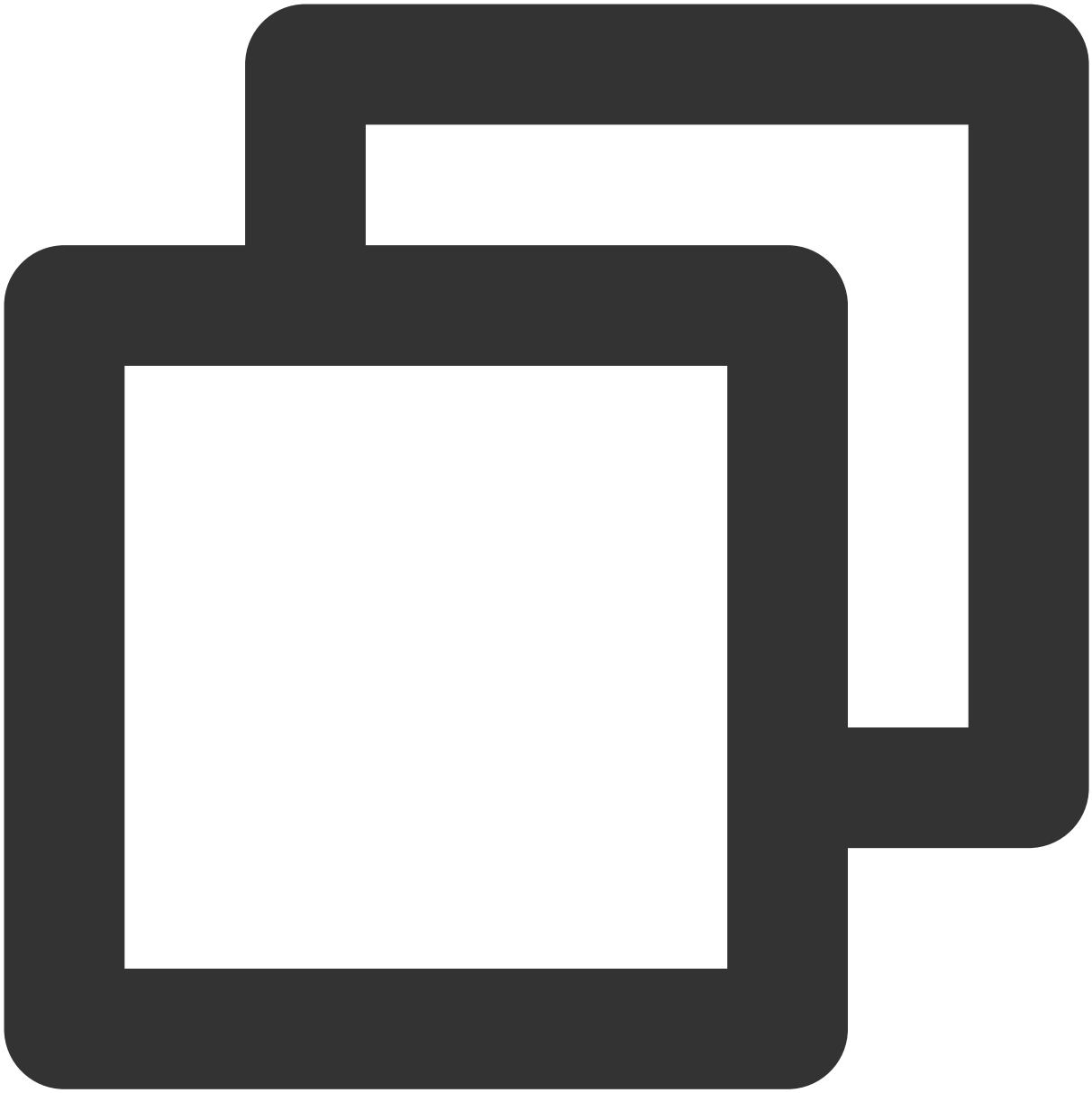
```
# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentClou
dSDKException
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
# 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
cred = credential.Credential("secretId", "secretKey")

#实例化要请求产品(以cdb为例)的client对象
client = cdb_client.CdbClient(cred, "ap-shanghai")

#实例化一个请求对象:req = models.ModifyInstanceParamRequest()
req = models.DescribeDBInstanceRebootTimeRequest()
req.InstanceIds = ["cdb-1y6g3zj8"]


# 通过client对象调用想要访问的接口，需要传入请求对象
resp = client.DescribeDBInstanceRebootTime(req)

# 输出json格式的字符串回包
print(resp.to_json_string())
except TencentCloudSDKException as err:
print(err)
```

