# 1.ShardingSphere-Proxy组件调研

## 前言

调研出发点：至少满足实际项目需求。

目的：完成千万级表数据拆分。

## 问题调研

这里调研ShardingSphere-Proxy在拆分过程可能会面临各种各样的问题，具体有那些呢？

- 问题一 表查询问题：在使用MySQL Client去连接ShardingSphere-Proxy时，非分片表与分片表是否都可以查询得出来呢？
- 问题二 数据库脚本版本管理：当使用ShardingSphere-Proxy时，原有基于Flyway进行数据库脚本版本管理可否继续兼容？
- 问题三 表拆分选择问题：对于某张表来说，有些医院可能需要拆分，有些医院不需要拆分，如何满足？
- 问题四 不支持功能问题：对于ShardingSphere-Proxy来说本身存在一些不支持的功能，那么实际需求又是必须的，为此，第一，需要整理实际需求中使用到的功能点有哪些在ShardingSphere-Proxy中不支持，第二，如何解决？
- 问题五 同步程序数据同步问题：对于非分片表和分片表而言，之前数据都是经同步程序通过而来，在使用ShardingSphere-Proxy后，是否有影响呢？如果有？如何解决？
- 问题六 数据迁移问题：对于存量数据如何进行数据同步到分片表中，并保证数据一致性？
- 问题七 单库事务问题（限定条件：单库）：仅单表，分表和单表，仅分表三个事务场景调研。

那么针对以上问题，组件是否已经存在解决方案呢？如何没有现有的方案，那么该如何解决呢？接下来主要描述的就是整个技术调研过程。

**下述问题调研中使用的项目如无特殊说明，均是使用EMR~**

## 问题一：表查询问题

**问题描述**：在使用MySQL Client去连接ShardingSphere-Proxy时，非分片表与分片表是否都可以查询得出来呢?

假设我们需要进行表水平拆分，那么除了需要配置必要的分片配置外，为了在使用MySQL Client去连接ShardingSphere-Proxy时，非分片表都可以查询出来。

### 1.基于配置文件的方式

分片配置文件路径：${SHARDING_PROXY_HOME}/config/config-sharding.yaml，只需要新增如下配置：

```yaml
databaseName: windranger_emr
# 重点就是SINGLE部分的配置
rules:
- !SINGLE
  tables:
    - "*.*"
- !SHARDING
 tables:
   pat_inhos_order_group:
    # 此处省略分片配置
```

已经验证可以解决此问题。

### 2.基于DistSQL的方式

```
# 创建逻辑库
CREATE DATABASE IF NOT EXISTS windranger_emr;
# 引用逻辑库
USE windranger_emr;
# 注册存储单元
REGISTER STORAGE UNIT IF NOT EXISTS windranger_emr_0 (
    HOST="10.2.3.167",
    PORT=3306,
```

```
    DB="windranger_emr",
    USER="user",
    PASSWORD="Lachesis-mh_1024"
);
# 加载全部单表
LOAD SINGLE TABLE *.*;
```

## 问题二：数据库脚本版本管理

**问题描述**：当使用ShardingSphere-Proxy时，原有基于Flyway进行数据库脚本版本管理可否继续兼容?

### 不改原有配置的情况下是否可行?

显然不可行，因为中间多了一层代理，在项目中连接的URL地址需要换成了代理的地址。

### 替换代理地址后是否可行?

替换代理地址：

```yaml
spring:
  datasource:
    url: jdbc:mysql://10.2.3.167:3307/windranger_emr
    username: root
    password: root
```

此处为了减少改动，我们将ShardingSphere-Proxy中的逻辑库名与物理库名设置为一样~

不可行，具体的报错信息：

**项目启动错误日志**

```
SQL State  : HY000
Error Code : 30000
```

```
Message    : Unknown exception: org.apache.calcite.rex.RexSubQuery cannot be cast to org.apache.calcite.rex.RexLocalRef

    at org.flywaydb.core.internal.database.base.Schema.exists(Schema.java:75)
    at org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:73)
    at org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:69)
    at org.flywaydb.core.internal.jdbc.TransactionTemplate.execute(TransactionTemplate.java:74)
    at org.flywaydb.core.internal.command.DbSchemas.create(DbSchemas.java:69)
    at org.flywaydb.core.Flyway$1.execute(Flyway.java:1369)
    at org.flywaydb.core.Flyway$1.execute(Flyway.java:1356)
    at org.flywaydb.core.Flyway.execute(Flyway.java:1711)
    at org.flywaydb.core.Flyway.migrate(Flyway.java:1356)
    at com.lachesis.molecule.common.database.DbVersionMigrationMonitor.run(DbVersionMigrationMonitor.java:45)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:498)
    at org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProcessor$LifecycleElement.invoke(InitDestroyAnnotationBeanPostProcessor.java:363)
    at org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProcessor$LifecycleMetadata.invokeInitMethods(InitDestroyAnnotationBeanPostProcessor.java:307)
    at org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProcessor.postProcessBeforeInitialization(InitDestroyAnnotationBeanPostProcessor.java:136)
    ... 16 common frames omitted
Caused by: java.sql.SQLException: Unknown exception: org.apache.calcite.rex.RexSubQuery cannot be cast to org.apache.calcite.rex.RexLocalRef
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:129)
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
    at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:122)
```

```
27      at
com.mysql.cj.jdbc.ClientPreparedStatement.executeInternal(ClientPreparedStatem
ent.java:974)
28      at
com.mysql.cj.jdbc.ClientPreparedStatement.executeQuery(ClientPreparedStatement
.java:1024)
29      at
com.alibaba.druid.pool.DruidPooledPreparedStatement.executeQuery(DruidPooledPr
eparedStatement.java:227)
30      at
org.flywaydb.core.internal.jdbc.JdbcTemplate.queryForInt(JdbcTemplate.java:139
)
31      at
org.flywaydb.core.internal.database.mysql.MySQLSchema.doExists(MySQLSchema.jav
a:44)
32      at org.flywaydb.core.internal.database.base.Schema.exists(Schema.java:73)
33      ... 32 common frames omitted
```

可以看到上述报错有一个schema是否存在的校验，存在直接返回true，不存在会进行schema创建，由于ShardingSphere-Proxy对information_schema的查询支持比较弱，比如

ShardingSphere-Proxy

ShardingSphere-Proxy 的定位为透明化的数据库代理，理论上支持任何使用 MySQL、PostgreSQL、openGauss 协议的客户端操作数据，对异构语言、运维场景更友好。

- 应用场景
- 使用限制
- 前提条件
- 操作步骤

## 使用限制

ShardingSphere-Proxy 对系统库/表（如 information_schema、pg_catalog）支持有限，通过部分图形化数据库客户端连接 Proxy 时，可能客户端或 Proxy 会有错误提示。可以使用命令行客户端（`mysql`、`psql`、`gsql` 等）连接 Proxy 验证功能。

```
mysql> SELECT schema_name FROM information_schema.schemata;
ERROR 30000 (HY000): Unknown exception: Index: 1, Size: 1
mysql> SELECT schema_name FROM information_schema.schemata WHERE schema_name='test';
+--------------------+
| SCHEMA_NAME        |
+--------------------+
| shardingsphere     |
| performance_schema |
| information_schema |
| test               |
| mysql              |
| sys                |
+--------------------+
6 rows in set (0.00 sec)
```

对于SQL：SELECT schema_name FROM information_schema.schemata WHERE schema_name='test'; 是可以执行成功的，但是条件并没有生效~

如果将条件去掉呢？就会报错：ERROR 3000~

因此，尝试通过修改Flyway的校验逻辑：

**org.flywaydb.core.internal.database.mysql.MySQLSchema**

```
@Override
protected boolean doExists() throws SQLException {
    try {
        return jdbcTemplate.queryForInt("SELECT (SELECT 1 FROM information_schema.schemata WHERE schema_name=? LIMIT 1)", name) > 0;
    } catch (Exception e) {
        String querySql = String.format("SELECT schema_name FROM information_schema.schemata WHERE schema_name='%s'", name);
        List<String> schemaNames = jdbcTemplate.query(querySql,  new RowMapper<String>() {
            @Override
            public String mapRow(ResultSet rs) throws SQLException {
                return rs.getString("schema_name");
            }
        });
        return schemaNames.contains(name);
    }
}
```

