# 腾讯云数据仓库 TCHouse-D

# 最佳实践

腾讯云

# 文档目录

# 最佳实践
# 基本功能使用

最近更新时间：2023-11-15 18:13:12

## 建表

### 数据模型选择

Doris 数据模型上目前分为三类: AGGREGATE KEY，UNIQUE KEY，DUPLICATE KEY。三种模型中数据都是按 KEY 进行排序。

1. AGGREGATE KEY
   AGGREGATE KEY 相同时，新旧记录进行聚合，目前支持的聚合函数有 SUM，MIN，MAX，REPLACE。
   AGGREGATE KEY 模型可以提前聚合数据, 适合报表和多维分析业务。

```
CREATE TABLE site_visit
(
 siteid      INT,
 city        SMALLINT,
 username    VARCHAR(32),
 pv BIGINT   SUM DEFAULT '0'
)
AGGREGATE KEY(siteid, city, username)
DISTRIBUTED BY HASH(siteid) BUCKETS 10;
```

2. UNIQUE KEY
   UNIQUE KEY 相同时，新记录覆盖旧记录。目前 UNIQUE KEY 实现上和 AGGREGATE KEY 的 REPLACE 聚合方法一样，二者本质上相同。适用于有更新需求的分析业务。

```
CREATE TABLE sales_order
(
    orderid     BIGINT,
    status      TINYINT,
    username    VARCHAR(32),
    amount      BIGINT DEFAULT '0'
)
UNIQUE KEY(orderid)
DISTRIBUTED BY HASH(orderid) BUCKETS 10;
```

3. DUPLICATE KEY

只指定排序列，相同的行不会合并。适用于数据无需提前聚合的分析业务。

```
CREATE TABLE session_data
(
    visitorid   SMALLINT,
    sessionid   BIGINT,
    visittime   DATETIME,
    city        CHAR(20),
    province    CHAR(20),
    ip          varchar(32),
    brower      CHAR(20),
    url         VARCHAR(1024)
)
DUPLICATE KEY(visitorid, sessionid)
DISTRIBUTED BY HASH(sessionid, visitorid) BUCKETS 10;
```

## 大宽表与 Star Schema

业务方建表时，为了和前端业务适配，往往不对维度信息和指标信息加以区分，而将 Schema 定义成大宽表。对于 Doris 而言，这类大宽表往往性能不尽如人意。

- Schema 中字段数比较多，聚合模型中可能 key 列比较多，导入过程中需要排序的列会增加。
- 维度信息更新会反应到整张表中，而更新的频率直接影响查询的效率。

使用过程中，建议用户尽量使用 Star Schema 区分维度表和指标表。频繁更新的维度表也可以放在 MySQL 外部表中。而如果只有少量更新, 可以直接放在 Doris 中。在 Doris 中存储维度表时，可对维度表设置更多的副本，提升 Join 的性能。

## 分区和分桶

Doris 支持两级分区存储，第一层为分区(partition)，目前支持 RANGE 分区和 LIST 分区两种类型，第二层为 HASH 分桶(bucket)。

1. 分区(partition)
   分区用于将数据划分成不同区间，逻辑上可以理解为将原始表划分成了多个子表。可以方便的按分区对数据进行管理，例如，删除数据时，更加迅速。
   1.1 RANGE 分区。
   业务上，多数用户会选择采用按时间进行partition。
   1.2 LIST 分区。
   业务上，用户可以选择城市或者其他枚举值进行 partition。
2. HASH 分桶(bucket)。
   根据 hash 值将数据划分成不同的 bucket。
   - 建议采用区分度大的列做分桶， 避免出现数据倾斜。
   - 为方便数据恢复，建议单个 bucket 的 size 不要太大, 保持在 10GB 以内，所以建表或增加 partition 时

请合理考虑 bucket 数目，其中不同 partition 可指定不同的 buckets 数。

## 稀疏索引和 Bloom Filter

Doris 对数据进行有序存储，在数据有序的基础上为其建立稀疏索引，索引粒度为 block(1024行)。
稀疏索引选取 schema 中固定长度的前缀作为索引内容, 目前 Doris 选取 36 个字节的前缀作为索引。

- 建表时建议将查询中常见的过滤字段放在 Schema 的前面，区分度越大，频次越高的查询字段越往前放。
- 这其中有一个特殊的地方，就是 varchar 类型的字段。varchar 类型字段只能作为稀疏索引的最后一个字段。索引会在 varchar 处截断，因此 varchar 如果出现在前面，可能索引的长度可能不足 36 个字节。具体可以参阅 数据模型 、ROLLUP 及前缀索引 。
- 除稀疏索引之外，Doris 还提供 bloomfilter 索引， bloomfilter 索引对区分度比较大的列过滤效果明显。 如果考虑到 varchar 不能放在稀疏索引中，可以建立 bloomfilter 索引。

## 物化视图(rollup)

Rollup 本质上可以理解为原始表(Base Table)的一个物化索引。建立 Rollup 时可只选取 Base Table 中的部分列作为 Schema。Schema 中的字段顺序也可与 Base Table 不同。
下列情形可以考虑建立 Rollup：

1. Base Table 中数据聚合度不高。
   这一般是因 Base Table 有区分度比较大的字段而导致。此时可以考虑选取部分列，建立 Rollup。
   如对于 `site_visit` 表：

   ```
   site_visit(siteid, city, username, pv)
   ```

   siteid 可能导致数据聚合度不高，如果业务方经常根据城市统计pv需求，可以建立一个只有 city, pv 的 Rollup：

   ```
   ALTER TABLE site_visit ADD ROLLUP rollup_city(city, pv);
   ```

2. Base Table 中的前缀索引无法命中。
   这一般是 Base Table 的建表方式无法覆盖所有的查询模式。此时可以考虑调整列顺序，建立 Rollup。
   如对于 session_data 表：

   ```
   session_data(visitorid, sessionid, visittime, city, province, ip, brower, url)
   ```

   如果除了通过 visitorid 分析访问情况外，还有通过 brower， province 分析的情形，可以单独建立 Rollup。

   ```
   ALTER TABLE session_data ADD ROLLUP rollup_brower(brower,province,ip,url)
   DUPLICATE KEY(brower,province);
   ```

# Schema Change

Doris 中目前进行 Schema Change 的方式有三种：Sorted Schema Change，Direct Schema Change, Linked Schema Change。

1. Sorted Schema Change

   改变了列的排序方式，需对数据进行重新排序。例如删除排序列中的一列, 字段重排序。

   ```
   ALTER TABLE site_visit DROP COLUMN city;
   ```

2. Direct Schema Change: 无需重新排序，但是需要对数据做一次转换。例如修改列的类型，在稀疏索引中加一列等。

   ```
   ALTER TABLE site_visit MODIFY COLUMN username varchar(64);
   ```

3. Linked Schema Change： 无需转换数据，直接完成。例如加列操作。

   ```
   ALTER TABLE site_visit ADD COLUMN click bigint SUM default '0';
   ```

   建表时建议考虑好 Schema，这样在进行 Schema Change 时可以加快速度。

# 高级特性使用

最近更新时间：2022-03-14 17:19:44

这里我们介绍 Doris 的一些高级特性。

## 表结构变更

使用 ALTER TABLE 命令可以修改表的 Schema，包括如下修改：

- 增加列。
- 删除列。
- 修改列类型。
- 改变列顺序。

以下举例说明。
原表 table1 的 Schema 如下:

```
+----------+-------------+------+-------+---------+-------+
| Field    | Type        | Null | Key   | Default | Extra |
+----------+-------------+------+-------+---------+-------+
| siteid   | int(11)     | No   | true  | 10      |       |
| citycode | smallint(6) | No   | true  | N/A     |       |
| username | varchar(32) | No   | true  |         |       |
| pv       | bigint(20)  | No   | false | 0       | SUM   |
+----------+-------------+------+-------+---------+-------+
```

我们新增一列 uv，类型为 BIGINT，聚合类型为 SUM，默认值为 0:
`ALTER TABLE table1 ADD COLUMN uv BIGINT SUM DEFAULT '0' after pv;`
提交成功后，可以通过以下命令查看作业进度:
`SHOW ALTER TABLE COLUMN;`
当作业状态为 FINISHED，则表示作业完成。新的 Schema 已生效。
ALTER TABLE 完成之后, 可以通过 `DESC TABLE` 查看最新的 Schema。

```
mysql> DESC table1;
+----------+-------------+------+-------+---------+-------+
| Field    | Type        | Null | Key   | Default | Extra |
+----------+-------------+------+-------+---------+-------+
| siteid   | int(11)     | No   | true  | 10      |       |
| citycode | smallint(6) | No   | true  | N/A     |       |
| username | varchar(32) | No   | true  |         |       |
| pv       | bigint(20)  | No   | false | 0       | SUM   |
| uv       | bigint(20)  | No   | false | 0       | SUM   |
```

```
+----------+-------------+------+-------+---------+-------+
5 rows in set (0.00 sec)
```

可以使用以下命令取消当前正在执行的作业: `CANCEL ALTER TABLE COLUMN FROM table1`
更多帮助，可以参阅 `HELP ALTER TABLE` 。

## Rollup

Rollup 可以理解为 Table 的一个物化索引结构。**物化** 是因为其数据在物理上独立存储，而 **索引** 的意思是，Rollup可以调整列顺序以增加前缀索引的命中率，也可以减少key列以增加数据的聚合度。
以下举例说明。
原表 table1的 Schema如下:

```
+----------+-------------+------+-------+---------+-------+
| Field    | Type        | Null | Key   | Default | Extra |
+----------+-------------+------+-------+---------+-------+
| siteid   | int(11)     | No   | true  | 10      |       |
| citycode | smallint(6) | No   | true  | N/A     |       |
| username | varchar(32) | No   | true  |         |       |
| pv       | bigint(20)  | No   | false | 0       | SUM   |
| uv       | bigint(20)  | No   | false | 0       | SUM   |
+----------+-------------+------+-------+---------+-------+
```

对于 table1 明细数据是 siteid, citycode, username 三者构成一组 key，从而对 pv 字段进行聚合；如果业务方经常有看城市 pv 总量的需求，可以建立一个只有 citycode, pv 的rollup。

`ALTER TABLE table1 ADD ROLLUP rollup_city(citycode, pv);`

提交成功后，可以通过以下命令查看作业进度： `SHOW ALTER TABLE ROLLUP;` ，当作业状态为 FINISHED，则表示作业完成。
Rollup 建立完成之后可以使用 `DESC table1 ALL` 查看表的 Rollup 信息。

```
mysql> desc table1 all;
+-------------+---------+-------------+------+-------+---------+-------+
| IndexName   | Field   | Type        | Null | Key   | Default | Extra |
+-------------+---------+-------------+------+-------+---------+-------+
| table1      | siteid  | int(11)     | No   | true  | 10      |       |
|             | citycode | smallint(6) | No   | true  | N/A     |       |
|             | username | varchar(32) | No   | true  |         |       |
|             | pv      | bigint(20)  | No   | false | 0       | SUM   |
|             | uv      | bigint(20)  | No   | false | 0       | SUM   |
|             |         |             |      |       |         |       |
| rollup_city | citycode | smallint(6) | No   | true  | N/A     |       |
|             | pv      | bigint(20)  | No   | false | 0       | SUM   |
+-------------+---------+-------------+------+-------+---------+-------+
```

```
8 rows in set (0.01 sec)
```

可以使用以下命令取消当前正在执行的作业： `CANCEL ALTER TABLE ROLLUP FROM table1;` 。
Rollup 建立之后，查询不需要指定 Rollup 进行查询。还是指定原有表进行查询即可。程序会自动判断是否应该使用 Rollup。是否命中 Rollup可以通过 `EXPLAIN your_sql;` 命令进行查看。
更多帮助，可以参阅 `HELP ALTER TABLE` 。

# 数据表的查询

## 内存限制

为了防止用户的一个查询可能因为消耗内存过大。查询进行了内存控制，一个查询任务，在单个 BE 节点上默认使用不超过2GB 内存。
用户在使用时，如果发现报 `Memory limit exceeded` 错误，一般是超过内存限制了。遇到内存超限时，用户应该尽量通过优化自己的 sql 语句来解决。如果确切发现2GB 内存不能满足，可以手动设置内存参数。
显示查询内存限制:

```
mysql> SHOW VARIABLES LIKE "%mem_limit%";
+---------------+------------+
| Variable_name | Value      |
+---------------+------------+
| exec_mem_limit| 2147483648 |
+---------------+------------+
1 row in set (0.00 sec)
```

`exec_mem_limit` 的单位是 byte，可以通过 `SET` 命令改变 `exec_mem_limit` 的值。如改为 8GB。
`SET exec_mem_limit = 8589934592;`

```
mysql> SHOW VARIABLES LIKE "%mem_limit%";
+---------------+------------+
| Variable_name | Value      |
+---------------+------------+
| exec_mem_limit| 8589934592 |
+---------------+------------+
1 row in set (0.00 sec)
```

**说明**

- 以上该修改为 session 级别，仅在当前连接 session 内有效。断开重连则会变回默认值。
- 如果需要修改全局变量，可以这样设置： `SET GLOBAL exec_mem_limit = 8589934592;` 。设置完成后，断开 session 重新登录，参数将永久生效。

### 查询超时

当前默认查询时间设置为最长为 300 秒，如果一个查询在 300 秒内没有完成，则查询会被 Doris 系统 cancel 掉。用户可以通过这个参数来定制自己应用的超时时间，实现类似 wait(timeout) 的阻塞方式。
查看当前超时设置:

```
mysql> SHOW VARIABLES LIKE "%query_timeout%";
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| QUERY_TIMEOUT | 300   |
+---------------+-------+
1 row in set (0.00 sec)
```

修改超时时间到1分钟:

```
SET query_timeout = 60;
```

> **说明**
>
> - 当前超时的检查间隔为 5 秒，所以小于 5 秒的超时不会太准确。
> - 以上修改同样为 session 级别。可以通过 `SET GLOBAL` 修改全局有效。

## Broadcast/Shuffle Join

系统默认实现 Join 的方式，是将小表进行条件过滤后，将其广播到大表所在的各个节点上，形成一个内存 Hash 表，然后流式读出大表的数据进行Hash Join。但是如果当小表过滤后的数据量无法放入内存的话，此时 Join 将无法完成，通常的报错应该是首先造成内存超限。
如果遇到上述情况，建议显式指定 Shuffle Join，也被称作 Partitioned Join。即将小表和大表都按照 Join 的 key 进行 Hash，然后进行分布式的 Join。这个对内存的消耗就会分摊到集群的所有计算节点上。
Doris会自动尝试进行 Broadcast Join，如果预估小表过大则会自动切换至 Shuffle Join。注意，如果此时显式指定了 Broadcast Join 也会自动切换至 Shuffle Join。
使用 Broadcast Join（默认）:

```
mysql> select sum(table1.pv) from table1 join table2 where table1.siteid = 2;
+--------------------+
| sum(`table1`.`pv`) |
+--------------------+
|                 10 |
+--------------------+
1 row in set (0.20 sec)
```

使用 Broadcast Join（显式指定）:

```
mysql> select sum(table1.pv) from table1 join [broadcast] table2 where table1.siteid
= 2;
+--------------------+
| sum(`table1`.`pv`) |
+--------------------+
|                 10 |
+--------------------+
1 row in set (0.20 sec)
```

使用 Shuffle Join:

```
mysql> select sum(table1.pv) from table1 join [shuffle] table2 where table1.siteid =
2;
+--------------------+
| sum(`table1`.`pv`) |
+--------------------+
|                 10 |
+--------------------+
1 row in set (0.15 sec)
```

## 查询重试和高可用

当部署多个 FE 节点时，用户可以在多个 FE 之上部署负载均衡层来实现 Doris 的高可用。
以下提供一些高可用的方案：

**第一种**

自己在应用层代码进行重试和负载均衡。例如发现一个连接挂掉，就自动在其他连接上进行重试。应用层代码重试需要应用自己配置多个doris前端节点地址。

**第二种**

如果使用 mysql jdbc connector 来连接Doris，可以使用 jdbc 的自动重试机制：

```
jdbc:mysql://[host:port],[host:port].../[database][?propertyName1][=propertyValue1]
[&propertyName2][=propertyValue2]...
```

**第三种**

应用可以连接到和应用部署到同一机器上的 MySQL Proxy，通过配置 MySQL Proxy 的 Failover 和 Load Balance 功能来达到目的。

http://dev.mysql.com/doc/refman/5.6/en/mysql-proxy-using.html

# 资源规格选型及调优建议

最近更新时间：2023-09-05 12:26:24

本文将为您介绍如何选择腾讯云数据仓库 TCHouse-D 的实例规格，并会给出资源不足时的调优建议。

## 资源规格及适配场景

购买 Doris 集群时，需要选择 FE 节点、BE 节点的计算资源规格和存储资源规格，并选择是否开启高可用。

### 资源规格及建议场景

| 机型类型 | 计算节点规格 | 建议存储类型 | 建议场景 |
| --- | --- | --- | --- |
| 标准型 | 4核16G | • 高性能云硬盘<br>• SSD 云硬盘<br>• 增强型 SSD 云硬盘 | 仅限于 **POC 功能测试或个人学习使用**，主要用于体验测试产品能力。 |
| | 8核32G | • 高性能云硬盘<br>• SSD 云硬盘<br>• 增强型 SSD 云硬盘 | 推荐用于**测试环境**，可支持中等数据规模、较复杂的数据分析 |
| | 16核64G | • 高性能云硬盘<br>• SSD云硬盘<br>• 增强型 SSD 云硬盘 | 推荐用于**生产环境**，可支持较大规模、较复杂场景的数据分析，及高并发场景 |
| | 32核128G | • 高性能云硬盘<br>• SSD 云硬盘<br>• 增强型 SSD 云硬 | 生产环境推荐配置，可支持大量高复杂度数据分析，高并发等场景 |

| | | | |
|---|---|---|---|
| | | 盘 | |
| | 64核<br>256G | • 高性能云硬盘<br>• SSD云硬盘<br>• 增强型SSD 云硬盘 | 适合企业级平台建设，适用于高并发场景，大规模企业核心数据平台推荐选择。 |
| 高性能型 | 16核<br>64G | 本地盘 | 生产场景对数据吞吐有强要求时，可以选择此机型。<br>**说明：此机型选定后不支持升降配。** |
| | 32核<br>128G | 本地盘 | |
| | 64核<br>256G | 本地盘 | |

## 高可用及节点数量建议

| 场景 | 建议最小 FE 节点数 | 建议最小 BE 节点数 |
|---|---|---|
| POC 功能测试 | 1个 | 3个 |
| 生产场景 | 建议开启高可用，最少3个FE节点 | 最少3个 BE 节点，按需扩容 |

## 资源规格选型参考

> ⚠ **注意：**
> 以下内容仅供参考，不同业务场景下性能可能会有较大的差异。

1. **场景一：产品功能验证，进行简单数据分析**
   FE：不开启高可用，单节点4核16G
   BE：3节点，每个节点4核16G
2. **场景二：中小规模数据简单查询，如百GB数据量级，1000QPS以下**
   FE：不开启高可用，单节点8核32G
   BE：3节点，每个节点8核32G
3. **场景三：生产场景，TB级数据量，涉及多表关联、GROUP BY 等复杂查询**
   FE：开启高可用，3节点，每个节点16核64G
   BE：3节点，每个节点16核64G
4. **场景四：生产业务，TB级数据量，涉及复杂查询，涉及大量高并发点查**
   FE：开启高可用，3节点，每个节点16核64G

BE：6节点，每个节点16核64G

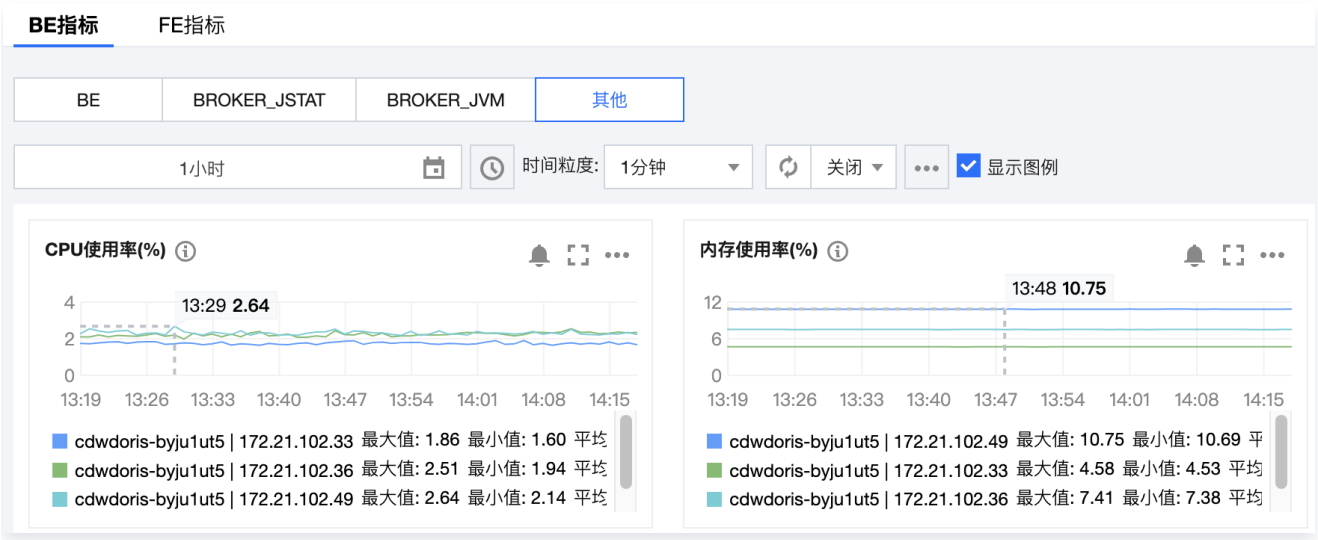
## 资源监控及调优建议

大批量数据导入、数据查询、并发查询、多表关联 join 等操作都会导致 CPU、内存的大量占用，若CPU/内存使用率持续超过85%会导致集群不稳定，建议优化业务或变配。

