## 需求介绍

双库热切，昆仑数据库Klustron与另一个数据库同时运行，应用将读写流量随时切换到另一个运行的数据库上。另一个库包括MySQL，ORACLE。

## 方案：CDC数据同步架构

### 架构简介

两种异构数据库以MySQL和Klustron为例，应用将数据写入到MySQL中，通过CDC数据同步方式，将MySQL中的数据实时同步到Klustron上，由于数据是实时同步的，在允许数据同步最大流量情况内，并保证数据都同步完成，即MySQL与Klustron两库的数据量一致后，应用读流量可以切换到Klustron中来。

此架构中，由于MySQL与Klustron之间使用的是CDC数据同步，MySQL担任主集群，Klustron担任的是备集群。应用无需关心数据库侧的同步问题，应用只需要负责正常写入即可。当故障发生时，可以将读写请求切换到Klustron上。需要实时观测CDC的同步情况，一旦出现同步告警，需立即介入处理跟进解决。例如出现数据同步中断，或者同步流量过大，导致同步速度慢等。

### 架构特点

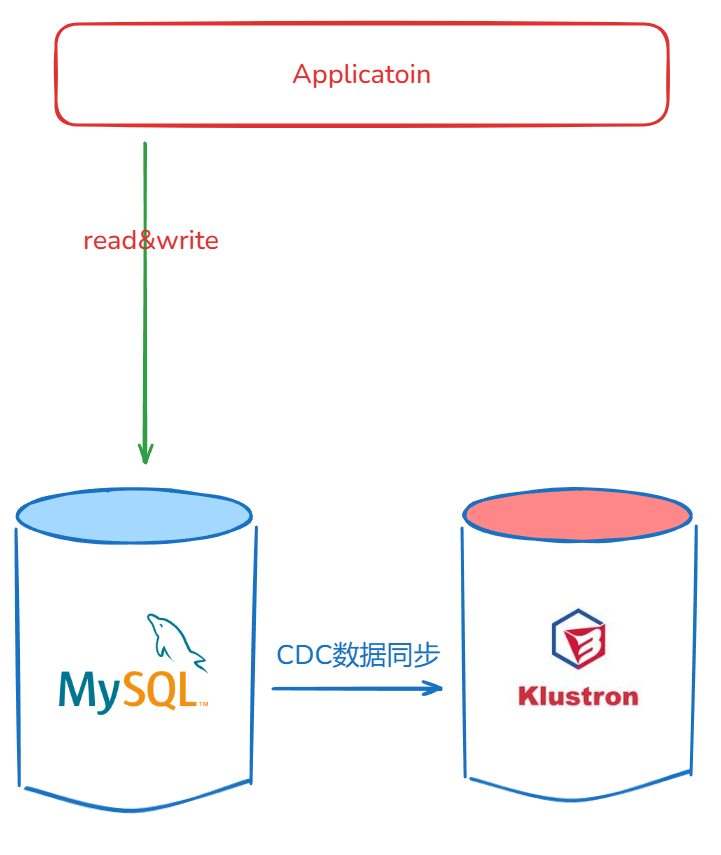
对应用透明，CDC数据同步需要保障数据实时同步和数据一致性。

### 架构关键点

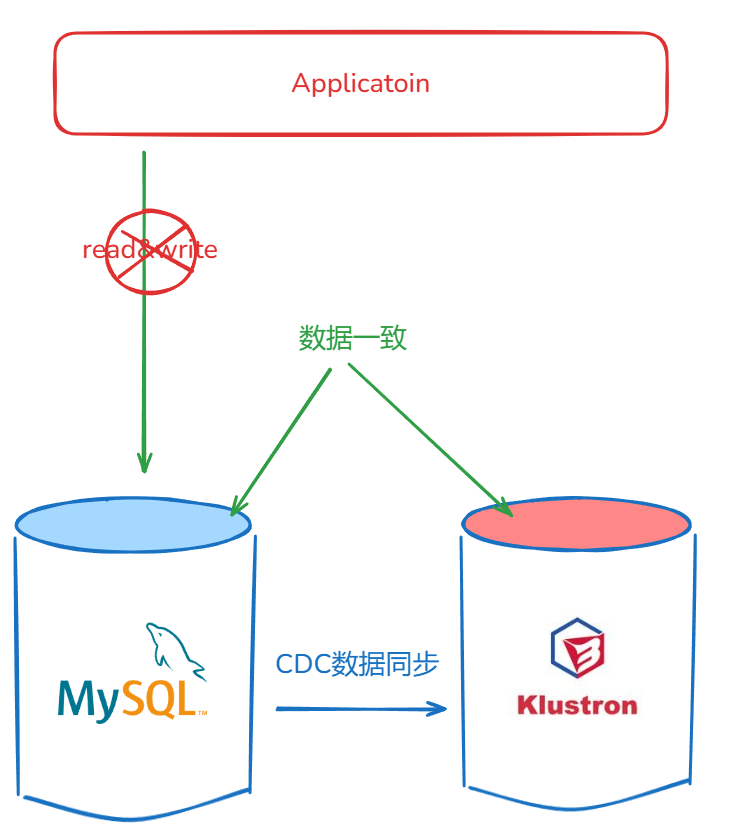
1. CDC数据同步延迟情况需要实时观测；
2. MySQL数据库与Klustron数据库中的数据需要定时校验，保证数据的一致性。

### 整个同步流程及切换过程图示

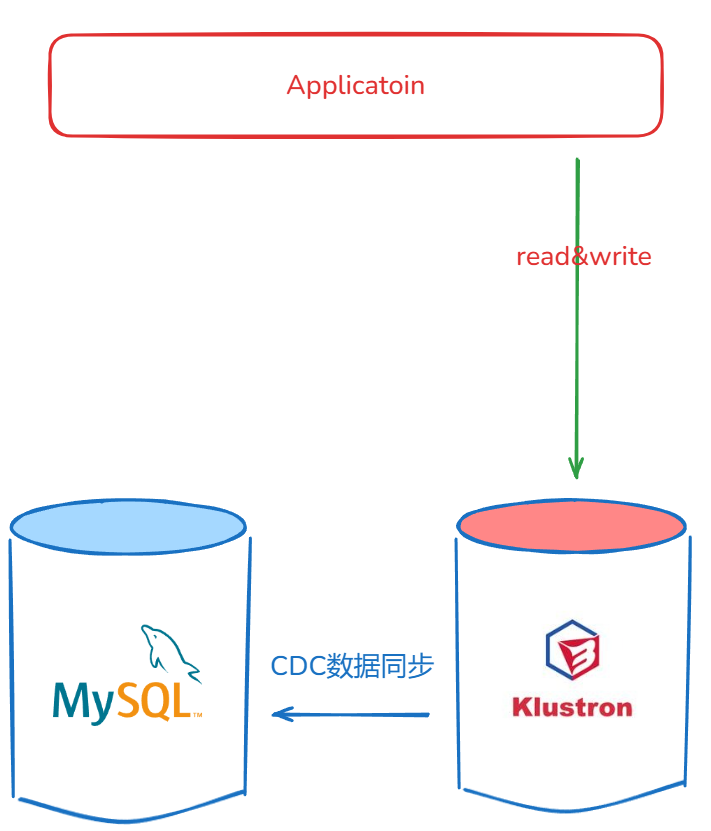
1. **全量+增量，数据正常同步过程，应用读写请求到MySQL中，由CDC将数据同步到Klustron**



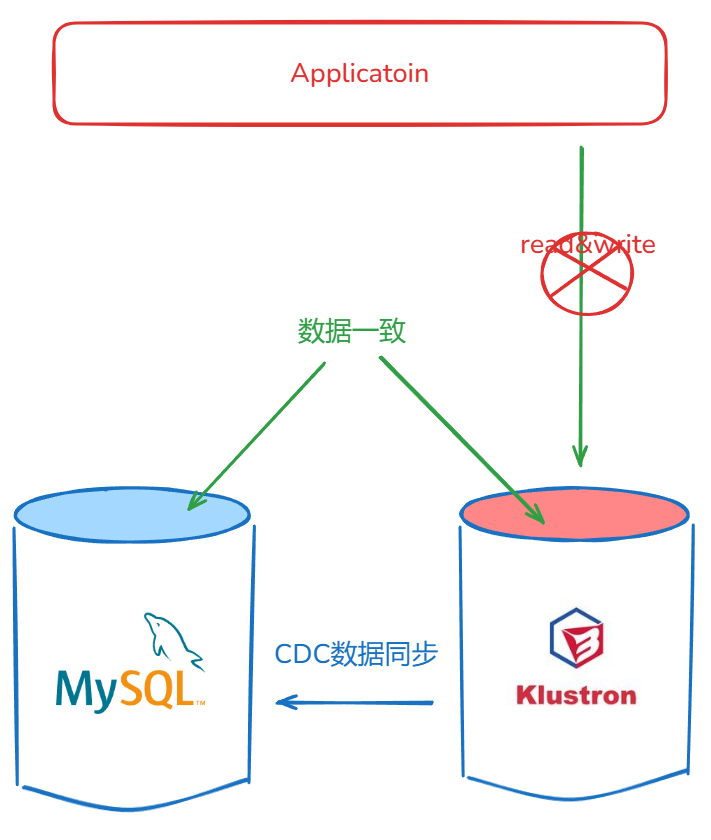
**（2）全量+增量，应用准备切换，停止对的MySQL请求，待MySQL与Klustron数据一致，进行切换**