发现修改后，又会有新的错误：

**项目启动错误日志**

```
SQL State  : HY000
Error Code : 30000
Message    : Unknown exception: Index: 1, Size: 1

        at org.flywaydb.core.internal.database.base.Schema.exists(Schema.java:75)
        at org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:73)
        at org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:69)
        at org.flywaydb.core.internal.jdbc.TransactionTemplate.execute(TransactionTemplat
```

```
e.java:74)
        at
org.flywaydb.core.internal.command.DbSchemas.create(DbSchemas.java:69)
        at org.flywaydb.core.Flyway$1.execute(Flyway.java:1369)
        at org.flywaydb.core.Flyway$1.execute(Flyway.java:1356)
        at org.flywaydb.core.Flyway.execute(Flyway.java:1711)
        at org.flywaydb.core.Flyway.migrate(Flyway.java:1356)
        at
com.lachesis.molecule.common.database.DbVersionMigrationMonitor.run(DbVersionM
igrationMonitor.java:45)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.j
ava:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor$LifecycleElement.invoke(InitDestroyAnnotationBeanPostProcessor.java:363)
        at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor$LifecycleMetadata.invokeInitMethods(InitDestroyAnnotationBeanPostProcess
or.java:307)
        at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor.postProcessBeforeInitialization(InitDestroyAnnotationBeanPostProcessor.j
ava:136)
        ... 16 common frames omitted
Caused by: java.sql.SQLException: Unknown exception: Index: 1, Size: 1
        at
com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:129)
        at
com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
        at
com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExcept
ionsMapping.java:122)
        at
com.mysql.cj.jdbc.StatementImpl.executeQuery(StatementImpl.java:1218)
        at
com.alibaba.druid.pool.DruidPooledStatement.executeQuery(DruidPooledStatement.
java:138)
        at
org.flywaydb.core.internal.jdbc.JdbcTemplate.query(JdbcTemplate.java:364)
        at
org.flywaydb.core.internal.database.mysql.MySQLSchema.doExists(MySQLSchema.jav
a:51)
```

```
31          at
    org.flywaydb.core.internal.database.base.Schema.exists(Schema.java:73)
32          ... 32 common frames omitted
```

最后，只能想着 直接绕开schema校验与创建逻辑：

**org.flywaydb.core.internal.database.mysql.MySQLSchema**

```
1   @Override
2   protected boolean doExists() throws SQLException {
3       return;
4   }
```

这样子改的初衷是：一般极少可能存在创建schema的情况，所以想着牺牲一点能力（虽然schema校验执行不了，但是可以创建），保证Flyway与ShardingSphere-Proxy的兼容性~

最终，发现可以Flyway可以执行成功了~

**上面尽管成功了，但是是非分片表的SQL脚本版本管理，而对于分片表是否能执行成功呢？**

为了解决这个问题，就必须如何通过SQL脚本实现分片表DDL或者DML操作呢？就是说，我执行了一条表创建语句，最终在ShardingSphere-Proxy组件是否可以协助根据分片规则创建对应分片数量的表呢？

答案是肯定的，这里就需要引入ShardingSphere-Proxy强大的DistSQL了，它是从5.0.0-Beta版本之后引入进来的，那它到底能做什么呢？可以参见 官方文档说明 ，后续补充一些案例到Confluence中，在此我们仅先测试一下分片表的创建，是否可以借助Flyway实现呢？

下面贴出创建分片表的SQL脚本：

```
1   CREATE DATABASE IF NOT EXISTS windranger_emr;
2
3   USE windranger_emr;
4
5   REGISTER STORAGE UNIT IF NOT EXISTS windranger_emr_0 (
6       HOST="10.2.3.167",
```

```
    PORT=3306,
    DB="windranger_emr",
    USER="user",
    PASSWORD="Lachesis-mh_1024"
);

LOAD SINGLE TABLE *.*;

CREATE SHARDING TABLE RULE IF NOT EXISTS t_order_item (
    STORAGE_UNITS(ds_0),
    SHARDING_COLUMN=order_id,
    TYPE(NAME=MOD,PROPERTIES("sharding-count"=4)),
    KEY_GENERATE_STRATEGY(COLUMN=order_id,TYPE(NAME=SNOWFLAKE))
);

CREATE TABLE IF NOT EXISTS `t_order_item`(
    `id`             bigint unsigned NOT NULL AUTO_INCREMENT,
    `order_id`       bigint unsigned NOT NULL DEFAULT '0',
    `order_item_id`  bigint unsigned NOT NULL DEFAULT '0',
    `order_item_name` varchar(100)   NOT NULL DEFAULT '',
    `create_time`    bigint          NOT NULL DEFAULT '0',
    `update_time`    bigint          NOT NULL DEFAULT '0',
    PRIMARY KEY (`id`)
) ENGINE = InnoDB;
```

启动EMR项目，发现新的报错：

**项目启动错误日志**

> 展开源码

```
SQL State  : null
Error Code : 0
Message    : wait millis 10014, active 0, maxActive 8

        at org.flywaydb.core.internal.jdbc.JdbcUtils.openConnection(JdbcUtils.java:60)
        at org.flywaydb.core.internal.database.DatabaseFactory.createDatabase(DatabaseFactory.java:72)
        at org.flywaydb.core.Flyway.execute(Flyway.java:1670)
        at org.flywaydb.core.Flyway.migrate(Flyway.java:1356)
```

```
        at com.lachesis.molecule.common.database.DbVersionMigrationMonitor.run(DbVersionMigrationMonitor.java:45)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProcessor$LifecycleElement.invoke(InitDestroyAnnotationBeanPostProcessor.java:363)
        at org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProcessor$LifecycleMetadata.invokeInitMethods(InitDestroyAnnotationBeanPostProcessor.java:307)
        at org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProcessor.postProcessBeforeInitialization(InitDestroyAnnotationBeanPostProcessor.java:136)
        ... 16 common frames omitted
Caused by: com.alibaba.druid.pool.GetConnectionTimeoutException: wait millis 10014, active 0, maxActive 8
        at com.alibaba.druid.pool.DruidDataSource.getConnectionInternal(DruidDataSource.java:1190)
        at com.alibaba.druid.pool.DruidDataSource.getConnectionDirect(DruidDataSource.java:1014)
        at com.alibaba.druid.pool.DruidDataSource.getConnection(DruidDataSource.java:994)
        at com.alibaba.druid.pool.DruidDataSource.getConnection(DruidDataSource.java:984)
        at com.alibaba.druid.pool.DruidDataSource.getConnection(DruidDataSource.java:103)
        at org.flywaydb.core.internal.jdbc.JdbcUtils.openConnection(JdbcUtils.java:56)
        ... 27 common frames omitted
Caused by: java.sql.SQLSyntaxErrorException: Unknown database 'windranger_emr'
        at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:120)
        at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
        at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:122)
```

```
30          at
com.mysql.cj.jdbc.ConnectionImpl.createNewIO(ConnectionImpl.java:835)
31          at com.mysql.cj.jdbc.ConnectionImpl.<init>(ConnectionImpl.java:455)
32          at
com.mysql.cj.jdbc.ConnectionImpl.getInstance(ConnectionImpl.java:240)
33          at
com.mysql.cj.jdbc.NonRegisteringDriver.connect(NonRegisteringDriver.java:207)
34          at
com.alibaba.druid.pool.DruidAbstractDataSource.createPhysicalConnection(DruidA
bstractDataSource.java:1421)
35          at
com.alibaba.druid.pool.DruidAbstractDataSource.createPhysicalConnection(DruidA
bstractDataSource.java:1477)
36          at
com.alibaba.druid.pool.DruidDataSource$CreateConnectionThread.run(DruidDataSou
rce.java:1998)
```

**疑问：其实在使用MySQL CLI时，上述脚本都是依次可以执行的，为什么Flyway中执行就报错了呢？**

这是因为在Flyway框架执行时会有各种各样的查询，比如我要创建逻辑库windranger_emr，但是呢？Flyway在连接的时候去连接windranger_emr库，查询windranger_emr库是否存在时就会因为ShardingSphere-Proxy一些限制，导致查询失败~

