

# 《软件架构与中间件》实验指导书

## ——实验三：数据分库分表的设计与实现

### ➤ 实验 1 Mycat 使用

#### 1.1 基本开发环境准备

- 安装 MySQL Server 5.5 及以上
- 安装 jdk1.6 以上（建议使用 1.7 或 1.8）
- 安装 Mycat

■ 下载地址 <http://dl.mycat.io/1.6.7.1/>

■ 下载完成后解压到合适位置

#### 1.2 Mycat 目录说明

/bin ：启动目录

/conf ：配置目录存放配置文件

--server.xml：是 Mycat 服务器参数调整和用户授权的配置文件。

--schema.xml：是逻辑库定义和表以及分片定义的配置文件。

--rule.xml：是分片规则的配置文件，分片规则的具体一些参数信息单

独存放为文件，也在这个目录下，配置文件修改需要重启 MyCAT。

#### 1.3 Mycat 配置

- Mycat 系统参数配置（/conf/server.xml）

所有的 Mycat 参数变量都是配置在 server.xml 文件中，system 标签下配置所有的参数，如果需要配置某个变量添加相应的配置即可，例如添加启动端口 8066，默认为 8066。

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE mycat:server SYSTEM "server.dtd">
3. <mycat:server xmlns:mycat="http://io.mycat/">
4.     <system>
5.         <property name="serverPort">8066</property>
6.         <!-- 0 为需要密码登陆、1 为不需要密码登陆 ,默认为 0, 设置为 1 则需要指定默认
           账户-->
7.         <property name="nonePasswordLogin">0</property>
8.         <property name="useHandshakeV10">1</property>
9.     </system>
10.    <!-- 设置登陆 Mycat 的用户名,密码,逻辑库 -->
11.    <user name="root" defaultAccount="true">
12.        <property name="password">123456</property>
13.        <property name="schemas">TESTDB</property>
14.    </user>
15. </mycat:server>

```

- 逻辑库、表分片配置 (/conf/schema.xml)

Mycat 作为一个中间件，实现 MySQL 协议，那么对前端应用连接来说就是一个数据库，也就有数据库的配置，Mycat 的数据库配置是在 schema.xml 中配置，配置好后映射到 server.xml 里面的用户就可以了。

注意：WriteHost 中 url,username,password 按照自己实际物理数据库配置填写。

```

1. <?xml version="1.0"?>
2. <!DOCTYPE mycat:schema SYSTEM "schema.dtd">
3. <mycat:schema xmlns:mycat="http://io.mycat/">
4.    <!-- 配置逻辑库 TESTDB -->
5.    <schema name="TESTDB" checkSQLschema="false" sqlMaxLimit="100">
6.        <!-- 逻辑表配置, name 代表表名, dataNode 代表对应的分片 -->
7.        <table name="users" primaryKey="id" dataNode="dn1" />
8.        <table name="item" primaryKey="id" dataNode="dn2,dn3" rule="rule1" />
9.    </schema>
10.    <!-- 设置 dataNode 对应的数据库,及 mycat 连接的地址 dataHost -->
11.    <!-- 配置分片(dataNode) -->
12.    <!-- 表切分后需要配置映射到哪个数据库中, Mycat 的分片实际上就是库的别名 -
       -->
13.    <!-- 例如上面例子配置了三个分片 dn1,dn2,dn3 分别对应到物理机映射
       dataHost localhost1 的两个库上 -->
14.    <dataNode name="dn1" dataHost="localhost1" database="db1" />

```

