

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？如果没有现有的方案，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：

```
1  databaseName: windranger_emr
2  # 重点就是SINGLE部分的配置
3  rules:
4    - !SINGLE
5      tables:
6        - "*"
7    - !SHARDING
8      tables:
9        pat_inhos_order_group:
10       # 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

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

```
9      DB="windranger_emr",
10      USER="user",
11      PASSWORD="Lachesis-mh_1024"
12 );
13 # 加载全部单表
14 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：

```
1  spring:
2    datasource:
3      url: jdbc:mysql://10.2.3.167:3307/windranger_emr
4      username: root
5      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

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

```
3 Message : Unknown exception: org.apache.calcite.rex.RexSubQuery cannot be
  cast to org.apache.calcite.rex.RexLocalRef
4
5     at org.flywaydb.core.internal.database.base.Schema.exists(Schema.java:75)
6     at org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:73)
7     at org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:69)
8     at
  org.flywaydb.core.internal.jdbc.TransactionTemplate.execute(TransactionTemplat
  e.java:74)
9     at org.flywaydb.core.internal.command.DbSchemas.create(DbSchemas.java:69)
10    at org.flywaydb.core.Flyway$1.execute(Flyway.java:1369)
11    at org.flywaydb.core.Flyway$1.execute(Flyway.java:1356)
12    at org.flywaydb.core.Flyway.execute(Flyway.java:1711)
13    at org.flywaydb.core.Flyway.migrate(Flyway.java:1356)
14    at
  com.lachesis.molecule.common.database.DbVersionMigrationMonitor.run(DbVersionM
  igrationMonitor.java:45)
15    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
16    at
  sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
17    at
  sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.j
  ava:43)
18    at java.lang.reflect.Method.invoke(Method.java:498)
19    at
  org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
  essor$LifecycleElement.invoke(InitDestroyAnnotationBeanPostProcessor.java:363)
20    at
  org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
  essor$LifecycleMetadata.invokeInitMethods(InitDestroyAnnotationBeanPostProcess
  or.java:307)
21    at
  org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
  essor.postProcessBeforeInitialization(InitDestroyAnnotationBeanPostProcessor.j
  ava:136)
22    ... 16 common frames omitted
23 Caused by: java.sql.SQLException: Unknown exception:
  org.apache.calcite.rex.RexSubQuery cannot be cast to
  org.apache.calcite.rex.RexLocalRef
24    at
  com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:129)
25    at
  com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
26    at
  com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExcept
  ionsMapping.java:122)
```

```

27      at
com.mysql.cj.jdbc.ClientPreparedStatement.executeInternal(ClientPreparedStatement
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。

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、gspl等）连接 Proxy 验证功能。

- 应用场景
- 使用限制
- 前提条件
- 操作步骤

```

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

```
1  @Override
2  protected boolean doExists() throws SQLException {
3      try {
4          return jdbcTemplate.queryForInt("SELECT (SELECT 1 FROM
information_schema.schemata WHERE schema_name=? LIMIT 1)", name) > 0;
5      } catch (Exception e) {
6          String querySql = String.format("SELECT schema_name FROM
information_schema.schemata WHERE schema_name='%s'", name);
7          List<String> schemaNames = jdbcTemplate.query(querySql, new
RowMapper<String>() {
8              @Override
9              public String mapRow(ResultSet rs) throws SQLException {
10                 return rs.getString("schema_name");
11             }
12         });
13         return schemaNames.contains(name);
14     }
15 }
```

发现修改后, 又会有新的错误:

项目启动错误日志

```
1  SQL State : HY000
2  Error Code : 30000
3  Message : Unknown exception: Index: 1, Size: 1
4
5      at
org.flywaydb.core.internal.database.base.Schema.exists(Schema.java:75)
6      at
org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:73)
7      at
org.flywaydb.core.internal.command.DbSchemas$1.call(DbSchemas.java:69)
8      at
org.flywaydb.core.internal.jdbc.TransactionTemplate.execute(TransactionTemplat
```

```
e.java:74)
9      at
org.flywaydb.core.internal.command.DbSchemas.create(DbSchemas.java:69)
10      at org.flywaydb.core.Flyway$1.execute(Flyway.java:1369)
11      at org.flywaydb.core.Flyway$1.execute(Flyway.java:1356)
12      at org.flywaydb.core.Flyway.execute(Flyway.java:1711)
13      at org.flywaydb.core.Flyway.migrate(Flyway.java:1356)
14      at
com.lachesis.molecule.common.database.DbVersionMigrationMonitor.run(DbVersionM
igrationMonitor.java:45)
15      at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
16      at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
17      at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.j
ava:43)
18      at java.lang.reflect.Method.invoke(Method.java:498)
19      at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor$LifecycleElement.invoke(InitDestroyAnnotationBeanPostProcessor.java:363)
20      at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor$LifecycleMetadata.invokeInitMethods(InitDestroyAnnotationBeanPostProcess
or.java:307)
21      at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor.postProcessBeforeInitialization(InitDestroyAnnotationBeanPostProcessor.j
ava:136)
22      ... 16 common frames omitted
23      Caused by: java.sql.SQLException: Unknown exception: Index: 1, Size: 1
24      at
com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:129)
25      at
com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
26      at
com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExcept
ionsMapping.java:122)
27      at
com.mysql.cj.jdbc.StatementImpl.executeQuery(StatementImpl.java:1218)
28      at
com.alibaba.druid.pool.DruidPooledStatement.executeQuery(DruidPooledStatement.
java:138)
29      at
org.flywaydb.core.internal.jdbc.JdbcTemplate.query(JdbcTemplate.java:364)
30      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",
```