类似查询在Flyway中还有，就不一一列举了~

**换个思路：我们借助原生JDBC去实现调用下上述脚本，看能否成功？猜测是成功的，毕竟都是遵循SQL协议的，接下来进行验证猜想。**

验证代码：

```
1   public class JdbcTestMain {
2       public static void main(String[] args) {
3           String jdbcUrl = "jdbc:mysql://10.2.3.167:3307/windranger_emr?
createDatabaseIfNotExist=true";
4           String username = "root";
5           String password = "root";
6           try (Connection connection = DriverManager.getConnection(jdbcUrl,
username, password);
7                Statement statement = connection.createStatement()) {
8               // CREATE DATABASE
```

```
            statement.executeUpdate("CREATE DATABASE IF NOT EXISTS windranger_emr");
            statement.executeUpdate("USE windranger_emr");
            // REGISTER STORAGE UNIT
            statement.executeUpdate("REGISTER STORAGE UNIT IF NOT EXISTS ds_0 " +
                "(HOST='10.2.3.167', PORT=3306, DB='windranger_emr', USER='user', PASSWORD='Lachesis-mh_1024')");
            // CREATE SHARDING TABLE RULE
            statement.executeUpdate("CREATE SHARDING TABLE RULE IF NOT EXISTS t_order_item " +
                "(STORAGE_UNITS(ds_0), SHARDING_COLUMN=order_id, TYPE(NAME=MOD, PROPERTIES('sharding-count'=4)), " +
                "KEY_GENERATE_STRATEGY(COLUMN=order_id, TYPE(NAME=SNOWFLAKE)))");
            // CREATE TABLE
            statement.executeUpdate("CREATE TABLE IF NOT EXISTS `t_order_item` (" +
                "`id` bigint unsigned NOT NULL AUTO_INCREMENT," +
                "`order_id` bigint unsigned NOT NULL DEFAULT '0'," +
                "`order_item_id` bigint unsigned NOT NULL DEFAULT '0'," +
                "`order_item_name` varchar(100) NOT NULL DEFAULT ''," +
                "`create_time` bigint NOT NULL DEFAULT '0'," +
                "`update_time` bigint NOT NULL DEFAULT '0'," +
                "PRIMARY KEY (`id`)) ENGINE = InnoDB");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

通过验证，猜想是成功的。

## 总结

暂时先搁置Flyway的各种不兼容问题了，即使换一个数据库版本框架Liquibase，同样是有不兼容的情况存在。

那怎么办呢?

1. 先暂时将分片表相关脚本放置在一个项目中，不同版本，不同文件即可实现版本控制，且不依赖Flyway，这样子Flyway就执行不到，也不会影响程序正常启动~
2. 脚本版本控制容易，但是并没有生效，在不依赖Flyway的情况下，怎么生效呢？目前给出的兜底方案：

- 自动执行：自行实现分片表脚本的版本管理并执行，初版可以做的相对简单些~
- 手动执行：使用MySQL CLI去执行脚本

## 问题三：表拆分选择问题

**问题描述**：对于某张表来说，有些医院可能需要拆分，有些医院不需要拆分，如何满足？

这里存在两种情况：

1. 不存在拆分旧表，只需要创建分表（比如新上医院，预估的数据也比较多，初始化数据库时直接创建分表）
    a. 对于这种情况，基于上述问题二的调研结果，对于问题三来说，就相对容易了，因为问题二中是将单表和分表的脚本管理区分放置的，只需要分开放置即可。
3. 需要拆分旧表（这块内容在后续方案设计时会进行详细阐述）
    a. 对于这种情况就相对麻烦了，需要考略的因素：
        i. 旧表迁移
        ii. 单表和分表均会存在一段时间，后续通过数据同步，数据校验等阶段后，在切换时，需要有一套健壮的设计方案

## 问题四：不支持功能问题

**问题描述**：对于ShardingSphere-Proxy来说本身存在一些不支持的功能，那么实际需求又是必须的，为此，第一，需要整理实际需求中使用到的功能点有哪些在ShardingSphere-Proxy中不支持，第二，如何解决？

### ShardingSphere-Proxy限制功能

- **稳定支持**
  - 常规查询
  - 子查询
  - 分页查询
  - 运算表达式中包含分片键
  - LOAD DATA / LOAD XML
- **实验性支持**
  - 子查询
  - 跨库关联查询
- **不支持**
  - CASE WHEN
  - 分页查询
  - LOAD DATA / LOAD XML

### CASE WHEN

以下CASE WHEN语句不支持：

- CASE WHEN中包含子查询
- CASE WHEN中使用逻辑表名

### 分页查询

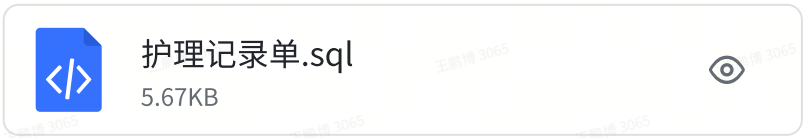
完全支持MySQL、PostgreSQL、openGauss， Oracle和SQLServer由于分页查询较为复杂，仅部分支持 。

5.x版本之前不支持UNION/UNION ALL和子查询，在最新版本是支持的！

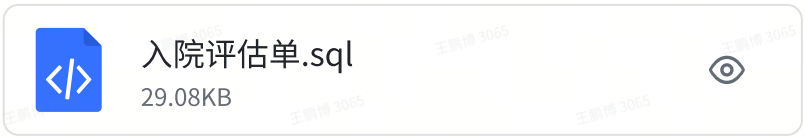
**限制功能对应项目使用场景**

1. 针对于一阶段的医嘱表（pat_inhos_order_group和pat_inhos_order）来说，业务代码中主要就是CASE WHEN场景，ShardingSphere-Proxy不支持哦~
2. 互联互通提供外部视图，主要就是CASE WHEN场景，ShardingSphere-Proxy不支持哦~

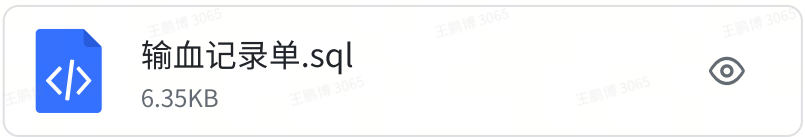
根据实施提供的视图里面含有的CASE WHEN使用场景，视图见附件：












上述互联互通SQL由实施王泽朝提供乌鲁木齐妇幼医院线上使用到的视图脚本。

## 问题五：同步程序数据同步问题

**问题描述**：对于非分片表和分片表而言，之前数据都是经同步程序通过而来，在使用ShardingSphere-Proxy后，是否有影响呢？如果有？如何解决？

**同步遇到问题一：分片键更新问题**

**错误日志**

```
Caused by: java.sql.SQLException: Can not update sharding value for table `pat_inhos_order_group`.
        at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:129)
        at com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
        at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:122)
        at com.mysql.cj.jdbc.ClientPreparedStatement.executeInternal(ClientPreparedStatement.java:974)
        at com.mysql.cj.jdbc.ClientPreparedStatement.execute(ClientPreparedStatement.java:391)
        at com.zaxxer.hikari.pool.ProxyPreparedStatement.execute(ProxyPreparedStatement.java:44)
        at com.zaxxer.hikari.pool.HikariProxyPreparedStatement.execute(HikariProxyPreparedStatement.java)
        at org.apache.ibatis.executor.statement.PreparedStatementHandler.update(PreparedStatementHandler.java:46)
        at org.apache.ibatis.executor.statement.RoutingStatementHandler.update(RoutingStatementHandler.java:74)
        at org.apache.ibatis.executor.SimpleExecutor.doUpdate(SimpleExecutor.java:50)
        at org.apache.ibatis.executor.BaseExecutor.update(BaseExecutor.java:117)
        at org.apache.ibatis.executor.CachingExecutor.update(CachingExecutor.java:76)
        at sun.reflect.GeneratedMethodAccessor293.invoke(Unknown Source)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.ibatis.plugin.Plugin.invoke(Plugin.java:63)
        at com.sun.proxy.$Proxy437.update(Unknown Source)
        at org.apache.ibatis.session.defaults.DefaultSqlSession.update(DefaultSqlSession.java:198)
        at sun.reflect.GeneratedMethodAccessor292.invoke(Unknown Source)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
```

```
22          at java.lang.reflect.Method.invoke(Method.java:498)
23          at
    org.mybatis.spring.SqlSessionTemplate$SqlSessionInterceptor.invoke(SqlSessionT
    emplate.java:433)
24          ... 123 more
```

该问题是因为在同步程序中PAT_INHOS_ORDER_GROUP是通过group_unique_code字段判断唯一性的，所以可能会存在相同group_unique_code时，更新inhos_code的情况（这里inhos_code为分片键），所以就会抛出上述异常~

## 问题六：数据迁移问题

**问题描述**：对于存量数据如何进行数据同步到分片表中，并保证数据一致性？

在同步完成之后，进行数据校验时，日志如下：

```
1   [INFO ] 2024-01-08 15:12:27.391 [ShardingSphere-pipeline-
    j0202p0000j0102p0000013efc3a9791b6876cc092ed25155f841-check-0]
    o.a.s.d.p.c.l.PipelineContextManagerLifecycleListener - mode type is not
    Cluster, mode type='Standalone', ignore
2   [INFO ] 2024-01-08 15:14:58.785 [ShardingSphere-Command-9]
    o.a.s.d.p.s.c.a.i.ConsistencyCheckJobAPI - check job already exists and
    status is not FINISHED,
    progress=Optional[ConsistencyCheckJobItemProgress(status=RUNNING,
    tableNames=windranger_emr_source.pat_inhos_order_group, ignoredTableNames=,
    checkedRecordsCount=0, recordsCount=28735078,
    checkBeginTimeMillis=1704697937731, checkEndTimeMillis=null,
    sourceTableCheckPositions={}, targetTableCheckPositions={},
    sourceDatabaseType=MySQL)]
```

啊这...，在启动的时候明明使用的是Cluster模式呀，这个报错没看懂~

所以查询数据一致性校验状态一直是这样子~

```
1   mysql> SHOW MIGRATION CHECK STATUS
    'j0102p0000013efc3a9791b6876cc092ed25155f84';
2   +-----------------------------------------------+--------+---------------------
    +--------+-------------------------------+-------------------------------+------
```

```
--------------------+-------------------------+----------------+-------------
----+----------------+-----------------+---------------+
3  | tables                                  | result | check_failed_tables
   | active | inventory_finished_percentage | inventory_remaining_seconds |
   incremental_idle_seconds | check_begin_time        | check_end_time |
   duration_seconds | algorithm_type | algorithm_props | error_message |
4  +-----------------------------------------+--------+--------------------
   +--------+-------------------------------+----------------------------+------
   --------------------+-------------------------+----------------+-------------
   ----+----------------+-----------------+---------------+
5  | windranger_emr_source.pat_inhos_order_group |        |
   | true   | 0                              | 0                          |
                    | 2024-01-08 15:12:17.731 |                | 0
       | CRC32_MATCH    |                 |               |
6  +-----------------------------------------+--------+--------------------
   +--------+-------------------------------+----------------------------+------
   --------------------+-------------------------+----------------+-------------
   ----+----------------+-----------------+---------------+
```

但是，数据还是同步过去了，就是目前使用 校验功能有问题 ~

## 问题七：单库事务问题

TODO

# 踩坑集锦

## 踩坑一：基于DistSQL创建的分片规则删除不了（难以复现）