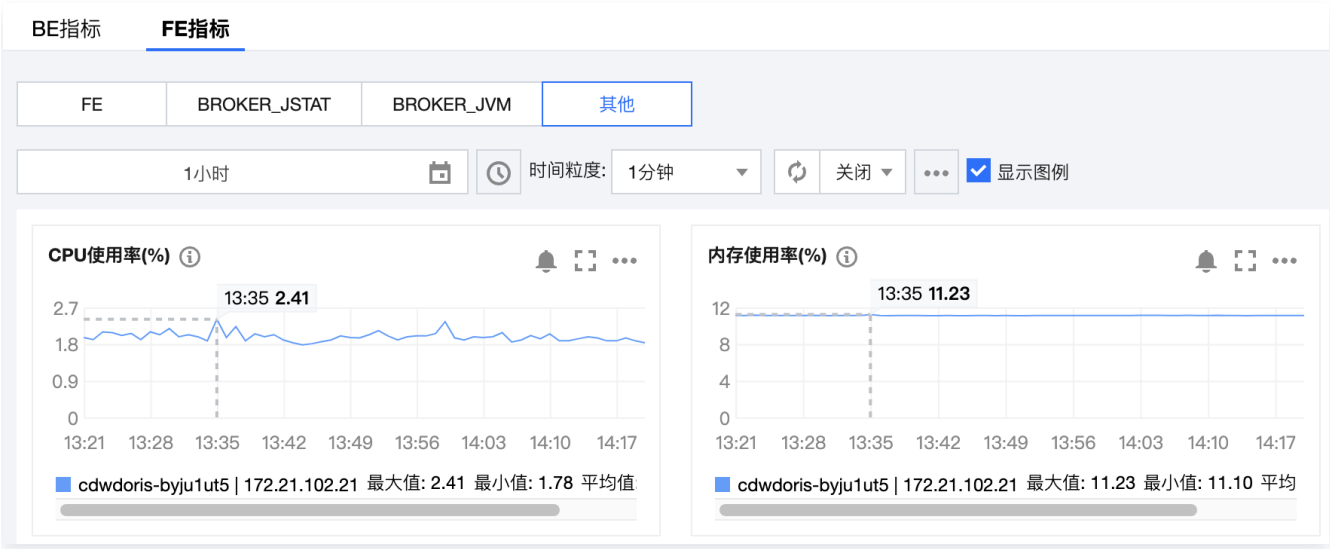
### 资源使用监控

可在**集群管理 > 集群监控**中查看 BE、FE 各节点的 CPU、内存使用情况，如下图所示。

- 集群监控 > BE 指标 > 其他




- 集群监控 > FE 指标 > 其他




## 资源扩容建议

FE 和 BE 的 CPU、内存使用率持续超过85%时，就需要考虑进行资源升配或扩容。

> **说明：**

导致 FE 和 BE 的 CPU、内存大量占用的主要原因如下：

- FE CPU 大量占用：多并发查询、大量复杂查询。
- FE 内存大量占用：元数据过多（分区不合理等）、频繁进行表删除等。
- BE CPU 大量占用：大量数据导入、大量复杂查询（如聚合查询）等。
- BE 内存大量占用：大量数据导入、大量复杂查询（如聚合查询）等。

| 常见场景 | 资源耗用表现 | 使用率持续超过85%时调优建议 |
| --- | --- | --- |
| 过多数据持续导入 | FE 和 BE 的 CPU、内存都会被大量占用 | • 如果是 FE 瓶颈：建议纵向升配<br>• 如果是 BE 瓶颈：建议纵向升配 |
| 点查较多/高并发 | FE 和 BE 的 CPU 都会被大量占用 | • 如果是 FE 瓶颈：建议纵向升配<br>• 如果是 BE 瓶颈：建议纵向升配 |
| 元数据频繁变更删除 | FE 内存大量占用 | • 建议 FE 纵向升配，增加内存 |
| 多表关联/聚合查询较多 | BE 的 CPU、内存会大量占用 | • 优先建议 BE 横向扩容，也可纵向升配 |
| 数据多并发度写入 | BE 的 CPU、内存会大量占用 | • 优先建议 BE 横向扩容，也可纵向升配 |

## 集群扩缩容注意事项

| 操作类型 | 注意事项 |
| --- | --- |
| 水平扩容 | • 水平扩容过程中，系统读写仍可进行，但是可能出现一些抖动，执行操作大约需要5 – 15分钟，请选择在非业务高峰期进行。<br>• 在数据存储量及查询量均相对增长时，优先选择水平扩容。 |
| 水平缩容 | • 只能每次选择一类节点进行缩容操作，如仅缩容 FE 或 仅缩容 BE。<br>• FE 缩容：可一次性缩容多个。<br>• BE 缩容：一次性缩容多个 BE 节点有可能导致数据丢失或时间过长，建议逐个缩容。<br>• 缩容过程中，系统读写仍可进行，但是可能出现一些抖动。 |
| 垂直升配/降配 | • 垂直变配系统不可读、不可写。<br>• 计算规格支持升配、降配；存储规格仅支持升配。<br>• 变配操作结果对集群所有节点均生效。 |

## 业务调优建议

| 调优类型 | 调优说明 |
| --- | --- |
| 使用建议 | • 如果经常对某列进行点查，且列的基数较高，建议在此列创建 bloom filter 索引。<br>• 如果经常对某表进行模式固定的聚合查询，建议在此表创建物化视图。<br>• 建议结合业务场景合理分区分桶，避免分区分桶过多占用FE内存。<br>• 普通数据探查的 sql，如果不需要全部数据，建议加上limit返回条数限制，也可加速查询。<br>• 导入数据建议用 CSV，避免 Json 数据格式。 |
| 尽量避免 | • 避免 select * 查询；<br>• 避免全局开profile（会带来较多资源开销，建议针对需要的 SQL 开 profile）<br>• 建表时：避免开启 merge_on_write（此功能暂不成熟）<br>• 建表时：避免开启 auto bucket（此功能暂不成熟）<br>• 建表时：避免开启动态 Schema 表（此功能暂不成熟）<br>• 避免多个大表 Join，涉及多个大表关联时：<br>  ○ 可转为大表两两 join，并使用 Colocation Join。<br>  ○ 或使用预聚合表、索引等进行查询加速。 |
| 参数调优 | • 一条 SQL 涉多并发时，建议调大 `parallel_fragment_exec_instance_num` 参数，此参数默认值200，可按倍数调大（如400、800），建议控制在2000以内。<br>• 建议控制 compaction 速度，若监控指标 `base_compaction_score` 超过200且持续上升的话（具体可在“集群监控-BE指标-BE”页查看），可以将 compaction_task_num_per_disk 参数配置调大（系统默认2，可调大至4或更多）。 |

# 表设计与数据导入

最近更新时间：2023-11-21 18:41:42

## 如何选择数据模型?

- 对于本身已完成聚合的数据，选择 Aggregate/Duplicate 模型均可。
- 对于清洗已完成的明细数据，若需要聚合后查询统计值，选择 Aggregate 模型。
- 对于清洗已完成的明细数据，若需要做明细级的查询，选择 Duplicate 模型（没有预聚合，聚合查询效率较低，可以通过物化视图，提升聚合查询效率）。
- 如果有**唯一的 Key**，需要按 Key 去重，选择 **Unique 模型**。

## 如何设置分区?

- 分区字段的值分布尽量分散，建议选择日期或 ID 等作为分区字段。

## 如何设置副本?

- 副本数量可以设为3（也可以设为2，但不建议设为1，原因是后续滚动升级会出现数据不可用状态）。

## 如何设置分桶?

- 采用一些 Hash 均匀的 Key 作为分桶 Key，避免数据倾斜。
- 创建的分桶数不宜过多或者过少，建议每个分桶最好保持在1 – 10G之间，对于小表，一般几个分桶已经足够。
- 分桶 Key 可以一个或者多个，多个保证数据分布更均衡，单个容易匹配命中（单个分桶一般选择区分度较高的 Key）。

## 如何选择字段格式?

- 表的每一个字段优先考虑使用**整型**的类型，而不是 string 等类型，能够极大地促进查询和版本合并效率。
- 对于浮点数使用 decimal，而不是 double，float。

## 数据导入注意事项

- 实时导入建议采用 Stream load，离线导入建议采用 Broker load，导入的基本原则就是批量导入，**减少并发，尽可能一次性导入尽量多的数据**，减少合并的成本，也尽量避免影响读的效率。（例如一分钟总导入次数不得超过20次，考虑各种并发在内，高频导入目前不适合）。**小文件太多可严重影响查询效率**。
- 对于作为 Hash key 的字段在数据导入的时候一定要注意 **NULL 值的过滤处理**，避免出现数据倾斜。出现数据倾斜将导致扩容机器**无法缓解集群压力**，另外容易导致**集群不稳定**。
- 数据导入是要**指定要导入的分区**，否则大表导入数据容易导致失败。
- 导入数据建议用 CSV，避免 JSON 数据格式。

## Schema Change 注意事项

为了集群稳定性考量和业务的实际需求，我们建议创建表之前做好字段类型的评估，业务修改表的 Schema，只建议 **Add Column** 操作，我们保障在尽可能短的时间内完成新列的增加。

## 数据清理注意事项

如果要清空分区的数据，建议优先考虑 **truncate** 操作（等同于 drop 分区，然后 create 分区的操作）而不是 delete 操作，delete 操作对查询性能有较大影响。

## 其他

对于不熟悉的操作，优先考虑在命令行键入 help keyword 寻求帮助（例如 help stream load 可以了解如何进行数据实时导入），或者 提交工单 以获得更深入的帮助来解决您的问题。

# 查询调优

最近更新时间：2023-11-21 18:41:42

## 基础查询调优

- 查询分区表时，一定要带上**分区字段**，Explain + SQL 可以协助用户分析查询了几个分区和 tablet 的数据。
- 查询 SQL 条件最好能命中分区 Key 和分桶 Key。
- 查询 SQL 条件最好能命中前缀索引。
- 由于 Doris 是列存数据库，当查询的字段足够多的时候，可能性能还不如行式存储，建议查询时尽可能选择具体的字段代替 *，在查询的最后方加上 **limit number** 的限制。
- 执行 Select 操作的时候尽可能**避免写成 function(column)= "xxxx"**的形式，这样将导致无法发挥 Doris 系统谓词下推的优势，左侧应为列名，右侧应该为可以计算展平的常数值。
- 查询尽可能避免使用 or，union all 的情况，在大多数场景下，考虑使用 **in 代替 or**。
- 普通数据探查的 SQL，如果不需要全部数据，建议加上 limit 返回条数限制，也可加速查询。

## Join 优化

- **Shuffle 方式优化**：效率为 **Colocate join > Bucket Shuffle > Shuffle > BroadCast**，具体参见 Bucket Shuffle Join 。
- **RuntimeFilter**：join 查询中，存在除了关联条件之外，右边有其他过滤条件。

## 使用 Rollup

- 查询无法覆盖基表前缀索引，通过 Rollup 调整 Key 顺序形成前缀索引。
- 对 Aggregate 表进行 Key 筛选聚合

## 使用物化视图

如果经常对某表进行模式固定的聚合查询，建议在此表创建物化视图；

- Rollup 支持的场景都能用；
- 对 Duplicate 表形成额外聚合。

详情请参见 物化视图 。

## 索引优化

- **Bitmap 索引**：取值基数比较小的列[100-100000]，查询条件命中列。
- **BloomFliter 索引**：如果经常对某列进行精确点查，且列的基数较高，建议在此列创建 Bloom filter 索引。

## 使用 Cache

- **PageCache**：此配置默认开启。

- **SqlCache**：此配置默认关闭。并发高，查询结果集较小时效果好。

# 建议规避的用法

最近更新时间：2023-11-28 16:14:11

## 建议避免的场景

- 避免在生产集群大规模周期性调度离线/批 ETL 作业（insert into select / create table as select），尤其在同一个集群中同时运行离线、在线业务，离线作业会占用较大资源从而影响在线业务的稳定性与性能。

> **说明：**
> 建议离线/在线业务通过不同的集群隔离，或提前通过 Spark 完成离线处理后，再将数据写入 Doris。

- 避免逐条 insert into：Doris 每个 insert into 都是一个事务，逐条写入可能导致并发超过事务上限。

> **说明：**
> 建议进行攒批，如每个 insert into 几十或上百条数据，以降低写入压力。

- **1.2内核版本**：尽量避免使用复杂数据类型（例如 MAP、ARRAY、STRUCT 等）。
- **1.2内核版本**：对复杂数据类型的支持不够完善，部分写入和查询可能会报错。

## 建议避免的查询

- 尽量避免在多列且数据规模较大的表上进行 select * 查询。
- 避免全局开 profile（这会带来较大的资源开销，因此建议仅对需要的 SQL 语句开启 profile）。
- 尽量避免多个大表 Join。

> **说明：**
> 涉及多个大表关联时，建议可转为大表两两 join 并使用 Colocation Join，或使用预聚合表、索引等进行查询加速。

## 建议避免的功能

- **1.2内核版本**：尽量避免开启 merge_on_write（此功能暂不成熟）。
- **1.2内核版本**：尽量避免开启 Light scheme change（此功能暂不成熟）。

# 通过 DataInLong 将 MySQL 数据同步至 TCHouse-D

## 概述

最近更新时间：2023-09-05 12:26:24

本文将为您介绍如何通过 DataInLong 数据集成将 MySQL 中的数据实时导入腾讯云数据仓库 TCHouse-D 中，此处以腾讯云数据库 MySQL 为例，介绍实时同步任务的配置及操作实践。

> **注意：**
> 建议提前合理分配及配置网络环境，确保 MySQL、TCHouse-D、数据集成资源之间网络互通。
> - 若 MySQL、TCHouse-D、DataInLong 集成资源 处于同一个 VPC 内：此时网络联通，可直接使用。（**推荐方式**）
> - 若与集成资源位于不同 VPC：需购买对等连接打通集成与数据源所在 VPC。
> - 若数据源位于 IDC 或其他经典网络环境下：可购买 VPN 连接或专线网关打通集成资源与 MySQL/TCHouse-D 集群所在 VPC。
> - 若数据源可开通公网：可购买 NAT 网关，允许集成资源通过网关连通数据源所在 VPC。NAT 网关配置流程请参见 集成资源组配置公网 。

### 步骤1：登录注册

登录腾讯云官网。如果没有账号，请参考 账号注册教程 。

### 步骤2：提前准备好 MySQL 数据库实例

数据集成目前支持 MySQL 数据库版本为：5.6，5.7，8.0.x。具体操作请参见 源端 MySQL 准备 。

### 步骤3：提前开通好集群

建议选择内核版本：1.1、1.2或以上。具体操作请参见 目标端 TCHouse-D 准备 。

### 步骤4：购买 DataInlong，并配置项目空间和集成资源

建议选择内核版本：1.1、1.2或以上。具体操作请参见 配置 DataInlong 项目空间及集成资源 。

### 步骤5：在 WeData 或 DataInlong 配置数据实时同步任务

- 需要进行单表同步时，可配置单表实时同步任务，具体操作请参见 配置单表实时同步任务 。
- 需要进行整库迁移时，可配置整库实时迁移任务，具体操作请参见 配置整库实时迁移任务 。

### 步骤6：进入运维中心进行任务运维

具体操作请参见 任务运维 。

# 源端 MySQL 准备

最近更新时间：2023-09-05 12:26:24

## 新建 MySQL 数据源实例

> **说明**
> 目前数据集成支持 MySQL 数据库版本为：5.6，5.7，8.0.x。

### 使用腾讯云 MySQL 时

1. 您可登录 云数据库 TencentDB 控制台，进入MySQL 实例列表。
2. 单击**新建**购买指定数据库版本的云实例。

> **注意：**
> 购买 MySQL 云实例时，建议配置 MySQL、Doris 处于同一个 VPC。


### 使用非腾讯云 MySQL 时

您可以在 MySQL 数据库中通过如下语句查看当前 MySQL 数据库版本，检查当前待同步的 MySQL 是否符合版本要求。

```
select version();
```

## 创建账号并赋权

> **注意：**
> 为保证实时数据同步顺利进行，您必须定义一个对 Debezium MySQL 连接器监控的所有数据库具有适当权限的 MySQL 用户。该 MySQL 账号必须拥有数据库的 SELECT、REPLICATION SLAVE 和 REPLICATION CLIENT 权限。

### 使用腾讯云 MySQL 时

1. 您可登录 云数据库 TencentDB 控制台，单击**实例 ID/名称**进入实例详情页。
2. 进入**数据库管理 > 账号管理** 页面，单击**创建账号**来新增账号，**修改权限**来配置账号权限。


### 使用非腾讯云 MySQL 时

您需要通过SQL语句授予并刷新账号权限。

```
mysql> GRANT SELECT, SHOW DATABASES, REPLICATION SLAVE, REPLICATION CLIENT
ON *.* TO 'user' IDENTIFIED BY 'password';
mysql>FLUSH PRIVILEGES;
```

> **说明：**
> 1. 启用 `scan.incremental.snapshot.enabled` 时不再需要 RELOAD 权限（默认启用）。
> 2. 查看更多关于权限说明。

## 创建数据库

### 使用腾讯云 MySQL 时

1. 您可登录 云数据库 TencentDB 控制台，单击**实例 ID/名称**进入实例详情页。
2. 进入**数据库管理 > 数据库列表** 页面，单击**创建数据库**来创建 MySQL数据库。


3. 在数据库登录跳转页面输入**账号管理**中已创建好的账号名和密码。


## 使用非腾讯云 MySQL 时

您可以通过 MySQL Client 等客户端进行建库操作。

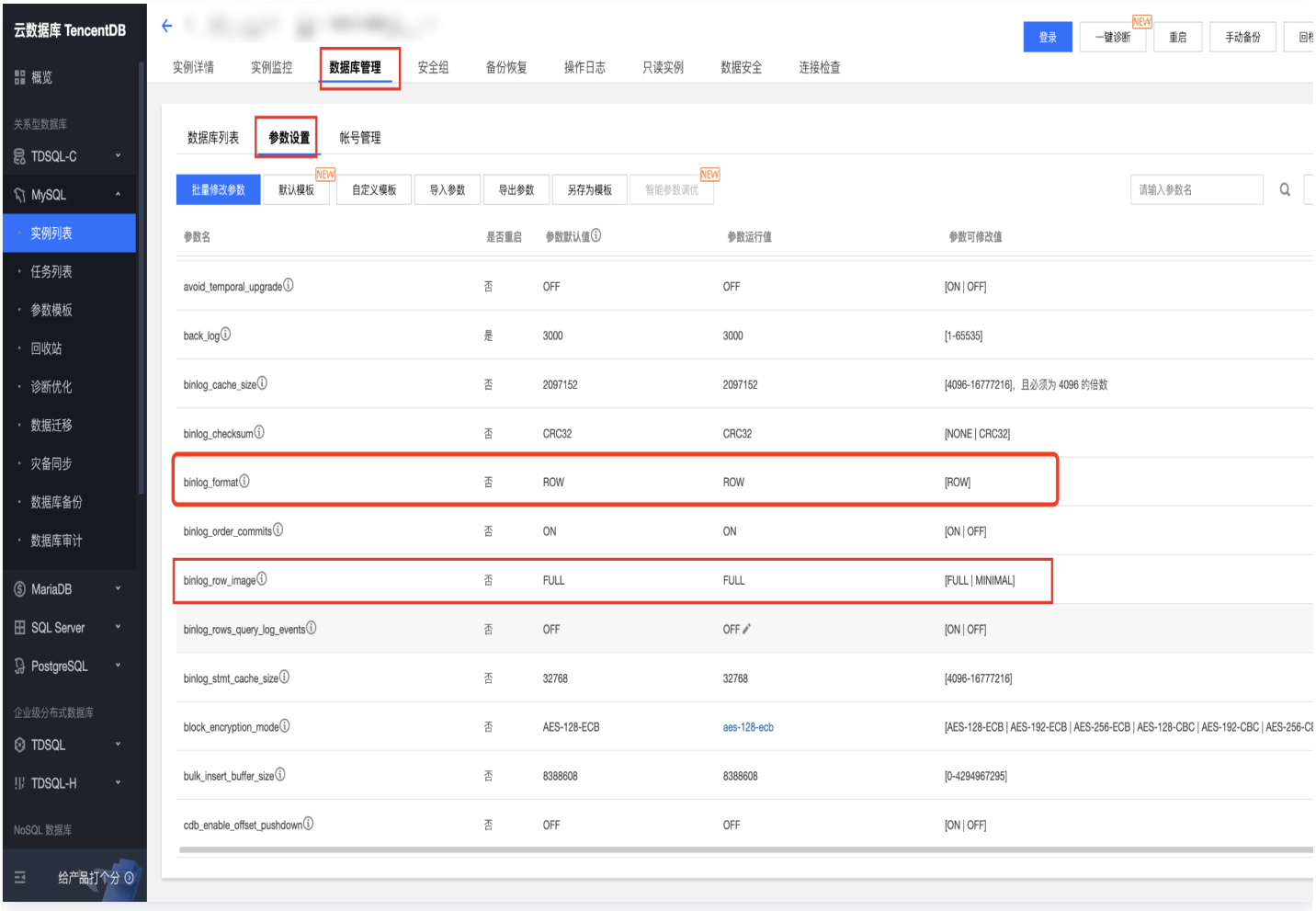
# 开启 Binlog 并确认 Binlog 格式

数据集成基于 MySQL binlog 进行数据同步时，要求 MySQL 服务器开启 binlog，并将 binlog 格式符配置为 ROW、将 binlog_row_image 配置格式为 FULL。

## 使用腾讯云 MySQL 时

腾讯云 MySQL 默认已开启 Binlog，可进入对应 MySQL 数据库管理-参数设置 页面设置并管理对应 binlog 参数：

> **说明：**
> MySQL 8.X 默认 binlog_format 为 ROW，无需额外配置。




## 使用非腾讯云 MySQL 时

可通过以下命令查看并配置 binlog 格式

```
show variables like "binlog_format";
show variables like "binlog_row_image";
```