备份任务

最近更新时间：2023-09-13 15:56:37

API	描述
CreateBackup	创建云数据库备份
DeleteBackup	删除云数据库备份
DescribeBackupConfig	查询云数据库备份配置信息
DescribeBackups	查询数据备份文件列表
DescribeBinlogs	查询二进制日志
DescribeSlowLogs	查询慢查询日志
ModifyBackupConfig	修改数据库备份配置

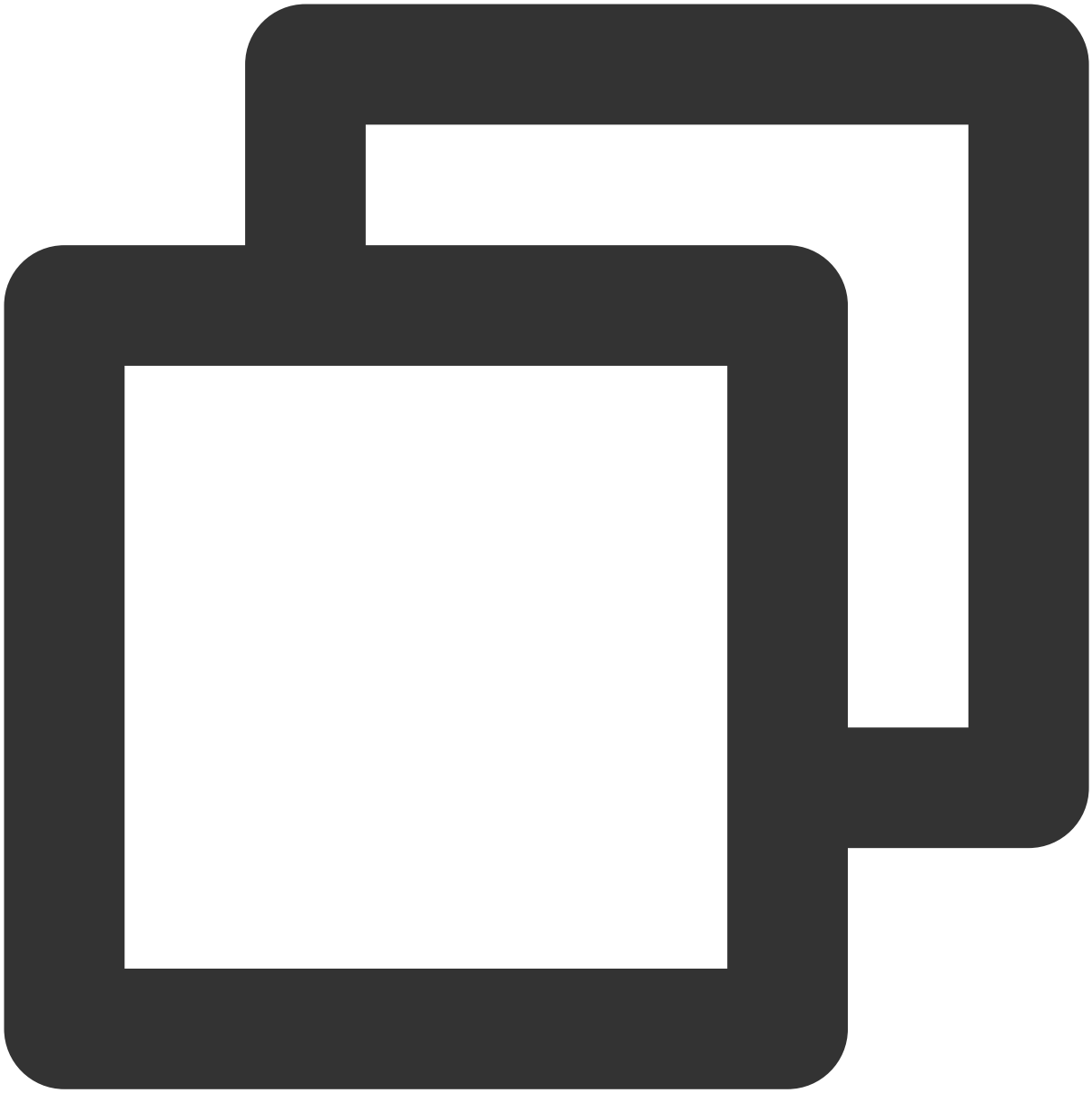
CreateBackup 创建云数据库备份



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
```

```
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.CreateBackupRequest()
    req.InstanceId = "cdb-7ghaiocc"
    req.BackupMethod = "logical"

    print req
    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.CreateBackup(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```

