

《基于 Dubbo 的分布式系统架构视频教程》高可用架构篇

MyCat 在 MySQL 主从复制基础上实现读写分离

一、环境

操作系统: CentOS-6.6-x86_64-bin-DVD1.iso

JDK 版本: jdk1.7.0_45

MyCat 版本: Mycat-server-1.4-release-20151019230038-linux.tar.gz

MyCat 节点 IP: 192.168.1.203 主机名: edu-mycat-01 主机配置: 4 核 CPU、4G 内存

MySQL 版本: mysql-5.6.26.tar.gz

主节点 IP: 192.168.1.205 主机名: edu-mysql-01 主机配置: 4 核 CPU、4G 内存

从节点 IP: 192.168.1.206 主机名: edu-mysql-02 主机配置: 4 核 CPU、4G 内存

二、依赖课程

《高可用架构篇--第 13 节--MySQL 源码编译安装 (CentOS-6.6+MySQL-5.6)》

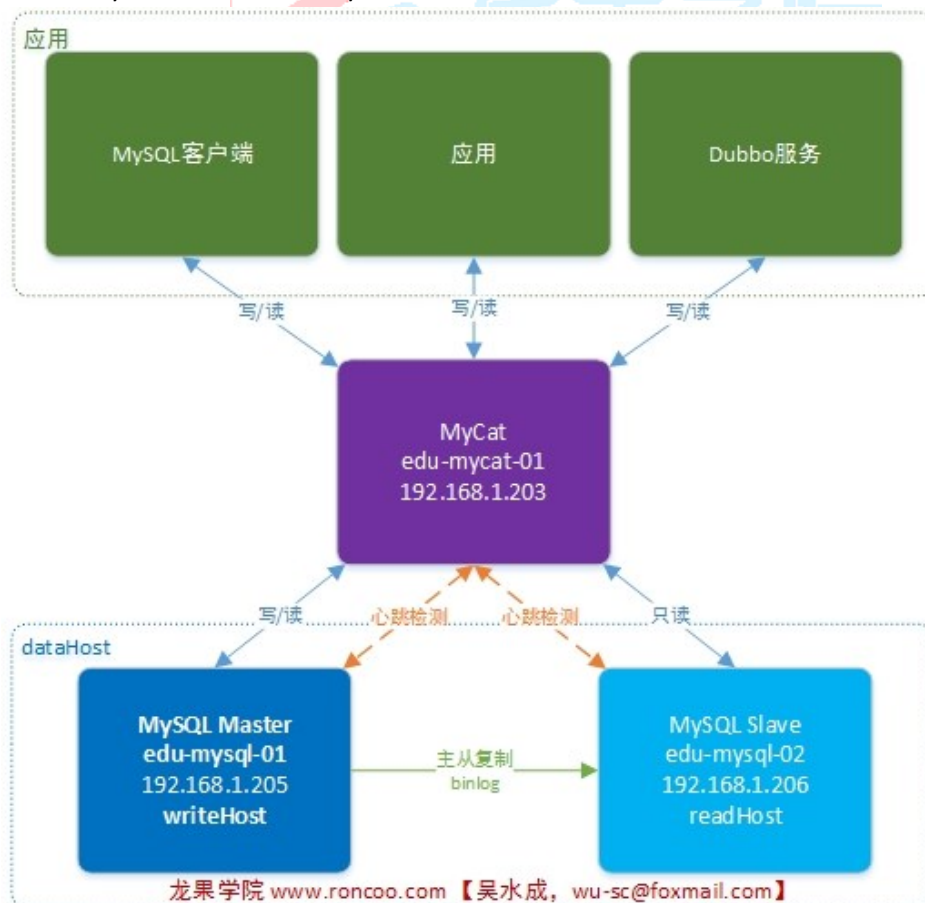
《高可用架构篇--第 14 节--MySQL 主从复制的配置 (CentOS-6.6+MySQL-5.6)》

注意: 上一节课中讲到的 MySQL 主从复制配置, 在用 MyCat 做主从读写分离或其结合实际项目场景应用中, 主从复制配置还需要按实际需求情况进行调整。

(调整后的主从数据库 my.cnf 配置文件, 随视频教程压缩包提供)

