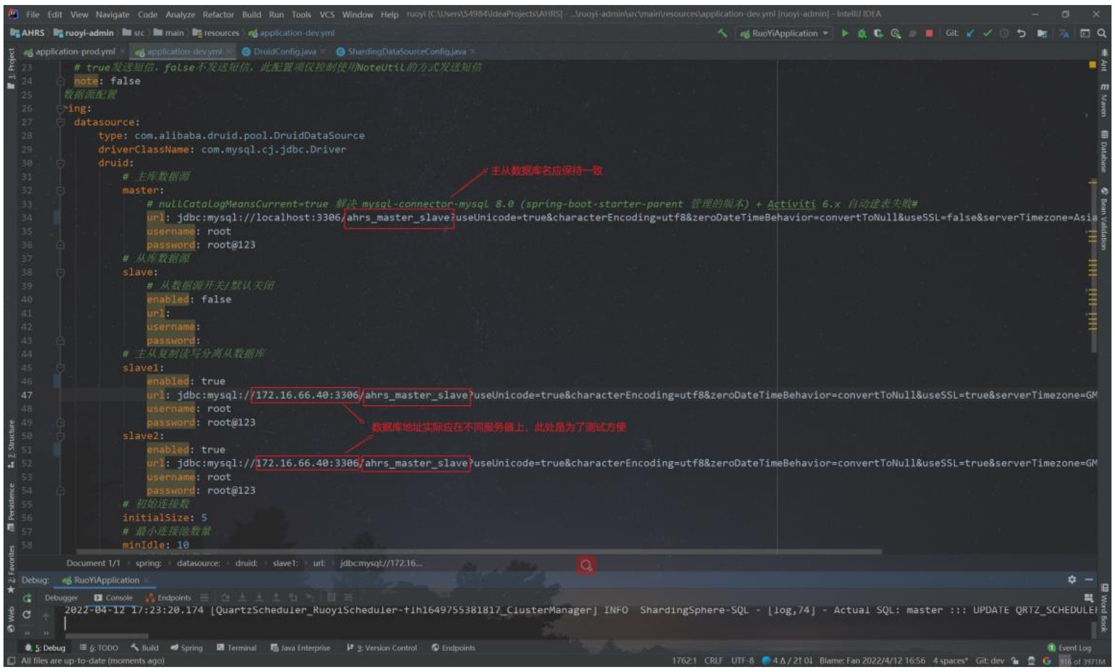
第一步、配置好 mysql 主从复制

第二步、在 spring 配置文件 application 中设置从库



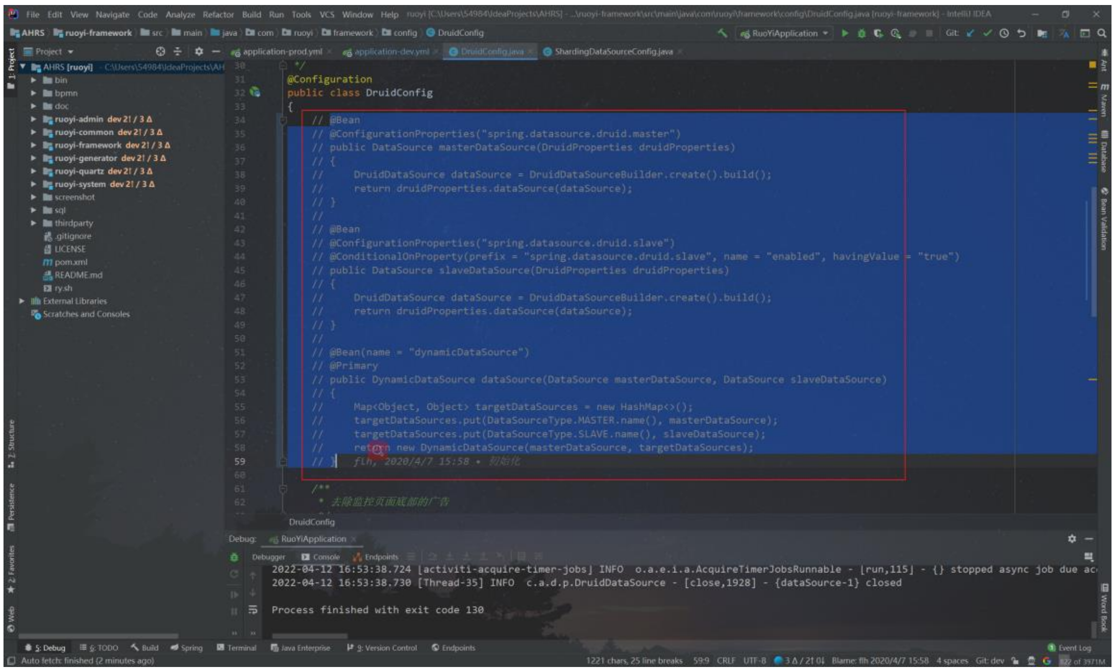
```yaml
    # true发送短信，false不发送短信，此配置项仅控制使用NoteUtil的方式发送短信
    note: false
  数据源配置
ing:
  datasource:
      type: com.alibaba.druid.pool.DruidDataSource
      driverClassName: com.mysql.cj.jdbc.Driver
      druid:
          # 主库数据源
          master:
              # nullCatalogMeansCurrent=true 解决 mysql-connector-mysql 8.0 (spring-boot-starter-parent 管理的版本) + Activiti 6.x 自动建表失败#
              url: jdbc:mysql://localhost:3306/ahrs_master_slave?useUnicode=true&characterEncoding=utf8&zeroDateTimeBehavior=convertToNull&useSSL=false&serverTimezone=Asia
              username: root
              password: root@123
          # 从库数据源
          slave:
              # 从数据源开关/默认关闭
              enabled: false
              url:
              username:
              password:
          # 主从复制读写分离从数据库
          slave1:
              enabled: true
              url: jdbc:mysql://172.16.66.40:3306/ahrs_master_slave?useUnicode=true&characterEncoding=utf8&zeroDateTimeBehavior=convertToNull&useSSL=true&serverTimezone=GM
              username: root
              password: root@123
          slave2:
              enabled: true
              url: jdbc:mysql://172.16.66.40:3306/ahrs_master_slave?useUnicode=true&characterEncoding=utf8&zeroDateTimeBehavior=convertToNull&useSSL=true&serverTimezone=GM
              username: root
              password: root@123
          # 初始连接数
          initialSize: 5
          # 最小连接池数量