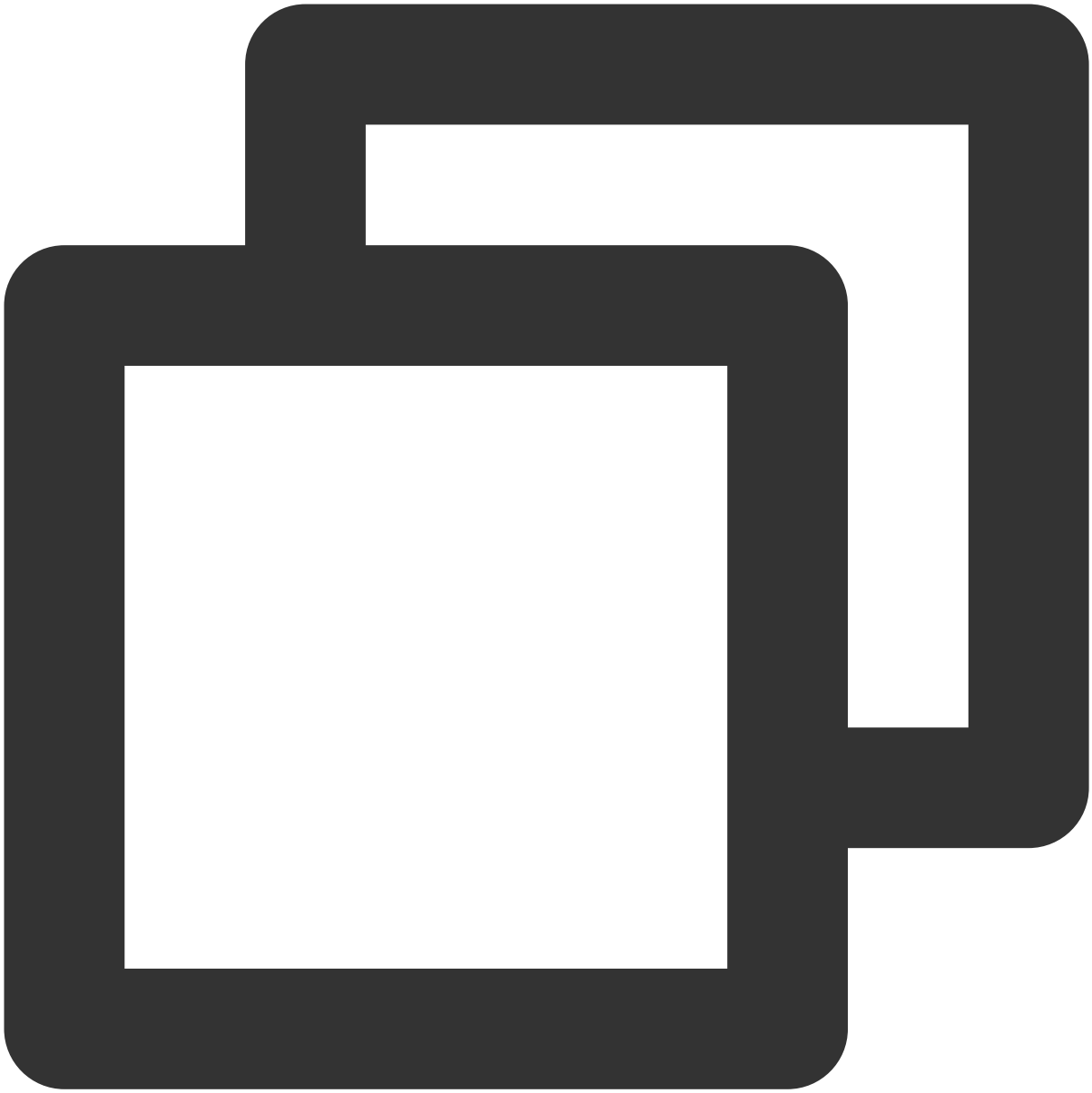
DeleteBackup 删除云数据库备份



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
```

```
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")



    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.DeleteBackupRequest()

    req.InstanceId = "cdb-7ghaiocc"
    #print  req.BackupId
    req.BackupId = 105119782


    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.DeleteBackup(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```