三、MyCat 介绍 (MyCat 官网: <http://mycat.org.cn/>)

MyCat 的读写分离是基于后端 MySQL 集群的主从同步来实现的, 而 MyCat 提供语句的分发功能。MyCat1.4 开始支持 MySQL 主从复制状态绑定的读写分离机制, 让读更加安全可靠。



四、MyCat 的安装

1、设置 MyCat 的主机名和 IP 与主机名的映射

```
# vi /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=edu-mycat-01
# vi /etc/hosts
127.0.0.1 edu-mycat-01
192.168.1.203 edu-mycat-01
192.168.1.205 edu-mysql-01
192.168.1.206 edu-mysql-02
```

2、因为 MyCat 是用 Java 开发的, 因此 MyCat 运行需要安装 JDK (准确来说是 JRE 就够了), 并且需要 JDK1.7 或以上版本

```
# vi /etc/profile
## java env
export JAVA_HOME=/usr/local/java/jdk1.7.0_72
export JRE_HOME=$JAVA_HOME/jre
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar:$JRE_HOME/lib/rt.jar
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
# source /etc/profile
# java -version
```

3、创建 mycat 用户并设置密码

```
# useradd mycat
# passwd mycat
```

4、上传安装包 [Mycat-server-1.4-release-20151019230038-linux.tar.gz](#) 到 MyCat 服务器中的 `/home/mycat` 目录, 并解压并移动到 `/usr/local/mycat` 目录

```
$ tar -zxvf Mycat-server-1.4-release-20151019230038-linux.tar.gz
```

```
[mycat@edu-mycat-01 ~]$ pwd
/home/mycat
[mycat@edu-mycat-01 ~]$ ll
total 7968
drwxrwxr-x. 7 mycat mycat 4096 Feb 29 05:36 mycat
-rw-rw-r--. 1 mycat mycat 8152178 Nov 3 00:14 Mycat-server-1.4-release-20151019230038-linux.tar.gz
[mycat@edu-mycat-01 ~]$
```

```
$ su root
```

Password:

```
# mv /home/mycat/mycat /usr/local/
# cd /usr/local/mycat/
# ll
```

```
[root@edu-mycat-01 mycat]# ll
total 24
drwxrwxr-x. 2 mycat mycat 4096 Feb 29 05:36 bin
drwxrwxr-x. 2 mycat mycat 4096 Jun 17 2015 catlet
drwxrwxr-x. 2 mycat mycat 4096 Feb 29 05:36 conf
drwxrwxr-x. 2 mycat mycat 4096 Feb 29 05:36 lib
drwxrwxr-x. 2 mycat mycat 4096 Jun 17 2015 logs
-rwxrwxr-x. 1 mycat mycat 217 Oct 19 23:00 version.txt
[root@edu-mycat-01 mycat]#
```

5、设置 MyCat 的环境变量

```
# vi /etc/profile
## mycat env
export MYCAT_HOME=/usr/local/mycat
export PATH=$PATH:$MYCAT_HOME/bin
# source /etc/profile
```

五、配置 MyCat

1、在配置 MyCat 前, 请确认 MySQL 的主从复制安装配置已完成并正常运行。MySQL 主从数据的同步在 MySQL 中配置, MyCat 不负责数据同步的问题。

补充:

(1) MySQL 主从复制配置中, 如果涉及到函数或存储过程的同步复制, 需要在/etc/my.cnf 中的[mysqld]段中增加配置 `log_bin_trust_function_creators=true` 或在客户端中设置 `set global log_bin_trust_function_creators = 1;`

(2) 如果要做读写分离下的主从切换, 那么从节点也有可能会变为写节点, 因此从节点就不能设置为只读 `read_only=1` 。

(3) Linux 版本的 MySQL, 需要设置为 MySQL 大小写不敏感, 否则可能会发生找不到表的问题。可在/etc/my.cnf 的[mysqld]段中增加 `lower_case_table_names=1` 。