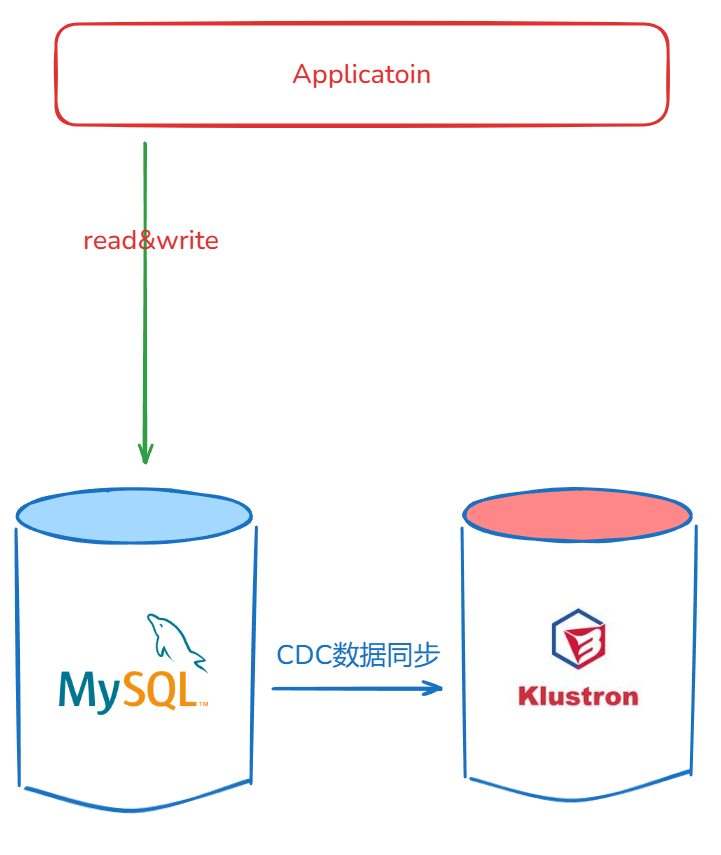
**（3）切换后，数据从Klustron同步到MySQL中，应用路由切换到Klustron**



**（4）如果后续需要将应用切换到MySQL，先停止对Klustron请求，待Klustron与MySQL数据保持一致，最后进行切换。**



**（5）切换完成之后，数据从MySQL同步到Klustron中，应用路由切换到MySQL**



以上的整个切换之后的流向为：

1. 发起同步任务，全量+增量，MySQL -> Klustron
2. 第一次切换，增量，Klustron -> MySQL
3. 第二次切换，增量，MySQL -> Klustron

## XPanel数据同步任务操作步骤

### 无proxysql集群发起任务前的准备工作

以MySQL同步到Klustron为例

**（1）在Klustron集群的计算节点上配置如下参数：**

连接计算节点：

psql -h192.168.0.17 -p23001 -Uabc postgres

alter system set enable\_collation\_pushdown =false;

alter system set storage\_table\_charset ='utf8';

alter system set storage\_table\_collate ='utf8\_general\_ci';

重启计算节点

cd /data/ha/kunlun/instance\_binaries/computer/23001/kunlun-server-1.3.2/bin

./pg\_ctl -D /data/ha/kunlun/server\_datadir/23001 restart

验证修改成功：

psql -h192.168.0.17 -p23001 -Uabc postgres

show enable\_collation\_pushdown ;

show storage\_table\_charset ;

show storage\_table\_collate ;

**（2）在Klustron中创建好需要同步的库；**

具体的create语句以实际为主。

**（3）为源和目标端创建cdc同步状态表；**

例如mysql和Klustron，在需要同步的库下创建，如果有多个库，选择一个即可，mysql和Klustron选择对应的库需要一一对应。

建表语句：

CREATE TABLE `kunlun\_cdc\_dump\_state` (

`job\_id` varchar(64) not NULL,

`repl\_info` varchar(1024) default '',

`commit\_sql\_num` int default 0,

PRIMARY KEY(`job\_id`)

);

### 有proxysql集群发起任务前的准备工作

1. **连接到集群创建库和表**

通过元数据集群登录查询proxysql集群的登录信息，执行如下语句即可：

mysql -upgx -ppgx\_pwd -h192.168.0.17 -P23301 kunlun\_metadata\_db -N -e "select concat(\"mysql \",\" -h \",hostaddr,\" -P \",port,\" -u \",user\_name,\" -p\",passwd)from proxysql\_nodes;"

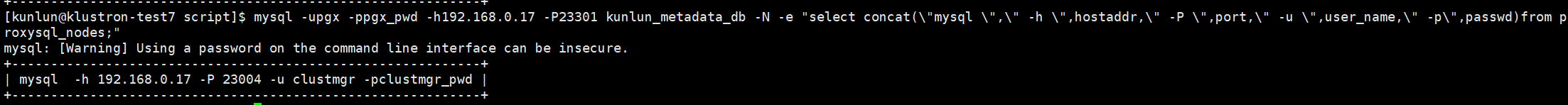
**参数介绍：**

-h修改为元数据主节点IP

-P 修改为元数据主节点端口

其他均可不变。

示例：



1. **连接登录proxysql集群，创建需要同步多个库其中的一个**

这里只创建一个库的意义在于，将CDC的状态表放到该库中，同步任务发起时需要指定该库的位置（例如DB.table），且在增量同步时，会将写入成功记录的位点写入到CDC状态表中，用于增量断点续传。该库可以是业务库，也可以是单独创建来存放CDC状态表的库。

建表语句如下：

CREATE TABLE `kunlun\_cdc\_dump\_state` (

`job\_id` varchar(64) not NULL,

`repl\_info` varchar(1024) default '',

`commit\_sql\_num` int default 0,

PRIMARY KEY(`job\_id`)

);

### 用户权限

发起数据同步前，需要确认在目标端，例如从mysql到Klustron（包括有或无proxysql），那么Klustron上的用户所需要的权限为：

全量：RELOAD, REPLICATION，示例：GRANT RELOAD, REPLICATION CLIENT ON \*.\* TO 'mysql1'@'192.168.0.%' ;