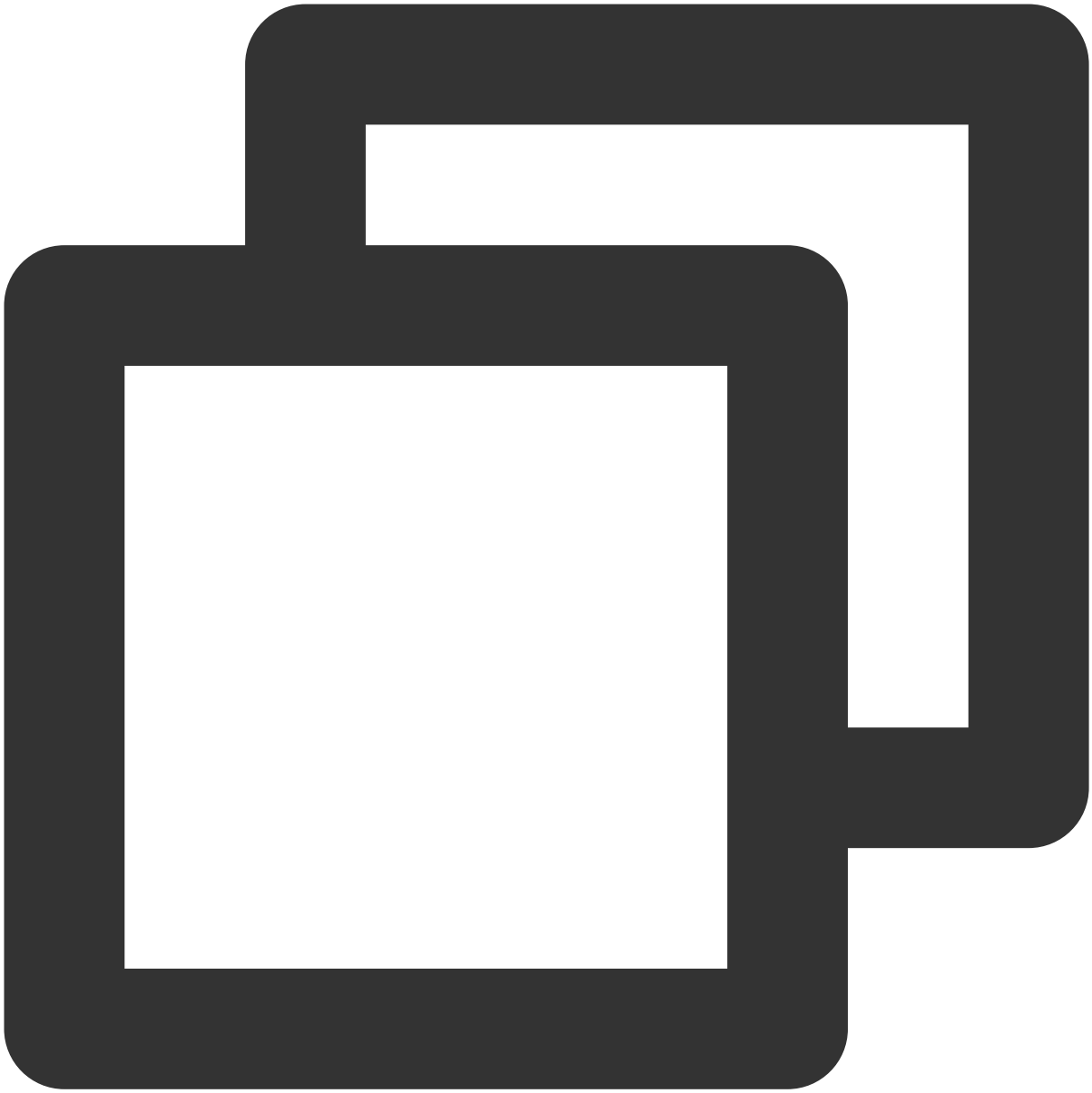
DescribeBackupConfig 查询云数据库备份配置信息



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
```

```
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.DescribeBackupConfigRequest()
    req.InstanceId = "cdb-7ghaiocc"

    print req
    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.DescribeBackupConfig(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```