```
1  mysql> show databases;
2  +--------------------+
3  | schema_name        |
4  +--------------------+
5  | information_schema |
6  | mysql              |
7  | performance_schema |
```

```
| sharding_db        |
| shardingsphere     |
| sys                |
| test               |
| windranger_emr     |
+--------------------+
mysql> use sharding_db;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> SHOW SHARDING TABLE RULES;
+--------------+--------------------+----------------------+--------------------
----+--------------------------+-----------------------------------+-----------
-----------------------+---------------------+----------------------+-------
-----------------------+-----------------------------------+--------------------
-+--------------------+---------------------+--------------+-----------------
---+
| table        | actual_data_nodes | actual_data_sources |
database_strategy_type | database_sharding_column |
database_sharding_algorithm_type | database_sharding_algorithm_props |
table_strategy_type | table_sharding_column | table_sharding_algorithm_type |
table_sharding_algorithm_props | key_generate_column | key_generator_type |
key_generator_props | auditor_types | allow_hint_disable |
+--------------+--------------------+----------------------+--------------------
----+--------------------------+-----------------------------------+-----------
-----------------------+---------------------+----------------------+-------
-----------------------+-----------------------------------+--------------------
-+--------------------+---------------------+--------------+-----------------
---+
| t_order_item |                  | ds_0                 |
    |                          |                                   |
                          | STANDARD            | order_id               | mod
                          | sharding-count=4               | order_id
 | snowflake          |                      |                |
   |
+--------------+--------------------+----------------------+--------------------
----+--------------------------+-----------------------------------+-----------
-----------------------+---------------------+----------------------+-------
-----------------------+-----------------------------------+--------------------
-+--------------------+---------------------+--------------+-----------------
---+
1 row in set (0.00 sec)

mysql> DROP SHARDING TABLE RULE t_order_item;
Query OK, 0 rows affected (0.03 sec)

```

```
mysql> SHOW SHARDING TABLE RULES;
+--------------+-------------------+--------------------+-------------------
----+--------------------------+-----------------------------------+-----------
-----------------------+---------------------+-----------------------+-------
-----------------------+-----------------------------------+-------------------
-+--------------------+--------------------+--------------+-----------------
---+
| table        | actual_data_nodes | actual_data_sources |
database_strategy_type | database_sharding_column |
database_sharding_algorithm_type | database_sharding_algorithm_props |
table_strategy_type | table_sharding_column | table_sharding_algorithm_type |
table_sharding_algorithm_props | key_generate_column | key_generator_type |
key_generator_props | auditor_types | allow_hint_disable |
+--------------+-------------------+--------------------+-------------------
----+--------------------------+-----------------------------------+-----------
-----------------------+---------------------+-----------------------+-------
-----------------------+-----------------------------------+-------------------
-+--------------------+--------------------+--------------+-----------------
---+
| t_order_item |                | ds_0                |
    |                        |                                   |
                        | STANDARD            | order_id              | mod
                        | sharding-count=4              | order_id
 | snowflake         |                    |              |
   |
+--------------+-------------------+--------------------+-------------------
----+--------------------------+-----------------------------------+-----------
-----------------------+---------------------+-----------------------+-------
-----------------------+-----------------------------------+-------------------
-+--------------------+--------------------+--------------+-----------------
---+
1 row in set (0.00 sec)
```

可以明明是可以查到该分片规则的，但是就是删除不了~

这个问题难以复现~

## 踩坑二：基于DistSQL创建的分片规则不支持创建表时同时创建索引

**前提**：准备好一个初始化好的环境。

- 如果是基于ZooKeeper的Cluster配置的化，清空ZK对应节点下的所有数据。
- 确保分片表在库中不存在。

**复现步骤**：

```
SHOW DATABASES;

CREATE DATABASE windranger_emr;

USE windranger_emr;

-- 注册存储单元
REGISTER STORAGE UNIT IF NOT EXISTS ds_0 (
    URL="jdbc:mysql://10.2.3.167:3306/windranger_emr?characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true",
    USER="user",
    PASSWORD="Lachesis-mh_1024",
    PROPERTIES("minPoolSize"="1","maxPoolSize"="20","idleTimeout"="60000")
);

-- 自动分片规则
CREATE SHARDING TABLE RULE IF NOT EXISTS pat_inhos_order_group (
    STORAGE_UNITS(ds_0),
    SHARDING_COLUMN=inhos_code,TYPE(NAME="hash_mod",PROPERTIES("sharding-count"="4"))
);

CREATE TABLE IF NOT EXISTS `pat_inhos_order_group` (
  `seq_id` bigint(20) NOT NULL AUTO_INCREMENT COMMENT '自增长编号',
  `group_unique_code` varchar(60) NOT NULL COMMENT '批次唯一编号',
  `order_group_no` varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
  `inhos_code` varchar(20) DEFAULT '' COMMENT '病人住院号',
  `orderbar` varchar(40) DEFAULT '' COMMENT '医嘱条码',
  `package_bar` varchar(50) DEFAULT NULL COMMENT '药箱条码',
  `plan_time` datetime DEFAULT NULL COMMENT '计划执行时间',
  `order_sort_no` int(11) DEFAULT NULL COMMENT '医嘱排序编号',
  `source_type` varchar(20) DEFAULT '' COMMENT '数据来源',
  `isprint` int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
  `execute_status` int(11) NOT NULL DEFAULT '0' COMMENT '执行状态  0 - 未执行, 1- 执行中,  2 - 已执行,  3-停止，4-作废',
  `execute_date` datetime DEFAULT NULL COMMENT '执行时间',
  `print_date` datetime DEFAULT NULL COMMENT '打印时间',
  `execute_person` varchar(20) DEFAULT '' COMMENT '执行人',
  `print_person` varchar(20) DEFAULT '' COMMENT '打印人',
  `apply_time` datetime DEFAULT NULL COMMENT '开立时间',
  `is_dispensed` int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是,  0 - 否',
  `remark` varchar(100) DEFAULT '' COMMENT ' 医嘱备注',
```

```
  `reason` varchar(200) DEFAULT NULL COMMENT '手动或作废执行的原因',
  `execute_type` int(11) DEFAULT '0' COMMENT '执行方式 0-无效， 1- 扫描执行， 2 - 手动执行',
  `start_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始执行人',
  `start_execute_date` datetime DEFAULT NULL COMMENT '手动补录开始执行时间',
  `start_check_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始核对人',
  `end_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录结束执行人',
  `end_execute_date` datetime DEFAULT NULL COMMENT '手动补录结束执行时间',
  `create_time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建时间',
  `create_person` varchar(50) NOT NULL DEFAULT '' COMMENT '创建人',
  `update_time` datetime DEFAULT NULL COMMENT '修改时间',
  `update_person` varchar(50) DEFAULT NULL COMMENT '修改人',
  PRIMARY KEY (`seq_id`),
  UNIQUE KEY `uq_pat_inhos_order_group_1` (`group_unique_code`),
  KEY `idx_pat_inhos_order_group_1` (`order_group_no`),
  KEY `idx_pat_inhos_order_group_2` (`orderbar`),
  KEY `idx_pat_inhos_order_group_3` (`execute_date`),
  KEY `idx_pat_inhos_order_group_4` (`plan_time`),
  KEY `idx_idx_pat_inhos_order_group_5` (`inhos_code`),
  KEY `idx_pat_inhos_order_group_6`
(`inhos_code`,`plan_time`,`execute_status`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='医嘱批次表';
```

**错误日志**

```
ERROR 20001 (44000): Can not get uniformed table structure for logic table `pat_inhos_order_group`, it has different meta data of actual tables are as follows:
actual table: pat_inhos_order_group, meta data: TableMetaData(name=pat_inhos_order_group, columns=[ColumnMetaData(name=seq_id, dataType=-5, primaryKey=true, generated=false, caseSensitive=false, visible=true, unsigned=false, nullable=false), ColumnMetaData(name=group_unique_code, dataType=12, primaryKey=false, generated=false, caseSensitive=false, visible=true, unsigne
```

但是创建非分片表是可以的，接着只需要将表名改下，继续执行：

```
CREATE TABLE IF NOT EXISTS `pat_inhos_order_group_t1` (
  `seq_id` bigint(20) NOT NULL AUTO_INCREMENT COMMENT '自增长编号',
  `group_unique_code` varchar(60) NOT NULL COMMENT '批次唯一编号',
```