```

7      PORT=3306,
8      DB="windranger_emr",
9      USER="user",
10     PASSWORD="Lachesis-mh_1024"
11 );
12
13 LOAD SINGLE TABLE *.*;
14
15 CREATE SHARDING TABLE RULE IF NOT EXISTS t_order_item (
16     STORAGE_UNITS(ds_0),
17     SHARDING_COLUMN=order_id,
18     TYPE(NAME=MOD,PROPERTIES("sharding-count"=4)),
19     KEY_GENERATE_STRATEGY(COLUMN=order_id,TYPE(NAME=SNOWFLAKE))
20 );
21
22 CREATE TABLE IF NOT EXISTS `t_order_item`(
23     `id`                bigint unsigned NOT NULL AUTO_INCREMENT,
24     `order_id`          bigint unsigned NOT NULL DEFAULT '0',
25     `order_item_id`     bigint unsigned NOT NULL DEFAULT '0',
26     `order_item_name`   varchar(100)    NOT NULL DEFAULT '',
27     `create_time`       bigint          NOT NULL DEFAULT '0',
28     `update_time`       bigint          NOT NULL DEFAULT '0',
29     PRIMARY KEY (`id`)
30 ) 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

```

1  SQL State : null
2  Error Code : 0
3  Message : wait millis 10014, active 0, maxActive 8
4
5      at
6      org.flywaydb.core.internal.jdbc.JdbcUtils.openConnection(JdbcUtils.java:60)
7      at
8      org.flywaydb.core.internal.database.DatabaseFactory.createDatabase(DatabaseFactory.java:72)
9      at org.flywaydb.core.Flyway.execute(Flyway.java:1670)
10     at org.flywaydb.core.Flyway.migrate(Flyway.java:1356)

```

```
9         at
com.lachesis.molecule.common.database.DbVersionMigrationMonitor.run(DbVersionM
igrationMonitor.java:45)
10         at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
11         at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
12         at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.j
ava:43)
13         at java.lang.reflect.Method.invoke(Method.java:498)
14         at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor$LifecycleElement.invoke(InitDestroyAnnotationBeanPostProcessor.java:363)
15         at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor$LifecycleMetadata.invokeInitMethods(InitDestroyAnnotationBeanPostProcess
or.java:307)
16         at
org.springframework.beans.factory.annotation.InitDestroyAnnotationBeanPostProc
essor.postProcessBeforeInitialization(InitDestroyAnnotationBeanPostProcessor.j
ava:136)
17         ... 16 common frames omitted
18 Caused by: com.alibaba.druid.pool.GetConnectionTimeoutException: wait millis
10014, active 0, maxActive 8
19         at
com.alibaba.druid.pool.DruidDataSource.getConnectionInternal(DruidDataSource.j
ava:1190)
20         at
com.alibaba.druid.pool.DruidDataSource.getConnectionDirect(DruidDataSource.jav
a:1014)
21         at
com.alibaba.druid.pool.DruidDataSource.getConnection(DruidDataSource.java:994)
22         at
com.alibaba.druid.pool.DruidDataSource.getConnection(DruidDataSource.java:984)
23         at
com.alibaba.druid.pool.DruidDataSource.getConnection(DruidDataSource.java:103)
24         at
org.flywaydb.core.internal.jdbc.JdbcUtils.openConnection(JdbcUtils.java:56)
25         ... 27 common frames omitted
26 Caused by: java.sql.SQLException: Unknown database 'windranger_emr'
27         at
com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:120)
28         at
com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
29         at
com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExcept
ionsMapping.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