          minIdle: 10
```

主从数据库名应保持一致

数据库地址实际应在不同服务器上，此处是为了测试方便

```
2022-04-12 17:23:20.174 [QuartzScheduler_RuoyiScheduler-fih1649755381817_ClusterManager] INFO  ShardingSphere-SQL - [log,74] - Actual SQL: master ::: UPDATE QRTZ_SCHEDULE
```

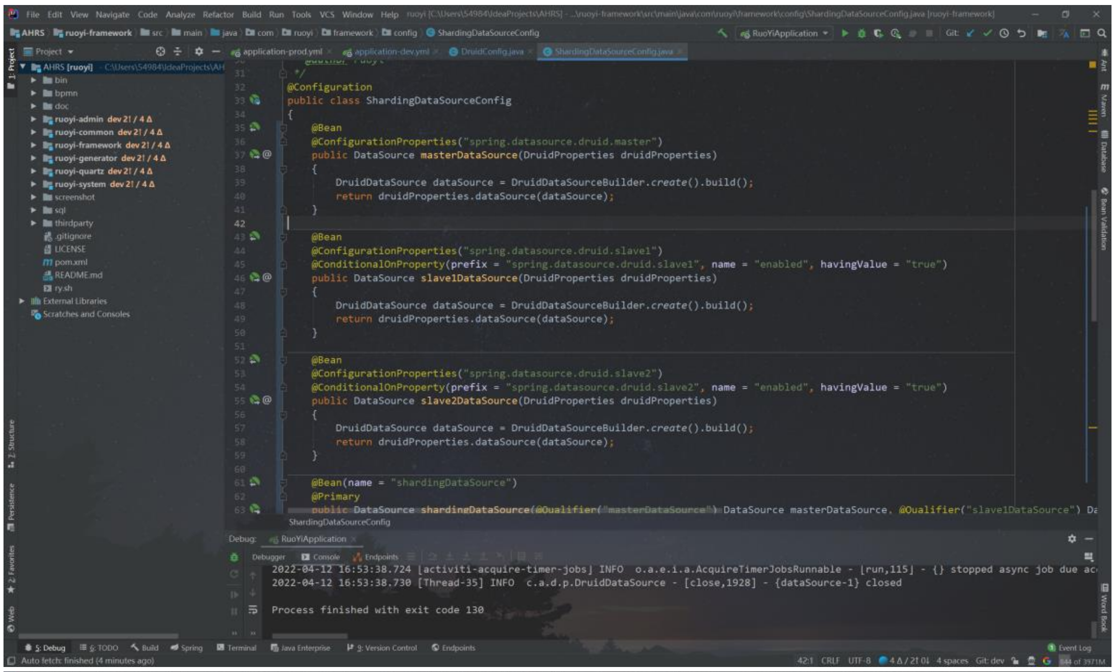
第三步、注释多数据源配置文件



```java
@Configuration
public class DruidConfig
{
//  @Bean
//  @ConfigurationProperties("spring.datasource.druid.master")
//  public DataSource masterDataSource(DruidProperties druidProperties)
//  {
//      DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
//      return druidProperties.dataSource(dataSource);
//  }
//
//  @Bean
//  @ConfigurationProperties("spring.datasource.druid.slave")
//  @ConditionalOnProperty(prefix = "spring.datasource.druid.slave", name = "enabled", havingValue = "true")
//  public DataSource slaveDataSource(DruidProperties druidProperties)
//  {
//      DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
//      return druidProperties.dataSource(dataSource);
//  }
//
//  @Bean(name = "dynamicDataSource")
//  @Primary
//  public DynamicDataSource dataSource(DataSource masterDataSource, DataSource slaveDataSource)
//  {
//      Map<Object, Object> targetDataSources = new HashMap<>();
//      targetDataSources.put(DataSourceType.MASTER.name(), masterDataSource);
//      targetDataSources.put(DataSourceType.SLAVE.name(), slaveDataSource);
//      return new DynamicDataSource(masterDataSource, targetDataSources);
//  }   fih, 2020/4/7 15:58 • 初始化

    /**
     * 去除监控页面底部的广告
```

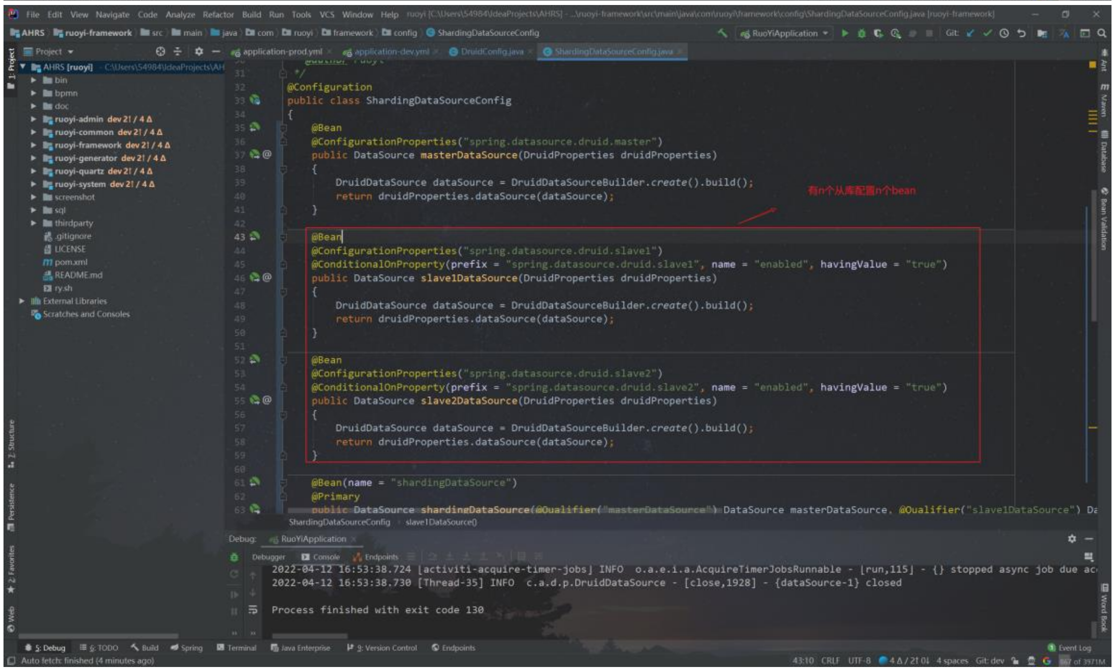