增量：由客户环境中的写请求用户确定。

初始化工作，在Klustron集群中分为**无proxysql集群**和**有proxysql**集群两种情况。

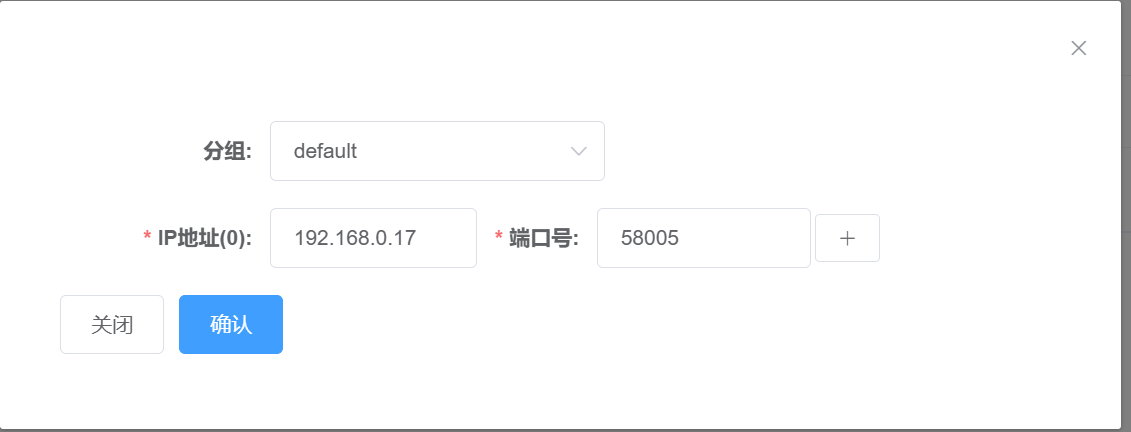
### XPanel上发起数据同步任务的操作

#### 全量

**以有proxysql集群为例，进行如下操作：**

1. 注册CDC服务信息

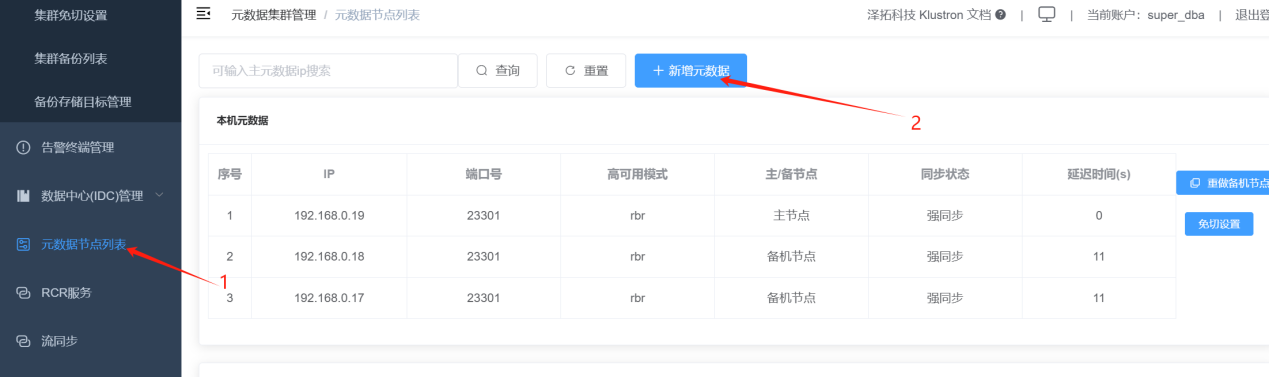




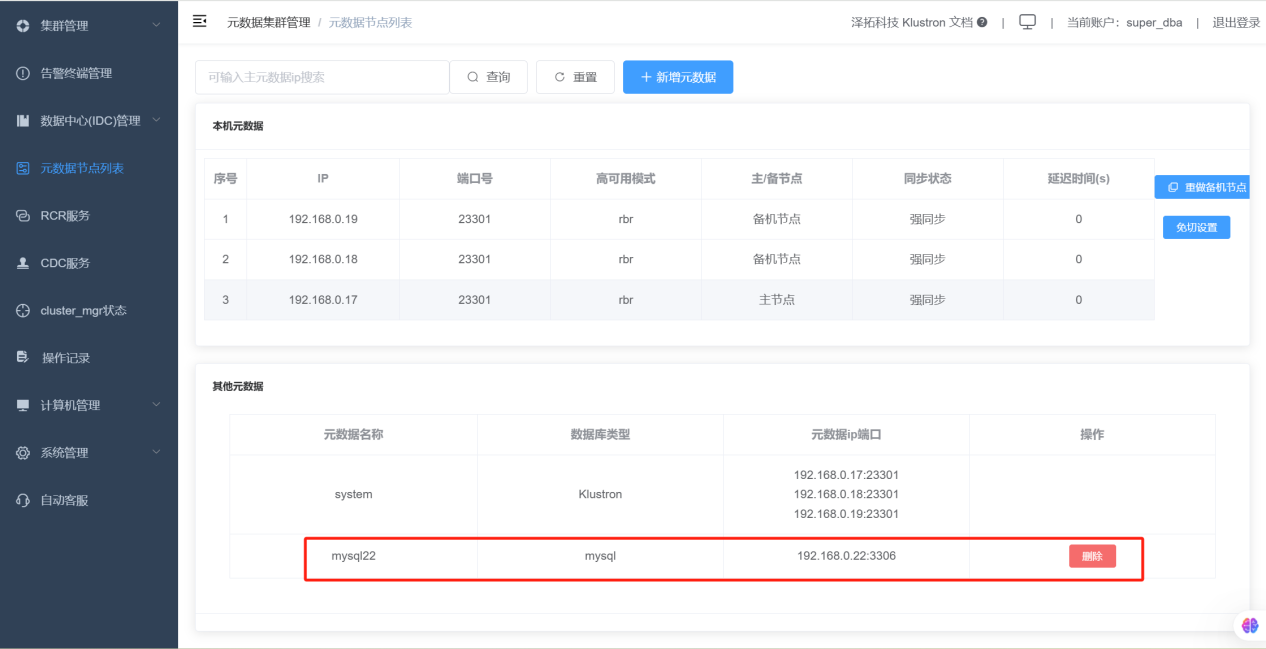


1. 注册目标数据库信息内容，以mysql为例

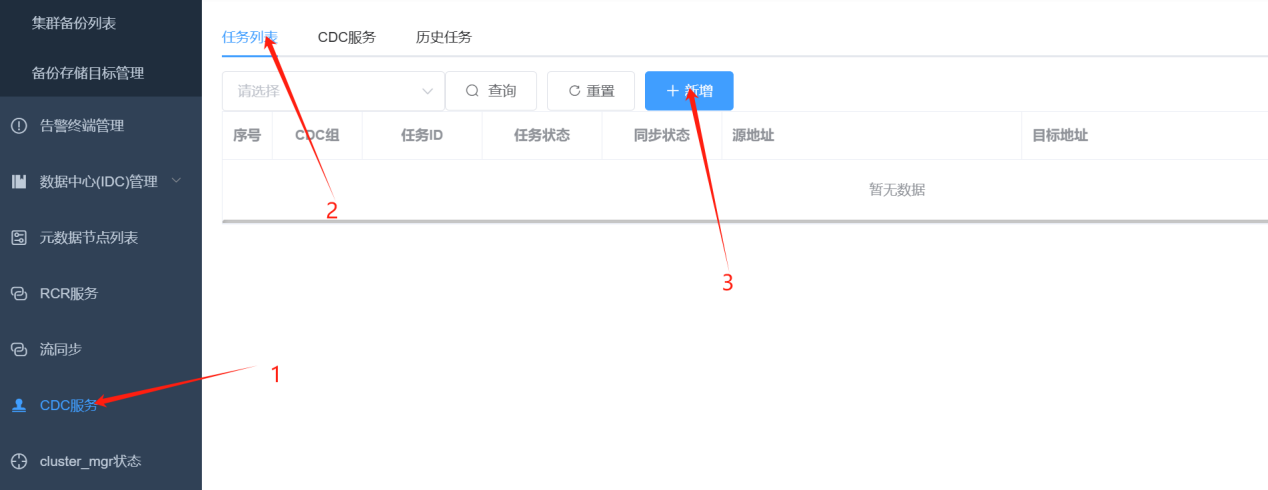