```

15.     <dataNode name="dn2" dataHost="localhost1" database="db2" />
16.     <dataNode name="dn3" dataHost="localhost1" database="db3" />
17.     <!-- 配置物理库分片映射 -->
18.     <!-- Mycat 作为数据库代理需要逻辑库，逻辑用户，表切分后需要配置分片，分片也就需
        要映射到真实的物理主机上 -->
19.     <dataHost name="localhost1" maxCon="1000" minCon="10" balance="0"
20.         writeType="0" dbType="mysql" dbDriver="native" switchType="1"
        slaveThreshold="100">
21.         <!-- heartbeat 标签代表 Mycat 需要对物理库心跳检测的语句 -->
22.         <heartbeat>select user()</heartbeat>
23.         <!-- writeHost 此标签代表 一个逻辑主机（dataHost）对应的后端的物理主机映
            射 -->
24.         <writeHost host="hostM1" url="localhost:3306" user="root" password="
            123456" />
25.     </dataHost>
26. </mycat:schema>

```

- Mycat 表切分规则配置 （/conf/rule.xml）

数据切分中作为表切分规则中最重要的配置，表的切分方式决定了数据切分后的性能好坏，因此也是最重要的配置。

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE mycat:rule SYSTEM "rule.dtd">
3. <mycat:rule xmlns:mycat="http://io.mycat/">
4.     <!-- name 为 schema.xml 中 table 标签中对应的 rule="name1" ,也就是配置表的分
        片规则 -->
5.     <!-- columns 是表的切分字段 -->
6.     <!-- algorithm 是规则对应的切分规则：映射到 function 的 name -->
7.     <tableRule name="rule1">
8.         <rule>
9.             <columns>id</columns>
10.            <algorithm>func1</algorithm>
11.        </rule>
12.    </tableRule>
13.    <!-- function 配置是分片规则的配置 -->
14.    <!-- name 为切分规则的名称，名字任意取，但是需要与 tableRule 中匹配 -->
15.    <!-- class 是切分规则对应的切分类，写死，需要哪种规则则配置哪种 -->
16.    <!-- property 标签是切分规则对应的不同属性，不同的切分规则配置不同 -->
17.    <function name="func1" class="io.mycat.route.function.PartitionByLong">
18.        <property name="partitionCount">2</property>
19.        <property name="partitionLength">512</property>
20.    </function>

```

21. `</mycat:rule>`

## 1.4 物理数据库配置

- 登录物理数据库 MySQL，创建数据库与表

```
1. drop database if exists db1;
2. create database db1;
3. use db1;
4. CREATE TABLE users (
5.     id INT NOT NULL AUTO_INCREMENT,
6.     name varchar(50) NOT NULL default '',
7.     indate DATETIME NOT NULL default CURRENT_TIMESTAMP,
8.     PRIMARY KEY (id)
9. )AUTO_INCREMENT= 1 ENGINE=InnoDB DEFAULT CHARSET=utf8;
10.
11. drop database if exists db2;
12. create database db2;
13. use db2;
14. CREATE TABLE item (
15.     id INT NOT NULL AUTO_INCREMENT,
16.     value INT NOT NULL default 0,
17.     indate DATETIME NOT NULL default CURRENT_TIMESTAMP,
18.     PRIMARY KEY (id)
19. )AUTO_INCREMENT= 1 ENGINE=InnoDB DEFAULT CHARSET=utf8;
20.
21. drop database if exists db3;
22. create database db3;
23. use db3;
24. CREATE TABLE item (
25.     id INT NOT NULL AUTO_INCREMENT,
26.     value INT NOT NULL default 0,
27.     indate DATETIME NOT NULL default CURRENT_TIMESTAMP,
28.     PRIMARY KEY (id)
29. )AUTO_INCREMENT= 1 ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

## 1.5 启动 Mycat 服务，测试插入数据

- 运行 Mycat

linux:

```
1. ./mycat start # 启动
```

2. `./mycat stop` # 停止
3. `./mycat console` # 前台运行
4. `./mycat restart` # 重启服务
5. `./mycat pause` # 暂停
6. `./mycat status` # 查看启动状态

windows:

直接运行 Mycat 安装目录下/bin/startup\_nowrap.bat, 如果出现闪退, 在 cmd 命令行运行, 查看出错原因。

提示 MyCAT Server startup successfully. 则启动成功。

- 连接 Mycat

1. `.\mysql.exe -uroot -p123456 -h localhost -P8066 -DTESTDB --default-auth=mysql_native_password`

```
PS C:\Program Files\MySQL\MySQL Server 8.0\bin> .\mysql.exe -uroot -p123456 -h localhost -P8066 -DTESTDB --default-auth=mysql_native_password
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.6.29-mycat-1.6.7.1-release-20190213150257 MyCat Server (OpenCloudDB)
Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>
```

- 插入数据

连接 Mycat 后测试插入三条数据：

1. `mysql> use TESTDB;`
2. Database changed
3. `mysql> show tables;`
4. +-----+
5. | Tables in TESTDB |
6. +-----+
7. | item |
8. | users |
9. +-----+
10. 2 rows in set (0.00 sec)
- 11.
12. `mysql> insert into users(name) values('zhangsan');`
13. Query OK, 1 row affected (0.12 sec)