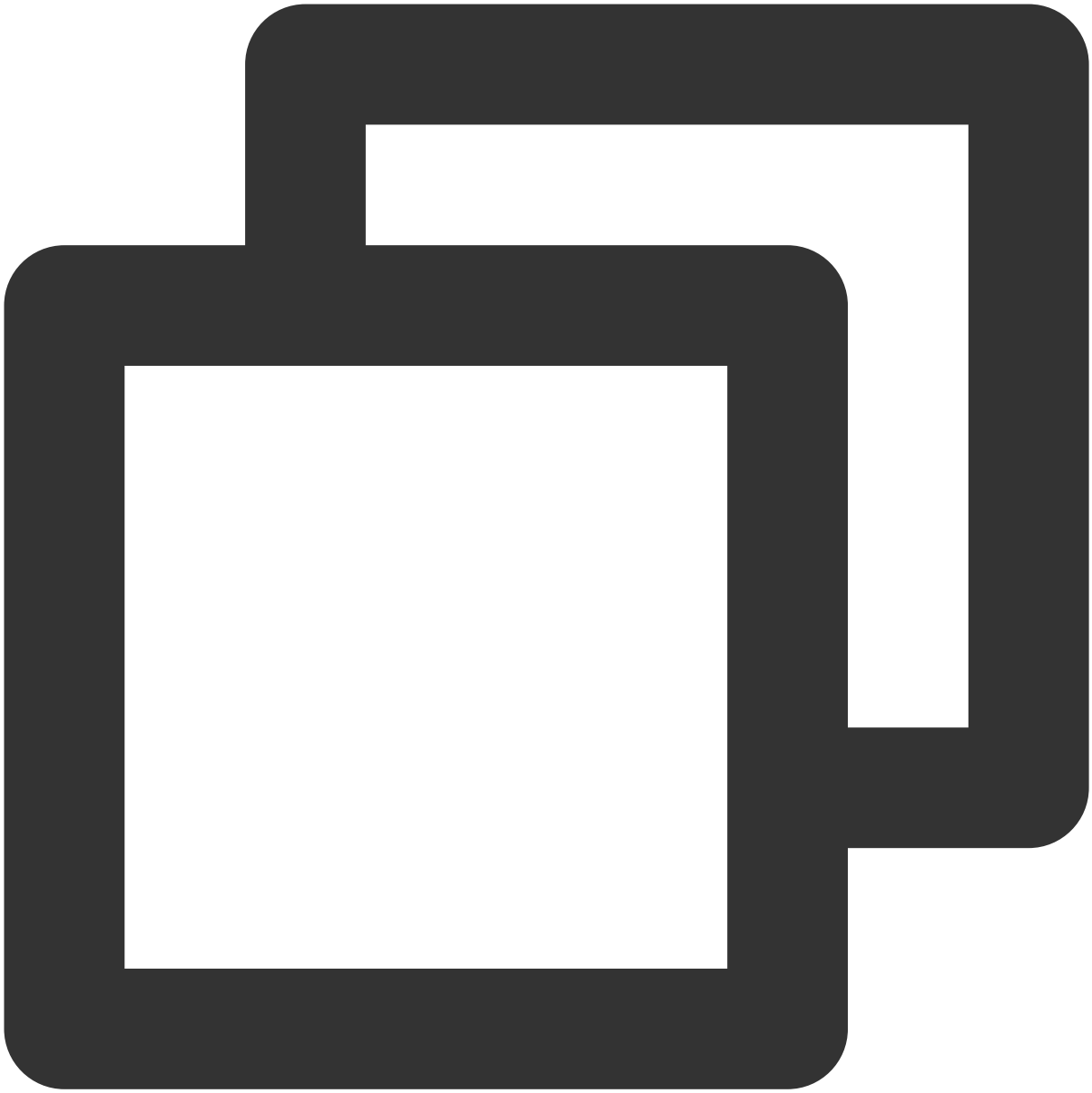
DescribeBackups 查询数据备份文件列表



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
```

```
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.DescribeBackupsRequest()
    req.InstanceId = "cdb-7ghaiocc"



    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.DescribeBackups(req)
    print resp

