

# Spring Boot 版 Sharding-JDBC 读写分离

作者: [MingGH](#)

原文链接: <https://ld246.com/article/1601984716814>

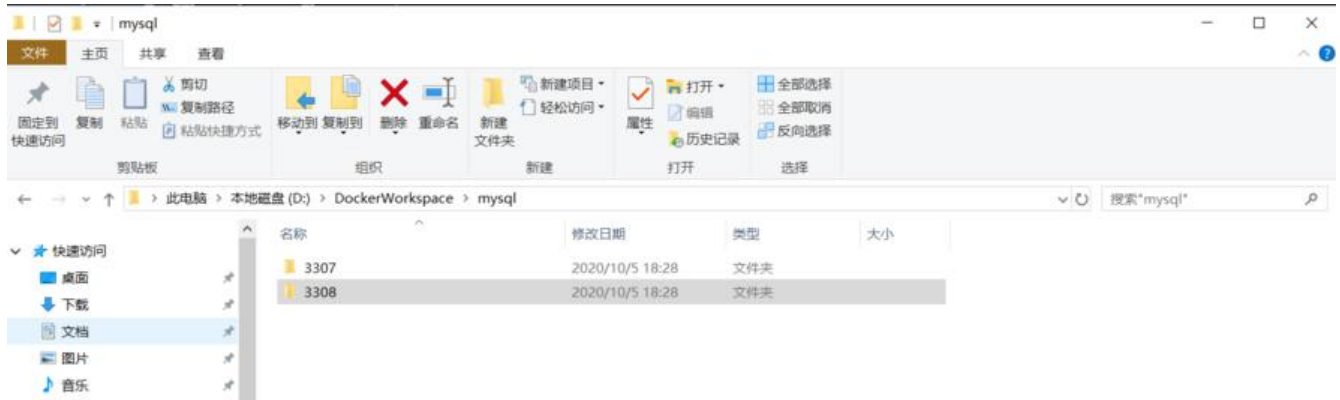
来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

# 一、搭建docker+mysql主从复制

本文采用的是windows上进行安装的docker,mysql使用的是5.6版本

## 1.1 分别创建主从对应的mysql配置文件



### 1.1.1 主mysql

D:\DockerWorkspace\mysql\3307\con\mysql.cnf

```
[mysqld]
user=mysql
log-bin=mysql-bin
server-id=1
character-set-server=utf8mb4
default_authentication_plugin=mysql_native_password
table_definition_cache=400
[client]
default-character-set=utf8mb4
[mysql]
default-character-set=utf8mb4
```

### 1.1.2 从mysql

D:\DockerWorkspace\mysql\3308\conf\mysql.cnf

```
[mysqld]
user=mysql
server-id=2
character-set-server=utf8mb4
default_authentication_plugin=mysql_native_password
table_definition_cache=400
[client]
default-character-set=utf8mb4
[mysql]
default-character-set=utf8mb4
```

## 1.2 创建docker容器与容器之间的网络

以创建一个新的Bridge网络实现两个容器之间的互相通信

参考博客:[Docker网络——实现容器间通信、容器与外网通信以及容器的跨主机访问](#)

```
docker network create mysql
```

## 1.3 docker安装mysql

创建主mysql,配置详情

- name 给创建的容器取别名
- p 端口映射
- e 指定容器内的环境变量
- v 挂载宿主机指定的文件, 这里挂载的是宿主机的mysql配置
- network 创建容器时使用Bridge网络 参考博客: [Docker network整理](#)
- network-alias 设置网络别名
- d 默认不会进入容器,容器后台运行

### 1.2.1 主mysql

```
docker run --name mysql-3307 -p 3307:3306 -e MYSQL_ROOT_PASSWORD=123456 -v D:\DockerWorkspace\mysql\3307\conf\mysql.cnf:/etc/mysql/conf.d/mysql.cnf --network mysql --network-alias mysql-master -d mysql:5.6
```