```
2022-04-12 16:53:38.724 [activiti-acquire-timer-jobs] INFO  o.a.e.i.a.AcquireTimerJobsRunnable - [run,115] - {} stopped async job due ac
2022-04-12 16:53:38.730 [Thread-35] INFO  c.a.d.p.DruidDataSource - [close,1928] - {dataSource-1} closed

Process finished with exit code 130
```

第四步、放开 ShardingDataSourceConfig 配置文件注释并修改对应从库信息



```java
 */
@Configuration
public class ShardingDataSourceConfig
{
    @Bean
    @ConfigurationProperties("spring.datasource.druid.master")
    public DataSource masterDataSource(DruidProperties druidProperties)
    {
        DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
        return druidProperties.dataSource(dataSource);
    }

    @Bean
    @ConfigurationProperties("spring.datasource.druid.slave1")
    @ConditionalOnProperty(prefix = "spring.datasource.druid.slave1", name = "enabled", havingValue = "true")
    public DataSource slave1DataSource(DruidProperties druidProperties)
    {
        DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
        return druidProperties.dataSource(dataSource);
    }

    @Bean
    @ConfigurationProperties("spring.datasource.druid.slave2")
    @ConditionalOnProperty(prefix = "spring.datasource.druid.slave2", name = "enabled", havingValue = "true")
    public DataSource slave2DataSource(DruidProperties druidProperties)
    {
        DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
        return druidProperties.dataSource(dataSource);
    }

    @Bean(name = "shardingDataSource")
    @Primary