```
  `order_group_no` varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
  `inhos_code` varchar(20) DEFAULT '' COMMENT '病人住院号',
  `orderbar` varchar(40) DEFAULT '' COMMENT '医嘱条码',
  `package_bar` varchar(50) DEFAULT NULL COMMENT '药箱条码',
  `plan_time` datetime DEFAULT NULL COMMENT '计划执行时间',
  `order_sort_no` int(11) DEFAULT NULL COMMENT '医嘱排序编号',
  `source_type` varchar(20) DEFAULT '' COMMENT '数据来源',
  `isprint` int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
  `execute_status` int(11) NOT NULL DEFAULT '0' COMMENT '执行状态  0 - 未执行, 1- 执行中， 2 - 已执行， 3-停止，4-作废',
  `execute_date` datetime DEFAULT NULL COMMENT '执行时间',
  `print_date` datetime DEFAULT NULL COMMENT '打印时间',
  `execute_person` varchar(20) DEFAULT '' COMMENT '执行人',
  `print_person` varchar(20) DEFAULT '' COMMENT '打印人',
  `apply_time` datetime DEFAULT NULL COMMENT '开立时间',
  `is_dispensed` int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是， 0 - 否',
  `remark` varchar(100) DEFAULT '' COMMENT ' 医嘱备注',
  `reason` varchar(200) DEFAULT NULL COMMENT '手动或作废执行的原因',
  `execute_type` int(11) DEFAULT '0' COMMENT '执行方式 0-无效， 1- 扫描执行， 2 - 手动执行',
  `start_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始执行人',
  `start_execute_date` datetime DEFAULT NULL COMMENT '手动补录开始执行时间',
  `start_check_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始核对人',
  `end_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录结束执行人',
  `end_execute_date` datetime DEFAULT NULL COMMENT '手动补录结束执行时间',
  `create_time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建时间',
  `create_person` varchar(50) NOT NULL DEFAULT '' COMMENT '创建人',
  `update_time` datetime DEFAULT NULL COMMENT '修改时间',
  `update_person` varchar(50) DEFAULT NULL COMMENT '修改人',
  PRIMARY KEY (`seq_id`),
  UNIQUE KEY `uq_pat_inhos_order_group_1` (`group_unique_code`),
  KEY `idx_pat_inhos_order_group_1` (`order_group_no`),
  KEY `idx_pat_inhos_order_group_2` (`orderbar`),
  KEY `idx_pat_inhos_order_group_3` (`execute_date`),
  KEY `idx_pat_inhos_order_group_4` (`plan_time`),
  KEY `idx_idx_pat_inhos_order_group_5` (`inhos_code`),
  KEY `idx_pat_inhos_order_group_6` (`inhos_code`,`plan_time`,`execute_status`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='医嘱批次表';
```

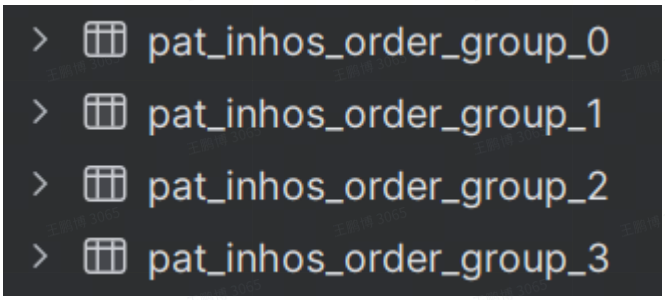
查看表：

```
mysql> show tables;
```

```
+--------------------------+
| Tables_in_windranger_emr |
+--------------------------+
| pat_inhos_order_group_t1 |
+--------------------------+
1 row in set (0.01 sec)
```

只有非分片表创建成功了，但是非分片表并没有成功。

注意：虽然分片表没有创建成功，但是在物理库中已经存在分片表。可以借助客户端工具查看：




已找到问题原因，是因为server.xml配置中开启了如下配置：

```
props:
  check-table-metadata-enabled: true
```

索引和表分开创建就可以执行成功~

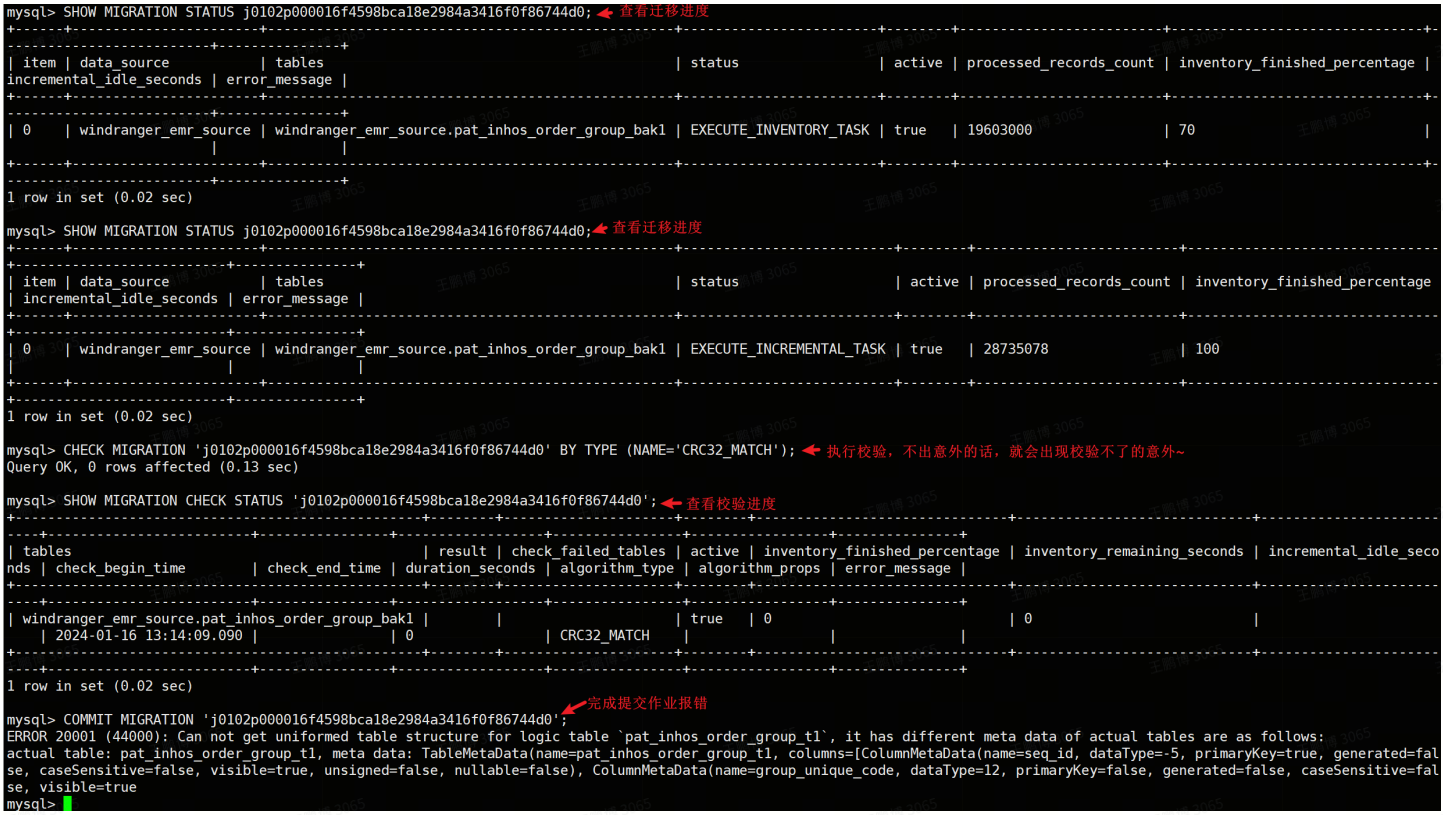
疑惑：至于这个配置具体作用是什么？ 官方的解释 没有看懂（字面意思能理解，但是为什么同时创建索引就报错呢？）

如果关闭该配置后（默认是关闭的），目前也没发现有什么异常情况，只能说暂时规避了此问题而已~

| | | | |
|---|---|---|---|
| check-table-metadata-enabled (?) | boolean | 在程序启动和更新时，是否检查分片元数据的结构一致性 | false |

## 踩坑三：基于DistSQL进行数据迁移后执行提交报错

需求背景：将pat_inhos_order_group_bak1表数据迁移至分片表pat_inhos_order_group_t1_0~pat_inhos_order_group_t1_3中。




同样执行刷新元数据也会报错：

```
mysql> REFRESH TABLE METADATA;
ERROR 20001 (44000): Can not get uniformed table structure for logic table `pat_inhos_order_group_t1`, it has different meta data of actual tables are as follows:
actual table: pat_inhos_order_group_t1, meta data: TableMetaData(name=pat_inhos_order_group_t1, columns=[ColumnMetaData(name=seq_id, dataType=-5, primaryKey=true, generated=false, caseSensitive=false, visible=true, unsigned=false, nullable=false), ColumnMetaData(name=group_unique_code, dataType=12, primaryKey=false, generated=false, caseSensitive=false, visible=true
```

## 踩坑四：基于配置文件创建的分片规则不支持创建表时同时创建索引

复现步骤：

**conf/server.xml**

展开源码

```yaml
mode:
  type: Standalone
  repository:
    type: JDBC

authority:
  users:
    - user: root@%
      password: root
    - user: sharding
      password: sharding
  privilege:
    type: ALL_PERMITTED

props:
  system-log-level: DEBUG
  sql-show: true
  check-table-metadata-enabled: true
```

### conf/config-sharding.xml

展开源码