## 创建数据表

### 使用腾讯云 MySQL 时

1. 您可进入 数据库管理，登录腾讯云 MySQL 。


2. 在数据库管理页面新建数据表。


## 使用非腾讯云 MySQL 时

您可以通过 MySQL Client 等客户端进行建表操作。

# 目标端 Doris 准备

最近更新时间：2023-09-05 12:33:42

## 购买 Doris 集群

- 购买方式一：进入腾讯云官网，登录后在官网首页单击**立即选购**。
- 购买方式二：登录腾讯云数据仓库 TCHouse-D 控制台，在集群列表页新建集群。


- 进入集群购买页面，按需选购 FE、BE 的集群资源规格，详细操作步骤可参见 新建集群 。

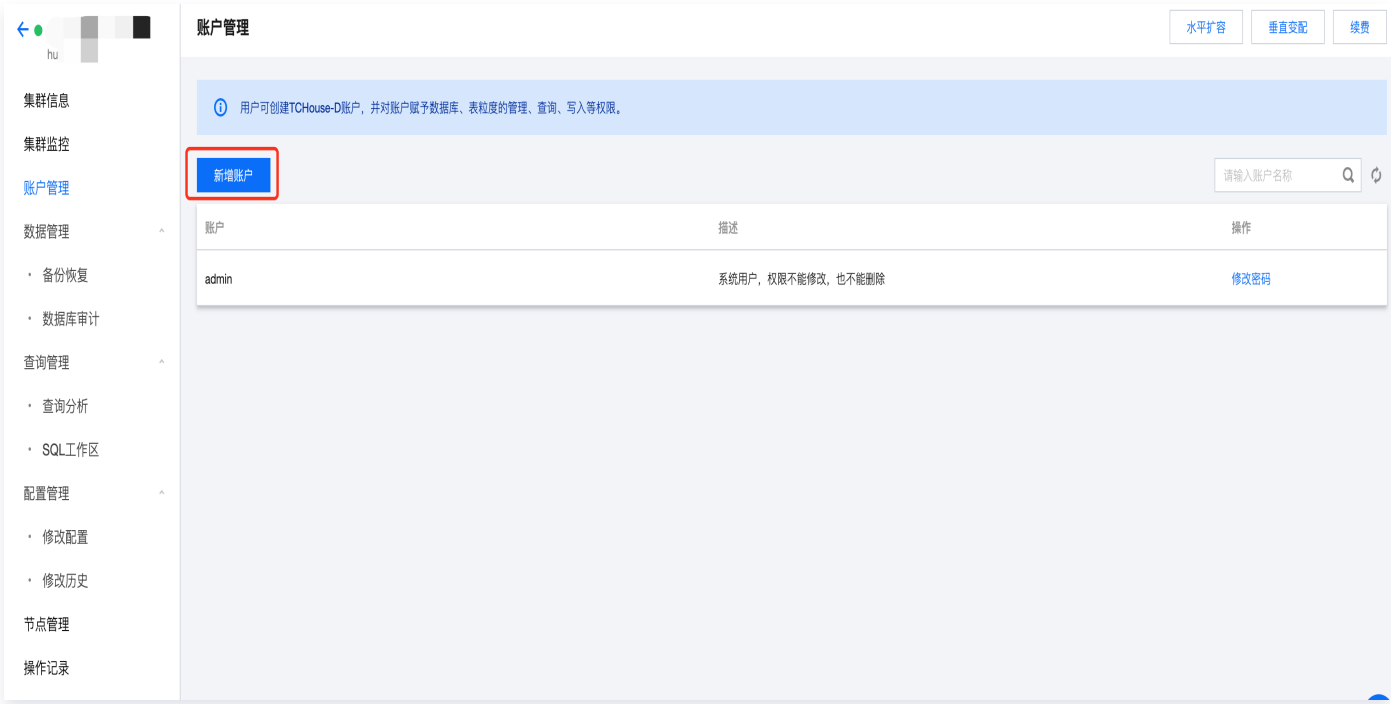
> **注意：**
> - 建议生产集群配置如下：
>  - FE：开启高可用，3节点，每个节点16核64G。
>  - BE：3节点，每个节点16核64G。
> - 购买 Doris 集群时，建议配置 MySQL、Doris 处于同一个 VPC 内。

## 创建账户并赋权

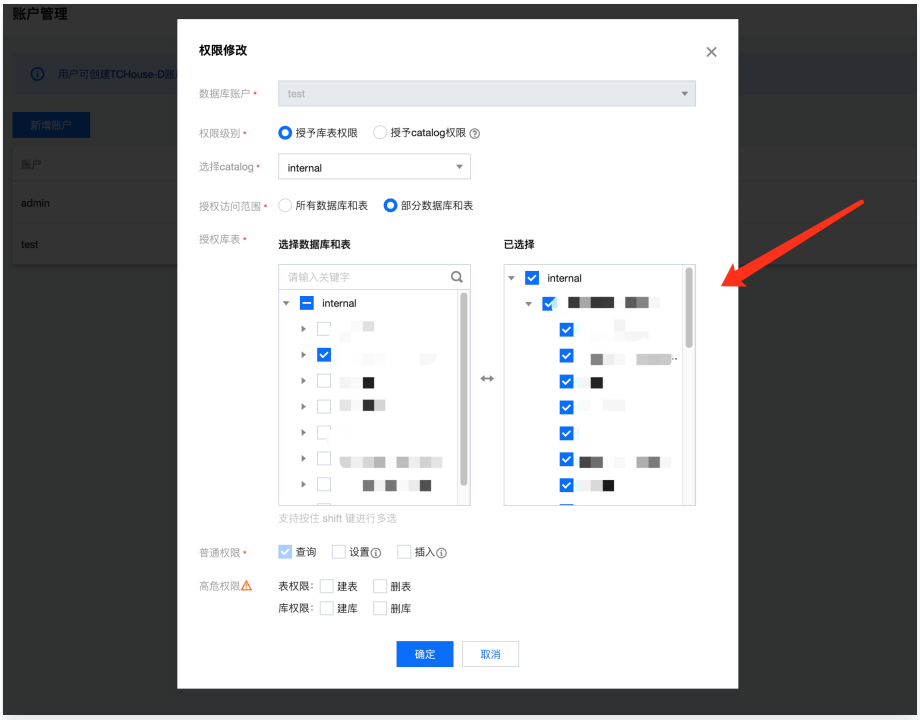
1. 集群购买完毕后，可进入 腾讯云数据仓库 TCHouse-D 控制台 集群列表，单击**集群 ID/名称**进行集群管理。


2. 可在“集群管理”模块查看集群具体信息、查看 BE/FE 监控指标、配置备份恢复策略等，详细操作说明可参见 集群常规操作 。
3. 为更方便的进行 Doris 库表操作，需进入集群详情页，单击**账户管理**，创建库表并进行用户授权。单击**新增账**

户，输入账户、名称后即可完成账号创建。




4. 账号创建完毕后，在账户列表单击**修改权限**按钮，即可为账号赋予相应的数据权限，支持为账户授予全部库表权限或指定表的权限，详细操作请参见 账户管理和权限管理 。




**注意：**

授权所有数据库和表后，也将获取外部数据源的权限，但外部数据源仅支持查询，不支持插入、增删库表等。

# 进入 SQL 工作区


- 进入 SQL 工作区 时，需指定集群和登录账号。
  - Admin 用户登录后，默认有所有库表的权限。
  - 其他账号登录后，只能操作自己有权限的库表（可参见上文账号管理）。

> **说明：**
> **若忘记密码：**
> - 普通账户可在 **集群详情页 > 账户管理** 中直接重置密码。
> - Admin 账户可通过 提交工单 联系我们重置密码。

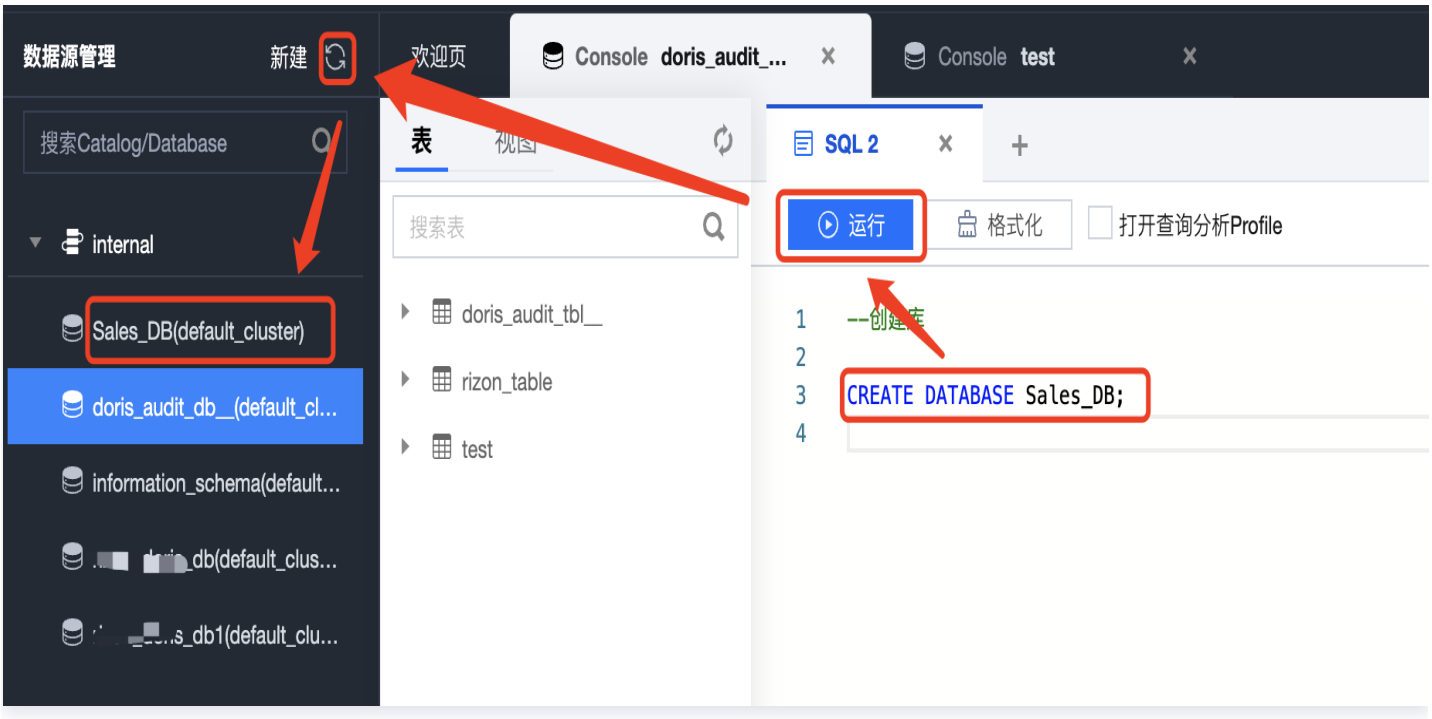

# 创建 Doris 数据库

> **注意：**
> 集群创建成功后，会初始化内置catalog（Internal）和2个系统数据库，请勿对系统数据库进行增删改等操作：
> 1. 系统库 doris_audit_db__：审计日志数据库，用于记录 Doris 系统的操作日志和安全事件。通过审计日志，可以追踪系统的操作历史和安全事件，保证系统的安全性和可靠性。
> 2. 系统库 information_schema：即 Doris 的元数据数据库，可通过查询 information_schema 来获取系统的元数据信息，了解系统的结构和属性，方便进行数据管理和查询操作。

在任意库下的 SQL 编译框中输入建库的 SQL 语句，并单击**运行**后即可完成建库操作：

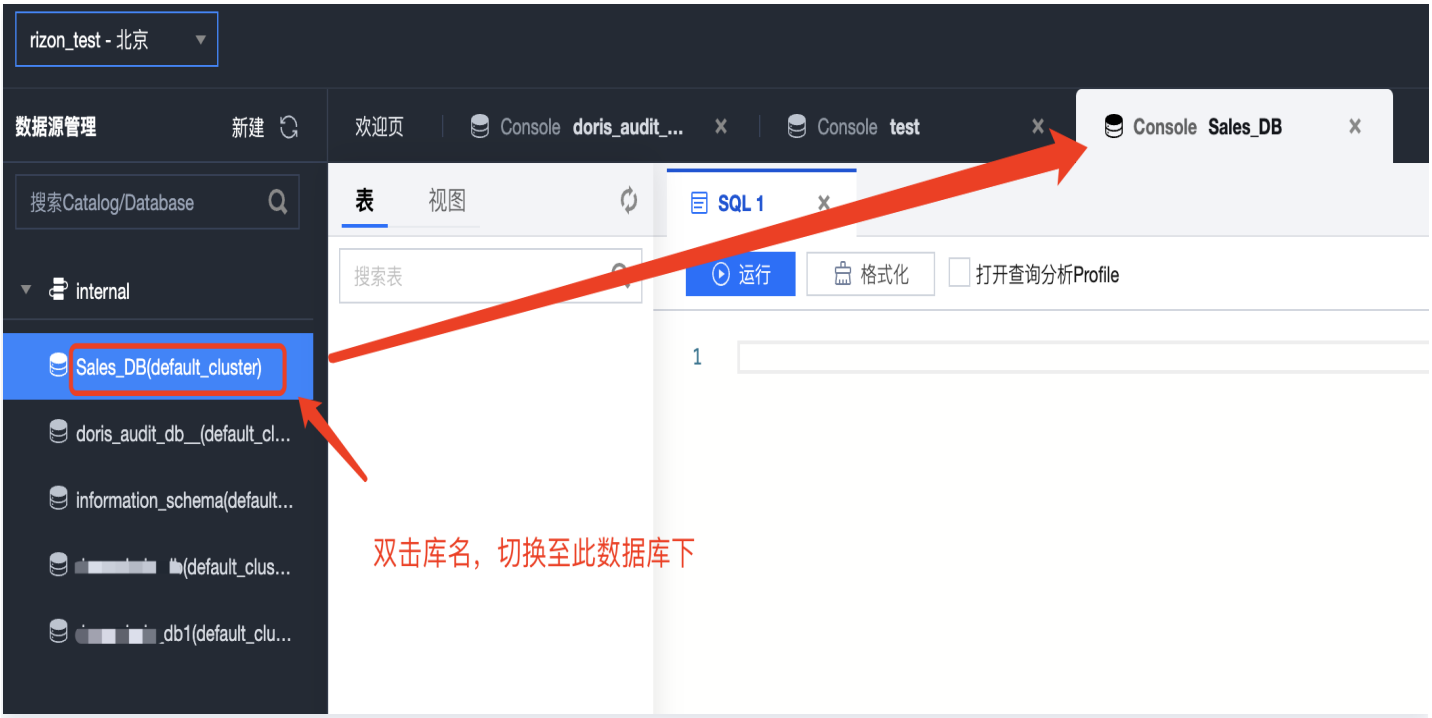
```
CREATE DATABASE Sales_DB;
```

建库完成后，单击数据源管理中的**刷新**，即可在目录树中回显新建的数据库。




## 创建 Doris 数据表

双击左侧库名，即可切换至此数据库下，然后可在弹出的 SQL 编译框中进行建表操作。




## Doris 表模型选择

Doris 支持 Aggregate、Unique 和 Duplicate三种表模型，数据模型在建表时就已经确定，且无法修改。所以，**选择一个合适的数据模型非常重要**。

> **说明：**
> 具体表模型说明可参考文档：Doris 数据表和数据模型 。MySQL 表同步 Doris 时的选型建议如下：
> - 源端有主键的：建议创建 Unique 表模型接入数据。
> - 源端没有主键的：建议创建 Duplicate 表模型接入数据。

> **注意：**
> - key 列在表中的顺序需要和在 key 定义的顺序保持一致。
> - String/text 类型的列不适合作为 key 列，可以转为 varchar 类型。

1. **Aggregate 模型**：可以通过预聚合，极大地降低聚合查询时所需扫描的数据量和查询的计算量，非常适合有固定模式的报表类查询场景，但该模型对 count( * ) 查询很不友好。

   建表 Demo 如下：

```
CREATE TABLE site_visit
(
```

```
    siteid      INT,
    city        SMALLINT,
    username    VARCHAR(32),
    pv BIGINT   SUM DEFAULT '0'
)
AGGREGATE KEY(siteid, city, username)
DISTRIBUTED BY HASH(siteid) BUCKETS 10;
```

2. **Unique 模型**：针对需要唯一主键约束的场景，Unique key 相同时，新记录覆盖旧记录，可以保证主键唯一性约束，适用于有更新需求的分析业务。

   建表 Demo 如下：

```
CREATE TABLE sales_order
(
    orderid     BIGINT,
    status      TINYINT,
    username    VARCHAR(32),
    amount      BIGINT DEFAULT '0'
)
UNIQUE KEY(orderid)
DISTRIBUTED BY HASH(orderid) BUCKETS 10;
```

3. **Duplicate 模型**：相同的行不会合并，适合任意维度的 Ad-hoc 查询。虽然无法利用预聚合的特性，但是不受聚合模型的约束，可以发挥列存模型的优势（列裁剪、向量执行等）。

   建表 Demo 如下：

```
CREATE TABLE session_data
(
    visitorid   SMALLINT,
    sessionid   BIGINT,
    visittime   DATETIME,
    city        CHAR(20),
    province    CHAR(20),
    ip          varchar(32),
    brower      CHAR(20),
    url         VARCHAR(1024)
)
DUPLICATE KEY(visitorid, sessionid)
DISTRIBUTED BY HASH(sessionid, visitorid) BUCKETS 10;
```

## Doris 表分区分桶建议

Doris 支持两层的数据划分，第一层是 Partition（分区），支持 Range 和 List 的划分方式。第二层是 Bucket（分桶），仅支持 Hash 的划分方式。一个表可以不分区，但必须分桶。

> **说明：**
> 具体分区、分桶说明可参考文档：数据分区和分桶，简而言之：
> - 数据量较小的场景：推荐使用一级分片表（即指不分区的分桶表）。
> - 数据量较大或数据有明确日期归属的场景（如事实表）：推荐使用两级分片表，即既分区又分桶的表。

1. 创建一级分片表的示例：

```
Create Table `user_login_new`
(
   `loginId` bigint NOT NULL COMMENT '登录Id',
   `userAcount` varchar(255)NOT NULL COMMENT '用户账号',
   `onlineTime` decimal(10, 2) NOT NULL DEFAULT '0.00' COMMENT '玩家在线时长，
单位：小时',
   `androidVersion` varchar
)
UNIQUE KEY (`loginid`)
COMMENT '用户归因登录表'
DISTRIBUTED BY HASH(`loginid`) buckets 8;
```

> **注意：**
> 创建分桶的语法：DISTRIBUTED BY HASH(`loginid`) buckets 8;
> 其中两个关键点：
> 1. 分桶列：分桶列有一定要求：必须是key列，类型不能是string,text。
> 2. 分桶数量：根据数据大小确定，最好保证分桶后每个桶的大小在1-10G。

2. 创建两级分片表的示例

```
Create Table `user_login_beifen`
(
   `loginId` bigint NOT NULL COMMENT '登录Id',
   `loginTime` date NOT NULL COMMENT '登录时间',
   `loginIp` varchar(255)NOT NULL COMMENT '登录ip',
   `loginProvince` varchar(255)COMMENT '登录地区',
   `onlineTime` decimal(10, 2) NOT NULL DEFAULT '0.00' COMMENT '玩家在线时长，
单位：小时'
)
UNIQUE KEY (`loginId`,`loginTime`)
COMMENT '用户归因登录表'
```

```
PARTITION BY RANGE(`loginTime`)
(
   PARTITION `p201701` VALUES LESS THAN ("2017-02-01"),
   PARTITION `p201702` VALUES LESS THAN ("2017-03-01"),
   PARTITION `p201703` VALUES LESS THAN ("2017-04-01")
)
DISTRIBUTED BY HASH(`loginId`) buckets 8;
```

**⚠ 注意：**

**上为创建 Range 类型分区的语法示例，有几个关键点：**

1. **分区列可以是一列或多列，但都需要是key列。**
2. **创建分区时不可添加范围重叠的分区。**
3. **数据写入前数据归属的分区要提前创建好。**

**以 Doris 也支持预先创建分区、自动创建分区，即动态分区特性：例子如下：**

```
Create Table user_login_beifen2
(
   loginId bigint NOT NULL COMMENT '登录Id',
   loginTime date NOT NULL COMMENT '登录时间',
   loginIp varchar(255)NOT NULL COMMENT '登录ip',
   loginProvince varchar(255)COMMENT '登录地区',
   onlineTime decimal(10, 2) NOT NULL DEFAULT '0.00' COMMENT '玩家在线时长，单
位：小时'
)
UNIQUE KEY (loginId,loginTime)
COMMENT '用户归因登录表'
PARTITION BY RANGE(loginTime)()
DISTRIBUTED BY HASH(loginId) buckets 8
PROPERTIES
(
   "dynamic_partition.time_unit" = "DAY",
   "dynamic_partition.start" = "-10",
   "dynamic_partition.end" = "3",
   "dynamic_partition.buckets" = "8",
   "dynamic_partition.create_history_partition" = "true",
   "dynamic_partition.prefix" = "p"
);
```

```
--上述建表demo预先创建好今天之前10天至后3天的分区表，按loginTime列分区，分区名以'p'开头，每个分区内分成8个桶。
```