    public DataSource shardingDataSource(@Qualifier("masterDataSource") DataSource masterDataSource, @Qualifier("slave1DataSource") Da
```

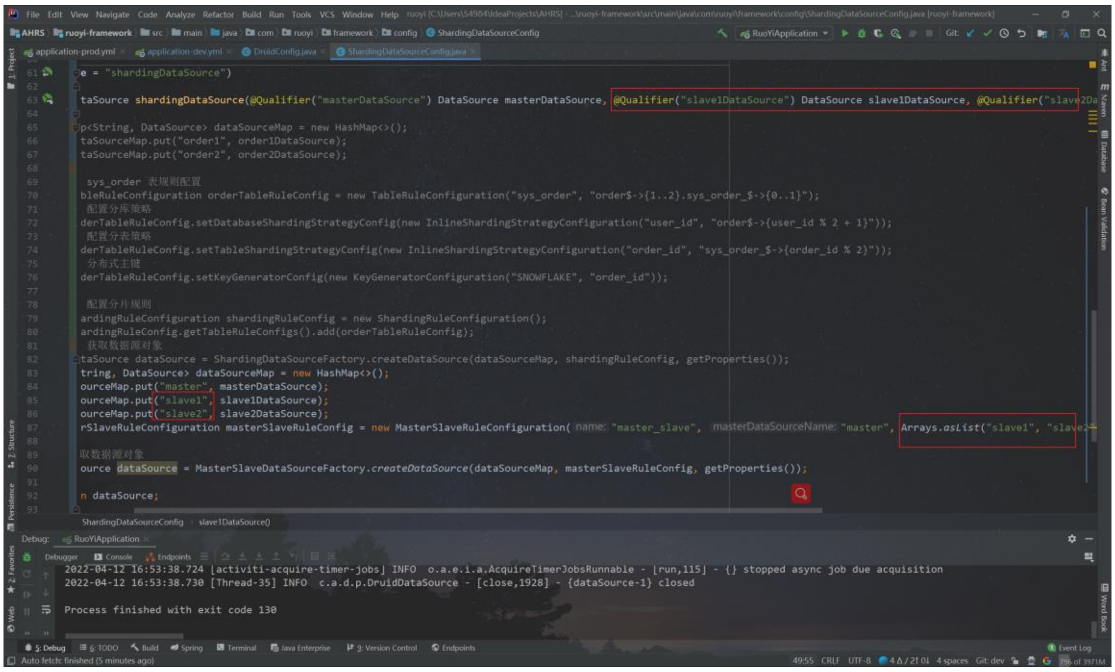
```
2022-04-12 16:53:38.724 [activiti-acquire-timer-jobs] INFO  o.a.e.i.a.AcquireTimerJobsRunnable - [run,115] - {} stopped async job due ac
2022-04-12 16:53:38.730 [Thread-35] INFO  c.a.d.p.DruidDataSource - [close,1928] - {dataSource-1} closed

Process finished with exit code 130
```



有n个从库配置n个bean

```java
 */
@Configuration
public class ShardingDataSourceConfig
{
    @Bean
    @ConfigurationProperties("spring.datasource.druid.master")
    public DataSource masterDataSource(DruidProperties druidProperties)
    {
        DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
        return druidProperties.dataSource(dataSource);
    }

    @Bean
    @ConfigurationProperties("spring.datasource.druid.slave1")
    @ConditionalOnProperty(prefix = "spring.datasource.druid.slave1", name = "enabled", havingValue = "true")
    public DataSource slave1DataSource(DruidProperties druidProperties)
    {
        DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
        return druidProperties.dataSource(dataSource);
    }

    @Bean
    @ConfigurationProperties("spring.datasource.druid.slave2")
    @ConditionalOnProperty(prefix = "spring.datasource.druid.slave2", name = "enabled", havingValue = "true")
    public DataSource slave2DataSource(DruidProperties druidProperties)
    {
        DruidDataSource dataSource = DruidDataSourceBuilder.create().build();
        return druidProperties.dataSource(dataSource);