```yaml
databaseName: windranger_emr

dataSources:
 ds_0:
   url: jdbc:mysql://10.2.3.167:3306/windranger_emr?characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true
   username: user
   password: Lachesis-mh_1024
   connectionTimeoutMilliseconds: 30000
   idleTimeoutMilliseconds: 60000
   maxLifetimeMilliseconds: 1800000
   maxPoolSize: 50
   minPoolSize: 1

rules:
```

```yaml
- !SINGLE
  tables:
    - "*.*"
- !SHARDING
  tables:
    pat_inhos_order_group_t2:
      actualDataNodes: ds_0.pat_inhos_order_group_t2_${0..3}
      tableStrategy:
        standard:
          shardingColumn: inhos_code
          shardingAlgorithmName: t_order_inline

  shardingAlgorithms:
    t_order_inline:
      type: INLINE
      props:
        algorithm-expression:
pat_inhos_order_group_t2_${Math.abs(inhos_code.hashCode())%4}
```

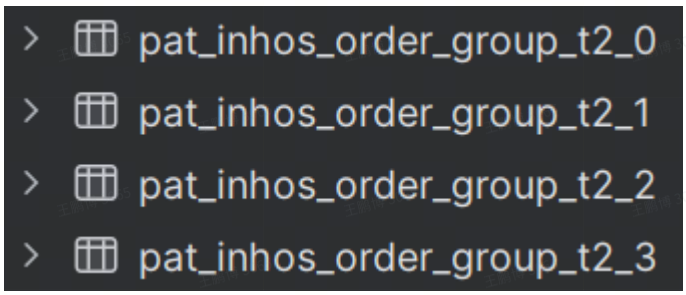
启动后，复现异常操作：

```
mysql> show databases;
+--------------------+
| schema_name        |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| shardingsphere     |
| sys                |
| windranger_emr     |
+--------------------+
6 rows in set (0.03 sec)

mysql> use windranger_emr;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select count(*) from pat_inhos_order_group_t2;
ERROR 10007 (42S02): Table or view `pat_inhos_order_group_t2` does not exist.
mysql> CREATE TABLE IF NOT EXISTS `pat_inhos_order_group_t2` (
    ->   `seq_id` bigint(20) NOT NULL AUTO_INCREMENT COMMENT '自增长编号',
    ->   `group_unique_code` varchar(60) NOT NULL COMMENT '批次唯一编号',
```

```
    ->   `order_group_no` varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
    ->   `inhos_code` varchar(20) DEFAULT '' COMMENT '病人住院号',
    ->   `orderbar` varchar(40) DEFAULT '' COMMENT '医嘱条码',
    ->   `package_bar` varchar(50) DEFAULT NULL COMMENT '药箱条码',
    ->   `plan_time` datetime DEFAULT NULL COMMENT '计划执行时间',
    ->   `order_sort_no` int(11) DEFAULT NULL COMMENT '医嘱排序编号',
    ->   `source_type` varchar(20) DEFAULT '' COMMENT '数据来源',
    ->   `isprint` int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
    ->   `execute_status` int(11) NOT NULL DEFAULT '0' COMMENT '执行状态  0 - 未执行， 1- 执行中， 2 - 已执行， 3-停止，4-作废',
    ->   `execute_date` datetime DEFAULT NULL COMMENT '执行时间',
    ->   `print_date` datetime DEFAULT NULL COMMENT '打印时间',
    ->   `execute_person` varchar(20) DEFAULT '' COMMENT '执行人',
    ->   `print_person` varchar(20) DEFAULT '' COMMENT '打印人',
    ->   `apply_time` datetime DEFAULT NULL COMMENT '开立时间',
    ->   `is_dispensed` int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是，0 - 否',
    ->   `remark` varchar(100) DEFAULT '' COMMENT ' 医嘱备注',
    ->   `reason` varchar(200) DEFAULT NULL COMMENT '手动或作废执行的原因',
    ->   `execute_type` int(11) DEFAULT '0' COMMENT '执行方式 0-无效， 1- 扫描执行， 2 - 手动执行',
    ->   `start_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始执行人',
    ->   `start_execute_date` datetime DEFAULT NULL COMMENT '手动补录开始执行时间',
    ->   `start_check_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始核对人',
    ->   `end_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录结束执行人',
    ->   `end_execute_date` datetime DEFAULT NULL COMMENT '手动补录结束执行时间',
    ->   `create_time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建时间',
    ->   `create_person` varchar(50) NOT NULL DEFAULT '' COMMENT '创建人',
    ->   `update_time` datetime DEFAULT NULL COMMENT '修改时间',
    ->   `update_person` varchar(50) DEFAULT NULL COMMENT '修改人',
    ->   PRIMARY KEY (`seq_id`),
    ->   UNIQUE KEY `uq_pat_inhos_order_group_1` (`group_unique_code`),
    ->   KEY `idx_pat_inhos_order_group_1` (`order_group_no`),
    ->   KEY `idx_pat_inhos_order_group_2` (`orderbar`),
    ->   KEY `idx_pat_inhos_order_group_3` (`execute_date`),
    ->   KEY `idx_pat_inhos_order_group_4` (`plan_time`),
    ->   KEY `idx_idx_pat_inhos_order_group_5` (`inhos_code`),
    ->   KEY `idx_pat_inhos_order_group_6` (`inhos_code`,`plan_time`,`execute_status`)
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='医嘱批次表';
```

```
ERROR 20001 (44000): Can not get uniformed table structure for logic table `pat_inhos_order_group_t2`, it has different meta data of actual tables are as follows:
actual table: pat_inhos_order_group_t2, meta data: TableMetaData(name=pat_inhos_order_group_t2, columns=[ColumnMetaData(name=seq_id, dataType=-5, primaryKey=true, generated=false, caseSensitive=false, visible=true, unsigned=false, nullable=false), ColumnMetaData(name=group_unique_code, dataType=12, primaryKey=false, generated=false, caseSensitive=false, visible=true
mysql>
```

但是分片表是创建成功了：




但是当我们在创建表的时候不创建索引，就会成功：

**注意：重置环境！！！**

**注意：重置环境！！！**

**注意：重置环境！！！**

```
mysql> show databases;
+--------------------+
| schema_name        |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| shardingsphere     |
| sys                |
| windranger_emr     |
+--------------------+
6 rows in set (0.01 sec)

mysql> use windranger_emr;
```

```
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> CREATE TABLE IF NOT EXISTS pat_inhos_order_group_t2 (
    ->   seq_id bigint(20) NOT NULL AUTO_INCREMENT COMMENT '自增长编号',
    ->   group_unique_code varchar(60) NOT NULL COMMENT '批次唯一编号',
    ->   order_group_no varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
    ->   inhos_code varchar(20) DEFAULT '' COMMENT '病人住院号',
    ->   orderbar varchar(40) DEFAULT '' COMMENT '医嘱条码',
    ->   package_bar varchar(50) DEFAULT NULL COMMENT '药箱条码',
    ->   plan_time datetime DEFAULT NULL COMMENT '计划执行时间',
    ->   order_sort_no int(11) DEFAULT NULL COMMENT '医嘱排序编号',
    ->   source_type varchar(20) DEFAULT '' COMMENT '数据来源',
    ->   isprint int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
    ->   execute_status int(11) NOT NULL DEFAULT '0' COMMENT '执行状态  0 - 未执行， 1- 执行中， 2 - 已执行， 3-停止，4-作废',
    ->   execute_date datetime DEFAULT NULL COMMENT '执行时间',
    ->   print_date datetime DEFAULT NULL COMMENT '打印时间',
    ->   execute_person varchar(20) DEFAULT '' COMMENT '执行人',
    ->   print_person varchar(20) DEFAULT '' COMMENT '打印人',
    ->   apply_time datetime DEFAULT NULL COMMENT '开立时间',
    ->   is_dispensed int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是， 0 - 否',
    ->   remark varchar(100) DEFAULT '' COMMENT ' 医嘱备注',
    ->   reason varchar(100) DEFAULT NULL COMMENT '手动或作废执行的原因',
    ->   execute_type int(11) DEFAULT '0' COMMENT '执行方式 0-无效， 1- 扫描执行， 2 - 手动执行',
    ->   start_execute_user varchar(20) DEFAULT NULL COMMENT '手动补录开始执行人',
    ->   start_execute_date datetime DEFAULT NULL COMMENT '手动补录开始执行时间',
    ->   start_check_user varchar(20) DEFAULT NULL COMMENT '手动补录开始核对人',
    ->   end_execute_user varchar(20) DEFAULT NULL COMMENT '手动补录结束执行人',
    ->   end_execute_date datetime DEFAULT NULL COMMENT '手动补录结束执行时间',
    ->   create_time datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建时间',
    ->   create_person varchar(50) NOT NULL DEFAULT '' COMMENT '创建人',
    ->   update_time datetime DEFAULT NULL COMMENT '修改时间',
    ->   update_person varchar(50) DEFAULT NULL COMMENT '修改人',
    ->   PRIMARY KEY (seq_id)
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='医嘱批次表';
Query OK, 0 rows affected (0.22 sec)

```