## Doris 索引使用建议

如果经常对某列进行精确匹配过滤并且列的基数比较高，建议在此列上创建 bloom filter 索引。

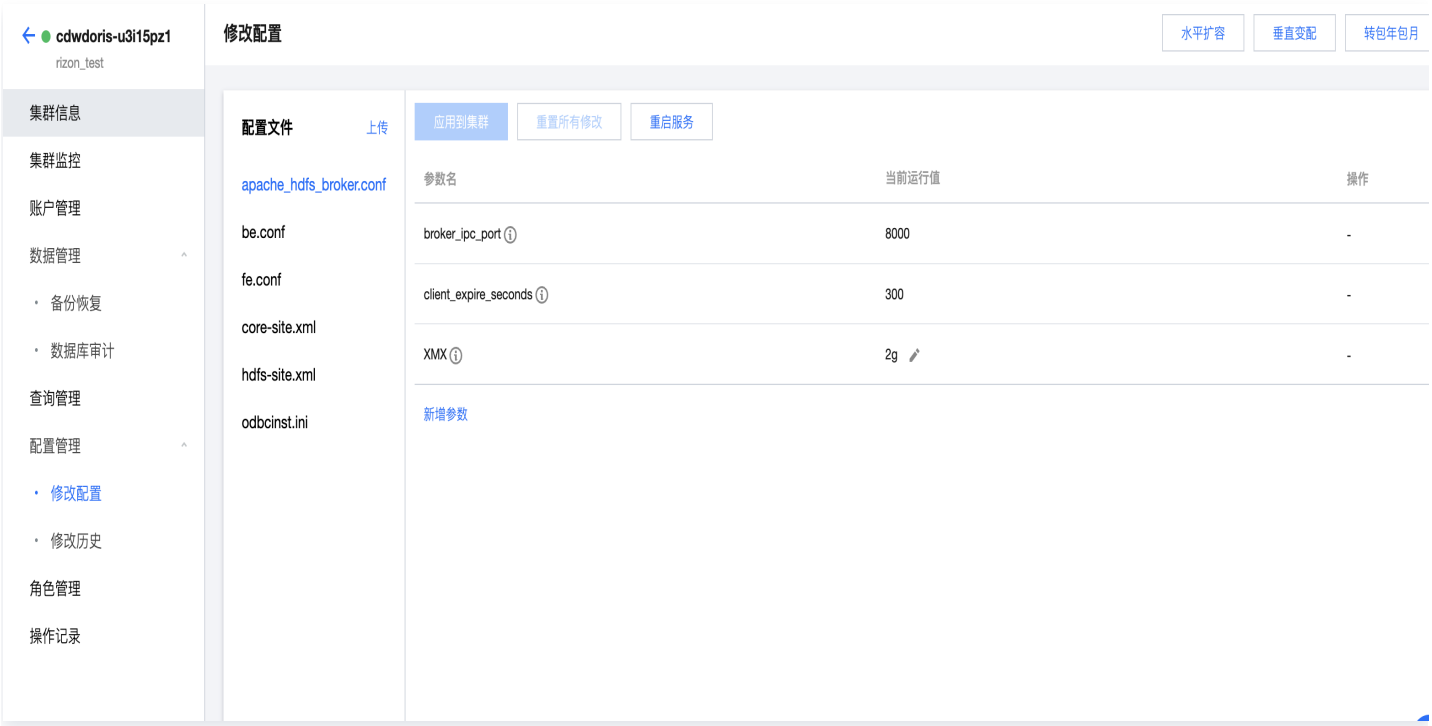
### 建表时尽量避免的操作

特别注意：截止当前最近 Doris 版本（1.2.4.2），以下表相关功能还不完善，**不建议生产使用。**

> ⚠ **注意：**
>
> - 不建议使用 “Merge-on-Write” 功能。
> - 不建议使用 “auto bucket” 功能。
> - 不建议使用 “动态Schema表” 功能。

## 集群配置建议

创建集群时，会初始化以下5个配置文件，说明如下：




| 配置文件 | 配置建议 |
| --- | --- |
| apache_hdfs_br | 建议保持默认配置不变 |

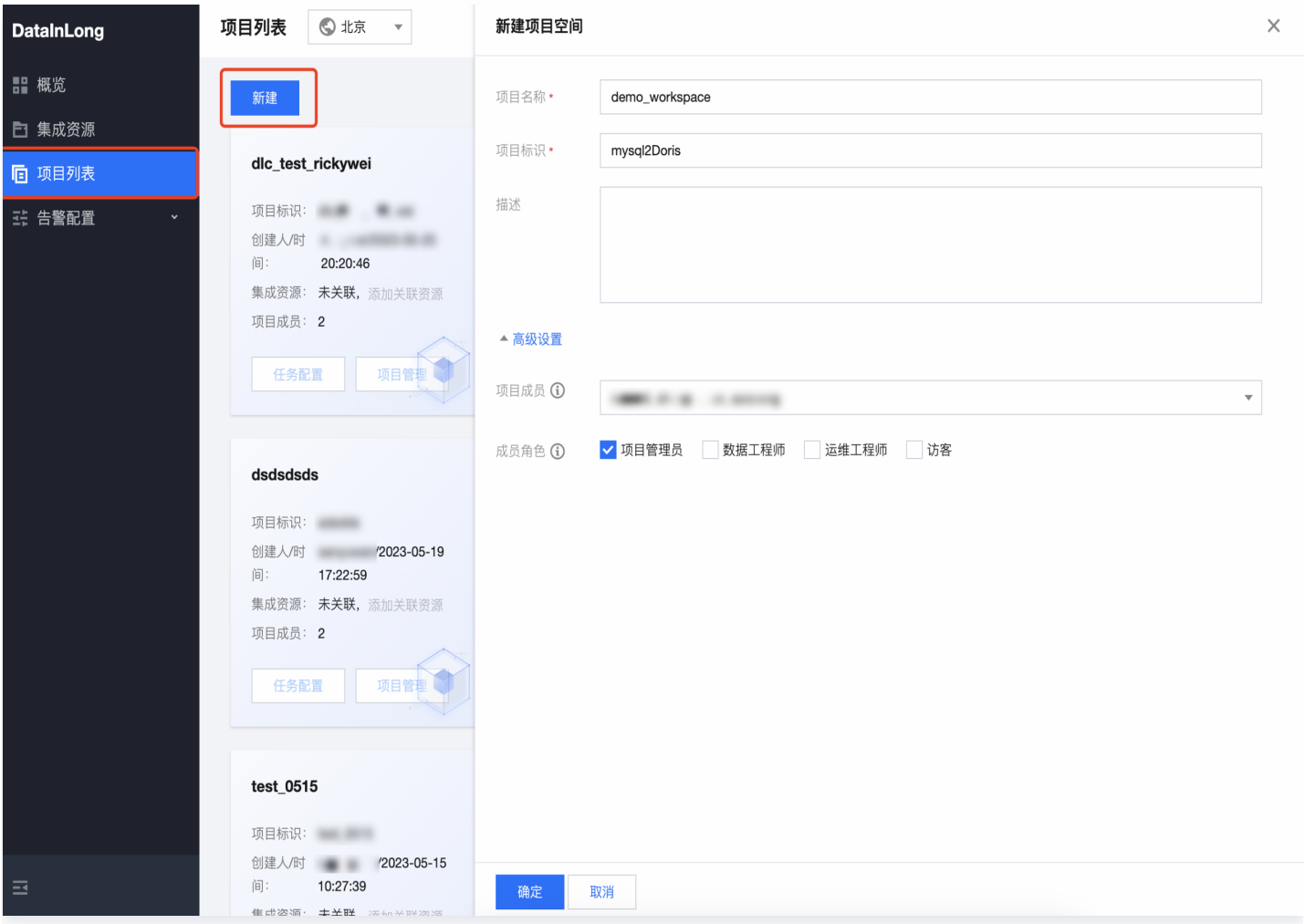
| | |
|---|---|
| oker.conf | |
| be.conf | 大多数配置保持默认配置不变，需特殊注意的参数如下：<br>• compaction_task_num_per_disk：每个磁盘可并发执行的 compaction 任务数量，默认值是2，如果想要提高导入速度可适当调大。具体可参考社区文档 BE 配置项 。<br>• disable_auto_compaction=true：建议不要调整。 |
| fe.conf | 大多数配置保持默认配置不变，需特殊注意的参数如下：<br>• max_running_txn_num_per_db: 控制同一个 DB 的并发导入个数，默认值100，当集群中有过多的导入任务正在运行时，可适当调大。 |
| core-site.xml | 建议保持默认配置不变。 |
| hdfs-site.xml | 建议保持默认配置不变。 |
| odbcinst.ini | 建议保持默认配置不变。 |

# 配置 DataInlong 项目空间及集成资源

最近更新时间：2023-07-14 15:11:24

## 创建项目空间

进入 DataInlong 控制台，单击**项目列表 > 新建**，创建并配置项目及所包含成员。




## 购买集成资源组并关联项目

1. 进入 DataInlong 控制台，选择集成资源并单击**创建**。


## 2. 购买并配置集成资源。


**说明：**

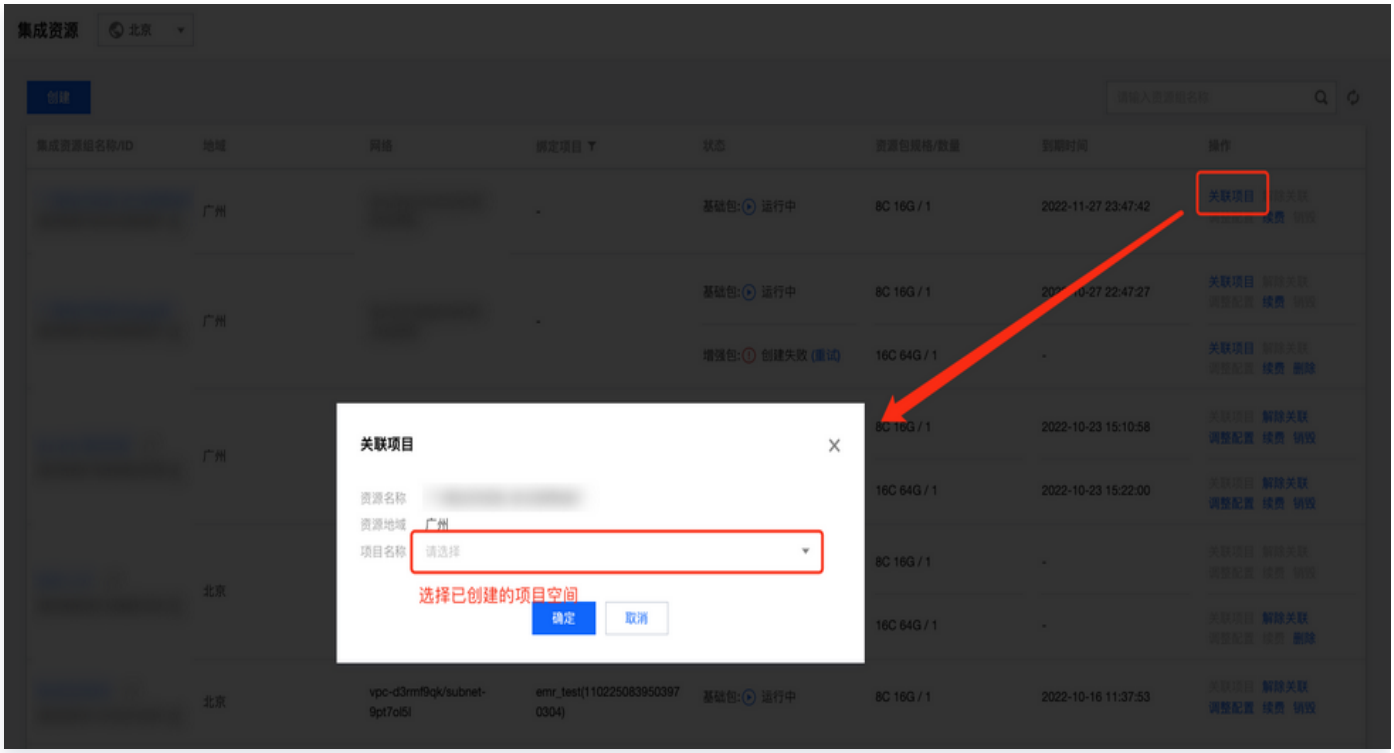
- 离线资源包与实时资源包可根据实际数据情况配置规格以及数量。
- 资源组网络建议和 MySQL 及 Doris 在同一个 VPC 下，若不在一个 VPC 下，可为 VPC 配置开

通公网，可参见 资源组配置公网 。

- 购买完成后，返回控制台并关联资源组与项目空间。

3. 关联资源组和项目空间

> **说明：**
> 若在购买页面内已经关联资源组与项目空间，可忽略此步骤。

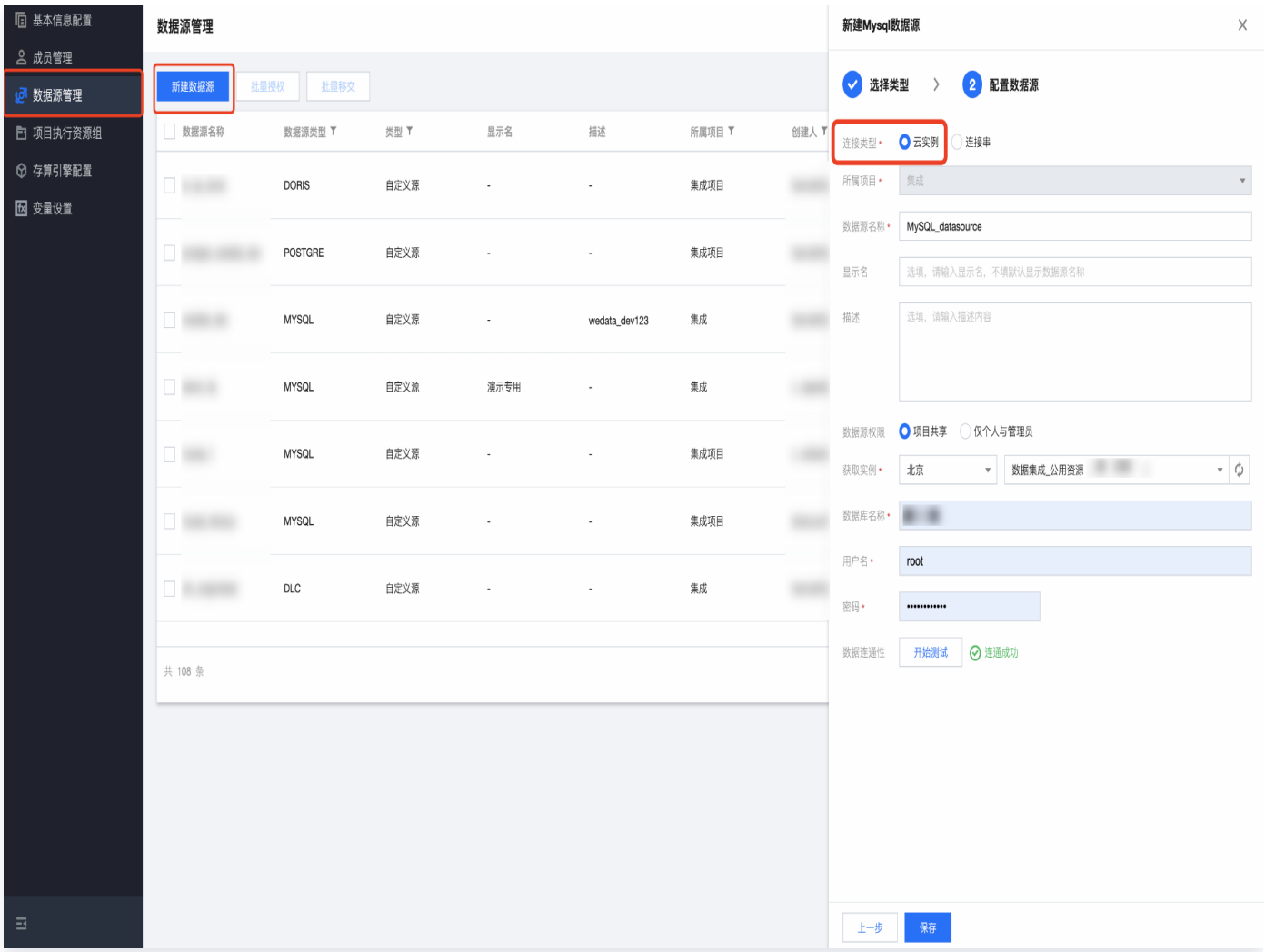



# 进入项目空间并注册数据源

## 配置 MySQL 数据源

支持注册腾讯云 MySQL 或本地自建 MySQL 数据源。进入 项目管理 模块，选择**数据源管理 > 新建数据源 > 选择 MySQL**。

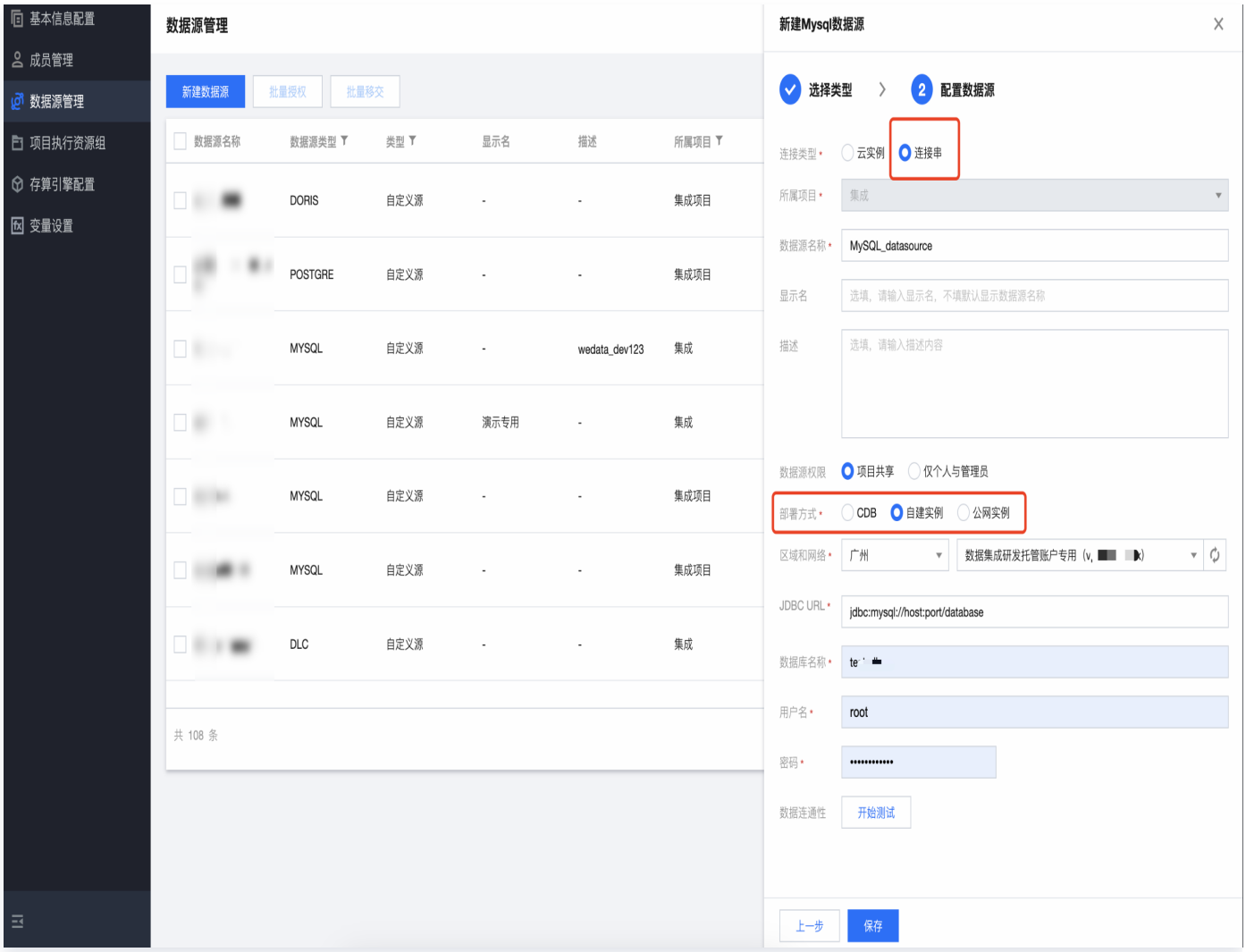
- 使用腾讯云 MySQL 时：可通过配置“云实例”直接关联已购买地域数据库云实例构建数据源。




| 参数 | 参数说明 |
| --- | --- |
| 连接类型 | 腾讯云 MySQL 数据库通过云实例方式添加，此方式下可直接获取当前账号下的 MySQL 数据。 |
| 数据源名称 | 新建的数据源的名称，由用户自定义且不可为空。命名以字母开头，可包含字母、数字、下划线。长度在20字符以内。 |
| 显示名 | 数据源在产品中使用时的显示名称，不填默认显示数据源名称。 |
| 描述 | 选填，对本数据源的描述。 |
| 数据源权限 | 项目共享表示当前数据源项目所有成员均可使用 ，仅个人和管理员表示该数据源仅创建人和项目管理员可用。 |
| 获取实例 | 选择账户下云数据库实例所在的地域、实例名称及 ID 信息。 |
| 数据库名 | 填入云实例下数据库名称；此数据库后续将作为后续数据源的默认数据库。 |

| | |
|---|---|
| 用户 | 连接数据库的用户名称。 |
| 密码 | 连接数据库的密码。 |
| 连通性 | 测试是否能够连通所配置的数据库。<br>**注意：**<br>• 若连通性测试不通过，可以继续创建数据源，但后续数据读写时会报错。<br>• 如果连通性测试不通过，可能是因为 WeData 被数据库所在网络防火墙禁止，请参见 添加腾讯云 MySQL 数据库安全组 。 |