### 1.2.2 从mysql

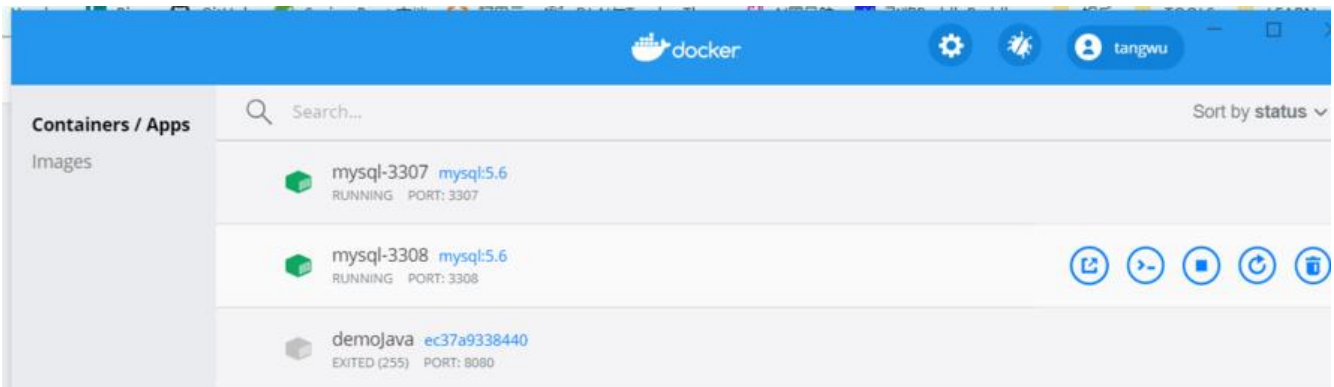
```
docker run --name mysql-3308 -p 3308:3306 -e MYSQL_ROOT_PASSWORD=123456 -v D:\DockerWorkspace\mysql\3308\conf\mysql.cnf:/etc/mysql/conf.d/mysql.cnf --network mysql --network-alias mysql-slave -d mysql:5.6
```

此时, 两个mysql的服务已经启动了

可以通过命令

```
docker ps -a
```

看到正在运行的容器, 如果是windows版本可以直接在面板看到



## 1.4 进入mysql容器进行配置

## 1.4.1 主mysql

```
docker exec -it e8c6 /bin/bash
```

在容器中 **连接mysql数据库**，密码根据安装mysql的时候进行对应修改

```
mysql -h127.0.0.1 -uroot -p123456
```

**授予 从mysql 复制权限**

```
GRANT REPLICATION SLAVE ON *.* TO 'root'@'%';
```

**刷新权限**

```
flush privileges;
```

**查看主服务器状态**

```
show master status;
```

```
mysql> show master status;
```

File	Position	Binlog_Do_DB	Binlog_Ignore_DB	Executed_Gtid_Set
mysql-bin.000004	1376			

```
1 row in set (0.00 sec)

mysql>
```

此时不要关闭窗口，需要记住 **File**列对应的名字，以及 **Position**对应的数字,这两个值在 **从mysql** 中行同步的时候需要用到

## 1.4.2 从mysql

新起一个dos窗口，进入容器

```
docker exec -it a8aa /bin/bash
```

**连接mysql数据库**

```
mysql -h127.0.0.1 -uroot -p123456
```

**配置同步数据**

- master\_host 主mysql名字，可填ip
- master\_user 连接主mysql用户
- master\_password 连接密码
- master\_log\_file 也就是在主mysql上使用show master status;获取到的File的值
- master\_log\_pos 从主mysql上哪个位置开始进行同步数据，如果填0，那么意味着从新开始同步，填在主mysql上使用show master status;获取到的Position的值意味着从当前位置进行同步

执行

```
change master to master_host='mysql-master',master_user='root',master_password='123456'  
master_log_file='mysql-bin.000004',master_log_pos=502,master_port=3306;
```