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```

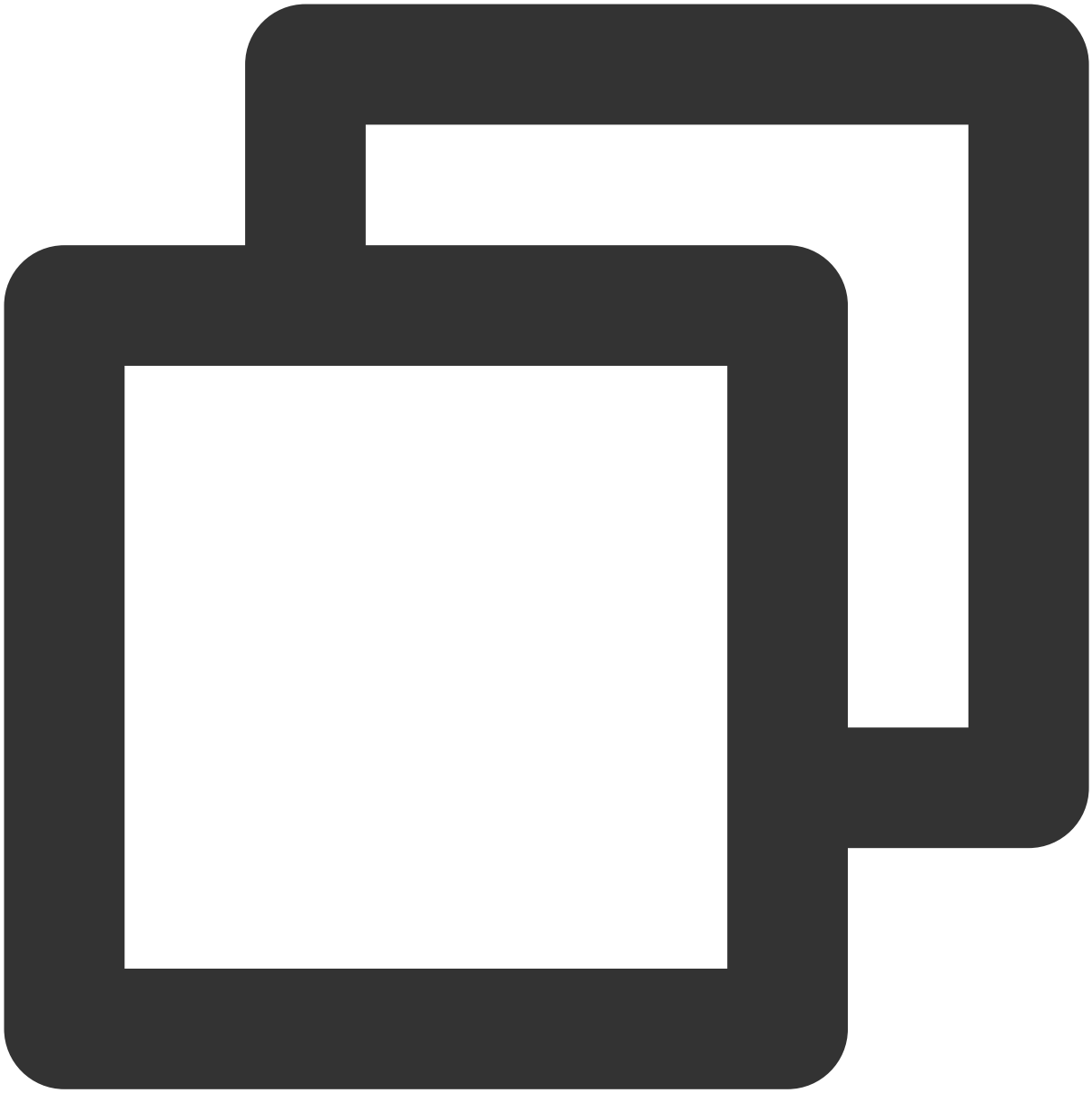
DescribeBinlogs 查询二进制日志



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
```

```
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.DescribeBinlogsRequest()
    req.InstanceId = "cdb-7ghaiocc"


    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.DescribeBinlogs(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```