- **使用非腾讯云 MySQL 时：通过“连接串” JDBC 方式添加自建数据库作为数据源。**

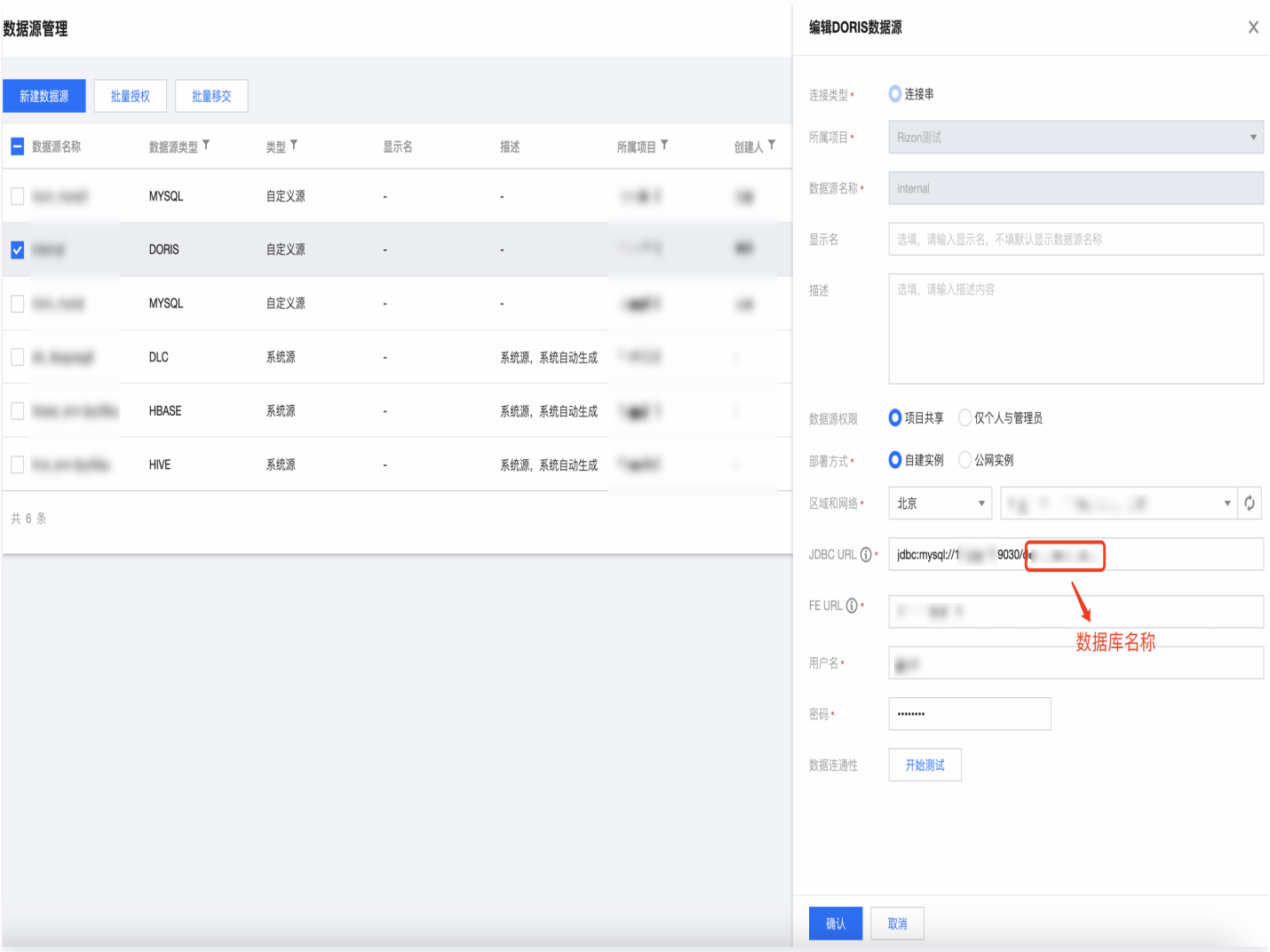



| 参数 | 参数说明 |
|---|---|
| 连接类型 | 非腾讯云数据库实例可通过连接串方式连接。 |
| 数据源名称 | 新建的数据源的名称，由用户自定义且不可为空。命名以字母开头，可包含字母、数字、下划线。长度在20字符以内。 |

| | |
|---|---|
| 显示名 | 数据源在产品中使用时的显示名称，不填默认显示数据源名称。 |
| 描述 | 选填，对本数据源的描述。 |
| 数据源权限 | 项目共享表示当前数据源项目所有成员均可使用 ，仅个人和管理员表示该数据源仅创建人和项目管理员可用。 |
| 部署方式 | • CDB：仅适用于使用腾讯云数据库。<br>• 自建实例：适用于自建且开通 VPC 环境内的 MySQL 集群。<br>• 公网实例：适用于开通了公网的 MySQL 集群。 |
| 数据库名 | 填入数据库名称；此数据库后续将作为后续数据源的默认数据库。 |
| 用户 | 连接数据库的用户名称。 |
| 密码 | 连接数据库的密码。 |
| 连通性 | 测试是否能够连通所配置的数据库。<br>**注意：若连通性测试不通过，可以继续创建数据源，但后续数据读写时会报错。** |

## 配置 Doris 数据源

进入 项目管理 模块，选择**数据源管理 > 新建数据源 > 选择 Doris**，配置数据源参数并在连通性测试成功后即可保存。



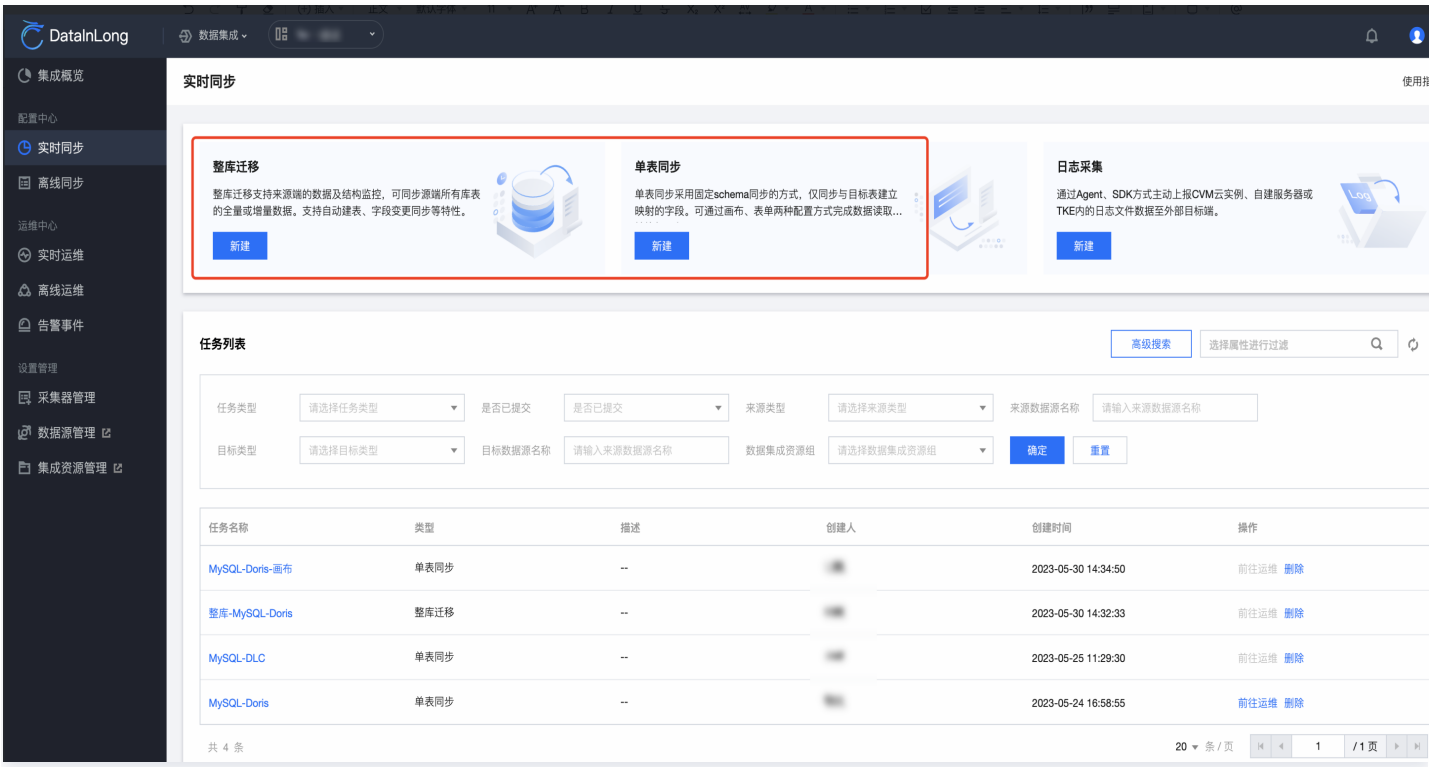

| 参数 | 说明 |
| --- | --- |
| 数据源名称 | 新建的数据源的名称，由用户自定义且不可为空。命名以字母开头，可包含字母、数字、下划线。长度在20字符以内。 |
| 描述 | 选填，对本数据源的描述。 |
| 数据源权限 | 项目共享表示当前数据源项目所有成员均可使用 ，仅个人和管理员表示该数据源仅创建人和项目管理员可用。 |
| 部署方式 | • 自建实例：位于 VPC 环境内 Doris 实例。<br>• 公网实例：可使用公网访问的实例。 |
| 区域 | 选择账户下云数据库实例所在的地域、实例名称及 ID 信息。 |

| | |
|---|---|
| 和网络 | |
| JDBC URL | 用于连接 Doris 数据源的连接串信息：<br>• 端口若为自建实例请填入内网IP地址和端口，多个地址间逗号（,）分隔，例如：jdbc:mysql://内网IP:port/参数。<br>• 若为公网实例请填入公网ip地址和端口，例如：jdbc:mysql://公网IP:port/参数。<br>注：上述“参数”填写数据源下的任一“数据库名称”即可，用于校验连通性。 |
| FE URL | 输入 fe http 地址，格式为：IP地址:http端口（无需 https:// 或 http://前缀），多个地址之间使用逗号（,）分隔，<br>例如：172.17.16.3:8030,172.17.16.4:8030。 |
| 用户名 | 连接数据源的用户名称。 |
| 密码 | 连接数据源的密码。 |
| 数据连通性 | 测试是否能够连通所配置的数据库。 |

# 配置单表实时同步任务

最近更新时间：2023-09-05 12:26:24

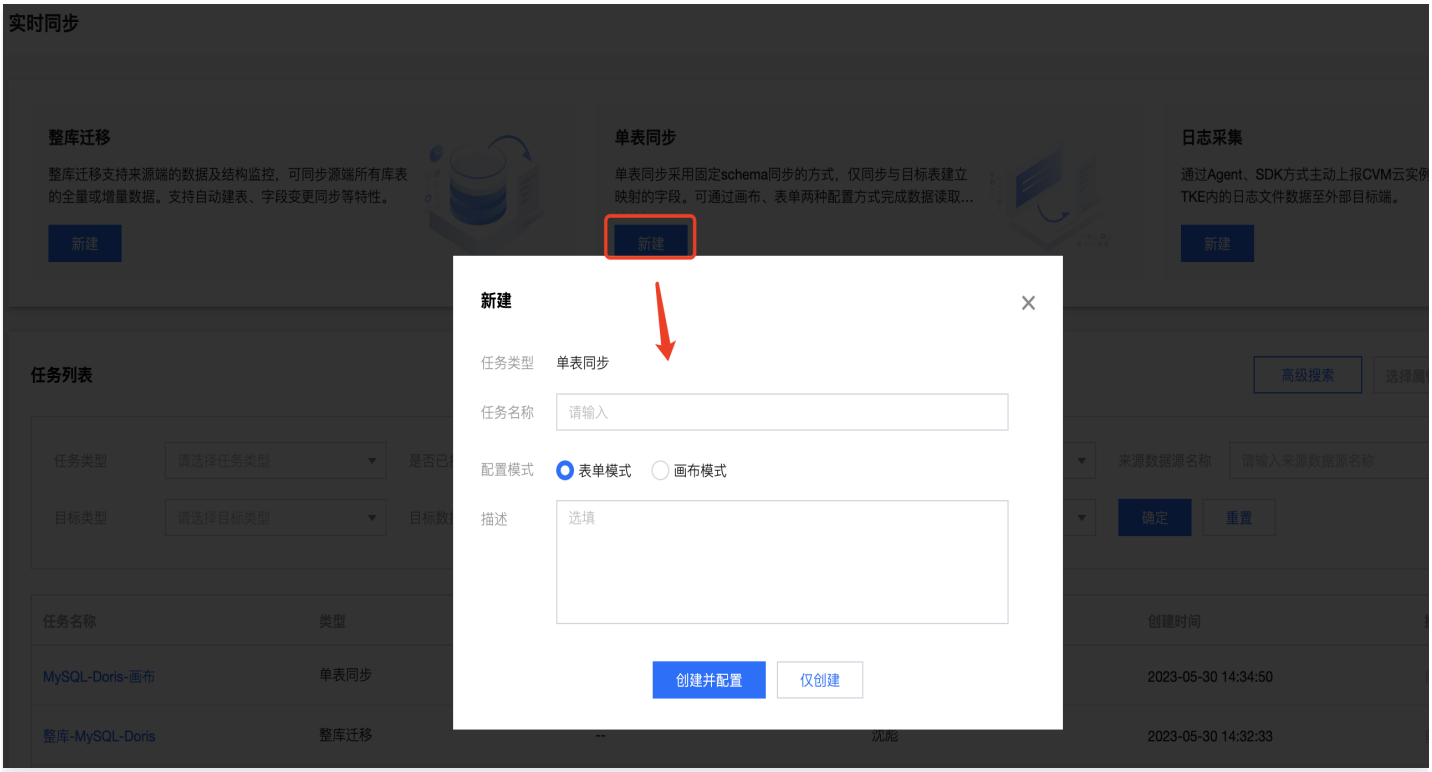
**数据集成支持单表同步、整库迁移两种数据同步方式，下文将介绍单表实时同步的操作方法。**




## 创建单表同步任务

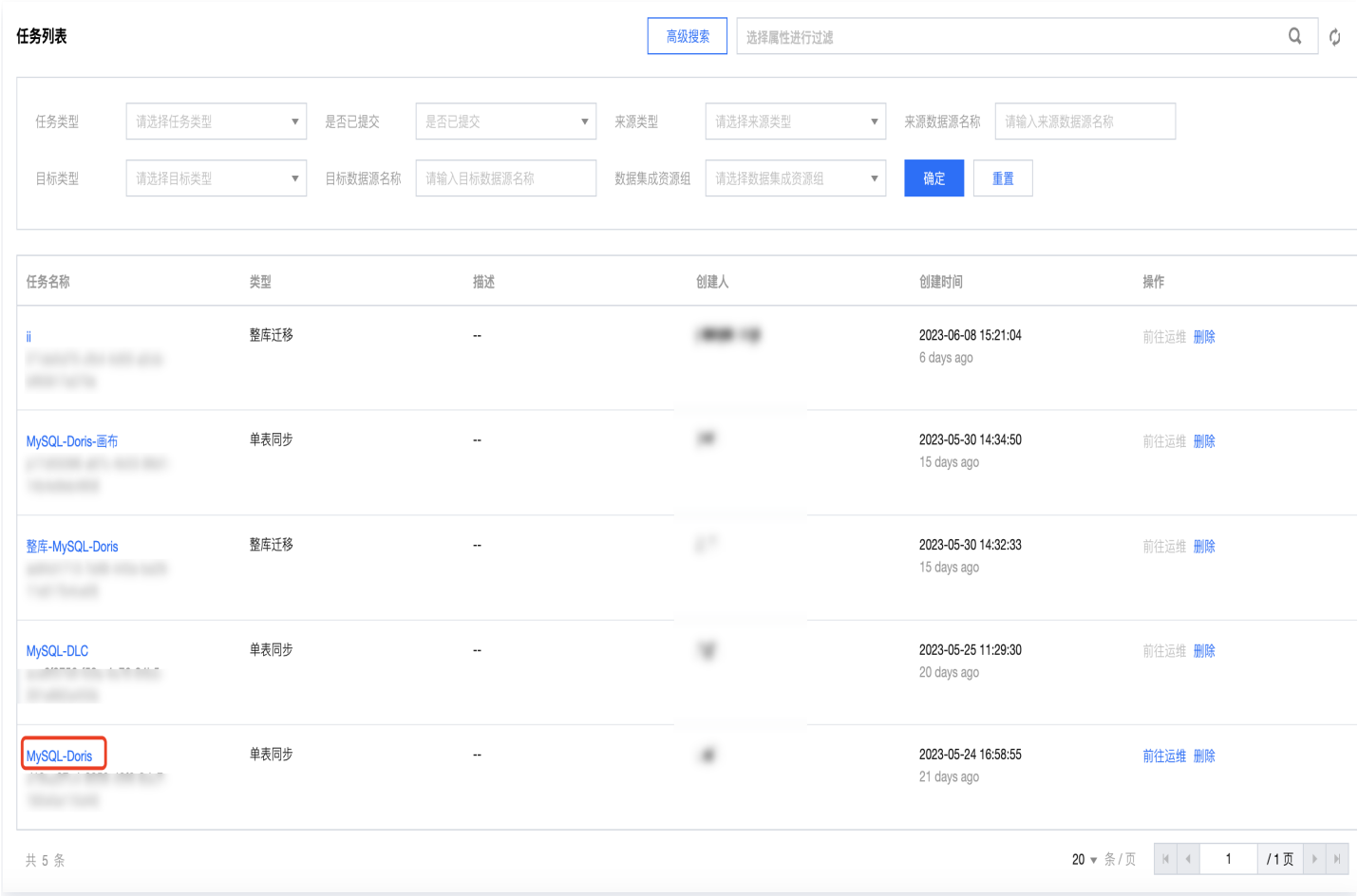
1. 进入**数据集成 > 实时同步**页面，单击**新建**创建单表同步任务。


2. 在弹出的提示框中输入任务名称和备注，选择表单模式或画布模式后（此处 demo 选择“单表模式”），单击“创建并配置”或“仅创建”完成任务创建。

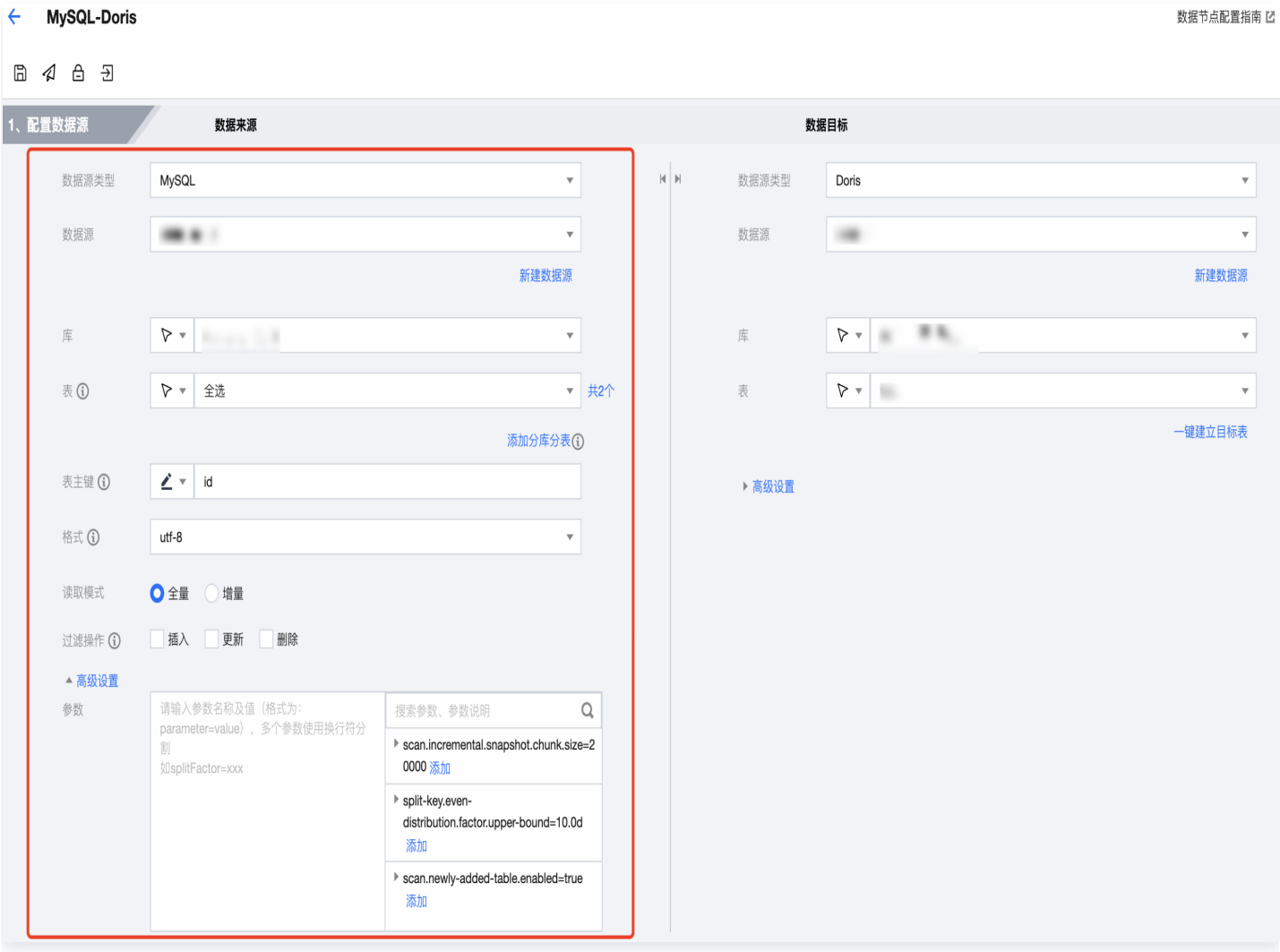



## 编辑任务

创建完成任务后可在任务列表页面单击新建的实时同步任务名称进入任务编辑界面，本文以单表模式为例，进行作业配置。


任务列表
高级搜索
选择属性进行过滤
任务类型
请选择任务类型
是否已提交
是否已提交
来源类型
请选择来源类型
来源数据源名称
请输入来源数据源名称
目标类型
请选择目标类型
目标数据源名称
请输入目标数据源名称
数据集成资源组
请选择数据集成资源组
确定
重置
任务名称
类型
描述
创建人
创建时间
操作
ii
整库迁移
--
2023-06-08 15:21:04
6 days ago
前往运维 删除
MySQL-Doris-画布
单表同步
--
2023-05-30 14:34:50
15 days ago
前往运维 删除
整库-MySQL-Doris
整库迁移
--
2023-05-30 14:32:33
15 days ago
前往运维 删除
MySQL-DLC
单表同步
--
2023-05-25 11:29:30
20 days ago
前往运维 删除
MySQL-Doris
单表同步
--
2023-05-24 16:58:55
21 days ago
前往运维 删除
共 5 条
20 条/页
1
/ 1 页