    }

    @Bean(name = "shardingDataSource")
    @Primary
    public DataSource shardingDataSource(@Qualifier("masterDataSource") DataSource masterDataSource, @Qualifier("slave1DataSource") Da
```

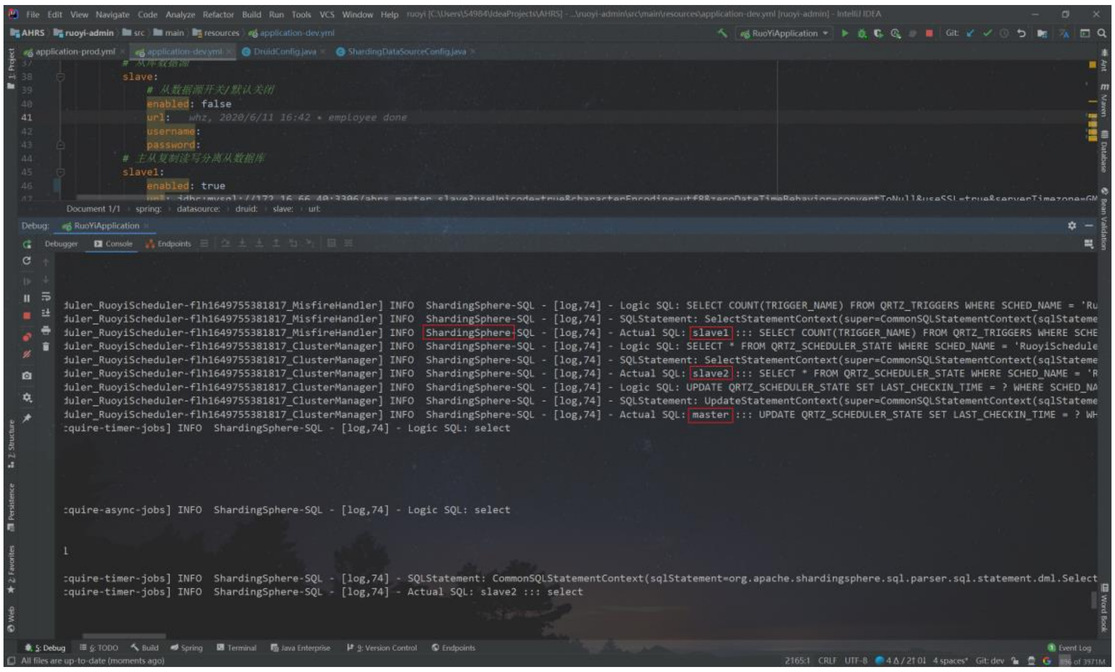
```
2022-04-12 16:53:38.724 [activiti-acquire-timer-jobs] INFO  o.a.e.i.a.AcquireTimerJobsRunnable - [run,115] - {} stopped async job due ac
2022-04-12 16:53:38.730 [Thread-35] INFO  c.a.d.p.DruidDataSource - [close,1928] - {dataSource-1} closed

Process finished with exit code 130
```



```java
e = "shardingDataSource")

taSource shardingDataSource(@Qualifier("masterDataSource") DataSource masterDataSource, @Qualifier("slave1DataSource") DataSource slave1DataSource, @Qualifier("slave2Da

p<String, DataSource> dataSourceMap = new HashMap<>();
taSourceMap.put("order1", order1DataSource);
taSourceMap.put("order2", order2DataSource);

 sys_order 表规则配置
bleRuleConfiguration orderTableRuleConfig = new TableRuleConfiguration("sys_order", "order$->{1..2}.sys_order_$->{0..1}");
 配置分库策略
derTableRuleConfig.setDatabaseShardingStrategyConfig(new InlineShardingStrategyConfiguration("user_id", "order$->{user_id % 2 + 1}"));
 配置分表策略
derTableRuleConfig.setTableShardingStrategyConfig(new InlineShardingStrategyConfiguration("order_id", "sys_order_$->{order_id % 2}"));
 分布式主键
derTableRuleConfig.setKeyGeneratorConfig(new KeyGeneratorConfiguration("SNOWFLAKE", "order_id"));

 配置分片规则
ardingRuleConfiguration shardingRuleConfig = new ShardingRuleConfiguration();
ardingRuleConfig.getTableRuleConfigs().add(orderTableRuleConfig);