```

```

9      statement.executeUpdate("CREATE DATABASE IF NOT EXISTS
windranger_emr");
10     statement.executeUpdate("USE windranger_emr");
11     // REGISTER STORAGE UNIT
12     statement.executeUpdate("REGISTER STORAGE UNIT IF NOT EXISTS ds_0
" +
13         "(HOST='10.2.3.167', PORT=3306, DB='windranger_emr',
USER='user', PASSWORD='Lachesis-mh_1024')");
14     // CREATE SHARDING TABLE RULE
15     statement.executeUpdate("CREATE SHARDING TABLE RULE IF NOT EXISTS
t_order_item " +
16         "(STORAGE_UNITS(ds_0), SHARDING_COLUMN=order_id,
TYPE(NAME=MOD, PROPERTIES('sharding-count'=4)), " +
17         "KEY_GENERATE_STRATEGY(COLUMN=order_id,
TYPE(NAME=SNOWFLAKE)))");
18     // CREATE TABLE
19     statement.executeUpdate("CREATE TABLE IF NOT EXISTS
`t_order_item` (" +
20         "`id` bigint unsigned NOT NULL AUTO_INCREMENT," +
21         "`order_id` bigint unsigned NOT NULL DEFAULT '0'," +
22         "`order_item_id` bigint unsigned NOT NULL DEFAULT '0'," +
23         "`order_item_name` varchar(100) NOT NULL DEFAULT ''," +
24         "`create_time` bigint NOT NULL DEFAULT '0'," +
25         "`update_time` bigint NOT NULL DEFAULT '0'," +
26         "PRIMARY KEY (`id`)) ENGINE = InnoDB");
27     } catch (SQLException e) {
28         e.printStackTrace();
29     }
30 }
31 }

```

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：

 护理记录单.sql
5.67KB

 患者个人信息.sql
2.65KB

 入院评估单.sql
29.08KB

 生命体征.sql
9.08KB

 输血记录单.sql
6.35KB

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

```
1  Caused by: java.sql.SQLException: Can not update sharding value for table
   `pat_inhos_order_group`.
2      at
   com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:129)
3      at
   com.mysql.cj.jdbc.exceptions.SQLError.createSQLException(SQLError.java:97)
4      at
   com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExcept
   ionsMapping.java:122)
5      at
   com.mysql.cj.jdbc.ClientPreparedStatement.executeInternal(ClientPreparedStatement
   ent.java:974)
6      at
   com.mysql.cj.jdbc.ClientPreparedStatement.execute(ClientPreparedStatement.java
   :391)
7      at
   com.zaxxer.hikari.pool.ProxyPreparedStatement.execute(ProxyPreparedStatement.j
   ava:44)
8      at
   com.zaxxer.hikari.pool.HikariProxyPreparedStatement.execute(HikariProxyPrepare
   dStatement.java)
9      at
   org.apache.ibatis.executor.statement.PreparedStatementHandler.update(PreparedS
   tatementHandler.java:46)
10     at
   org.apache.ibatis.executor.statement.RoutingStatementHandler.update(RoutingSta
   tementHandler.java:74)
11     at
   org.apache.ibatis.executor.SimpleExecutor.doUpdate(SimpleExecutor.java:50)
12     at
   org.apache.ibatis.executor.BaseExecutor.update(BaseExecutor.java:117)
13     at
   org.apache.ibatis.executor.CachingExecutor.update(CachingExecutor.java:76)
14     at sun.reflect.GeneratedMethodAccessor293.invoke(Unknown Source)
15     at
   sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.j
   ava:43)
16     at java.lang.reflect.Method.invoke(Method.java:498)
17     at org.apache.ibatis.plugin.Plugin.invoke(Plugin.java:63)
18     at com.sun.proxy.$Proxy437.update(Unknown Source)
19     at
   org.apache.ibatis.session.defaults.DefaultSqlSession.update(DefaultSqlSession.
   java:198)
20     at sun.reflect.GeneratedMethodAccessor292.invoke(Unknown Source)
21     at
   sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.j
   ava: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-
j0202p00000j0102p00000013efc3a9791b6876cc092ed25155f841-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
'j0102p00000013efc3a9791b6876cc092ed25155f84';
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 |
```

```

8 | sharding_db |
9 | shardingsphere |
10 | sys |
11 | test |
12 | windranger_emr |
13 +-----+
14 mysql> use sharding_db;
15 Database changed
16 mysql> show tables;
17 Empty set (0.00 sec)
18
19 mysql> SHOW SHARDING TABLE RULES;
20 +-----+-----+-----+-----+-----+
21 | table | actual_data_nodes | actual_data_sources |
22 | database_strategy_type | database_sharding_column |
23 | database_sharding_algorithm_type | database_sharding_algorithm_props |
24 | table_strategy_type | table_sharding_column | table_sharding_algorithm_type |
25 | table_sharding_algorithm_props | key_generate_column | key_generator_type |
26 | key_generator_props | auditor_types | allow_hint_disable |
27 +-----+-----+-----+-----+-----+
28 | t_order_item | | ds_0 |
29 | | | STANDARD | order_id | mod
30 | | | sharding-count=4 | order_id |
31 | snowflake | | | |
32 +-----+-----+-----+-----+-----+
33 1 row in set (0.00 sec)
34
35 mysql> DROP SHARDING TABLE RULE t_order_item;
36 Query OK, 0 rows affected (0.03 sec)
37