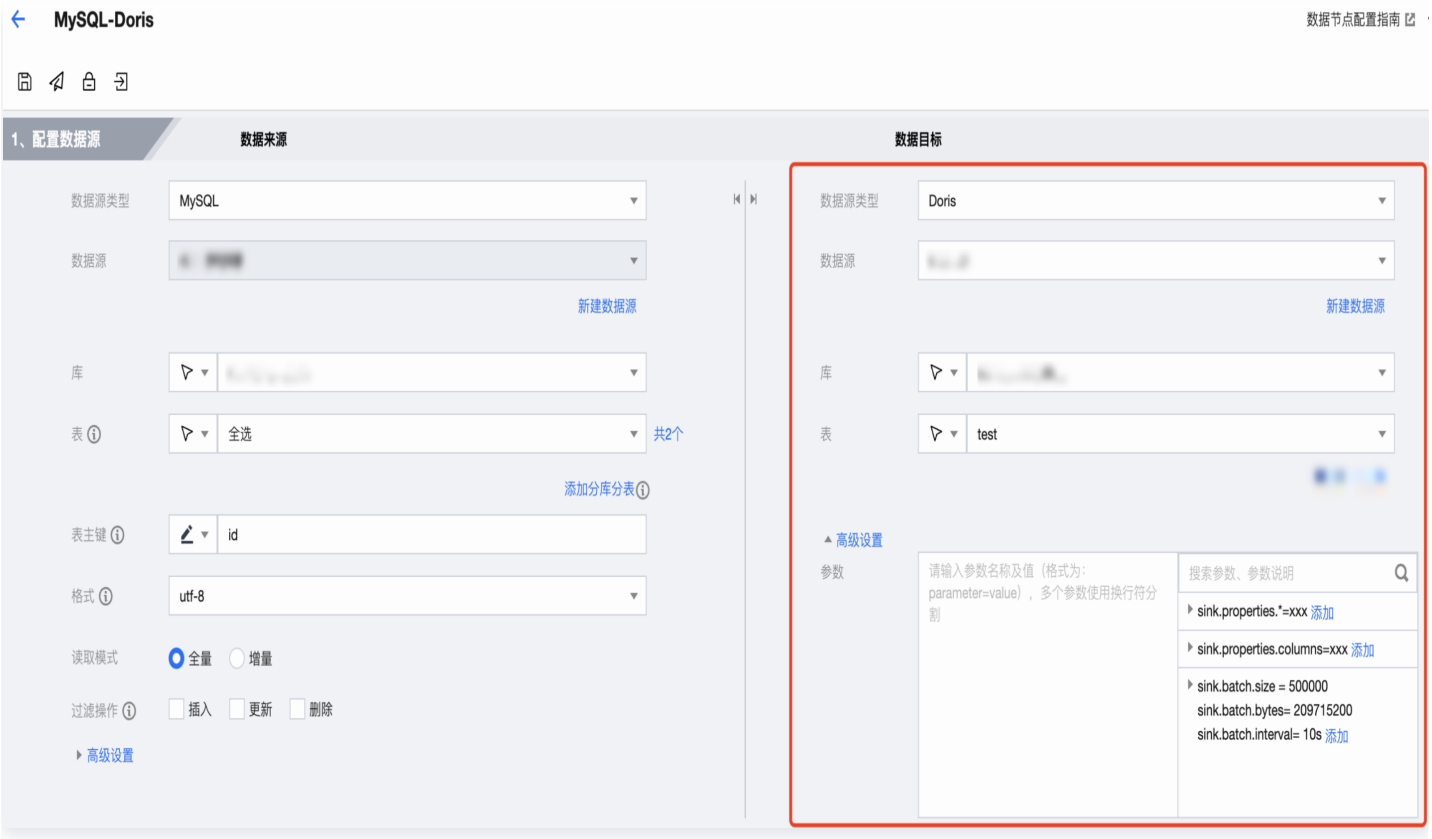
# 配置 MySQL 信息




| 参数 | 参数说明 |
| --- | --- |
| 数据源类型 | 可支持多种，此处选择 MySQL。 |
| 数据源 | 可选择上文“数据源管理”中已注册好的 MySQL 数据源。 |
| 库 | 单选，支持选择值、输入值/表达式两种方式：<br>• 选择值时：可通过下拉列表选择在“数据源管理”中注册的数据库。<br>• 输入值/表达式时：可手动输入选定数据源下的已有库。 |
| 表 | 可多选，支持选择值、输入值/表达式两种方式：<br>• 选择值时：可通过下拉列表选择。<br>• 输入值/表达式时：可手动输入。<br>**注意：选择多个表时需保证多表 schema一致，当选定的多个表的 schema 不一致时，将以选中的第一个表的 schema 为准。** |

| | |
|---|---|
| 表主键 | 支持选择值、输入值/表达式两种方式：<br>• 选择值时：可通过下拉列表选择。<br>• 输入值/表达式时：可手动输入。<br>**注意：分库分表模式下默认表 schema 一致，当选定的多个表的 schema 不一致时，系统将使用拉取的第一张表的主键。** |
| 读取模式 | • 全量：将同步库内历史数据，全量同步结束后，会继续同步增量数据。<br>• 增量：仅从任务启动后的 binlog cdc 位点开始同步数据。 |
| 过滤操作 | 支持插入、更新和删除三种操作，设置后将不同步指定操作类型的数据。 |
| 高级设置 | 当前算子的运行参数，具体参数说明请参见 实时节点高级参数 。 |

## 配置 Doris 信息

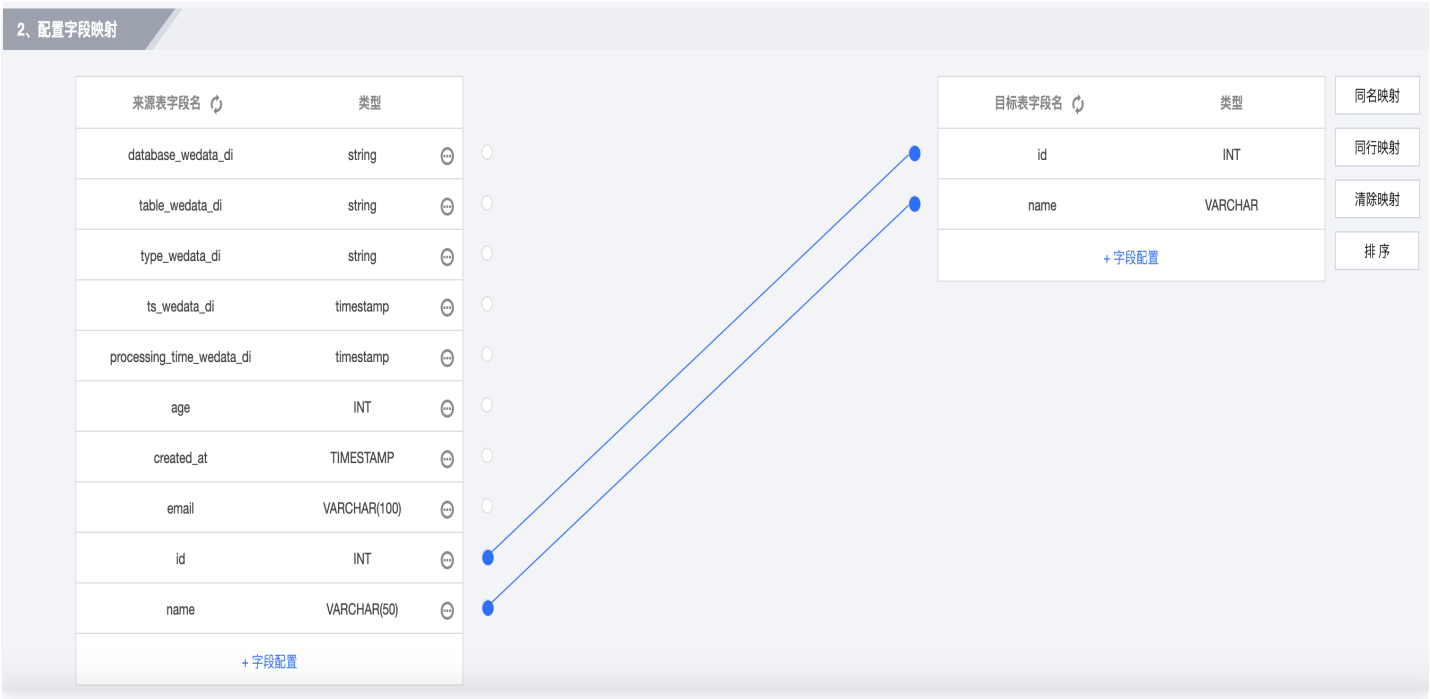



| 参数 | 参数说明 |
|---|---|
| 数据源类型 | 可支持多种，此处选择 Doris。 |
| | |

| | |
|---|---|
| 数据源 | 可选择上文“数据源管理”中已注册好的 Doris 数据源。 |
| 库 | 单选，支持选择值、输入值/表达式两种方式：<br>• 选择值时：可通过下拉列表选择在“数据源管理”中注册的数据库。<br>• 输入值/表达式时：可手动输入选定数据源下的已有库。 |
| 表 | 单选，支持选择值、输入值/表达式两种方式：<br>• 选择值时：可通过下拉列表选择。<br>• 输入值/表达式时：可手动输入。 |
| 过滤操作 | 支持插入、更新和删除三种操作，设置后将不同步指定操作类型的数据。 |
| 高级设置 | 当前算子的运行参数，具体参数说明请参见 实时节点高级参数 。 |

## 配置表字段映射

此处可设置来源和目标端数据对应关系，后续任务仅同步具有映射关系的字段之间的数据。




| 参数 | 参数说明 |
|---|---|
| 同名映射 | 即建立源端表、目标表同名字段的映射关系。 |
| 同行映射 | 即根据相同的行号建立源端表、目标表字段的映射关系。 |
| | |

| | |
|---|---|
| 手动映射 | 除同名映射、同行映射的快捷方式外，还支持手动连线的方式进行字段映射。 |
| 清除映射 | 即清除当前已创建好的映射关系。 |
| 排序 | 对当前字段映射进行格式化显示，点击后具有连线关系字段将显示为一行。<br>说明：此排序并不变更实际表内字段顺序。 |
| 字段配置 | 可单击**字段配置**手动添加字段名称及类型。<br>**说明：**<br>1. MySQL / Doris 已提供直接获取数据表结构能力，您可以直接使用或查看界面内已展示出的字段。<br>2. 源端提供 flink 函数对字段进行转换，可在添加字段内选择“函数”类型增加转换字段写入结果中。 |

**⚠ 注意：**

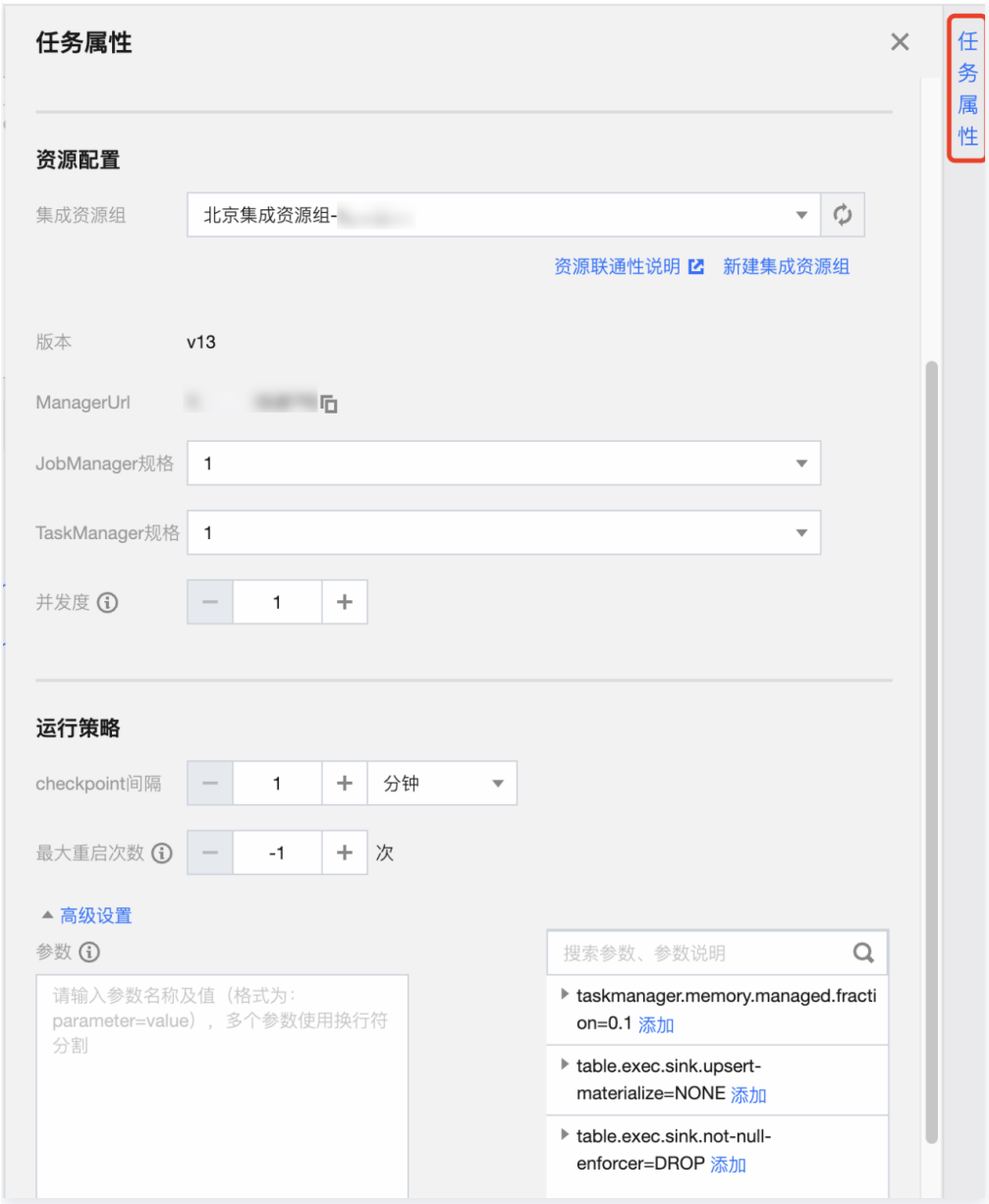
- **未配置映射关系的目标字段内容将为空。**
- **若来源字段类型与目标字段类型间无法转换时，可能会导致任务失败。**

MySQL -> Doris字段映射说明如下：

| 类型 | MySQL 数据类型 | 建议转化 Doris 数据类型 | 补充说明 |
|---|---|---|---|
| | BOOLEAN | BOOLEAN | - |
| 数值类型 | TINYINT | TINYINT | - |
| | SMALLINT | SMALLINT | - |
| | MEDIUMINT | INT | - |
| | INT | INT | - |
| | BIGINT | BIGINT | - |
| | UNSIGNED TINYINT | SMALLINT | - |
| | UNSIGNED MEDIUMINT | INT | - |
| | UNSIGNED INT | BIGINT | - |
| | UNSIGNED | LARGEINT | - |

| | | | |
|---|---|---|---|
| | BIGINT | | |
| | FLOAT | FLOAT | - |
| | DOUBLE | DOUBLE | - |
| | DECIMAL | DECIMALV3 | - |
| 日期时间类型 | YEAR | SMALLINT | - |
| | TIME | STRING | - |
| | DATE | DATEV2 | - |
| | DATETIME | DATETIMEV2 | - |
| | TIMESTAMP | DATETIMEV2 | **TIMESTAMP 字段数据会随着系统时区而改变但 DATETIME 字段数据不会，建议根据业务场景进行时区转化** |
| 字符串类型 | CHAR | CHAR | |
| | VARCHAR | VARCHAR | 如果 MySQL 字段长度超过65533，建议转化为 string |
| | TINYTEXT、TEXT | STRING | |
| | MEDIUMTEXT、LONGTEXT | STRING | MySQL 字段长度超过1048576 字节时可能精度丢失 |
| 二进制字符串 | TINYBLOB、BLOB | STRING | |
| | MEDIUMBLOB、LONGBLOB | STRING | MySQL 字段长度超过1048576 字节时可能精度丢失 |
| | BINARY、VARBINARY | STRING | |
| 其他 | JSON | STRING | MySQL 字段大小超过1M时可能精度丢失 |
| | SET、BIT | STRING | - |
| | ENUM | UNSUPPORTED | 暂不支持 |

## 配置任务属性

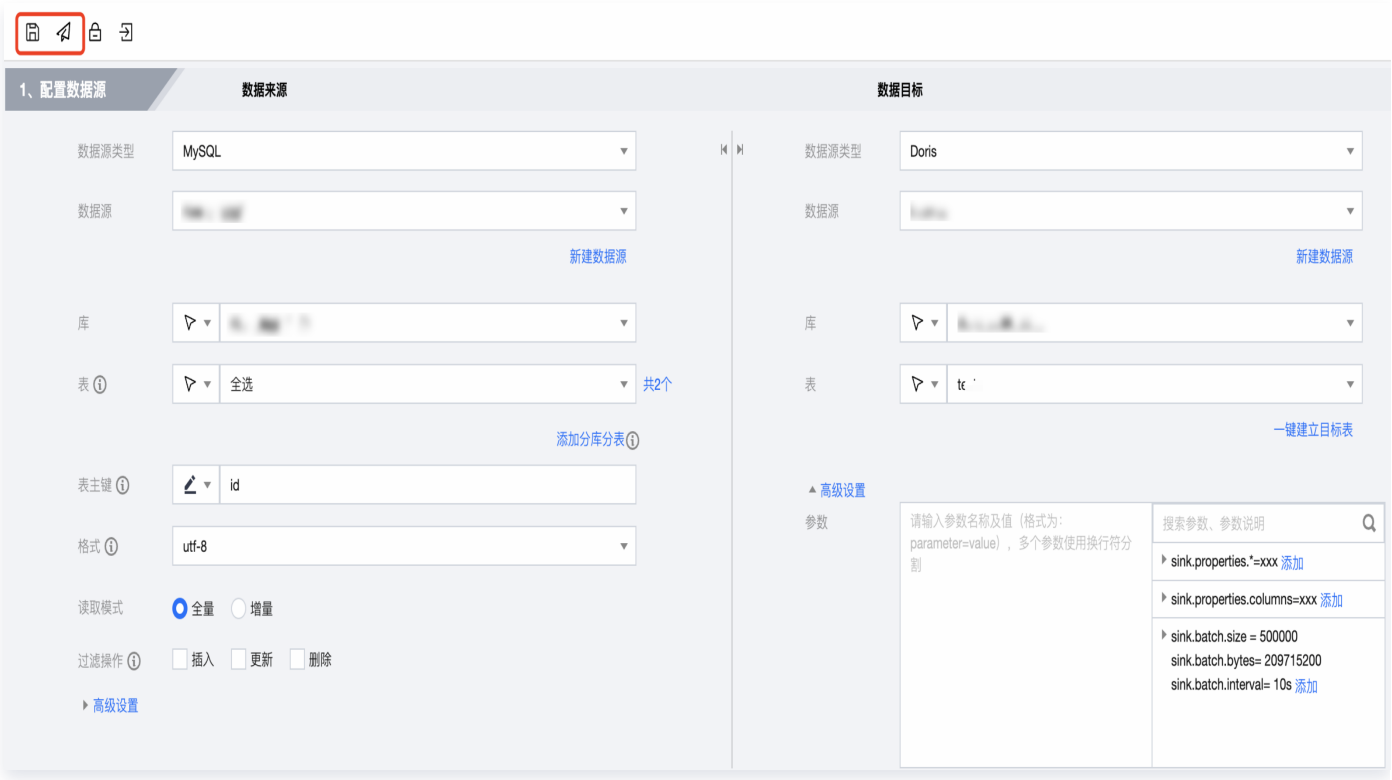



| 参数 | 参数说明 |
| --- | --- |
| 集成资源组 | 可选择绑定此项目的集成资源组，一个任务仅可绑定一个资源组。<br>若未购买资源组或未绑定资源组，请先进行绑定操作。 |
| JobManager 规格 | 支持0.25、0.5、1、2CU，设置后任务将默认占用此规格。<br>CU 任务实际占用 CU 数= JobManager 规格 + TaskManager 规格 × 并行度。 |
| TaskManager 规格 | 支持0.25、0.5、1、2CU，设置后任务将默认占用此规格 。<br>CU 任务实际占用 CU 数= JobManager 规格 + TaskManager 规格 × 并行度 |
| 并发度 | 每个算子的默认并行度，默认1。<br>任务实际占用CU数= JobManager 规格 + TaskManager 规格 × 并行度 |
| checkpoi | 当前任务提交的最大 checkpoint 间隔 |

| | |
|---|---|
| nt 间隔 | |
| 最大重启次数 | 设置在执行过程中发生故障时任务最大的重启阈值，若运行中重启次数超过此阈值，任务状态将置为“失败”。设置范围为[-1,100]， 阈值为0表示不重启，-1 表示不限制最大重启次数 |
| 高级设置 | 设置任务级别运行参数，具体参数说明请参见 实时节点高级参数 。 |