 获取数据源对象
taSource dataSource = ShardingDataSourceFactory.createDataSource(dataSourceMap, shardingRuleConfig, getProperties());
tring, DataSource> dataSourceMap = new HashMap<>();
ourceMap.put("master", masterDataSource);
ourceMap.put("slave1", slave1DataSource);
ourceMap.put("slave2", slave2DataSource);
rSlaveRuleConfiguration masterSlaveRuleConfig = new MasterSlaveRuleConfiguration( name: "master_slave",  masterDataSourceName: "master", Arrays.asList("slave1", "slave2

取数据源对象
ource dataSource = MasterSlaveDataSourceFactory.createDataSource(dataSourceMap, masterSlaveRuleConfig, getProperties());

n dataSource;
```

```
2022-04-12 16:53:38.724 [activiti-acquire-timer-jobs] INFO  o.a.e.i.a.AcquireTimerJobsRunnable - [run,115] - {} stopped async job due acquisition
2022-04-12 16:53:38.730 [Thread-35] INFO  c.a.d.p.DruidDataSource - [close,1928] - {dataSource-1} closed

Process finished with exit code 130
```

第五步、启动后确认成功



```yaml
      slave:
        # 从数据源开关/默认关闭
        enabled: false
        url:   whz, 2020/6/11 16:42 • employee done
        username:
        password:
      # 主从复制读写分离从数据库
      slave1:
        enabled: true
```

```
duler_RuoyiScheduler-flh1649755381817_MisfireHandler] INFO  ShardingSphere-SQL - [log,74] - Logic SQL: SELECT COUNT(TRIGGER_NAME) FROM QRTZ_TRIGGERS WHERE SCHED_NAME = 'Ru
duler_RuoyiScheduler-flh1649755381817_MisfireHandler] INFO  ShardingSphere-SQL - [log,74] - SQLStatement: SelectStatementContext(super=CommonSQLStatementContext(sqlStateme