```

```

30 mysql> SHOW SHARDING TABLE RULES;
31 +-----+-----+-----+-----+
32 | table | actual_data_nodes | actual_data_sources |
33 | database_strategy_type | database_sharding_column |
34 | database_sharding_algorithm_type | database_sharding_algorithm_props |
35 | table_strategy_type | table_sharding_column | table_sharding_algorithm_type |
36 | table_sharding_algorithm_props | key_generate_column | key_generator_type |
37 | key_generator_props | auditor_types | allow_hint_disable |
38 +-----+-----+-----+-----+
39 | t_order_item | | ds_0 |
40 | | | STANDARD | order_id | mod
41 | | | sharding-count=4 | order_id
42 | snowflake | | |
43 +-----+-----+-----+-----+
44 | 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：

```
1  SHOW DATABASES;
2
3  CREATE DATABASE windranger_emr;
4
5  USE windranger_emr;
6
7  -- 注册存储单元
8  REGISTER STORAGE UNIT IF NOT EXISTS ds_0 (
9      URL="jdbc:mysql://10.2.3.167:3306/windranger_emr?
characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true",
10     USER="user",
11     PASSWORD="Lachesis-mh_1024",
12     PROPERTIES("minPoolSize"="1","maxPoolSize"="20","idleTimeout"="60000")
13 );
14
15 -- 自动分片规则
16 CREATE SHARDING TABLE RULE IF NOT EXISTS pat_inhos_order_group (
17     STORAGE_UNITS(ds_0),
18     SHARDING_COLUMN=inhos_code,TYPE(NAME="hash_mod",PROPERTIES("sharding-
count"="4"))
19 );
20
21 CREATE TABLE IF NOT EXISTS `pat_inhos_order_group` (
22     `seq_id` bigint(20) NOT NULL AUTO_INCREMENT COMMENT '自增长编号',
23     `group_unique_code` varchar(60) NOT NULL COMMENT '批次唯一编号',
24     `order_group_no` varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
25     `inhos_code` varchar(20) DEFAULT '' COMMENT '病人住院号',
26     `orderbar` varchar(40) DEFAULT '' COMMENT '医嘱条码',
27     `package_bar` varchar(50) DEFAULT NULL COMMENT '药箱条码',
28     `plan_time` datetime DEFAULT NULL COMMENT '计划执行时间',
29     `order_sort_no` int(11) DEFAULT NULL COMMENT '医嘱排序编号',
30     `source_type` varchar(20) DEFAULT '' COMMENT '数据来源',
31     `isprint` int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
32     `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',
33     `execute_date` datetime DEFAULT NULL COMMENT '执行时间',
34     `print_date` datetime DEFAULT NULL COMMENT '打印时间',
35     `execute_person` varchar(20) DEFAULT '' COMMENT '执行人',
36     `print_person` varchar(20) DEFAULT '' COMMENT '打印人',
37     `apply_time` datetime DEFAULT NULL COMMENT '开立时间',
38     `is_dispensed` int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是, 0 -
否',
39     `remark` varchar(100) DEFAULT '' COMMENT '医嘱备注',
```

```

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

```

错误日志

```

1 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:
2 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：

```

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

```

```

4  `order_group_no` varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
5  `inhos_code` varchar(20) DEFAULT '' COMMENT '病人住院号',
6  `orderbar` varchar(40) DEFAULT '' COMMENT '医嘱条码',
7  `package_bar` varchar(50) DEFAULT NULL COMMENT '药箱条码',
8  `plan_time` datetime DEFAULT NULL COMMENT '计划执行时间',
9  `order_sort_no` int(11) DEFAULT NULL COMMENT '医嘱排序编号',
10 `source_type` varchar(20) DEFAULT '' COMMENT '数据来源',
11 `isprint` int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
12 `execute_status` int(11) NOT NULL DEFAULT '0' COMMENT '执行状态 0 - 未执行,
13 1- 执行中, 2 - 已执行, 3-停止, 4-作废',
14 `execute_date` datetime DEFAULT NULL COMMENT '执行时间',
15 `print_date` datetime DEFAULT NULL COMMENT '打印时间',
16 `execute_person` varchar(20) DEFAULT '' COMMENT '执行人',
17 `print_person` varchar(20) DEFAULT '' COMMENT '打印人',
18 `apply_time` datetime DEFAULT NULL COMMENT '开立时间',
19 `is_dispensed` int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是, 0 -
20 否',
21 `remark` varchar(100) DEFAULT '' COMMENT '医嘱备注',
22 `reason` varchar(200) DEFAULT NULL COMMENT '手动或作废执行的原因',
23 `execute_type` int(11) DEFAULT '0' COMMENT '执行方式 0-无效, 1- 扫描执行, 2
24 - 手动执行',
25 `start_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始执行人',
26 `start_execute_date` datetime DEFAULT NULL COMMENT '手动补录开始执行时间',
27 `start_check_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始核对人',
28 `end_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录结束执行人',
29 `end_execute_date` datetime DEFAULT NULL COMMENT '手动补录结束执行时间',
30 `create_time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建时间',
31 `create_person` varchar(50) NOT NULL DEFAULT '' COMMENT '创建人',
32 `update_time` datetime DEFAULT NULL COMMENT '修改时间',
33 `update_person` varchar(50) DEFAULT NULL COMMENT '修改人',
34 PRIMARY KEY (`seq_id`),
35 UNIQUE KEY `uq_pat_inhos_order_group_1` (`group_unique_code`),
36 KEY `idx_pat_inhos_order_group_1` (`order_group_no`),
37 KEY `idx_pat_inhos_order_group_2` (`orderbar`),
38 KEY `idx_pat_inhos_order_group_3` (`execute_date`),
39 KEY `idx_pat_inhos_order_group_4` (`plan_time`),
40 KEY `idx_idx_pat_inhos_order_group_5` (`inhos_code`),
41 KEY `idx_pat_inhos_order_group_6`
42 (`inhos_code`, `plan_time`, `execute_status`)
43 ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='医嘱批次表';