需要注意这里的用户权限问题，具体见用户权限章节说明。



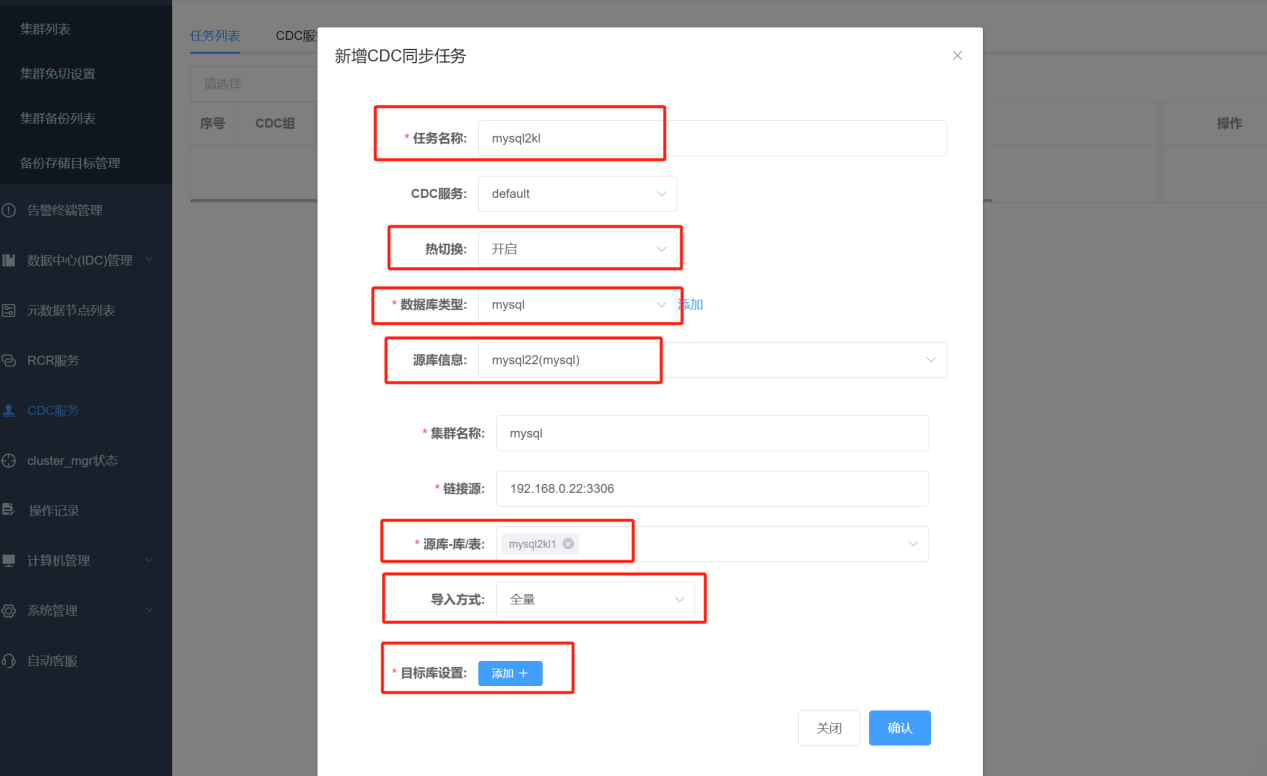




（3）配置并发起CDC全量数据同步任务，以MySQL同步到Klustron为例



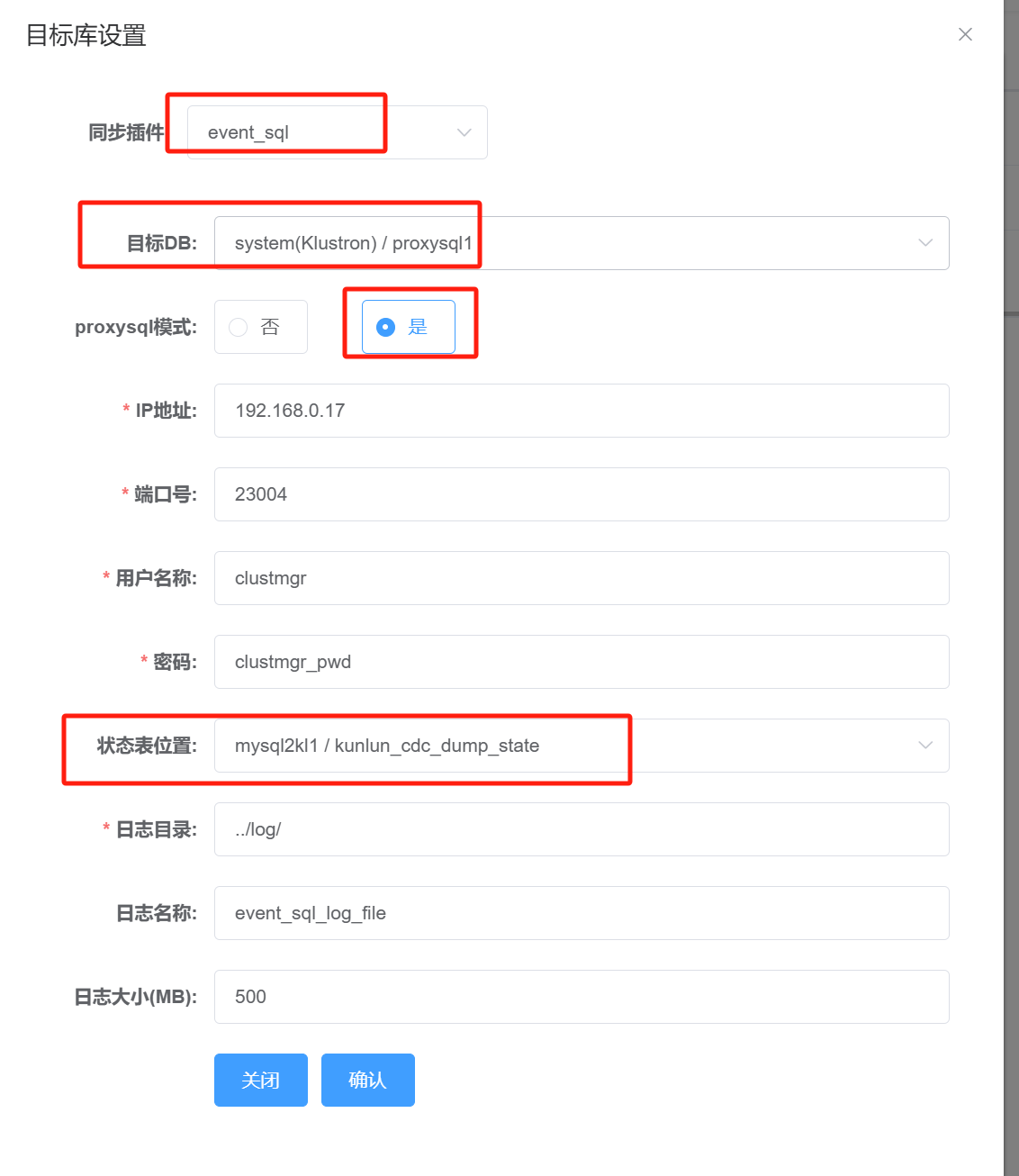
第一步：输入源库配置等信息



如图有7个地方需要根据实际情况填写或选择：

1. 任务名称：手动输入任务名称；
2. 热切换：选择开启；
3. 数据库类型：选择mysql（以mysql为例）；
4. 源库信息：选择对应mysql连接信息；
5. 源库-库/表：选择需要同步的库或表；
6. 导入方式：选择全量；
7. 目标库设置：见第二步骤。