duler_RuoyiScheduler-flh1649755381817_MisfireHandler] INFO  ShardingSphere-SQL - [log,74] - Actual SQL: slave1 ::: SELECT COUNT(TRIGGER_NAME) FROM QRTZ_TRIGGERS WHERE SCHE
duler_RuoyiScheduler-flh1649755381817_ClusterManager] INFO  ShardingSphere-SQL - [log,74] - Logic SQL: SELECT * FROM QRTZ_SCHEDULER_STATE WHERE SCHED_NAME = 'RuoyiSchedule
duler_RuoyiScheduler-flh1649755381817_ClusterManager] INFO  ShardingSphere-SQL - [log,74] - SQLStatement: SelectStatementContext(super=CommonSQLStatementContext(sqlStateme
duler_RuoyiScheduler-flh1649755381817_ClusterManager] INFO  ShardingSphere-SQL - [log,74] - Actual SQL: slave2 ::: SELECT * FROM QRTZ_SCHEDULER_STATE WHERE SCHED_NAME = 'R
duler_RuoyiScheduler-flh1649755381817_ClusterManager] INFO  ShardingSphere-SQL - [log,74] - Logic SQL: UPDATE QRTZ_SCHEDULER_STATE SET LAST_CHECKIN_TIME = ? WHERE SCHED_NA
duler_RuoyiScheduler-flh1649755381817_ClusterManager] INFO  ShardingSphere-SQL - [log,74] - SQLStatement: UpdateStatementContext(super=CommonSQLStatementContext(sqlStateme
duler_RuoyiScheduler-flh1649755381817_ClusterManager] INFO  ShardingSphere-SQL - [log,74] - Actual SQL: master ::: UPDATE QRTZ_SCHEDULER_STATE SET LAST_CHECKIN_TIME = ? WH
:quire-timer-jobs] INFO  ShardingSphere-SQL - [log,74] - Logic SQL: select





:quire-async-jobs] INFO  ShardingSphere-SQL - [log,74] - Logic SQL: select


1

:quire-timer-jobs] INFO  ShardingSphere-SQL - [log,74] - SQLStatement: CommonSQLStatementContext(sqlStatement=org.apache.shardingsphere.sql.parser.sql.statement.dml.Select
:quire-timer-jobs] INFO  ShardingSphere-SQL - [log,74] - Actual SQL: slave2 ::: select
```