```

查看表:

```
1 mysql> show tables;
```

```
2 +-----+
3 | Tables_in_windranger_emr |
4 +-----+
5 | pat_inhos_order_group_t1 |
6 +-----+
7 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：

```
> pat_inhos_order_group_0
> pat_inhos_order_group_1
> pat_inhos_order_group_2
> pat_inhos_order_group_3
```

已找到问题原因，是因为server.xml配置中开启了如下配置：

```
1 props:
2   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。

```
mysql> SHOW MIGRATION STATUS 'j0102p000016f4598bca18e2984a3416f0f86744d0';
+-----+-----+-----+-----+-----+-----+-----+-----+
| item | data_source | tables | status | active | processed_records_count | inventory_finished_percentage |
+-----+-----+-----+-----+-----+-----+-----+
| 0 | windranger_emr_source | windranger_emr_source.pat_inhos_order_group_bak1 | EXECUTE_INVENTORY_TASK | true | 19603000 | 70 |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.02 sec)

mysql> SHOW MIGRATION STATUS 'j0102p000016f4598bca18e2984a3416f0f86744d0';
+-----+-----+-----+-----+-----+-----+-----+-----+
| item | data_source | tables | status | active | processed_records_count | inventory_finished_percentage |
+-----+-----+-----+-----+-----+-----+-----+
| 0 | windranger_emr_source | windranger_emr_source.pat_inhos_order_group_bak1 | EXECUTE_INCREMENTAL_TASK | true | 28735078 | 100 |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.02 sec)

mysql> CHECK MIGRATION 'j0102p000016f4598bca18e2984a3416f0f86744d0' BY TYPE (NAME='CRC32_MATCH');
Query OK, 0 rows affected (0.13 sec)

mysql> SHOW MIGRATION CHECK STATUS 'j0102p000016f4598bca18e2984a3416f0f86744d0';
+-----+-----+-----+-----+-----+-----+-----+-----+
| tables | check_begin_time | check_end_time | result | check_failed_tables | active | inventory_finished_percentage | inventory_remaining_seconds | incremental_idle_seconds |
+-----+-----+-----+-----+-----+-----+-----+-----+
| windranger_emr_source.pat_inhos_order_group_bak1 | 2024-01-16 13:14:09.090 | 0 | CRC32_MATCH | true | 0 | 0 | 0 |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.02 sec)

mysql> COMMIT MIGRATION 'j0102p000016f4598bca18e2984a3416f0f86744d0';
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=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>
```

同样执行刷新元数据也会报错：

```
1 mysql> REFRESH TABLE METADATA;
2 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:
3 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

```
1  mode:
2    type: Standalone
3    repository:
4      type: JDBC
5
6  authority:
7    users:
8      - user: root@%
9        password: root
10     - user: sharding
11       password: sharding
12  privilege:
13    type: ALL_PERMITTED
14
15  props:
16    system-log-level: DEBUG
17    sql-show: true
18    check-table-metadata-enabled: true
```

conf/config-sharding.xml

展开源码

```
1  databaseName: windranger_emr
2
3  dataSources:
4    ds_0:
5      url: jdbc:mysql://10.2.3.167:3306/windranger_emr?
6      characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true
7      username: user
8      password: Lachesis-mh_1024
9      connectionTimeoutMilliseconds: 30000
10     idleTimeoutMilliseconds: 60000
11     maxLifetimeMilliseconds: 1800000
12     maxPoolSize: 50
13     minPoolSize: 1
14  rules:
```

```

15 - !SINGLE
16   tables:
17     - "*,*"
18 - !SHARDING
19   tables:
20     pat_inhos_order_group_t2:
21       actualDataNodes: ds_0.pat_inhos_order_group_t2_${0..3}
22       tableStrategy:
23         standard:
24           shardingColumn: inhos_code
25           shardingAlgorithmName: t_order_inline
26
27       shardingAlgorithms:
28         t_order_inline:
29           type: INLINE
30           props:
31             algorithm-expression:
pat_inhos_order_group_t2_${Math.abs(inhos_code.hashCode())%4}

```

启动后，复现异常操作：