```

14.
15. mysql> insert into item(id, value) values(1, 100);
16. Query OK, 1 row affected (0.09 sec)
17.
18. mysql> insert into item(id, value) values(512, 100);
19. Query OK, 1 row affected (0.07 sec)
20.
21. mysql> explain insert into users(name) values('zhangsan');
22. +-----+-----+
23. | DATA_NODE | SQL |
24. +-----+-----+
25. | dn1 | insert into users(name) values('zhangsan') |
26. +-----+-----+
27. 1 row in set (0.00 sec)
28.
29. mysql> explain insert into item(id, value) values(1, 100);
30. +-----+-----+
31. | DATA_NODE | SQL |
32. +-----+-----+
33. | dn2 | INSERT INTO item (id, value) VALUES ('1', 100) |
34. +-----+-----+
35. 1 row in set (0.00 sec)
36.
37. mysql> explain insert into item(id, value) values(512, 100);
38. +-----+-----+
39. | DATA_NODE | SQL |
40. +-----+-----+
41. | dn3 | INSERT INTO item (id, value) VALUES ('512', 100) |
42. +-----+-----+
43. 1 row in set (0.00 sec)

```

然后登录物理数据库，查看是否插入成功：

```

1. mysql> use db1;
2. Database changed
3. mysql> select * from users;
4. +----+-----+-----+
5. | id | name | indate |
6. +----+-----+-----+
7. | 11 | zhangsan | 2019-04-14 13:44:22 |
8. +----+-----+-----+
9. 1 row in set (0.07 sec)
10.
11. mysql> use db2;

```