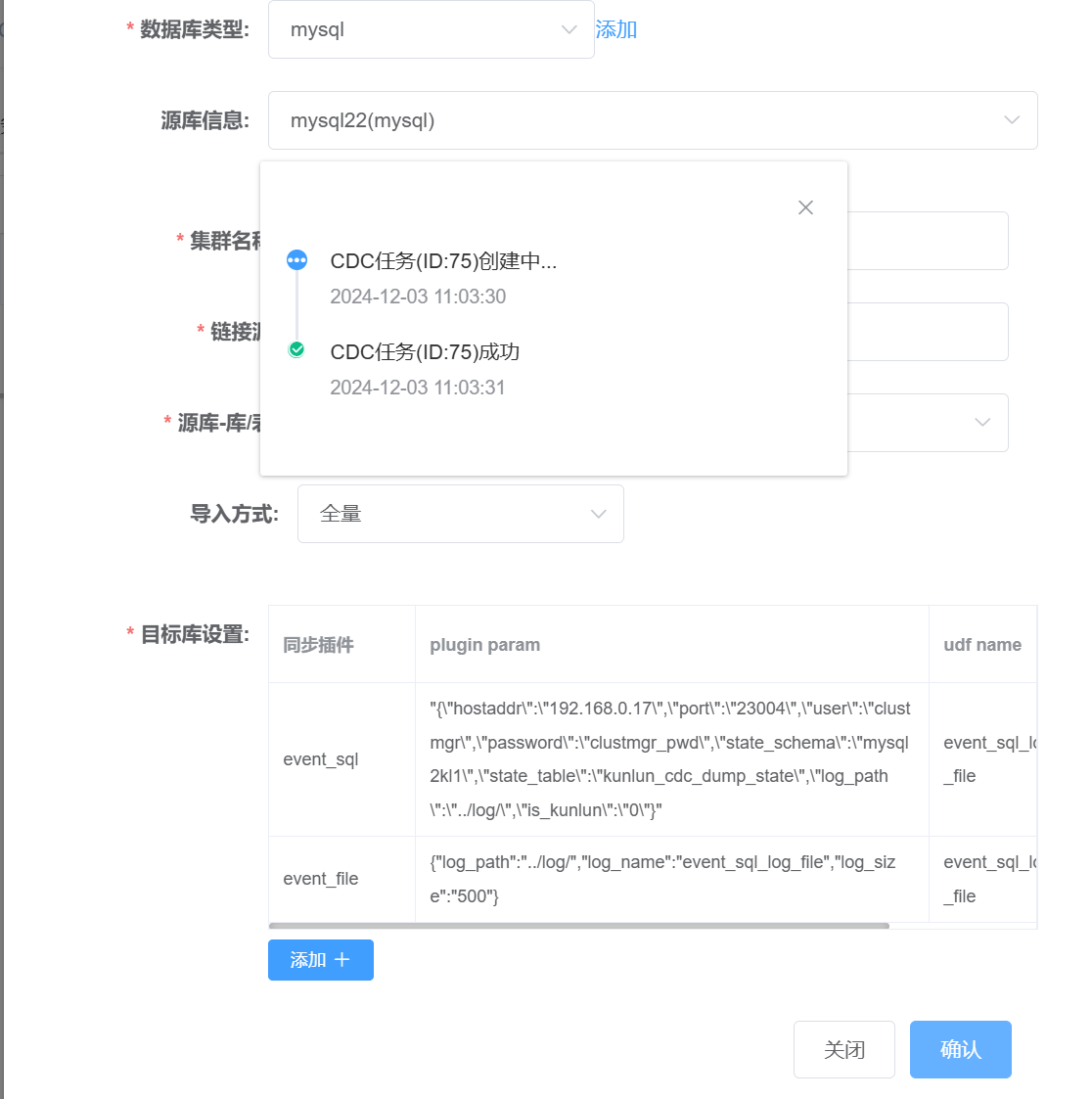
第二步：输入目标库配置等信息



如图有4个地方需要根据实际情况选择：

1. 同步插件：选择event\_sql；
2. 目标DB：选择对应的proxysql集群
3. proxysql模式：选择是；
4. 状态表位置：选择CDC状态表创建的位置。

发起任务成功：



（4）全量数据同步完成后，会继续同步增量数据



#### 增量

分如下3种情况，该三种情况的同步链路都是MySQL->Klustron proxysql集群：

（1）第一种情况，全量同步完成后，自动开启增量。

（2）第二种情况，手动全量同步完成后，在开启增量同步之前，源端无表结构的变化，需要开启增量。

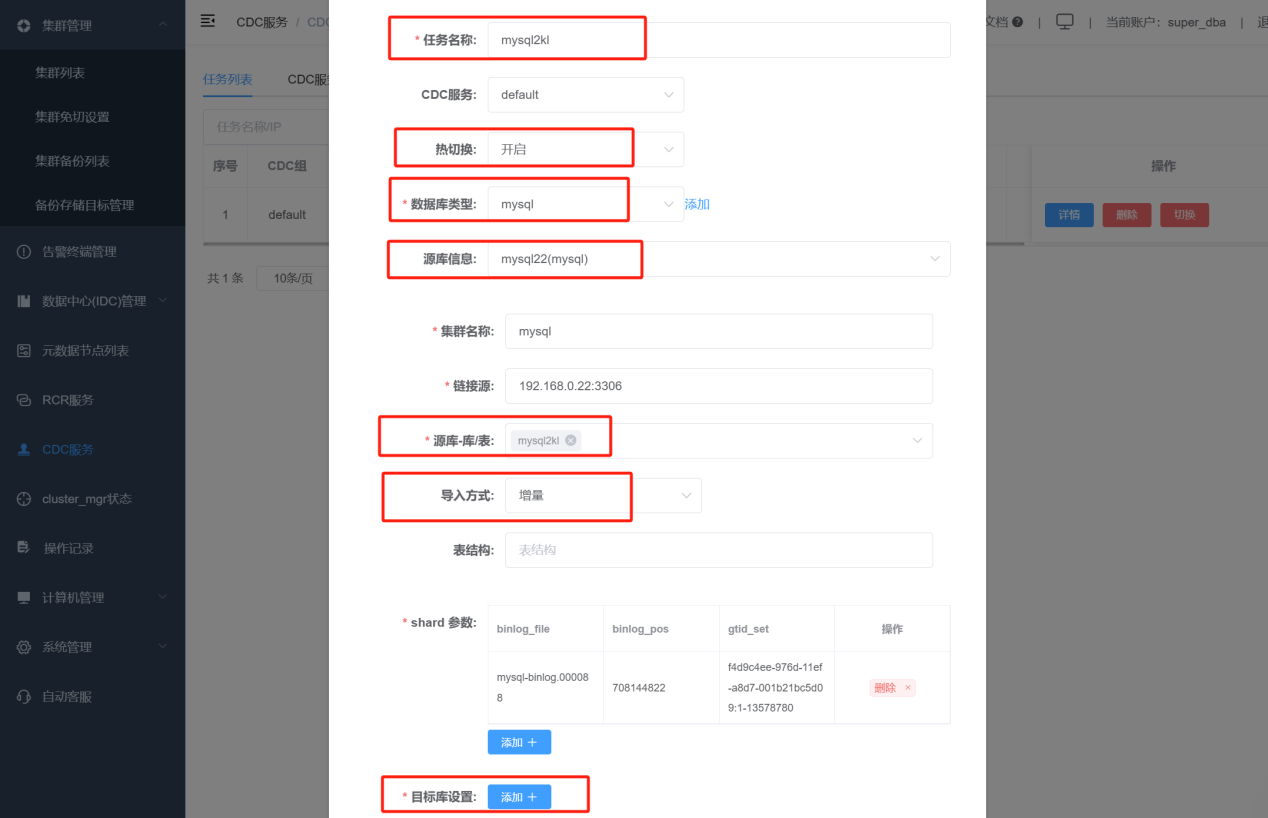
（3）第三种情况，手动全量同步完成后，在开启增量同步之前，源端有表结构的变化，需要开启增量。