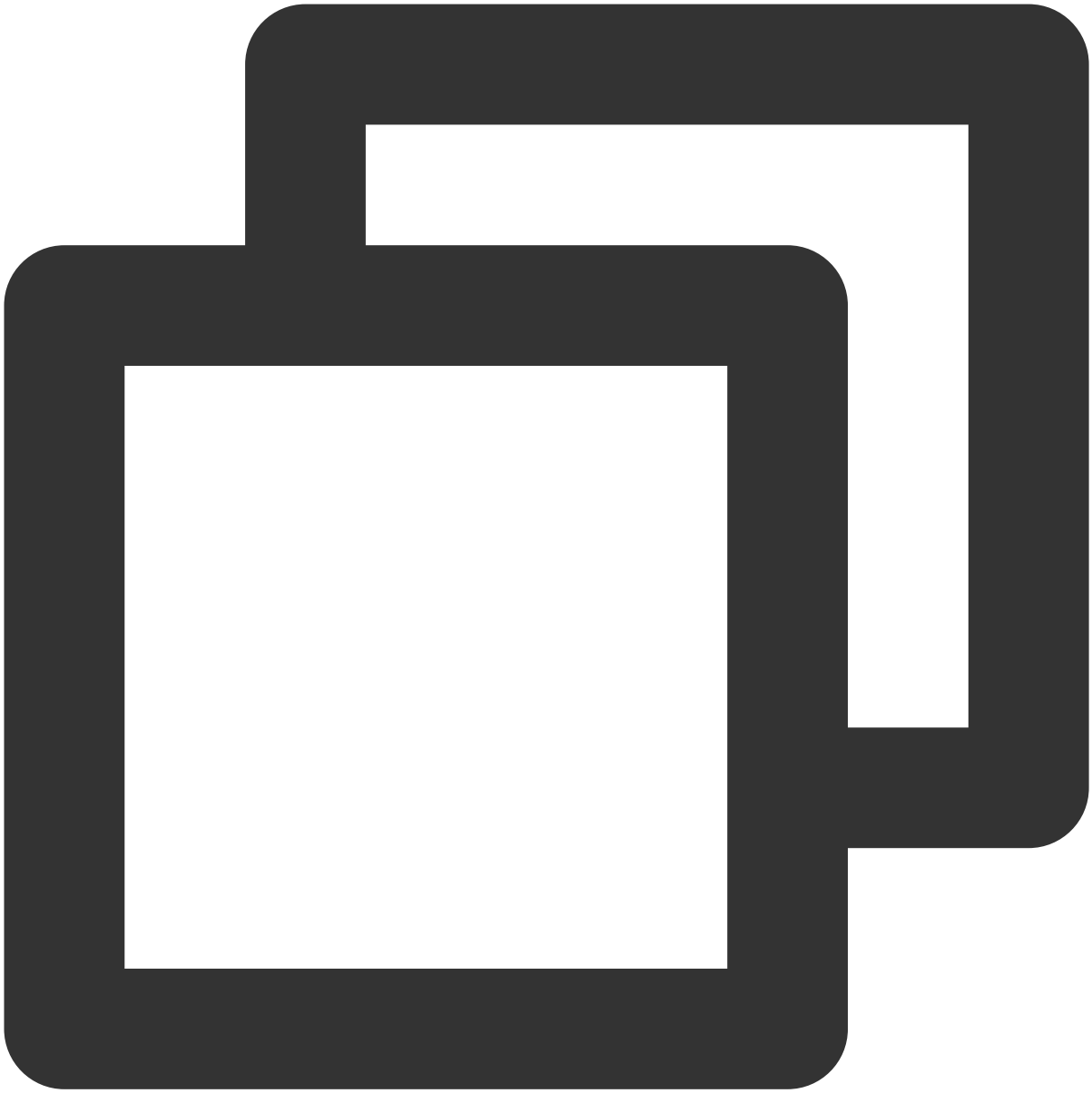
DescribeSlowLogs 查询慢查询日志



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
```

```
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.DescribeSlowLogsRequest()
    req.InstanceId = "cdb-7ghaiocc"


    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.DescribeSlowLogs(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```

ModifyBackupConfig 修改数据库备份配置



```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# 引入云API入口模块

from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import TencentCloudS
from tencentcloud.cdb.v20170320 import cdb_client, models

try:
    # 实例化一个认证对象，入参需要传入腾讯云账户secretId，secretKey
```

```
    cred = credential.Credential("secretId", "secretKey")

    #实例化要请求产品(以cdb为例)的client对象
    client = cdb_client.CdbClient(cred, "ap-shanghai")

    #实例化一个请求对象:req = models.ModifyInstanceParamRequest()
    req = models.ModifyBackupConfigRequest()
    req.InstanceId = "cdb-1y6g3zj8"
    req.ExpireDays = 10
    req.StartTime = "06:00-10:00"
    req.BackupMethod = "logical"
    print req


    # 通过client对象调用想要访问的接口，需要传入请求对象
    resp = client.ModifyBackupConfig(req)

    # 输出json格式的字符串回包
    print(resp.to_json_string())
except TencentCloudSDKException as err:
    print(err)
```