**启动slave** 也就是开启同步

```
start slave;
```

**查看slave状态**

```
show slave status\G;
```

当两个都处于YES的时候，意味着配置成功了

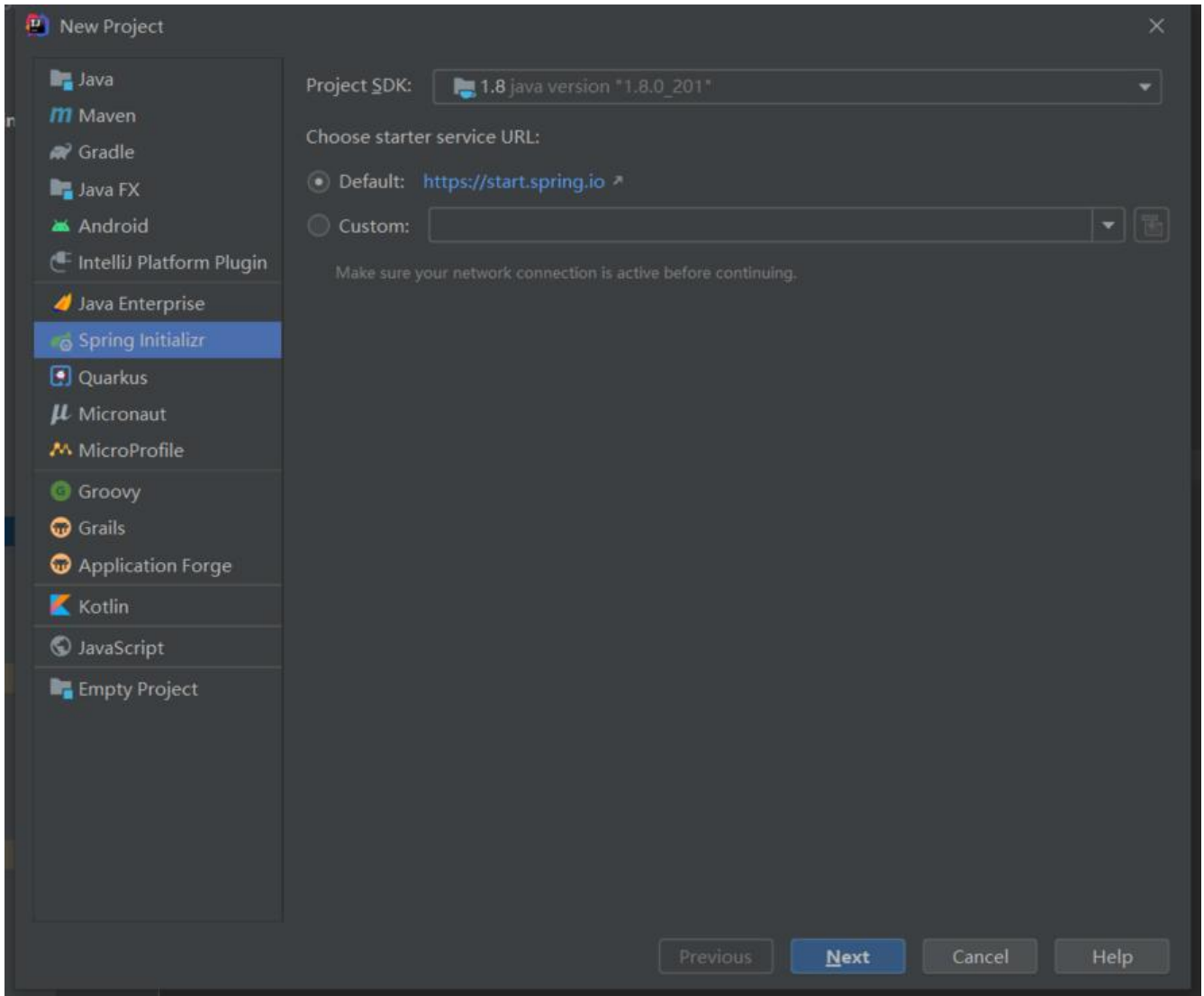
```
Read_Master_Log_Pos: 1376  
Relay_Log_File: mysqld-relay-bin.000002  
Relay_Log_Pos: 1012  
Relay_Master_Log_File: mysql-bin.000004  
Slave_IO_Running: Yes  
Slave_SQL_Running: No  
Replicate_Do_DB:  
Replicate_Ignore_DB:  
Replicate_Do_Table:  
Replicate_Ignore_Table:  
Replicate_Wild_Do_Table:  
Replicate_Wild_Ignore_Table:  
Last_Errno: 1007  
Last_Error: Error 'Can't create database 'temp2': database exists' on query. Default database: 'temp2'. Query: 'CREATE DATABASE `temp2`'  
Skip_Counter: 0  
Exec_Master_Log_Pos: 849  
Relay_Log_Space: 1713  
Until_Condition: None  
Until_Log_File:  
Until_Log_Pos: 0  
Master_SSL_Allowed: No  
Master_SSL_CA_File:  
Master_SSL_CA_Path:  
Master_SSL_Cert:  
Master_SSL_Cipher:  
Master_SSL_Key:
```