```
mysql> select count(*) from pat_inhos_order_group_t2;
+----------+
| count(*) |
+----------+
|        0 |
+----------+
1 row in set (0.07 sec)

mysql>
```

那如何在创建表的时候，还使创建索引不报错呢？与踩坑二类似，只需关闭server.xml中如下配置：

```
props:
  check-table-metadata-enabled: false
```

疑惑：至于这个配置具体作用是什么？ 官方的解释 没有看懂（字面意思能理解，但是为什么同时创建索引就报错呢？）

如果关闭该配置后（默认是关闭的），目前也没发现有什么异常情况，只能说暂时规避了此问题而已~

| | | | |
|---|---|---|---|
| check-table-metadata-enabled (?) | boolean | 在程序启动和更新时，是否检查分片元数据的结构一致性 | false |

## 踩坑五：基于配置文件创建的分片规则与基于DistSQL创建的分片规则冲突问题

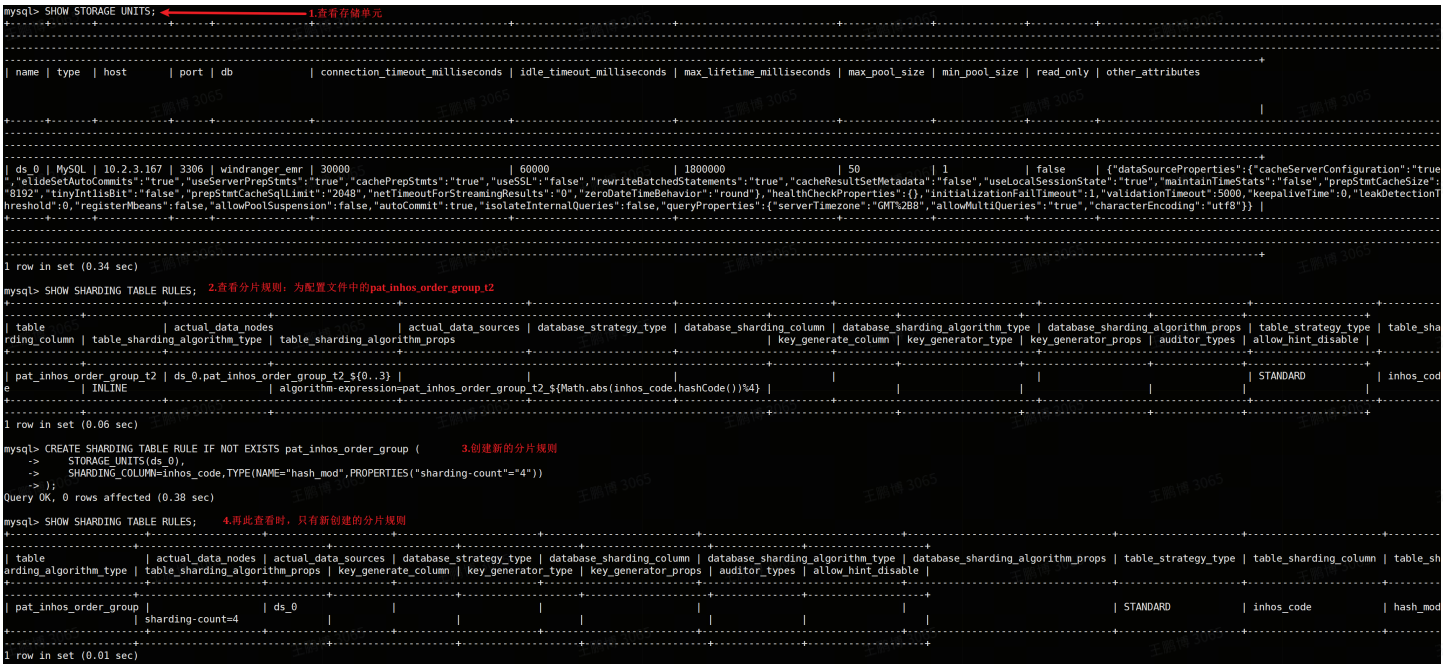
复现步骤：

**conf/config-sharding.xml**

> 展开源码

```
databaseName: windranger_emr

```

```yaml
dataSources:
 ds_0:
   url: jdbc:mysql://10.2.3.167:3306/windranger_emr?characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true
   username: user
   password: Lachesis-mh_1024
   connectionTimeoutMilliseconds: 30000
   idleTimeoutMilliseconds: 60000
   maxLifetimeMilliseconds: 1800000
   maxPoolSize: 50
   minPoolSize: 1

rules:
- !SINGLE
  tables:
    - "*.*"
- !SHARDING
  tables:
    pat_inhos_order_group_t2:
      actualDataNodes: ds_0.pat_inhos_order_group_t2_${0..3}
      tableStrategy:
        standard:
          shardingColumn: inhos_code
          shardingAlgorithmName: t_order_inline

  shardingAlgorithms:
    t_order_inline:
      type: INLINE
      props:
        algorithm-expression: pat_inhos_order_group_t2_${Math.abs(inhos_code.hashCode())%4}
```




# 踩坑六：Proxy并发批量插入报错

Proxy配置：

server.yaml
3.06KB

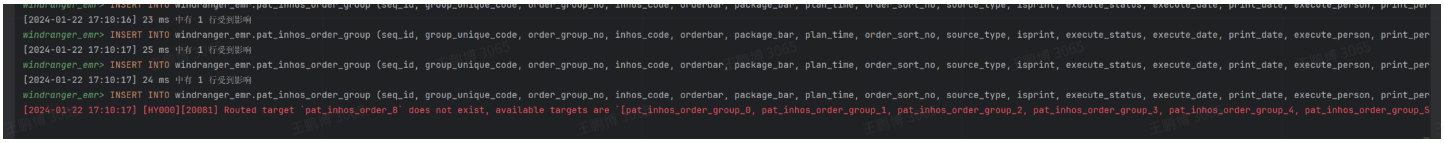
config-sharding.yaml
3.71KB

复现：准备两个批量执行SQL。

医嘱1000条INSERT.sql
1.44MB

医嘱批次1000条INSERT.sql
736.08KB

两个批量执行的SQL，同时执行时，有几率产生如下报错：



特别说明：手动批量执行，不一定能复现此问题。可以借助程序去进行测试，个人使用基于RestCloud工具，将emr的单表数据同步到Proxy代理的数据源的分库分表，类似问题必现。

**猜测Proxy端存在并发BUG！**

类似的报错主要有以下几种情况：

- 并发同步单表到单库分表（暂时未发现报错）
  - pat_inhos_order → ds0.pat_inhos_order_{0..63} ---------- **箭头含义：就是简单的将旧版数据（数据源见下文SOURCE）批量插入新表（数据源见下文TARGET，只不过新表是Proxy代理的单库分表或者分库分表，下文对于箭头不在赘述！）**
  - pat_inhos_order_group_new → ds0.pat_inhos_order_group _{0..63}
  - 以上两个任务 串行运行时
- 并发同步两个单表到各自的单库分表中（必现，且报错，导致批量插入终止）
  - pat_inhos_order → ds0.pat_inhos_order_{0..63}
  - pat_inhos_order_group_new → ds0.pat_inhos_order_group _{0..63}
  - 以上两个任务 并发运行时
- 并发同步单表到分库分表（必现，且报错，导致批量插入终止）
  - pat_inhos_order → ds_{0..3}.pat_inhos_order_{0..16}
  - 启动不同的任务，同步不同范围pat_inhos_order表数据

思考：所以基于Proxy代理数据源做分库分表，对于同步程序（不论新旧同步程序）来说，如果存在并发的情况，就会有概率出现数据同步不了，影响较大~

上述问题，是在数据迁移过程中遇到的，目前还无暇写代码去复现此问题，均是基于现有成熟工具复现。

**数据源信息：**

**SOURCE**

```
1  url: jdbc:mysql://10.2.3.173:3306/windranger_emr?
   characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true
2  username: user
3  password: Lachesis-mh_1024
```

**TARGET**