**第一种情况**

略，在XPaenl发起全量同步即可，无需其他操作。

**第二种情况**

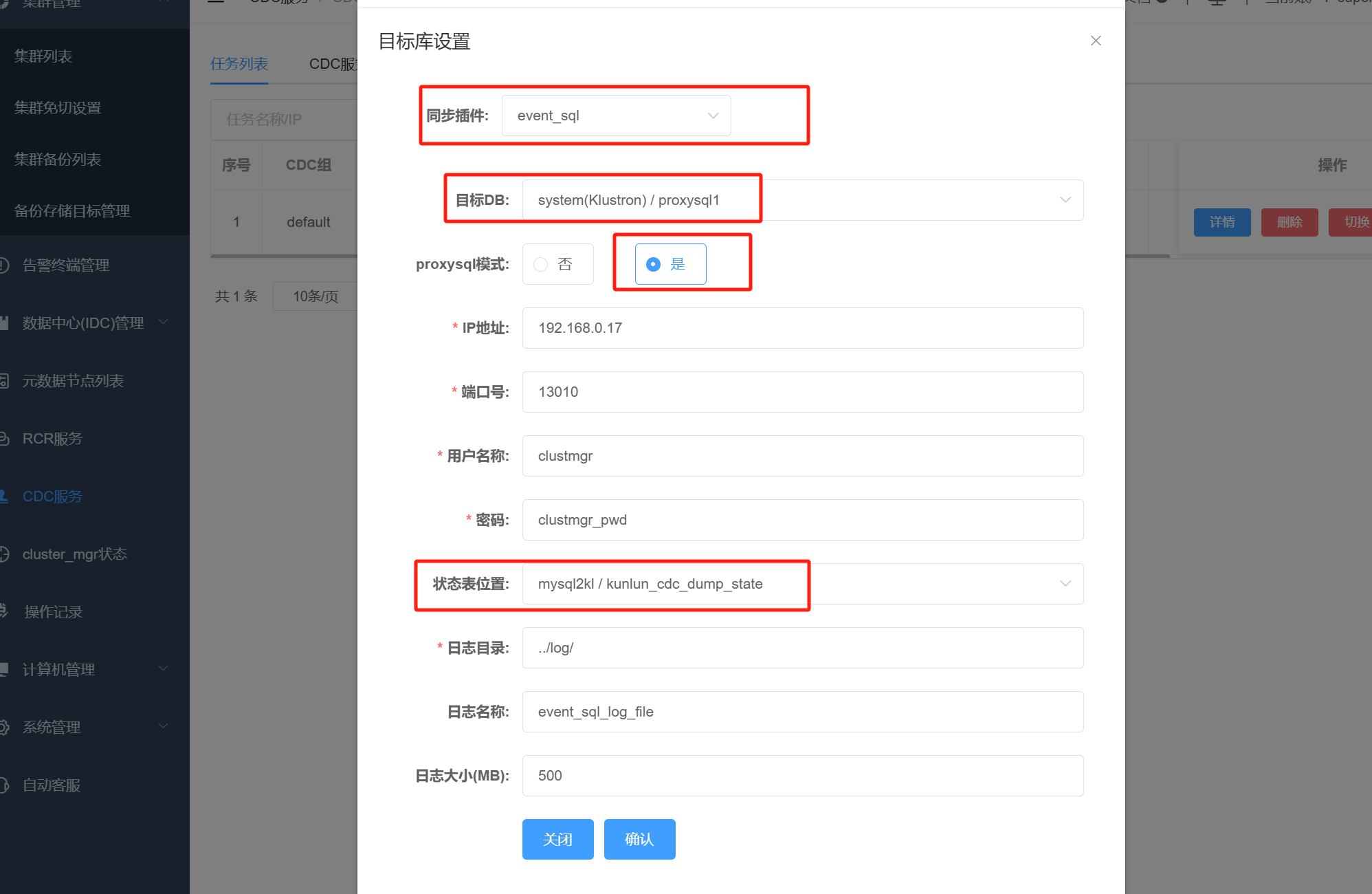
目前全量已经完成，源端也没有出现过DDL的变化，那么可根据如下步骤进行操作：



如图有7个地方需要根据实际情况填写或选择：

1. 任务名称：手动输入任务名称；
2. 热切换：选择开启；
3. 数据库类型：选择mysql（以mysql为例）；
4. 源库信息：选择对应mysql连接信息；
5. 源库-库/表：选择需要同步的库或表；
6. 导入方式：选择增量；
7. 目标库设置：见第二步骤。

第二步：输入目标库配置等信息



如图有4个地方需要根据实际情况选择：

1. 同步插件：选择event\_sql；
2. 目标DB：选择对应的proxysql集群；
3. proxysql模式：选择是；
4. 状态表位置：选择CDC状态表创建的位置。

点击确认后，创建CDC增量任务成功。

**第三种情况**

目前已经全量同步完成，但在全量完成后，源端有发生DDL语句，按照如下步骤进行操作：

**步骤1：**利用mydumper手动导出时的建表语句，根据如下格式拼接写入到schema.txt文件中，该文件需要在CDC服务（主节点）的机器上。

文件格式：

CREATE TABLE db\_name.table\_name1 (

....

);

CREATE TABLE db\_name.table\_name2 (

....

);

文件内容示例：

cat schema.txt

CREATE TABLE mysql2kl1.t2 (

`id` int(11) NOT NULL,

`name` varchar(23) DEFAULT NULL,

`imoney` int(11) DEFAULT NULL,

`c3` int(11) DEFAULT NULL

);

CREATE TABLE mysql2kl1.t3 (

`id` int(11) NOT NULL,

`name` varchar(23) DEFAULT NULL,

`imoney` int(11) DEFAULT NULL,

PRIMARY KEY (`id`)

);

CREATE TABLE mysql2kl1.z2 (

`c1` int(11) DEFAULT NULL,

`c2` int(11) DEFAULT NULL,

`c3` int(11) DEFAULT NULL

);