```

1  mysql> show databases;
2  +-----+
3  | schema_name |
4  +-----+
5  | information_schema |
6  | mysql |
7  | performance_schema |
8  | shardingsphere |
9  | sys |
10 | windranger_emr |
11 +-----+
12 6 rows in set (0.03 sec)
13
14 mysql> use windranger_emr;
15 Reading table information for completion of table and column names
16 You can turn off this feature to get a quicker startup with -A
17
18 Database changed
19 mysql> select count(*) from pat_inhos_order_group_t2;
20 ERROR 10007 (42S02): Table or view `pat_inhos_order_group_t2` does not exist.
21 mysql> CREATE TABLE IF NOT EXISTS `pat_inhos_order_group_t2` (
22   ->   `seq_id` bigint(20) NOT NULL AUTO_INCREMENT COMMENT '自增长编号',
23   ->   `group_unique_code` varchar(60) NOT NULL COMMENT '批次唯一编号',

```

```

24      -> `order_group_no` varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
25      -> `inhos_code` varchar(20) DEFAULT '' COMMENT '病人住院号',
26      -> `orderbar` varchar(40) DEFAULT '' COMMENT '医嘱条码',
27      -> `package_bar` varchar(50) DEFAULT NULL COMMENT '药箱条码',
28      -> `plan_time` datetime DEFAULT NULL COMMENT '计划执行时间',
29      -> `order_sort_no` int(11) DEFAULT NULL COMMENT '医嘱排序编号',
30      -> `source_type` varchar(20) DEFAULT '' COMMENT '数据来源',
31      -> `isprint` int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
32      -> `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',
33      -> `execute_date` datetime DEFAULT NULL COMMENT '执行时间',
34      -> `print_date` datetime DEFAULT NULL COMMENT '打印时间',
35      -> `execute_person` varchar(20) DEFAULT '' COMMENT '执行人',
36      -> `print_person` varchar(20) DEFAULT '' COMMENT '打印人',
37      -> `apply_time` datetime DEFAULT NULL COMMENT '开立时间',
38      -> `is_dispensed` int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是,
0 - 否',
39      -> `remark` varchar(100) DEFAULT '' COMMENT '医嘱备注',
40      -> `reason` varchar(200) DEFAULT NULL COMMENT '手动或作废执行的原因',
41      -> `execute_type` int(11) DEFAULT '0' COMMENT '执行方式 0-无效, 1- 扫描执
行, 2 - 手动执行',
42      -> `start_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始执行
人',
43      -> `start_execute_date` datetime DEFAULT NULL COMMENT '手动补录开始执行时
间',
44      -> `start_check_user` varchar(20) DEFAULT NULL COMMENT '手动补录开始核对
人',
45      -> `end_execute_user` varchar(20) DEFAULT NULL COMMENT '手动补录结束执行
人',
46      -> `end_execute_date` datetime DEFAULT NULL COMMENT '手动补录结束执行时
间',
47      -> `create_time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创
建时间',
48      -> `create_person` varchar(50) NOT NULL DEFAULT '' COMMENT '创建人',
49      -> `update_time` datetime DEFAULT NULL COMMENT '修改时间',
50      -> `update_person` varchar(50) DEFAULT NULL COMMENT '修改人',
51      -> PRIMARY KEY (`seq_id`),
52      -> UNIQUE KEY `uq_pat_inhos_order_group_1` (`group_unique_code`),
53      -> KEY `idx_pat_inhos_order_group_1` (`order_group_no`),
54      -> KEY `idx_pat_inhos_order_group_2` (`orderbar`),
55      -> KEY `idx_pat_inhos_order_group_3` (`execute_date`),
56      -> KEY `idx_pat_inhos_order_group_4` (`plan_time`),
57      -> KEY `idx_idx_pat_inhos_order_group_5` (`inhos_code`),
58      -> KEY `idx_pat_inhos_order_group_6`
(`inhos_code`, `plan_time`, `execute_status`)
59      -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='医嘱批次表';

```

```

60 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:
61 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
62 mysql>

```

但是分片表是创建成功了：

```

> pat_inhos_order_group_t2_0
> pat_inhos_order_group_t2_1
> pat_inhos_order_group_t2_2
> pat_inhos_order_group_t2_3

```

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！！

```

1  mysql> show databases;
2  +-----+
3  | schema_name |
4  +-----+
5  | information_schema |
6  | mysql |
7  | performance_schema |
8  | shardingsphere |
9  | sys |
10 | windranger_emr |
11 +-----+
12 6 rows in set (0.01 sec)
13
14 mysql> use windranger_emr;

```