```

12. Database changed
13. mysql> select * from item;
14. +-----+-----+-----+
15. | id | value | indate          |
16. +-----+-----+-----+
17. | 1 | 100 | 2019-04-14 13:44:35 |
18. +-----+-----+-----+
19. 1 row in set (0.06 sec)
20.
21. mysql> use db3;
22. Database changed
23. mysql> select * from item;
24. +-----+-----+-----+
25. | id | value | indate          |
26. +-----+-----+-----+
27. | 512 | 100 | 2019-04-14 13:44:49 |
28. +-----+-----+-----+
29. 1 row in set (0.06 sec)

```

插入的 users 表中的数据全部在 db1 中，而 item 表中的数据分布在 db2 和 db3 中。这样就根据实际的路由策略进行了分表。

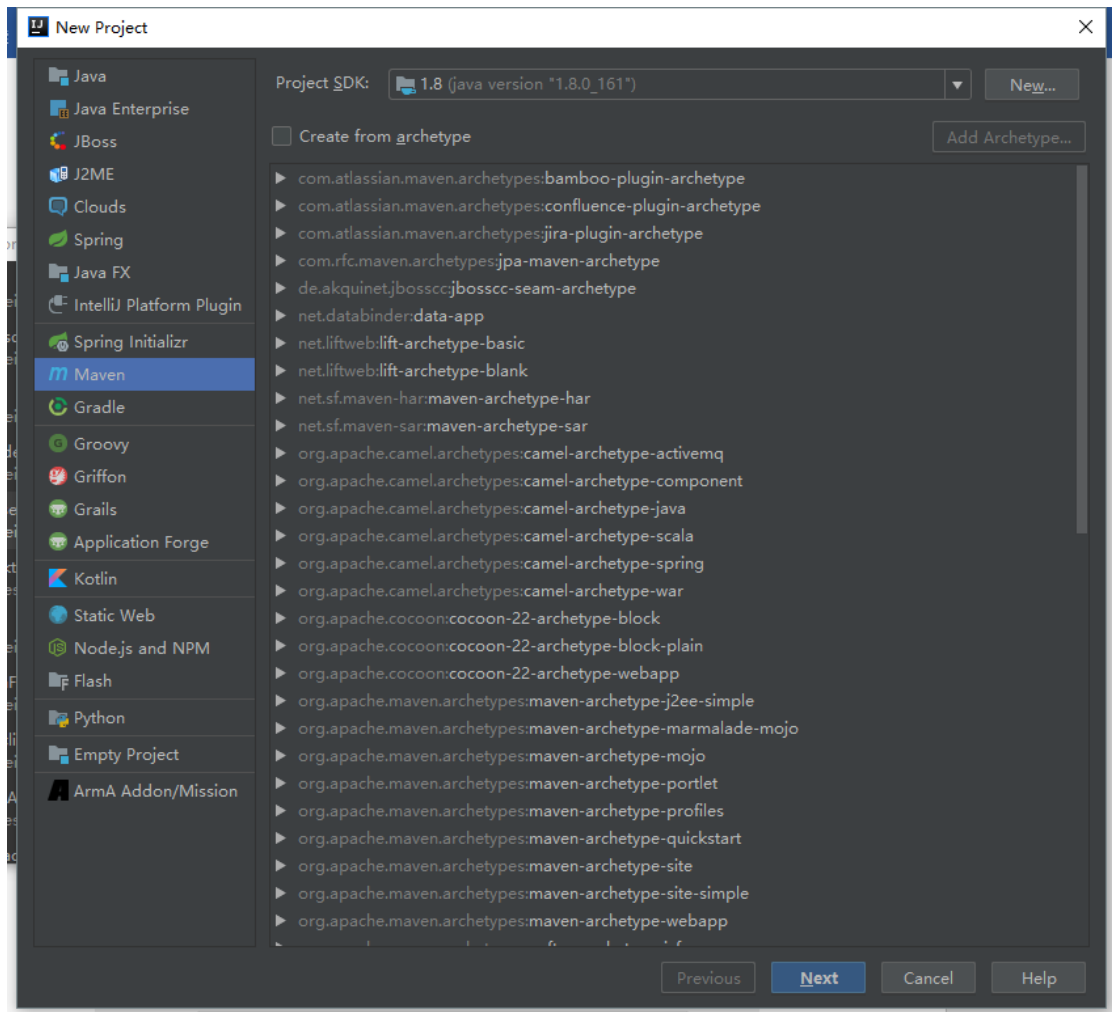
## ➤ 实验 2 Sharding-JDBC 使用

### 2.1 基本开发环境准备

- 安装 IntelliJ IDEA，并注册学生账户以获得免费使用权。
- 安装 jdk1.6 以上（建议使用 1.7 或 1.8）
- 安装 MySQL Server 5.5 及以上

### 2.2 项目创建

- 打开 idea，选择 Create New Project
- 项目配置，选择 Maven，Project SDK 选择 1.8，选择 Next



- 继续项目配置，选择 Next，选择默认项直到 Finish 生成项目。



- 配置依赖，修改 pom.xml，修改后内容如下：

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <project xmlns="http://maven.apache.org/POM/4.0.0"
3.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.
   apache.org/xsd/maven-4.0.0.xsd">
5.     <modelVersion>4.0.0</modelVersion>
6.
7.     <groupId>com.hit</groupId>

```



```

8.     <artifactId>shardingjdbcDemo</artifactId>
9.     <version>1.0-SNAPSHOT</version>
10.
11.     <dependencies>
12.         <dependency>
13.             <groupId>commons-dbcp</groupId>
14.             <artifactId>commons-dbcp</artifactId>
15.             <version>1.4</version>
16.         </dependency>
17.
18.         <dependency>
19.             <groupId>mysql</groupId>
20.             <artifactId>mysql-connector-java</artifactId>
21.             <version>5.1.6</version>
22.         </dependency>
23.
24.         <dependency>
25.             <groupId>io.shardingsphere</groupId>
26.             <artifactId>sharding-jdbc-core</artifactId>
27.             <version>3.1.0</version>
28.         </dependency>
29.     </dependencies>
30. </project>

```

配置好后等待 maven 导入依赖。

## 2.3 物理数据库配置

- 登录物理数据库 MySQL，创建数据库与表

```

1. drop database if exists db1;
2. create database db1;
3. use db1;
4. CREATE TABLE users (
5.     id INT NOT NULL AUTO_INCREMENT,
6.     name varchar(50) NOT NULL default '',
7.     PRIMARY KEY (id)
8. )AUTO_INCREMENT= 1 ENGINE=InnoDB DEFAULT CHARSET=utf8;
9.
10. drop database if exists db2;
11. create database db2;
12. use db2;
13. CREATE TABLE item (
14.     id INT NOT NULL AUTO_INCREMENT,