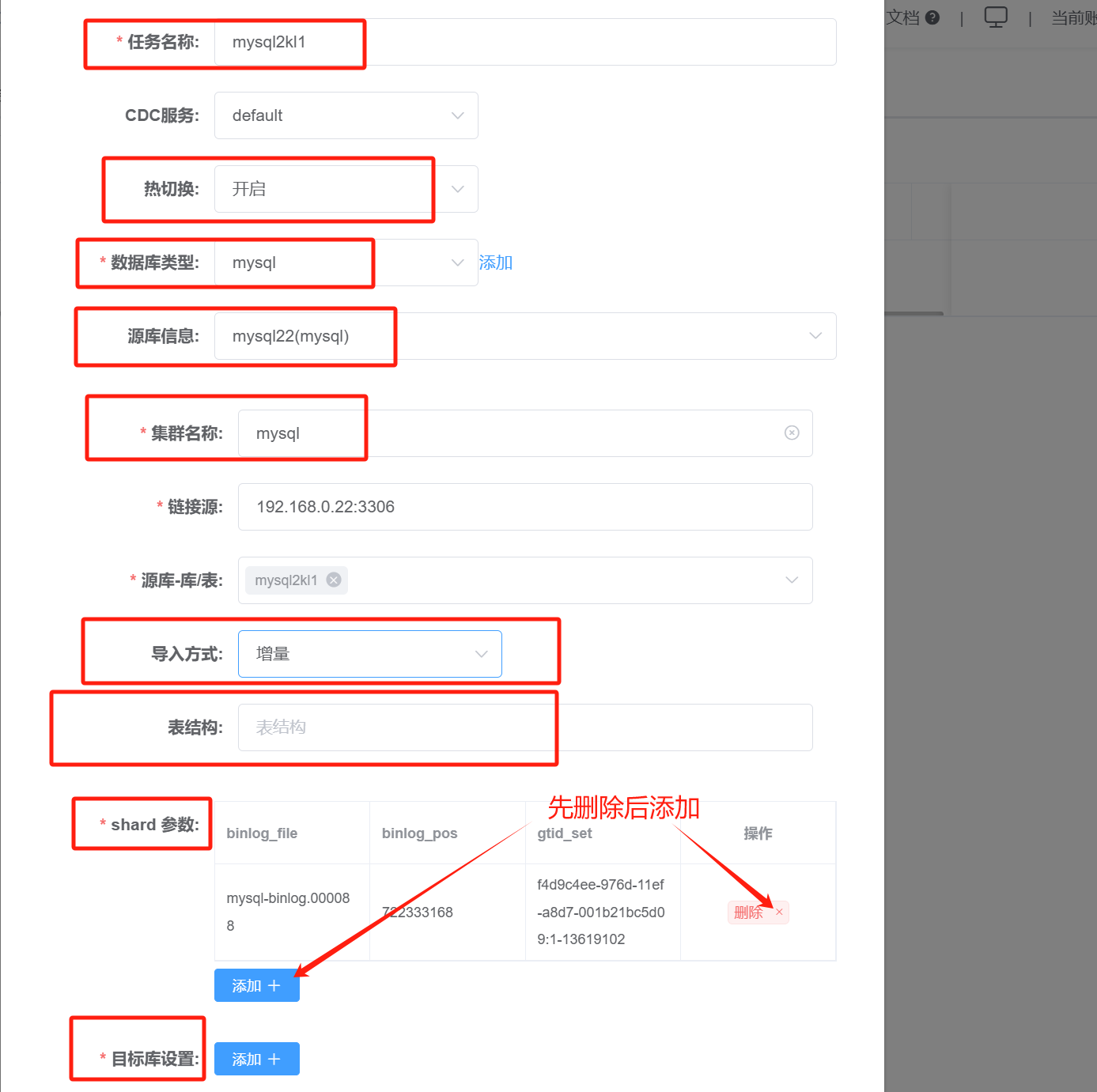
CREATE TABLE mysql2kl1.z3 (

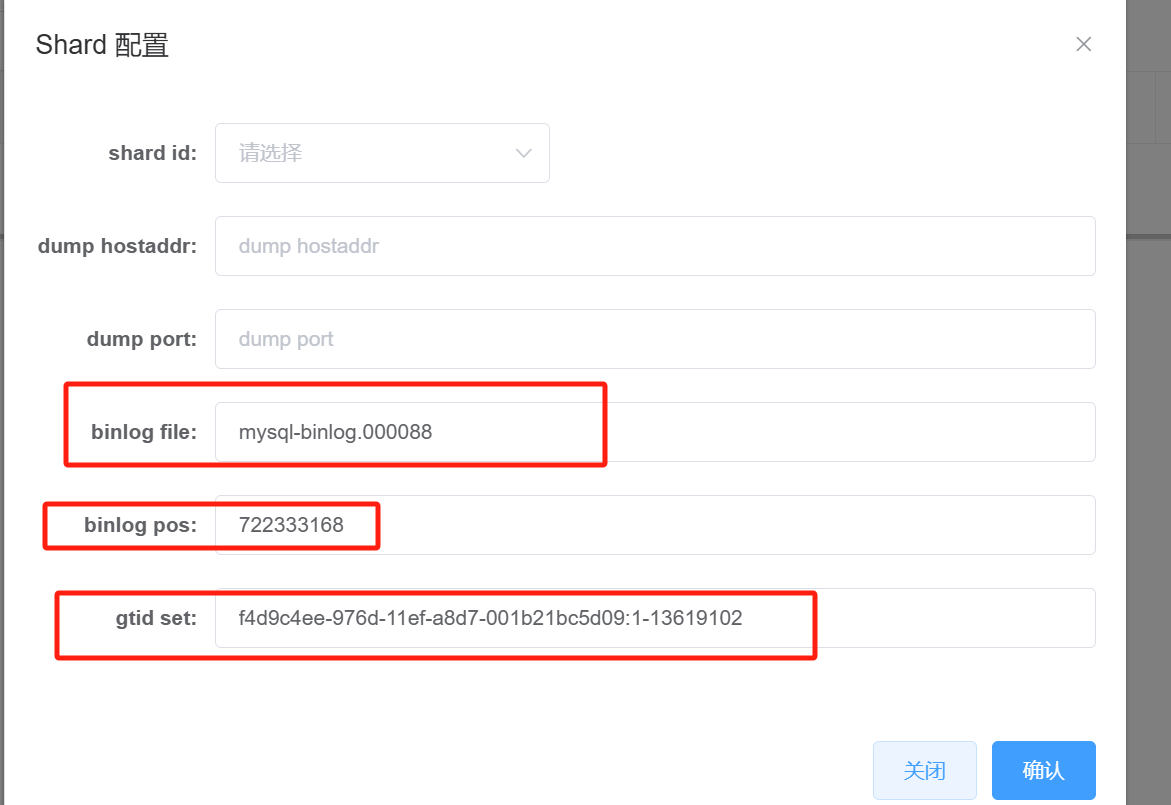
`c1` int(11) DEFAULT NULL,

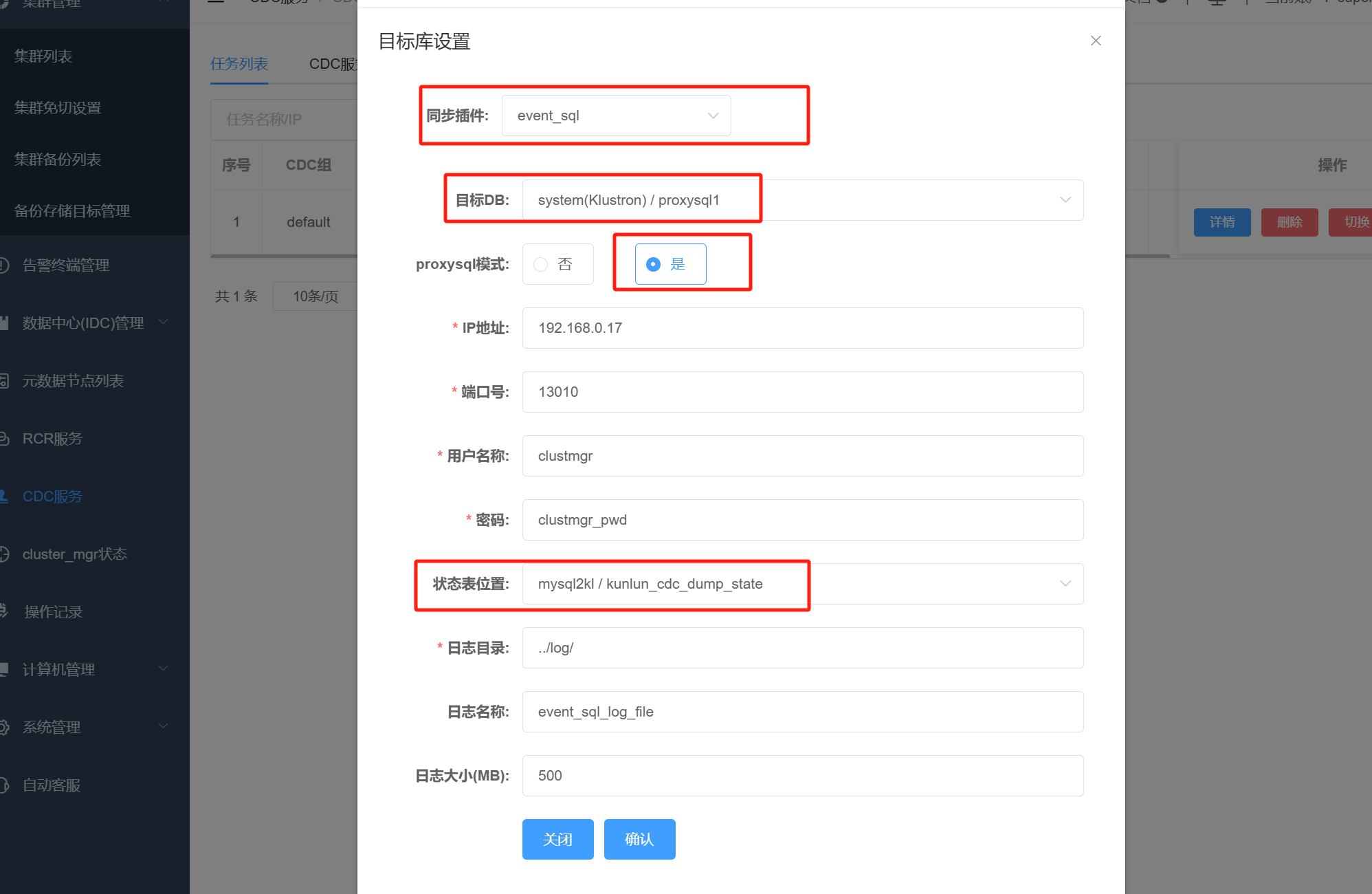
`c2` int(11) DEFAULT NULL

);

**步骤2：**XPanel上发起增量任务







如图有9个地方需要根据实际情况填写或选择：

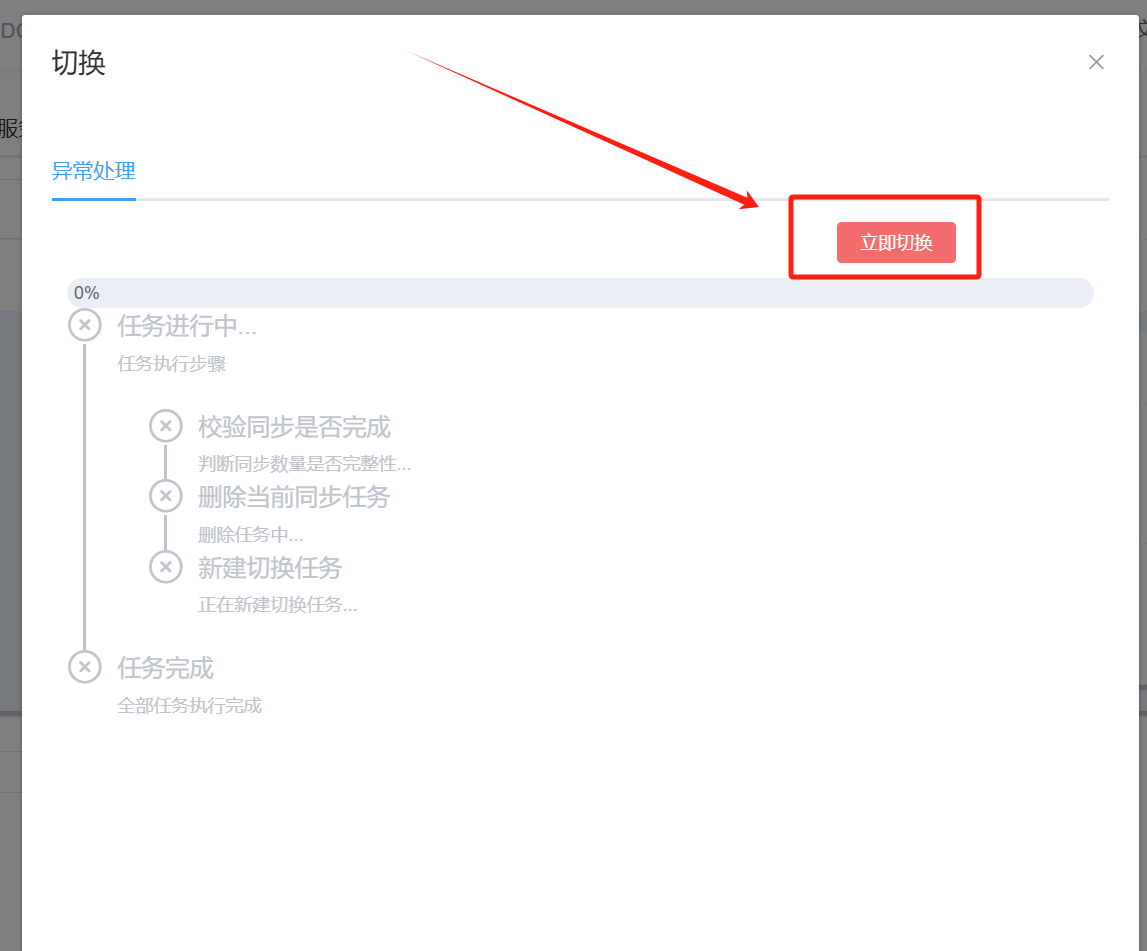
1. 任务名称：手动输入任务名称；
2. 热切换：选择开启；
3. 数据库类型：选择mysql（以mysql为例）；
4. 源库信息：选择对应mysql连接信息；
5. 源库-库/表：选择需要同步的库或表；
6. 导入方式：选择增量；
7. 表结构文件目录，根据步骤1操作，在此处填写表结构文件的绝对路径，例如：/nvme2/software/mysql2klschema.txt
8. Shard参数：先删除自动加载出来的shard参数，然后点击添加，填写mydumper全量导出完成后的metadata中的Binlog文件、偏移量Pos、GTID，最后点击确认。
9. 目标库设置：同步插件：选择event\_sql；目标DB：选择对应的proxysql集群；proxysql模式：选择是；状态表位置：选择CDC状态表创建的位置。