## 任务保存与提交

1. 配置完成后，单击页面左上角的**保存**按钮完成配置保存，再单击**提交**按钮完成作业启动。



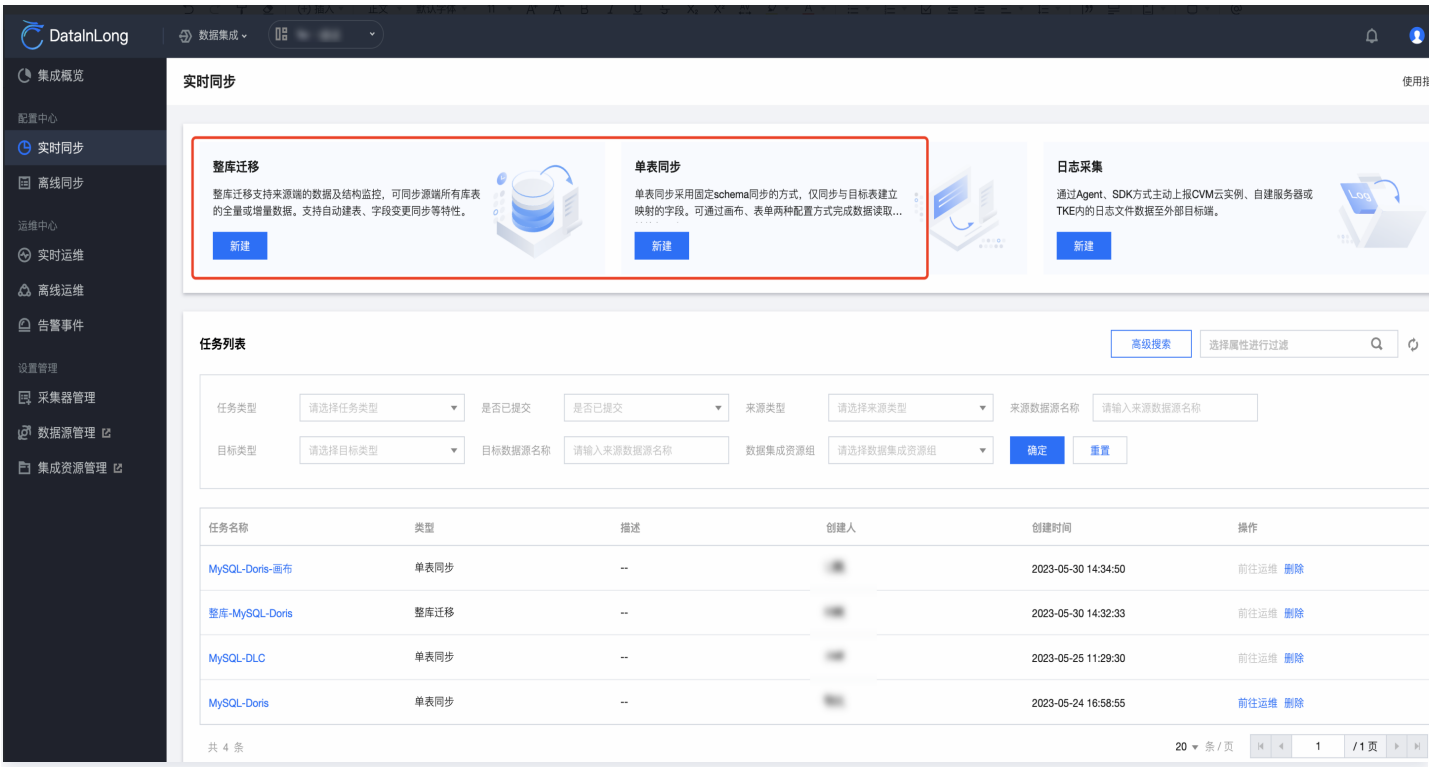

2. 作业启动前，会对必要配置进行校验，请确认无误后再提交。


数据来源
数据目标
Doris
检测到1个告警，建议您修复后再提交；若直接提交，可能造成任务失败
再次检测提交
直接提交
任务配置检测
来源配置 检测完成
目标配置 检测完成
映射关系配置 检测完成
资源组配置 检测完成
资源监测
资源状态检测 检测完成
资源余量检测 检测完成
资源连通性检测 警告
当前资源:北京集成资源组-kyluxpwv 与 数据源:rizon_mysql 网络不一致，可能会造成任务运行失败。请调整数据源与资源使用VPC或为网络配置公网
全选
增量
更新

# 配置整库实时迁移任务

最近更新时间：2023-09-05 12:26:24

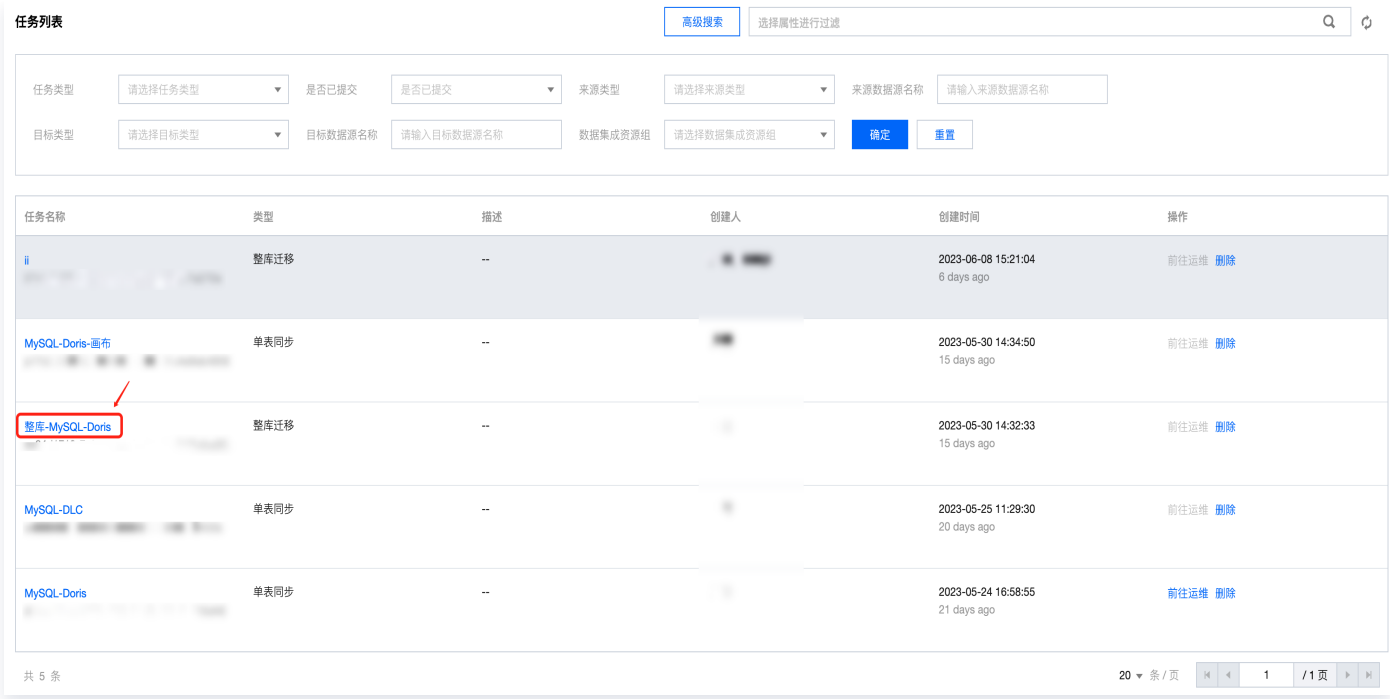
**数据集成**支持单表同步、整库迁移两种数据同步方式，下文将介绍**整库实时迁移**的操作方法。



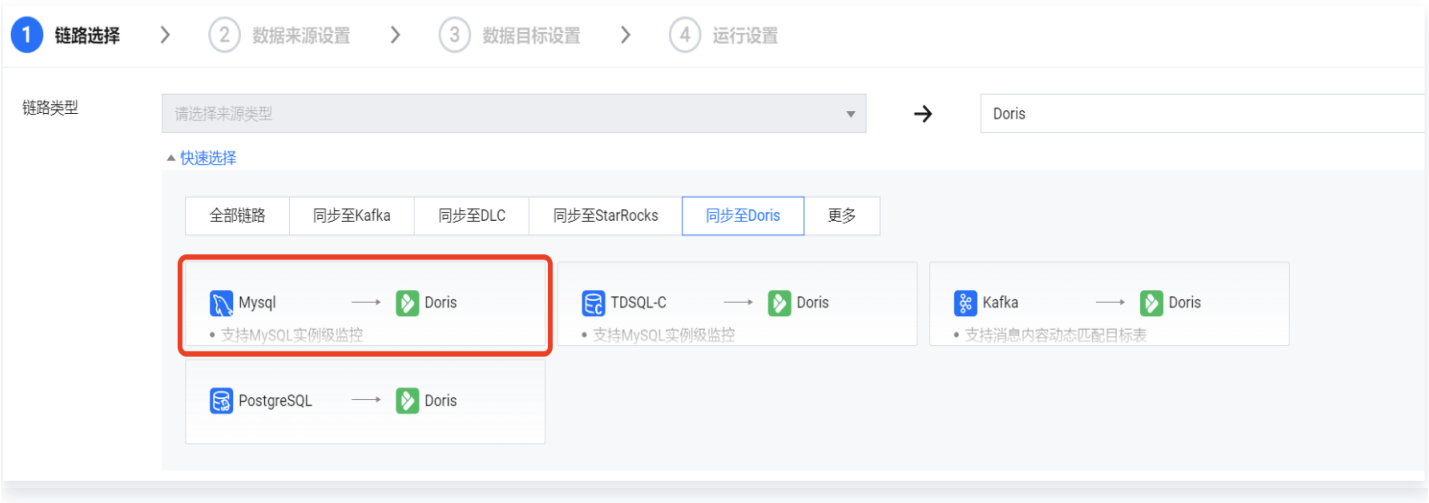

## 创建整库同步任务

1. 进入**数据集成 > 实时同步**页面，单击**新建**创建整库迁移任务。

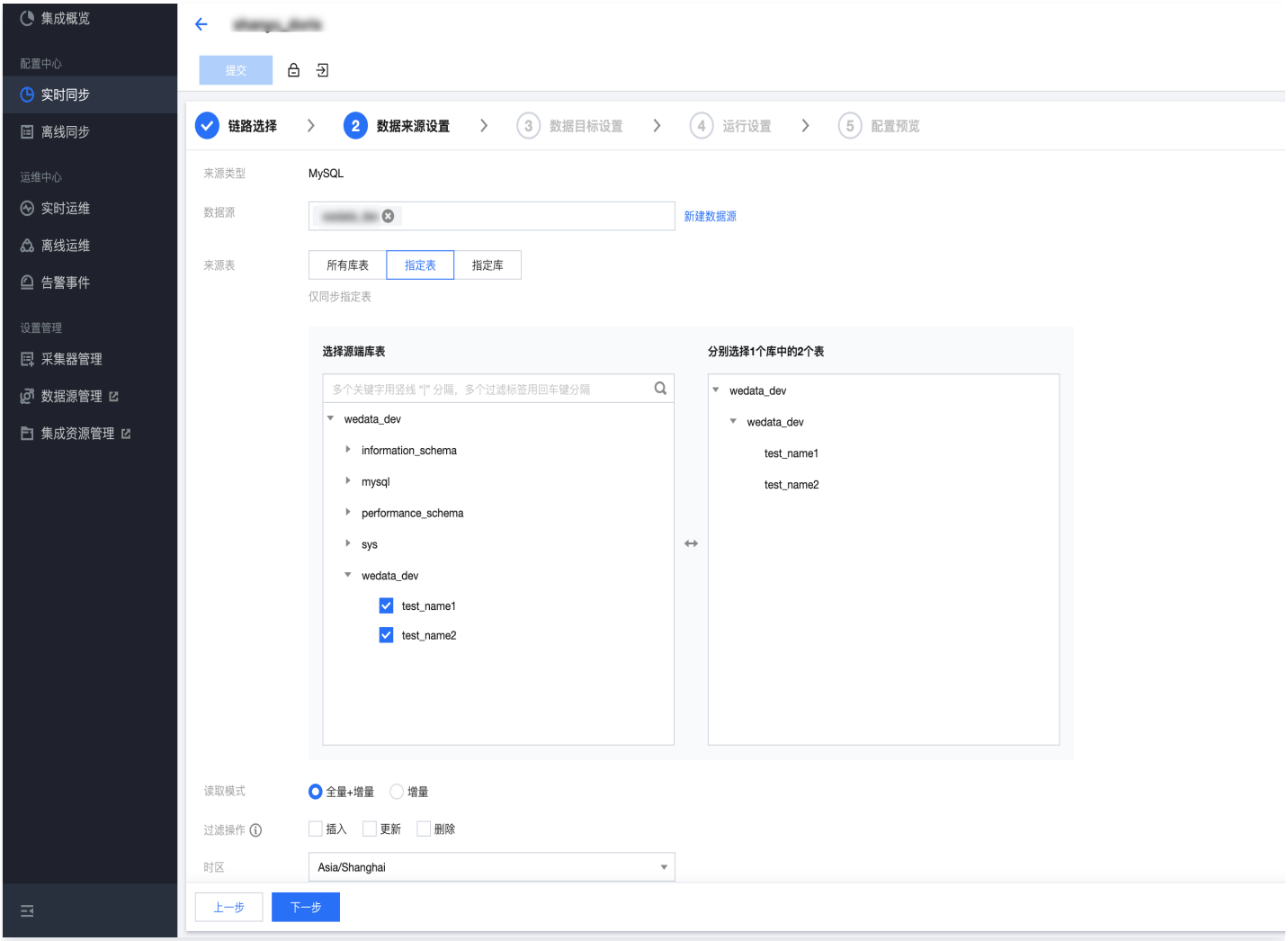

2. 创建完毕后，单击任务列表中的任务名称即可进行具体配置。




# 选择同步至 Doris 目标端的链路




# 配置 MySQL 源端读取多张表




| 参数 | 说明 |
| --- | --- |
| 数据源 | 选择需要同步的已配置好的 MySQL 数据源。 |
| 来源表 | • 所有库表：监控数据源下所有库。任务运行期间新增库、表默认将同步至目标端。<br>• 指定表：此选项下需指定到具体表名称，设置后任务仅同步指定表。<br>• 指定库：此选项下需指定具体库名、以表名正则表达式。设置后，任务运行期间符合表名表达式的新增表默认将同步至目标端。 |
| 读取模式 | • 全量+增量：数据同步分为全量和增量同步阶段，全量阶段完成后任务进入增量阶段。全量阶段将同步库内历史数据，增量阶段从任务启动后 binlog cdc 的位点开始同步。<br>• 增量：仅从任务启动后的 binlog cdc 位点开始同步数据。 |
| 过滤操作 | 支持插入、更新和删除三种操作，设置后将不同步指定操作类型的数据。 |

| | |
|---|---|
| 时区 | 设置日志时间所属时区，默认上海。 |

## 设置 Doris 目标写入方式


| 参数 | 说明 |
|---|---|
| 数据源 | 选择已经创建的 Doris 数据源。 |
| 库/表匹配策略 | 设置任务运行时 Doris 中数据库以及数据表对象的名称匹配规则：<br>• 与来源库/表同名：任务运行时系统将默认在目标数据源内匹配与来源库/表同名对象。<br>• 自定义：自定义规则支持设置来源与目标之间特殊关系，例如统一将源端库名或表名加上统一固定前缀或者后缀在写入目标库或表任务运行时。此策略下，任务运行时系统将默认根据命名规则匹配目标对象。 |
| 高级设置 – 参数 | 设置 Doris 写入端的运行参数，此参数可根据业务需求配置。<br>Doris 端已支持参数详情请参见实时节点高级参数。 |

## 配置运行资源和策略

- **集成资源配置**

  为当前任务关联的集成资源组，同时设定运行时 JM、TM 规格以及任务运行并行度。其中，当前任务实际运行时实际占用 CU 数= JobManager 规格 + TaskManager 规格 × 并行度。
- **消息处理策略**

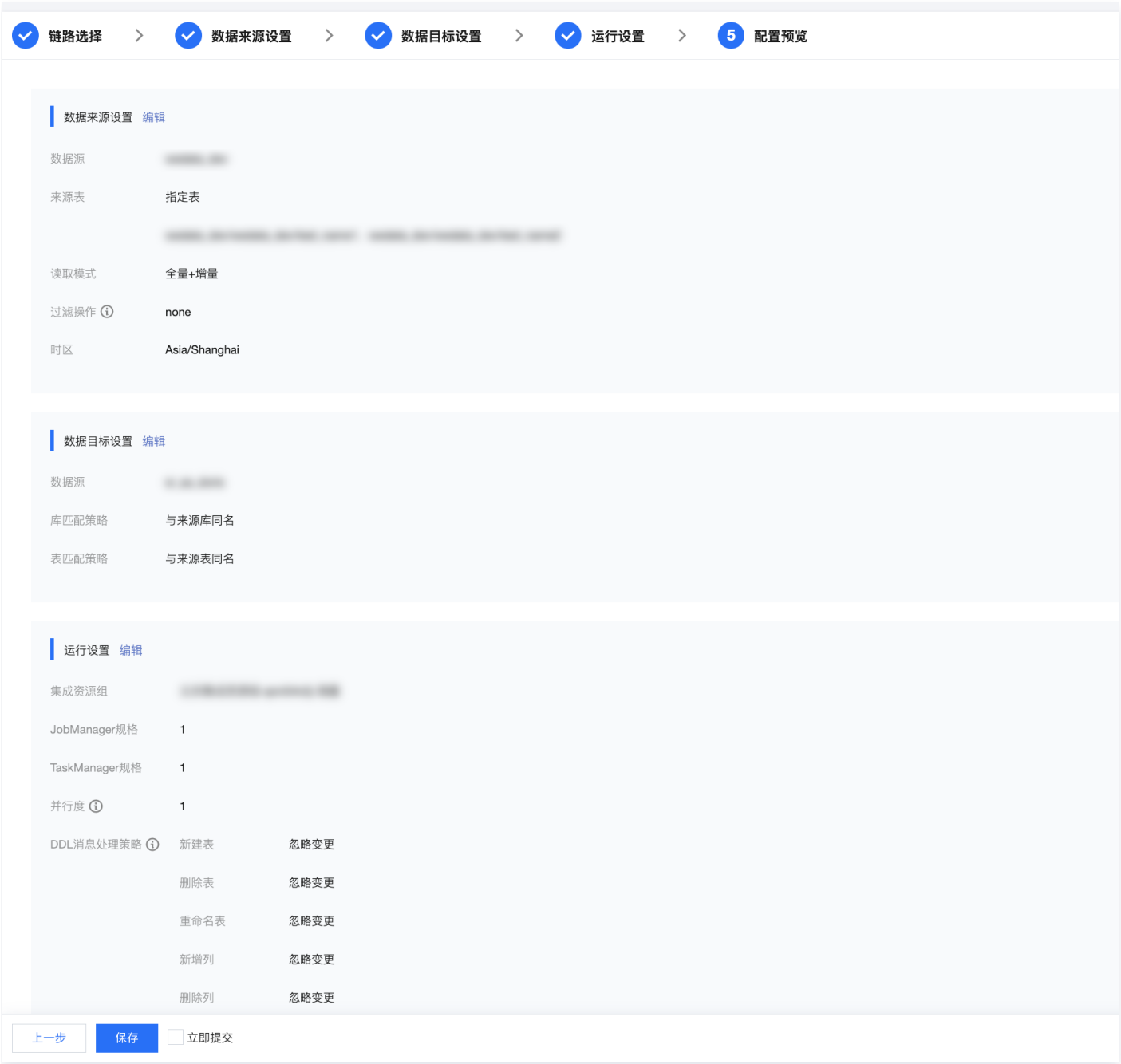
| 参数 | 策略名 | 策略说明 |
|---|---|---|

| | | |
|---|---|---|
| DDL 消息处理 | 新建表 | 1. 自动建表：当来源端被监控的库中出现新建表时，Doris 端将自动创建同结构的表及字段：<br>○ 若来源端表包含主键，任务默认创建 Unique key 模型表。<br>○ 若来源端表包含主键，任务默认创建 Duplicate 模型表。<br>2. 忽略变更：目标端忽略来源端的产生的 DDL 变更消息，Doris 端及日志不做任何响应或消息提醒。<br>3. 日志告警：目标端仅接收 DDL 变更消息，并在日志内打印消息内容，不触发新建表操作。<br>4. 任务出错：目标端接收 DDL 变更消息并持续重启任务，重启过程中任务日志报错并出现数据写入异常。 |
| | 新增列 | 1. 新增列：当来源端被监控的库中出现表增加字段时，Doris 端将自动同步新增同名字段。<br>2. 忽略变更：目标端忽略来源端的产生的 DDL 变更消息，Doris 端及日志不做任何响应或消息提醒。<br>3. 日志告警：目标端仅接收 DDL 变更消息，并在日志内打印消息内容。此策略并不触发新增列操作。<br>4. 任务出错：目标端接收 DDL 变更消息并持续重启任务，重启过程中任务日志报错并出现数据写入异常。 |
| | 删除表 | 除新建表、新增字段外其他 DDL 变更消息不支持自动响应，目前提供忽略变更、日志告警、任务出错三种策略选择：<br>1. 忽略变更：目标端忽略来源端的产生的 DDL 变更消息，Doris 端及日志不做任何响应或消息提醒。<br>2. 日志告警：目标端仅接收 DDL 变更消息，并在日志内打印消息内容。此策略并不触发新建表操作。<br>3. 任务出错：目标端接收 DDL 变更消息并持续重启任务，重启过程中任务日志报错并出现数据写入异常。 |
| | 重命名表 | |
| | 删除列 | |
| | 重命名列 | |
| | 修改列 | |
| | 删除列 | |
| 写入异常 | 部分停止 | 数据无法写入目标表时丢弃数据，后续该异常表对应的数据自动丢弃不再同步 |
| | 异常重启 | 任意表数据写入异常后任务将异常退出并自动重启。重启后任务将持续尝试写入，直到所有表均可正常同步。重启期间可能导致部分表数据重复写入。 |
| | 忽略异常 | 忽略表内无法写入的异常数据并标记为脏数据，任务继续读取并写入剩下的数据。 |
| 脏数据 | COS 归档 | 写入异常策略配置为 忽略异常 时，将未写入至目标端的数据同步写入到指定的 COS 桶及文件内。 |