```

```

15.     value INT NOT NULL default 0,
16.     PRIMARY KEY (id)
17. )AUTO_INCREMENT= 1 ENGINE=InnoDB DEFAULT CHARSET=utf8;
18.
19. drop database if exists db3;
20. create database db3;
21. use db3;
22. CREATE TABLE item (
23.     id INT NOT NULL AUTO_INCREMENT,
24.     value INT NOT NULL default 0,
25.     PRIMARY KEY (id)
26. )AUTO_INCREMENT= 1 ENGINE=InnoDB DEFAULT CHARSET=utf8;

```

## 2.4 程序编写

- 按照以下代码编写文件/src/main/java/Demo.java

```

1. import io.shardingsphere.shardingjdbc.api.yaml.YamlShardingDataSourceFactory
   ;
2.
3. import javax.sql.DataSource;
4. import java.io.File;
5. import java.io.IOException;
6. import java.sql.*;
7.
8. public class Demo {
9.     public static void main(String[] args) throws SQLException, IOException
   {
10.         File conf = new File("./src/main/resources/conf.yml");
11.         DataSource dataSource = YamlShardingDataSourceFactory.createDataSource(
   conf);
12.         Connection conn = dataSource.getConnection();
13.         Statement stmt = conn.createStatement();
14.
15.         System.out.println(stmt.executeUpdate("insert into users(id, name) v
   alues(1, 'zhangsan')"));
16.         System.out.println(stmt.executeUpdate("insert into item(id, value) v
   alues(1, 100)"));
17.         System.out.println(stmt.executeUpdate("insert into item(id, value) v
   alues(2, 200)"));
18.     }
19. }

```

- 按照以下代码编写文件/src/main/resources/conf.yml

按照自己的实际数据库配置填写 url, username, password 属性

```
1. dataSources: # 配置数据源列表,必须是有有效的 jdbc 配置,目前仅支持 MySQL 与 PostgreSQL
2.   db1: !!org.apache.commons.dbcp.BasicDataSource # 数据源名称
3.     driverClassName: com.mysql.jdbc.Driver
4.     url: jdbc:mysql://localhost:3306/db1 # 这里的要求合法的 jdbc 连接串即可,目前尚未兼容 MySQL 8.x
5.     username: root # MySQL 用户名
6.     password: 123456 # MySQL 用户的明文密码
7.   db2: !!org.apache.commons.dbcp.BasicDataSource
8.     driverClassName: com.mysql.jdbc.Driver
9.     url: jdbc:mysql://localhost:3306/db2
10.    username: root
11.    password: 123456
12.   db3: !!org.apache.commons.dbcp.BasicDataSource
13.     driverClassName: com.mysql.jdbc.Driver
14.     url: jdbc:mysql://localhost:3306/db3
15.     username: root
16.     password: 123456
17.
18. shardingRule: # sharding 的配置
19.   tables: # 配置表 sharding 的主要位置
20.     users:
21.       actualDataNodes: db1.users # sharding 表对应的数据源以及物理名称,需要用表达式处理,表示表实际上在哪些数据源存在
22.       item:
23.         actualDataNodes: db${2..3}.item # 表示存在 db2.item, db3.item
24.         databaseStrategy: # sharding 规则
25.           inline:
26.             shardingColumn: id # 列名
27.             algorithmExpression: db${id % 2 + 2} # 例如: id=1 时表示 db3
```

## 2.5 程序运行

- 运行 Demo.main(), 成功后登录物理数据库查看是否插入成功。

```
1. mysql> use db1;
2. Database changed
3. mysql> select * from users;
4. +----+-----+
5. | id | name   |
6. +----+-----+
7. |  1 | zhangsan |
8. +----+-----+
9. 1 row in set (0.00 sec)
10.
11. mysql> use db2;
12. Database changed
13. mysql> select * from item;
14. +----+-----+
15. | id | value |
16. +----+-----+
17. |  2 |   200 |
18. +----+-----+
19. 1 row in set (0.01 sec)
20.
21. mysql> use db3;
22. Database changed
23. mysql> select * from item;
24. +----+-----+
25. | id | value |
26. +----+-----+
27. |  1 |   100 |
28. +----+-----+
29. 1 row in set (0.00 sec)
```