此时当你在主mysql进行怎样的操作都会同步到从数据库

## 二、搭建Sharding-JDBC读写分离

官方参考文档: [SHARDING-JDBC](#)

### 2.1 创建SpringBoot项目，引入依赖



依赖

pom.xml

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>

<dependency>
  <groupId>org.mybatis.spring.boot</groupId>
  <artifactId>mybatis-spring-boot-starter</artifactId>
  <version>2.1.3</version>
</dependency>

<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
</dependency>

<dependency>
  <groupId>com.alibaba</groupId>
  <artifactId>druid-spring-boot-starter</artifactId>
```

```
    <version>1.1.16</version>
  </dependency>

  <dependency>
    <groupId>org.apache.shardingsphere</groupId>
    <artifactId>sharding-jdbc-spring-boot-starter</artifactId>
    <version>4.0.0-RC1</version>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
```

## 2.2 配置sharding-jdbc

```
application.properties
```

```
server.port=8080
```

```
#数据源名称，多数据源以逗号分隔
```

```
spring.shardingsphere.datasource.names=master,slave
```

```
# 主数据源
```

```
#数据库连接池类名称
```

```
spring.shardingsphere.datasource.master.type=com.alibaba.druid.pool.DruidDataSource
```

```
#数据库驱动类名
```

```
spring.shardingsphere.datasource.master.driver-class-name=com.mysql.jdbc.Driver
```

```
#数据库url连接
```

```
spring.shardingsphere.datasource.master.url=jdbc:mysql://localhost:3307/temp1?characterEncoding=utf-8
```

```
#数据库用户名
```

```
spring.shardingsphere.datasource.master.username=root
```

```
#数据库密码
```

```
spring.shardingsphere.datasource.master.password=123456
```

```
#从数据源
```

```
spring.shardingsphere.datasource.slave.type=com.alibaba.druid.pool.DruidDataSource
```

```
spring.shardingsphere.datasource.slave.driver-class-name=com.mysql.jdbc.Driver
```

```
spring.shardingsphere.datasource.slave.url=jdbc:mysql://localhost:3308/temp1?characterEncoding=utf-8
```

```
spring.shardingsphere.datasource.slave.username=root
```

```
spring.shardingsphere.datasource.slave.password=123456
```

```
# 读写分离配置
```

```
#从库负载均衡算法类型，可选值：ROUND_ROBIN，RANDOM。若`load-balance-algorithm-class-name`存在则忽略该配置
```

```
spring.shardingsphere.masterslave.load-balance-algorithm-type=round_robin
```

```
spring.shardingsphere.masterslave.name=dataSource
```

```
#主库数据源名
```

```
spring.shardingsphere.masterslave.master-data-source-name=master
```

```
#从库数据源名称列表
```

```
spring.shardingsphere.masterslave.slave-data-source-names=slave
```

```
#是否开启SQL显示, 默认值: false  
spring.shardingsphere.props.sql.show=true  
#mybatis配置  
mybatis.type-aliases-package=run.runnable.readwritesplit.entity
```

其实有这些就已经完成了, 剩下的不用做其他的配置, 因为sharding会帮我们自动插入数据时调用主, 查询数据时调用从库

源码: <https://github.com/MingGH/read-write-split>