| | | |
|---|---|---|
| | 不归档 | 不归档保存未写入的异常的数据 |

## 配置预览及提交

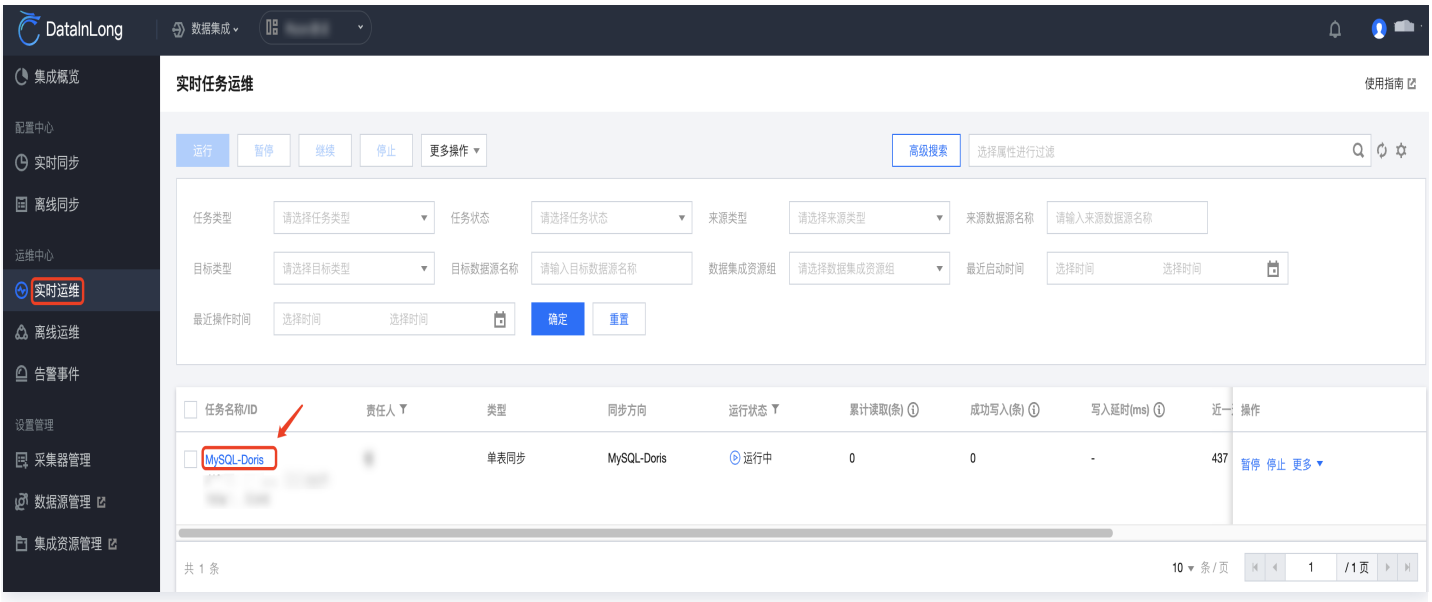
配置完成后可进行预览，确认后单击**保存**。



# 任务运维

最近更新时间：2023-09-05 12:26:24

**提交任务以后，可进入 数据集成 > 实时运维页面查看并监控当前任务状态、读写指标统计、日志及配置当前任务监控规则。**

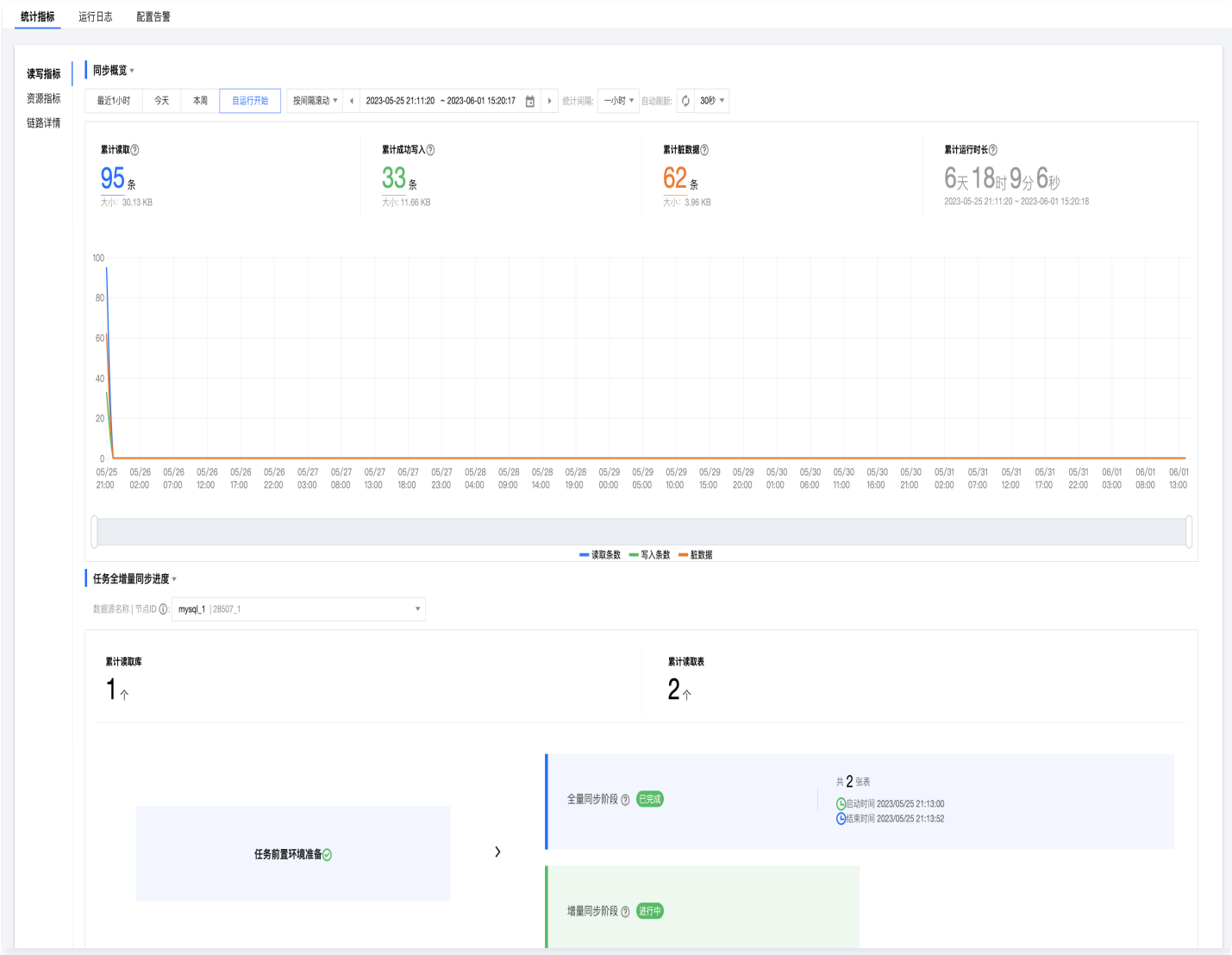



## 统计指标

**统计指标页面展示了任务内读写及资源运行情况。**

### 读写指标

**展示当前任务读写整体条数、全增量同步阶段、以及读写速度等。**

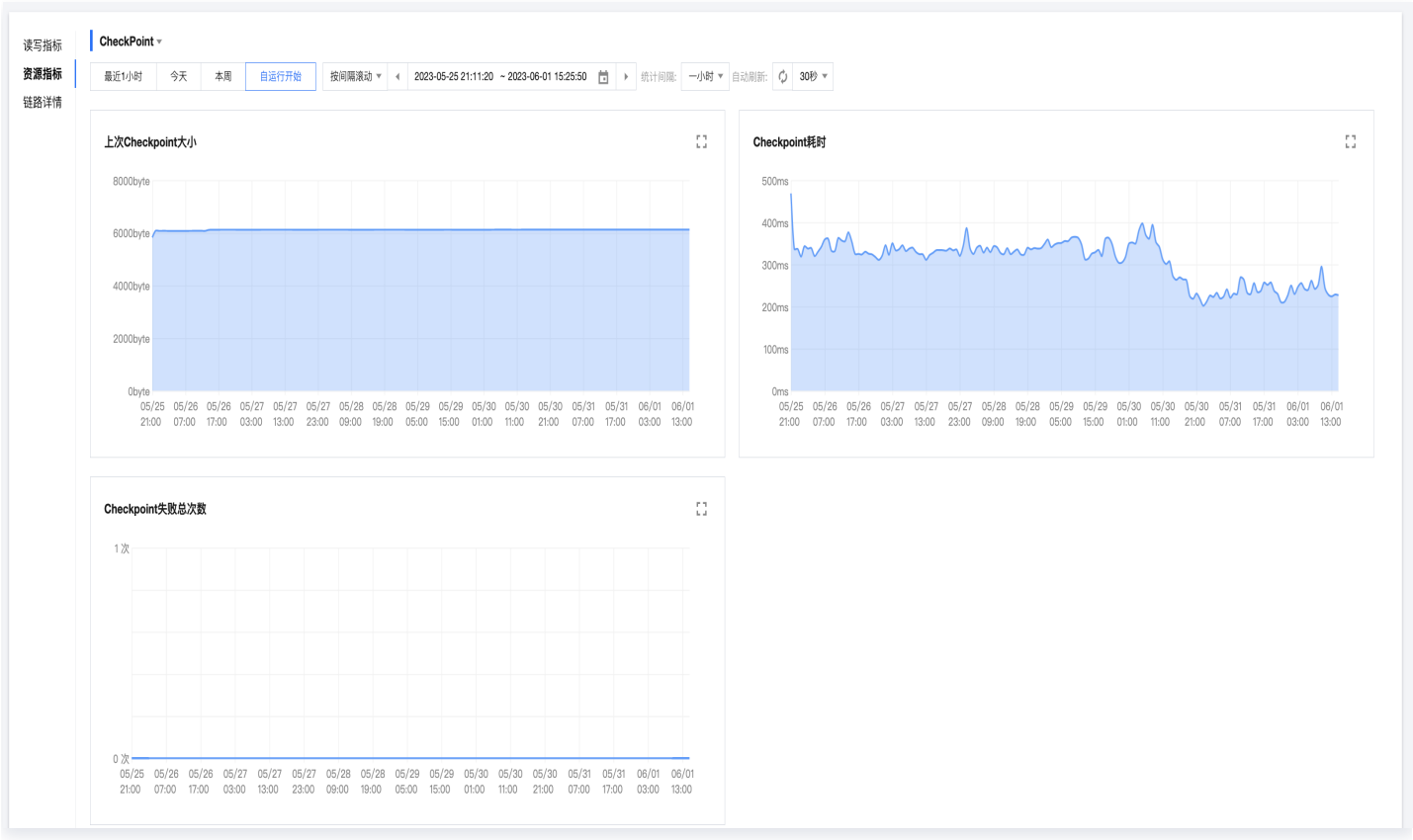



| | 指标参数 | 说明 |
| --- | --- | --- |
| 同步概览 | 累计读取 | 本次任务运行期间，从来源端实际读取数据条数。此指标不包含筛选过滤等方式剔除的数据总量。 |
| | 累计成功写入 | 本次任务运行期间，已读取的数据中成功写入到目标端的数据总量。 |
| | 累计脏数据 | 本次任务运行期间，已读取的数据中异常写入失败的数据总量。此指标不包含任务配置中主动忽略/过滤而导致未写入的数据，包括指定部分停止、异常重启等运行策略，以及数据过滤等。 |
| | 累计运行时长 | 本次任务启动后，累计总运行时长（包含暂停时间）。 |

| | | |
|---|---|---|
| | 累计读取库 | 本次任务运行期间，从来源端实际读取数据库数量。 |
| | 累计读取表 | 本次任务运行期间，从来源端实际读取数据表数量，并且分别全量同步阶段和增量同步阶段数量。 |
| 全增量同步进度 | 全量/增量状态 | 提供未启动、进行中和已完成三种状态。 |
| | 全量同步阶段 | 读取源端库表中的所有记录，本阶段内仅统计读取成功且有存量业务数据的表，并且同步展示增量启动时间、统计时间、全量结束时间。 |
| | 增量同步阶段 | 从 binlog 消费变更数据，本阶段内仅统计读取成功且有新增业务数据的表，并且同步展示增量启动时间。 |
| | 读取速度 | 读取速度 = 统计间隔内总读取条数/统计间隔。 |
| | 读取吞吐 | 读取吞吐 = 统计间隔内总读取总量/统计间隔。 |
| | 写入速度 | 写入速度 = 统计间隔内成功写入条数/统计间隔。 |
| 读写详情 | 写入吞吐 | 写入吞吐 = 统计间隔内成功写入总量/统计间隔。 |
| | 写入延时 | 来源Source端至写入Sink端之间的链路延迟，写入延时=系统时间-记录读取时间（读取端LatencyMarker时间戳）。 |
| | 作业重启次数 | 统计间隔内当前任务重启次数。 |

## 资源指标

展示当前任务使用资源情况。

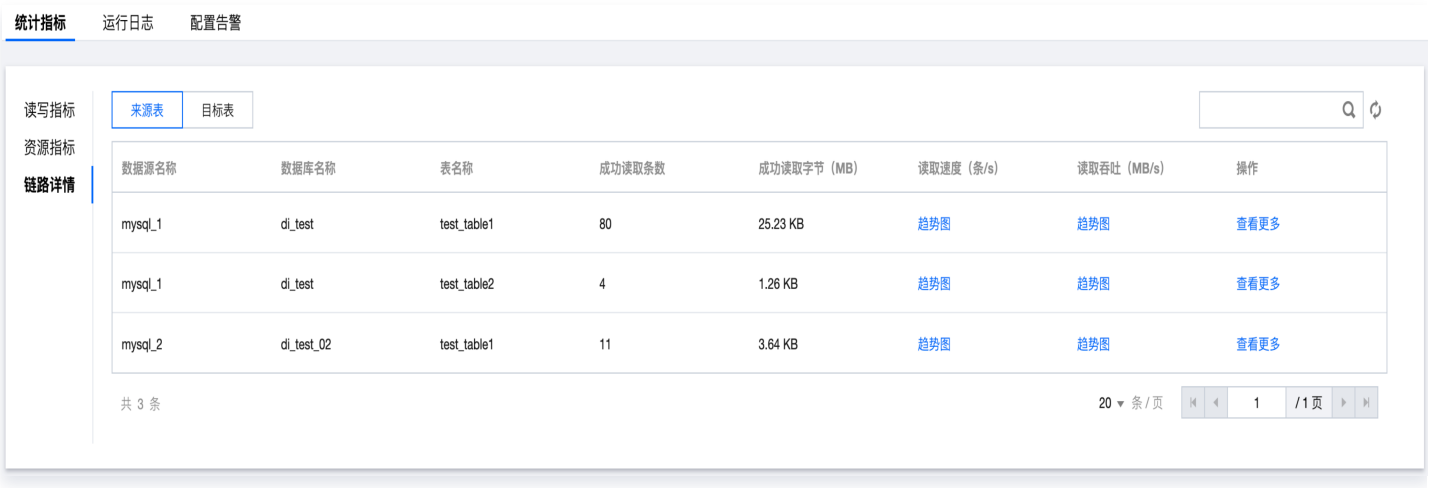



| | 指标参数 | 说明 |
| --- | --- | --- |
| CheckPoint | 上次Checkpoint 大小 | 当前作业最近一次的 Checkpoint 大小 |
| | Checkpoint 耗时 | 当前作业的 Checkpoint 耗时 |
| | Checkpoint 失败总次数 | 当前作业的 Checkpoint 的失败总次数 |
| TaskManager | TaskManager CPU 使用率 | 当前作业 TaskManager 的 CPU 使用率。 |
| | TaskManager 堆内存使用量 | 当前作业 TaskManager 堆内存的用量。 |
| | TaskManager 老年代总 GC 次数 | 当前作业 TaskManager 老年代 GC 次数。 |
| | TaskManager 老年代总 GC 时间 | 当前作业 TaskManager 老年代 GC 时间。 |
| | TaskManager 物理内存用量 | 当前作业 TaskManager 所在的 JVM 的物理内存用量（RSS），包括堆内、堆外、Native 等所有区域的总内存用量。 |

| | | |
|---|---|---|
| JobManager | JM CPU Load | TaskManager 维度的 JVM 最近 CPU 利用率。 |
| | JM Head Memory | TaskManager 维度的堆内存使用情况。 |
| | JM GC Count | TaskManager 维度的 Status.JVM.GarbageCollector.<GarbageCollector>.Count，GC（垃圾回收）次数。 |
| | JM GC Time | TaskManager 维度的 Status.JVM.GarbageCollector.<GarbageCollector>.Time，GC（垃圾回收）时间。 |

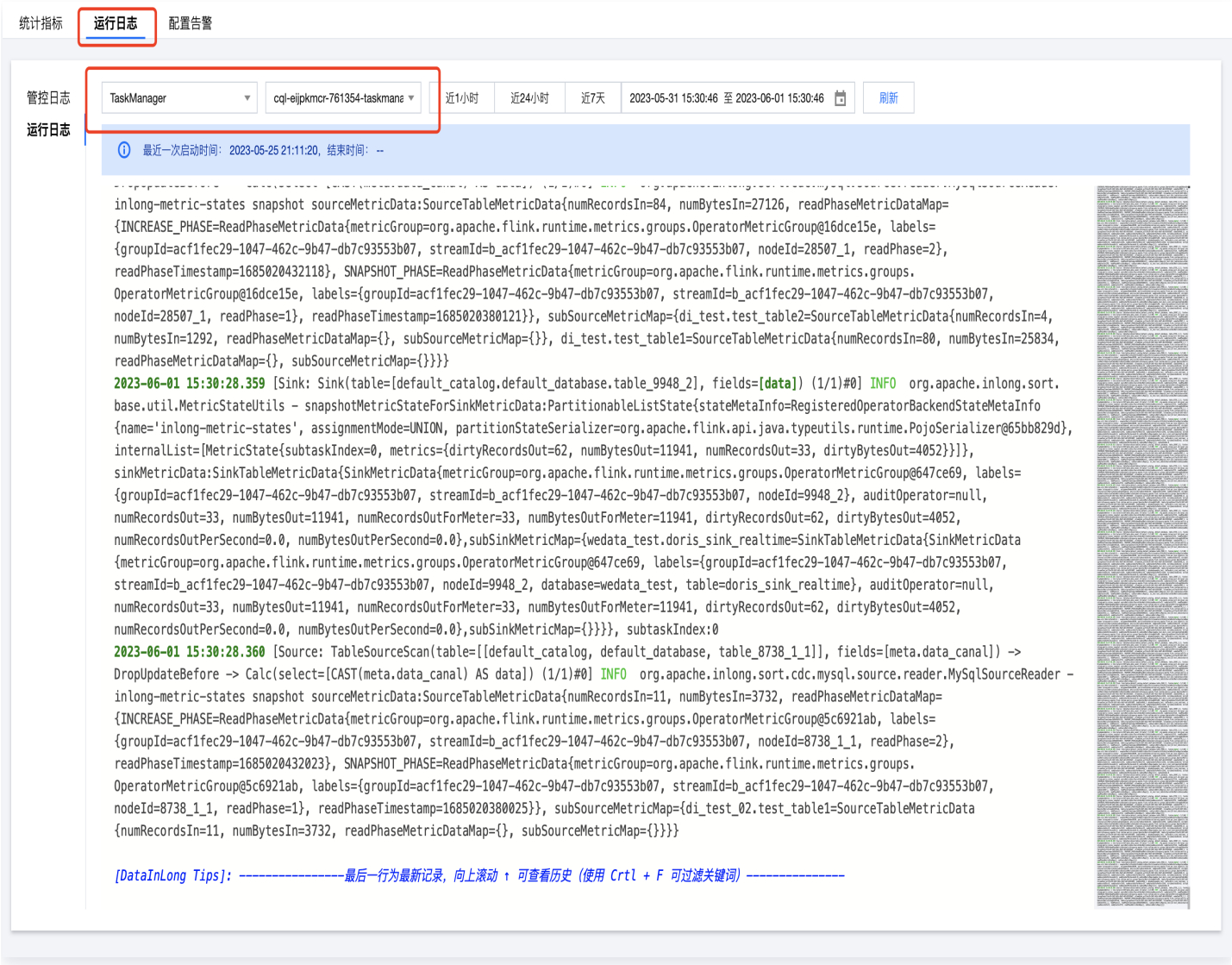
## 链路详情

展示整库任务下每张表的读写情况（仅整库同步时会展示此页面）。




## 运行日志

展示运行日志。




# 配置告警

配置告警页面支持对实时任务创建监控规则及告警渠道。


统计指标
运行日志
配置告警
创建告警规则
请输入规则名称
规则ID
规则名称
规则状态
告警级别
告警指标
指标阈值
告警方式
操作
a
关闭
普通
1、脏数据条数
2、脏数据字节数
1、脏数据条数大于1条后触发告警
2、脏数据字节数大于1字节后触发告警
邮件
查看告警事件 编辑 删除
共 1 条
10 条/页
1
/ 1 页

# 常见问题

最近更新时间：2023-06-15 10:55:02

## 导入任务过多，新导入任务提交报错 “current running txns on db xxx is xx, larger than limit xx”？

调整 fe 参数：max_running_txn_num_per_db，默认100，可适当调大，建议控制在500以内。

## 导入频率太快出现 err=[E-235] 错误?

- 参数调优建议：可通过适当调大 max_tablet_version_num 参数暂时解决，此参数默认200，建议控制在2000以内。
- 业务调优建议：降低导入频率才能根本解决这个问题。

## 导入文件过大，被参数限制。报错 “The size of this batch exceed the max size”？

调整 be 参数：streaming_load_max_mb，建议超过需要导入的文件大小。

## 导入数据报错：“[-238]”？

- 原因：-238 错误通常出现在同一批导入数据量过大的情况，从而导致某一个 tablet 的 Segment 文件过多（由 ）。
- 参数调优建议：可适当调大BE参数max_segment_num_per_rowset ，此参数默认值200，可按倍数调大（如400、800），建议控制在2000以内；
- 业务调优建议：建议减少一批次导入的数据量。

## 导入失败，报错：“too many filtered rows xxx, "ErrorURL":"或 Insert has filtered data in strict mode, tracking url=xxxx.”？

原因：表的 schema、分区等与导入的数据不匹配。可在 CDW Studio 或客户端执行 doris 命令查看具体原因：show load warnings on `<tracking url>`，<tracking url> 即为报错信息中返回的 error url。