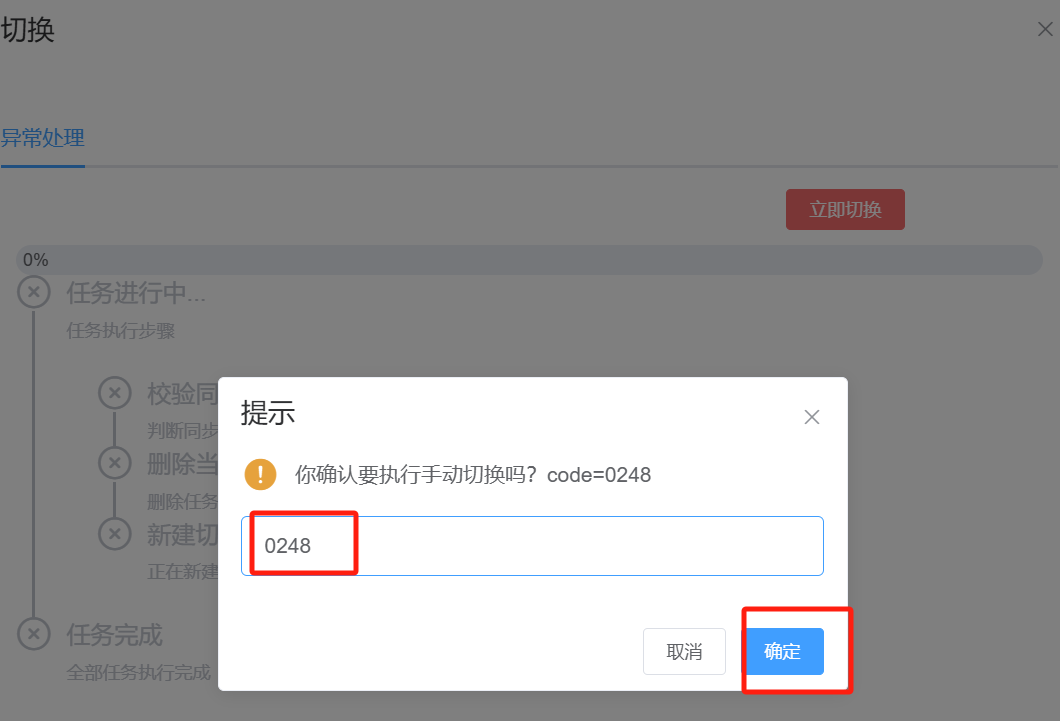
#### 热切换

根据实际情况进行数据同步源和目标端进行切换。**需要注意的是，在热切换前，业务需要停止对源的写入，保证切换后数据一致性。**切换完成后可以将业务写入路由到新的源端上。

当前的源为mysql，目标为proxysql集群：









切换后，源为proxysql集群，目标为mysql集群：



如果需要继续切换回原架构，即源为mysql，目标为proxysql，可以再次操作切换，切换的次数不限制。但需要注意切换前，需要保证当前数据源端无数据写入。

### 注意事项

1. CDC任务同步开启后，由于目前端是没有限制，即目标端不是只读，需要避免目标端有写请求，最终导致源端和目标端的数据不一致。
2. CDC重启后，全量任务会清空，需要清理目标端库表后，重新发起全量同步。
3. 开启event\_file之后，用于记录增量同步的日志，以便于排查增量同步异常引起的数据不一致情况。在CDC服务所在的机器，CDC服务日志目录磁盘空间容量需要注意，一些参考值：

* 200M全量数量同步，CDC需要1.6G左右的空间记录日志。
* 12G全量数量同步，CDC需要28G左右的空间记录日志。

1. 不允许执行跨库create table like语句，如果B库不存在，则同步失败，后续同步任务不会再进行。例如数据同步的是A库，B库再源端存在，但目标端不存在，如使用create table t3 like B.t3;则同步失败。原因：目标端中找不到B库，所以sql执行失败。
2. 在mysql作为源端或者目标端时，需要将参数explicit-defaults-for-timestamp设置为ON，执行语句为，set global explicit-defaults-for-timestamp=ON;
3. 对于timestamp数据类型的字段，强烈不建议使用current\_timestamp做default默认值，原因是有可能造成源端和目标端数据不一致的情况。
4. 如果表中存在float数据类型，包括float和float(M,D)，建议使用double类型替代，是double类型不能指定精度或位数。

## 数据校验

下载和操作见链接：

<https://zettatech.tpddns.cn:10443/wiki/person.winter.guide.mysqlchecker>

源端：MySQL 目标端：Klustron

参考命令：

命令实现指定整个库中的表进行对比

sudo python3 mysqlchecker.py --dump\_tables='mysql2kl.\*' --src\_host=192.168.0.22 --src\_port=3306 --src\_user=root --src\_pass=james123 --src\_db=mysql2kl --dest\_host=192.168.0.17 --dest\_port=13004 --dest\_user=clustmgr --dest\_pass=clustmgr\_pwd --dest\_db=mysql2kl --src=mysql --dest=proxysql --job\_name=mysql --skipcdc --debug

源端：Klustron 目标端：MySQL

参考命令：

命令实现为对比单个表的情况：

sudo python3 mysqlchecker.py --dump\_tables='spm.erp\_response\_log' --src\_host=192.168.0.17 --src\_port=13010 --src\_user=clustmgr --src\_pass=clustmgr\_pwd --src\_db=spm --dest\_host=192.168.0.22 --dest\_port=3306 --dest\_user=root --dest\_pass=james123 --dest\_db=spm --src=proxysql --dest=mysql --skipcdc --job\_name=klustron --debug

数据校验参考值：

## 功能测试内容

根据同步流程及切换过程，可进行功能验证测试

1. 两个库的全量数据一致、建立CDC数据同步链路（MySQL->Klustron）正常；
2. 持续的增量数据写入MySQL；
3. 重点，第一持续写入量增大时，CDC的延迟情况；第二，在持续写的情况下，突然停止写请求后，立即查看MySQL与Klustron的数据是否一致。
4. 如果数据一致，删除原CDC任务，建立新的CDC任务（Klustron->MySQL）后，再将写请求到Klustron中；
5. 重复（3）步骤；
6. 如果数据一致，删除CDC任务，建立新的CDC任务（MySQL->Klustron）后，再将写请求到MySQL中。

## 优化测试内容

测试CDC同步速度全量、增量：

全量数据量：10G、20G、30G、50G、100G、200G、300G、500G

增量数据量：10G、20G；或者数据行数。

同步链路包括：

1. MySQL -> Klustron
2. Klustron -> MySQL

## 故障场景测试

1. 测试内容：全量同步，mysql->proxysql集群，同步过程中，kill掉主proxysql集群的主，并且不让其启动，全量同步完成，对比数据源和目标数据是否一致。

预期结果：kill掉proxysql的主后主备发生切换，选出新主，然后继续同步数据，在选主期间，CDC是有报错连接不上目标端proxysql的，新主选举完成后，数据同步恢复正常。最终数据同步完成后，源和目标两端的数据一致。