```
url: jdbc:mysql://10.2.3.173:3307/windranger_emr?characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true
username: root
password: root
```

## 踩坑七：Proxy无法执行改写后的数据库脚本

对于SQL脚本，如果存在对于infomation_schema库的操作就会存在不支持的情况：

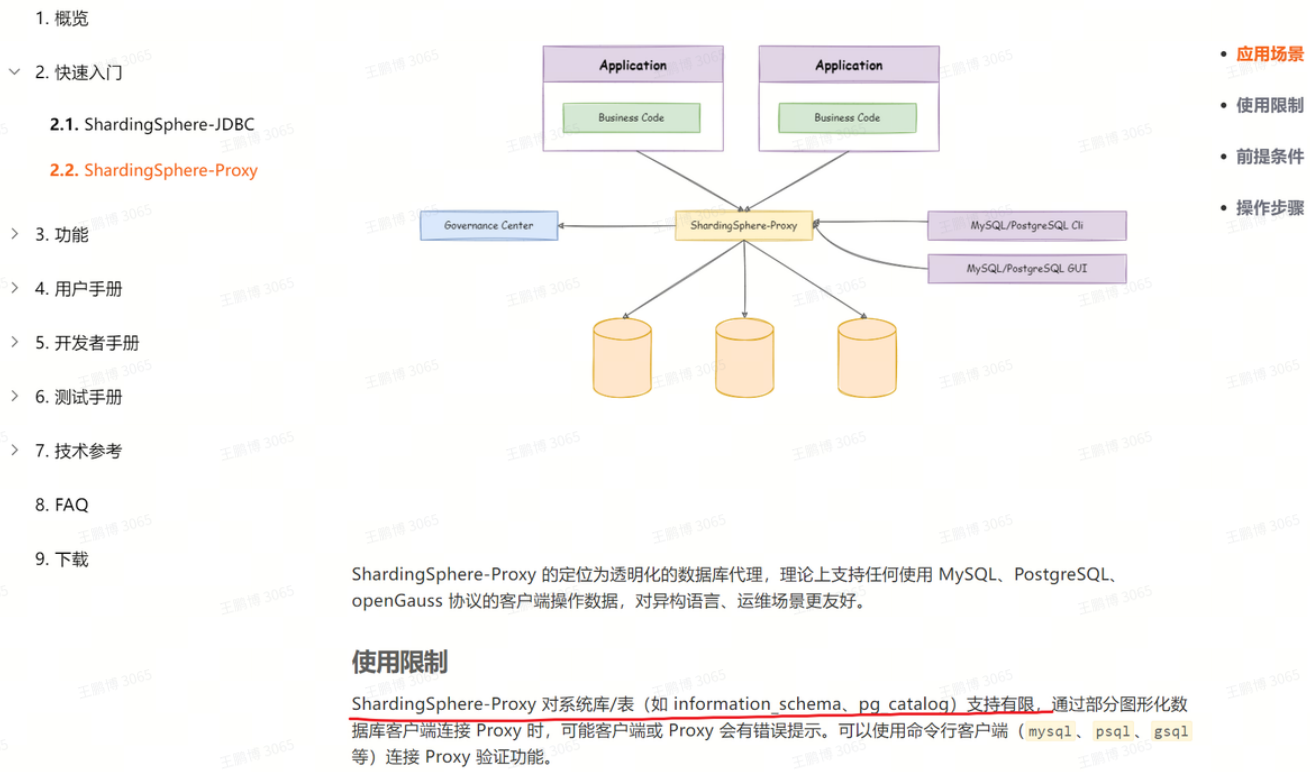
```
/*
  20220628 谭华建 需求详情: 4259 手术管理-优化 #10
*/

use windranger_emr;

drop procedure if exists p_schema_change;
DELIMITER ;;
create procedure p_schema_change(
                              in  i_in_change_type  int,            /*1 新增加字段; 2 删除字段; 3 创建索引; 4 删除索引; 5 创建唯一索引; 6 修改表名*/
                              in  s_in_table_schema varchar(50),
                              in  s_in_table_name   varchar(50),
                              in  s_in_column_name  varchar(50),
                              in  s_in_column_other varchar(200),    /*当添加字段时为字段的属性; 当创建或删除索引时为索引名称*/
                              out s_o_message       varchar(100)
                              )
BEGIN
/****************************************************************************************************
**    Name:       p_schema_change
**    Purpose:    实现(1 新增加字段; 2 删除字段; 3 创建索引; 4 删除索引; 5 创建唯一索引; 6 修改表名) 可重复执行
**
**
**    Revisions:
**    Ver         Date        Author               Description
**    ----------  ----------  -------------------  ------------------------------------
**    V1.0        2020-12-29  王进宝               1.Create the Procedure
**    Notes:      <1>实现(1 新增加字段; 2 删除字段; 3 创建索引; 4 删除索引; 5 创建唯一索引; 6 修改表名) 可重复执行
**
****************************************************************************************************/
    declare i_l_row int unsigned default 0;
    /*添加字段*/
    if i_in_change_type = 1 then
       select count(1) into i_l_row  from information_schema.COLUMNS where TABLE_SCHEMA=s_in_table_schema and TABLE_NAME=s_in_table_name and COLUMN_NAME=s_in_column_n
       if i_l_row = 0 then
          set @add_column_sql = concat("ALTER TABLE ",s_in_table_schema,".",s_in_table_name," ADD COLUMN ",s_in_column_name," ",s_in_column_other);

          prepare c_add_column_sql from @add_column_sql;
          execute c_add_column_sql;
          set s_o_message='添加字段成功。';
       end if;
    end if;

    /*删除字段*/
    if i_in_change_type = 2 then
       select count(1) into i_l_row  from information_schema.COLUMNS where TABLE_SCHEMA=s_in_table_schema and TABLE_NAME=s_in_table_name and COLUMN_NAME=s_in_column_n
       if i_l_row = 1 then
          set @del_column_sql = concat("ALTER TABLE ",s_in_table_schema,".",s_in_table_name," DROP COLUMN ",s_in_column_name,";");
```

原始文件路径：
http://10.2.3.111/backend_group/backend_application/emr/blob/master/windranger-emr-web/src/main/resources/database/migrations/V1_0_1_31__Create.sql

执行报错信息： [2024-01-23 11:16:50] [42S02][1146] Table 'COLUMNS' doesn't exist

原因见官网：

https://shardingsphere.apache.org/document/current/cn/quick-start/shardingsphere-proxy-quick-start/




# 总结

1. 功能限制
   - 不支持的SQL
     - 业务代码存在CASE WHEN需要整改
     - 互联互通SQL存在CASE WHEN的无法支持
   - 无法与Flyway进行脚本版本管理
   - 同步程序存在问题
     - 分片键不是判断数据唯一性时，如果存在分片键更新的情况，将会更新失败（同样的问题可能在业务系统也存在）
8. 稳定性
   - ~~DDL执行可能遇到一些奇怪的问题，不是很稳定~~
     - **不要开启元数据校验，开启后一些DDL操作会存在报错（见上踩坑二和踩坑四）**

- 目前主要集中在DistSQL支持的一些功能存在不稳定因素）

12. 健壮性
    - 基于DistSQL的数据迁移可以支持，但是校验存在BUG~
14. 性能问题
    - 不采用集群时，相当于所有需要分片的表都是通过Proxy进行计算分片，Proxy本身也是一个Java程序，会将各个端的请求压力集中到Proxy服务
    - 采用集群，服务部署资源成本提升~

# Original Confluence page attachments

| Name | Size | Created by | Created on | Labels | Comments |
|---|---|---|---|---|---|
| image-2024-1-23_11-22-… 272.80KB | 272.80 KB | 王鹏博 | 2024-01-23T11:22:53.000+08:00 | | |
| image-2024-1-23_11-20-… 292.46KB | 292.46 KB | 王鹏博 | 2024-01-23T11:20:50.000+08:00 | | |
| image-2024-1-23_11-19-… 97.90KB | 97.90 KB | 王鹏博 | 2024-01-23T11:19:45.000+08:00 | | |
| image-2024-1-22_18-32-… 169.78KB | 169.78 KB | 王鹏博 | 2024-01-22T18:30:53.000+08:00 | | |
| | 736.08 KB | 王鹏博 | 2024-01-22T17:17: | | |

| | | | | | |
|---|---|---|---|---|---|
| 医嘱批次1000条INSERT.sql 736.08KB | | | 56.000+08:00 | | |
| 医嘱1000条INSERT.sql 1.44MB | 1.44 MB | 王鹏博 | 2024-01-22T17:17:47.000+08:00 | | |
| server.yaml 3.06KB | 3.06 KB | 王鹏博 | 2024-01-22T17:17:34.000+08:00 | | |
| config-sharding.yaml 3.71KB | 3.71 KB | 王鹏博 | 2024-01-22T17:17:12.000+08:00 | | |
| image-2024-1-16_18-23-… 27.50KB | 27.50 KB | 王鹏博 | 2024-01-16T18:22:35.000+08:00 | | |
| image-2024-1-16_17-48-… 17.63KB | 17.62 KB | 王鹏博 | 2024-01-16T17:47:46.000+08:00 | | |
| image-2024-1-16_17-35-… 348.45KB | 348.45 KB | 王鹏博 | 2024-01-16T17:34:22.000+08:00 | | |
| image-2024-1-16_13-18-… 278.94KB | 278.94 KB | 王鹏博 | 2024-01-16T13:17:51.000+08:00 | | |
| image-2024-1-16_9-55-… 16.45KB | 16.45 KB | 王鹏博 | 2024-01-16T09:54:36.000+08:00 | | |
| 输血记录单.sql 6.35KB | 6.35 KB | 王鹏博 | 2024-01-08T16:06:42.000+08:00 | | |

| | | | | | |
|---|---|---|---|---|---|
| 生命体征.sql<br>9.08KB | 9.08 KB | 王鹏博 | 2024-01-08T16:06:33.000+08:00 | | |
| 入院评估单.sql<br>29.08KB | 29.08 KB | 王鹏博 | 2024-01-08T16:06:22.000+08:00 | | |
| 患者个人信息.sql<br>2.65KB | 2.65 KB | 王鹏博 | 2024-01-08T16:06:15.000+08:00 | | |
| 护理记录单.sql<br>5.67KB | 5.67 KB | 王鹏博 | 2024-01-08T16:05:44.000+08:00 | | |
| image-2024-1-8_15-49-…<br>25.90KB | 25.90 KB | 王鹏博 | 2024-01-08T15:49:17.000+08:00 | | |
| image-2024-1-8_9-29-53.png<br>57.78KB | 57.78 KB | 王鹏博 | 2024-01-08T09:29:53.000+08:00 | | |
| image-2024-1-4_15-57-2.png<br>25.95KB | 25.95 KB | 王鹏博 | 2024-01-05T14:33:57.000+08:00 | | |