```

15 Reading table information for completion of table and column names
16 You can turn off this feature to get a quicker startup with -A
17
18 Database changed
19 mysql> CREATE TABLE IF NOT EXISTS pat_inhos_order_group_t2 (
20     -> seq_id bigint(20) NOT NULL AUTO_INCREMENT COMMENT '自增长编号',
21     -> group_unique_code varchar(60) NOT NULL COMMENT '批次唯一编号',
22     -> order_group_no varchar(50) DEFAULT '' COMMENT '医嘱批次编号',
23     -> inhos_code varchar(20) DEFAULT '' COMMENT '病人住院号',
24     -> orderbar varchar(40) DEFAULT '' COMMENT '医嘱条码',
25     -> package_bar varchar(50) DEFAULT NULL COMMENT '药箱条码',
26     -> plan_time datetime DEFAULT NULL COMMENT '计划执行时间',
27     -> order_sort_no int(11) DEFAULT NULL COMMENT '医嘱排序编号',
28     -> source_type varchar(20) DEFAULT '' COMMENT '数据来源',
29     -> isprint int(11) NOT NULL DEFAULT '0' COMMENT '是否已打印 0 否; 1 是',
30     -> 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',
31     -> execute_date datetime DEFAULT NULL COMMENT '执行时间',
32     -> print_date datetime DEFAULT NULL COMMENT '打印时间',
33     -> execute_person varchar(20) DEFAULT '' COMMENT '执行人',
34     -> print_person varchar(20) DEFAULT '' COMMENT '打印人',
35     -> apply_time datetime DEFAULT NULL COMMENT '开立时间',
36     -> is_dispensed int(11) NOT NULL DEFAULT '0' COMMENT '是否已配药 1-是, 0
    - 否',
37     -> remark varchar(100) DEFAULT '' COMMENT '医嘱备注',
38     -> reason varchar(100) DEFAULT NULL COMMENT '手动或作废执行的原因',
39     -> execute_type int(11) DEFAULT '0' COMMENT '执行方式 0-无效, 1- 扫描执
    行, 2 - 手动执行',
40     -> start_execute_user varchar(20) DEFAULT NULL COMMENT '手动补录开始执行
    人',
41     -> start_execute_date datetime DEFAULT NULL COMMENT '手动补录开始执行时
    间',
42     -> start_check_user varchar(20) DEFAULT NULL COMMENT '手动补录开始核对
    人',
43     -> end_execute_user varchar(20) DEFAULT NULL COMMENT '手动补录结束执行
    人',
44     -> end_execute_date datetime DEFAULT NULL COMMENT '手动补录结束执行时间',
45     -> create_time datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建
    时间',
46     -> create_person varchar(50) NOT NULL DEFAULT '' COMMENT '创建人',
47     -> update_time datetime DEFAULT NULL COMMENT '修改时间',
48     -> update_person varchar(50) DEFAULT NULL COMMENT '修改人',
49     -> PRIMARY KEY (seq_id)
50     -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='医嘱批次表';
51 Query OK, 0 rows affected (0.22 sec)
52

```

```
53 mysql> select count(*) from pat_inhos_order_group_t2;
54 +-----+
55 | count(*) |
56 +-----+
57 |          0 |
58 +-----+
59 1 row in set (0.07 sec)
60
61 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：

```
1 props:
2   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

```
1   databaseName: windranger_emr
2
```

```
3  dataSources:
4    ds_0:
5      url: jdbc:mysql://10.2.3.167:3306/windranger_emr?
characterEncoding=utf8&serverTimezone=GMT%2B8&allowMultiQueries=true
6      username: user
7      password: Lachesis-mh_1024
8      connectionTimeoutMilliseconds: 30000
9      idleTimeoutMilliseconds: 60000
10     maxLifetimeMilliseconds: 1800000
11     maxPoolSize: 50
12     minPoolSize: 1
13
14  rules:
15    - !SINGLE
16      tables:
17        - "*"
18    - !SHARDING
19      tables:
20        pat_inhos_order_group_t2:
21          actualDataNodes: ds_0.pat_inhos_order_group_t2_${0..3}
22          tableStrategy:
23            standard:
24              shardingColumn: inhos_code
25              shardingAlgorithmName: t_order_inline
26
27          shardingAlgorithms:
28            t_order_inline:
29              type: INLINE
30              props:
31                algorithm-expression:
pat_inhos_order_group_t2_${Math.abs(inhos_code.hashCode())%4}
```



```
mysql> SHOW STORAGE UNITS; 1.查看存储单元
+-----+
| name | type | host | port | db | connection_timeout_milliseconds | idle_timeout_milliseconds | max_lifetime_milliseconds | max_pool_size | min_pool_size | read_only | other_attributes |
+-----+
| ds_0 | MySQL | 10.2.3.167 | 3306 | windranger_emr | 30000 | 60000 | 1800000 | 50 | 1 | false | {"dataSourceProperties":{"cacheServerConfiguration":{"true","elideSetAutoCommits":"true","useServerPrepStmts":"true","cachePrepStmts":"true","useSSL":"false","rewriteBatchedStatements":"true","cacheResultSetMetadata":"false","useLocalSessionState":"true","maintainTimeStats":"false","prepStmtCacheSize":"18192","tinyInttisBit":"false","prepStmtCacheSqlLimit":"2048","netTimeoutForStreamingResults":"0","zeroDateTmeBehavior":"round"},"healthCheckProperties":{"},"initializationFailTimeout":1,"validationTimeout":5000,"keepaliveTime":0,"leakDetectionThreshold":0,"registerMbeans":"false","allowPoolSuspension":"false","autoCommit":"true","isolateInternalQueries":"false","queryProperties":{"serverTimezone":"GMT+288","allowMultiQueries":"true","characterEncoding":"utf8"}}} |
+-----+
1 row in set (0.34 sec)

mysql> SHOW SHARDING TABLE RULES; 2.查看分片规则：为配置文件中的pat_inhos_order_group.t2
+-----+
| 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 | auditor_types | allow_hint_disable |
+-----+
| pat_inhos_order_group_t2 | ds_0_pat_inhos_order_group_t2 ${0..3} | | | | | | | | | | | | |
+-----+
1 row in set (0.06 sec)

mysql> CREATE SHARDING TABLE RULE IF NOT EXISTS pat_inhos_order_group ( 3.创建新的分片规则
-> STORAGE UNITS(ds_0);
-> SHARDING_COLUMN=inhos_code,TYPE(NAME="hash_mod",PROPERTIES("sharding-count"=4));
-> );
Query OK, 0 rows affected (0.38 sec)

mysql> SHOW SHARDING TABLE RULES; 4.再次查看时，只有新创建的分片规则
+-----+
| 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 | auditor_types | allow_hint_disable |
+-----+
| pat_inhos_order_group | | | | | | | | | | | | |
+-----+
1 row in set (0.01 sec)
```


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：

```
[2024-01-22 17:10:16] 23 ms 中有 1 行受到影响
windranger_emr> INSERT INTO windranger_emr.pat_inhos_order_group (seq_id, group_unique_code, order_group_no, inhos_code, orderbar, package_bar, plan_time, order_sort_no, source_type, isprint, execute_status, execute_date, print_date, execute_person, print_per
[2024-01-22 17:10:17] 25 ms 中有 1 行受到影响
windranger_emr> INSERT INTO windranger_emr.pat_inhos_order_group (seq_id, group_unique_code, order_group_no, inhos_code, orderbar, package_bar, plan_time, order_sort_no, source_type, isprint, execute_status, execute_date, print_date, execute_person, print_per
[2024-01-22 17:10:17] 24 ms 中有 1 行受到影响
windranger_emr> INSERT INTO windranger_emr.pat_inhos_order_group (seq_id, group_unique_code, order_group_no, inhos_code, orderbar, package_bar, plan_time, order_sort_no, source_type, isprint, execute_status, execute_date, print_date, execute_person, print_per
[2024-01-22 17:10:17] [HY000][20081] Routed target 'pat_inhos_order_0' does not exist, available targets are ['pat_inhos_order_group_0', 'pat_inhos_order_group_1', 'pat_inhos_order_group_2', 'pat_inhos_order_group_3', 'pat_inhos_order_group_4', 'pat_inhos_order_group_5']
```

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

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

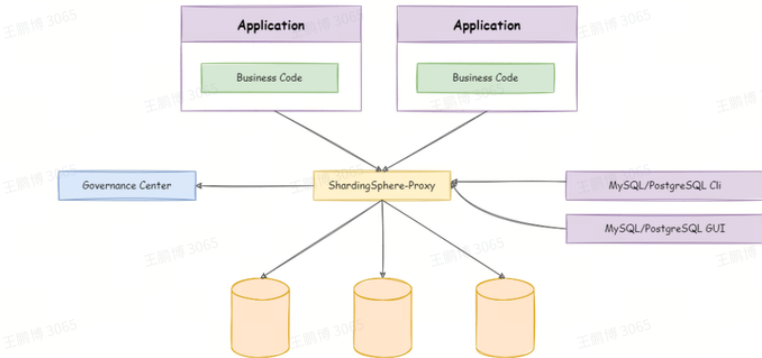
http://10.2.3.111/backend_group/backend_application/emr/blob/master/windranger-emr-web/src/main/resources/database/migrations/V1_0_1_31__Create.sql

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/>

- 1. 概览
- 2. 快速入门
 - 2.1. ShardingSphere-JDBC
 - 2.2. ShardingSphere-Proxy
- 3. 功能
- 4. 用户手册
- 5. 开发者手册
- 6. 测试手册
- 7. 技术参考
- 8. FAQ
- 9. 下载



- 应用场景
- 使用限制
- 前提条件
- 操作步骤

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。

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`、`gsqsl` 等）连接 Proxy 验证功能。

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
<div> image-2024-1-23_11-22-... 272.80KB</div>	272.80 KB	王鹏博	2024-01-23T11:22:53.000+08:00		
<div> image-2024-1-23_11-20-... 292.46KB</div>	292.46 KB	王鹏博	2024-01-23T11:20:50.000+08:00		
<div> image-2024-1-23_11-19-... 97.90KB</div>	97.90 KB	王鹏博	2024-01-23T11:19:45.000+08:00		
<div> image-2024-1-22_18-32-... 169.78KB</div>	169.78 KB	王鹏博	2024-01-22T18:30:53.000+08:00		
	736.08 KB	王鹏博	2024-01-22T17:17:		

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

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