2、配置 MyCat 的 schema.xml

schema.xml 是 MyCat 最重要的配置文件之一, 用于设置 MyCat 的逻辑库、表、数据节点、dataHost 等内容,

```
[mycat@edu-mycat-01 conf]$ cd /usr/local/mycat/conf/
[mycat@edu-mycat-01 conf]$ vi schema.xml
```

```
<?xml version="1.0"?>
<!DOCTYPE mycat:schema SYSTEM "schema.dtd">
<mycat:schema xmlns:mycat="http://org.opencloudb/">
  <!-- 定义MyCat的逻辑库, 逻辑库的概念与MySQL中的 database 概念相同 -->
  <!-- schema name="rc_schema1" checkSQLSchema="false" sqlMaxLimit="100" dataNode="rc_dn1"></schema -->
  <!-- schema name="pay_schema1" checkSQLSchema="false" sqlMaxLimit="100" dataNode="pay_dn1"></schema -->
  <schema name="rc_schema2" checkSQLSchema="false" sqlMaxLimit="100" dataNode="rc_dn2"></schema>
  <schema name="pay_schema2" checkSQLSchema="false" sqlMaxLimit="100" dataNode="pay_dn2"></schema>
  <!-- 其中checkSQLSchema表明是否检查并过滤SQL中包含schema的情况, 如逻辑库为 TESTDB, 则可能写为select * from
  TESTDB.edu_user, 此时会自动过滤TESTDB, SQL变为select * from edu_user, 若不会出现上述写法, 则可以关闭属性为false -->
  <!-- sqlMaxLimit默认返回的最大记录数限制, MyCat1.4版本里面, 用户的Limit参数会覆盖掉MyCat的sqlMaxLimit默认设置-->

  <!-- 定义MyCat的数据节点 -->
  <!-- dataNode name="rc_dn1" dataHost="dtHost1" database="roncoo" / -->
  <!-- dataNode name="pay_dn1" dataHost="dtHost1" database="edu_simple_pay" / -->
  <dataNode name="rc_dn2" dataHost="dtHost2" database="roncoo" />
  <dataNode name="pay_dn2" dataHost="dtHost2" database="edu_simple_pay" />
  <!-- dataNode 中的 name 数据表示节点名称, dataHost表示数据主机名称, database表示该节点要路由的数据库的名称 -->
```

```
<!-- dataHost配置的是实际的后端数据库集群（当然，也可以是非集群） -->
<!-- 注意：schema中的每一个dataHost中的host属性值必须唯一，否则会出现主从在所有dataHost中全部切换的现象 -->
<!-- 定义数据主机dtHost1，只连接到MySQL读写分离集群中的Master节点，不使用MyCat托管MySQL主从切换 -->
<!--
<dataHost name="dtHost1" maxCon="500" minCon="20" balance="0"

    writeType="0" dbType="mysql" dbDriver="native" switchType="1" slaveThreshold="100">

        <heartbeat>select user()</heartbeat>

        <writeHost host="hostM1" url="192.168.1.205:3306" user="root" password="www.roncoo.com" />
    </dataHost>
-->

<!-- 使用MyCat托管MySQL主从切换 -->
<!-- 定义数据主机dtHost2，连接到MySQL读写分离集群，并配置了读写分离和主从切换 -->
<dataHost name="dtHost2" maxCon="500" minCon="20" balance="1"

    writeType="0" dbType="mysql" dbDriver="native" switchType="2" slaveThreshold="100">

        <!-- 通过show slave status检测主从状态，当主宕机以后，发生切换，从变为主，原来的主变为从，这时候show slave
            status就会发生错误，因为原来的主没有开启slave，不建议直接使用switch操作，而是在DB中做主从对调。 -->

        <heartbeat>show slave status</heartbeat>

        <!-- can have multi write hosts -->

        <writeHost host="hostM2" url="192.168.1.205:3306" user="root" password="www.roncoo.com" />

        <writeHost host="hostS2" url="192.168.1.206:3306" user="root" password="www.roncoo.com" />
    </dataHost>

    <!-- 参数balance决定了哪些MySQL服务器参与到读SQL的负载均衡中 -->
    <!-- balance="0"，为不开启读写分离，所有读操作都发送到当前可用的writeHost上-->
    <!-- balance="1"，全部的readHost与stand by writeHost参与select语句的负载均衡-->
    <!-- balance="2"，所有读操作都随机的在writeHost、readHost上分发-->
    <!-- MyCat1.4版本中，若想支持MySQL一主一从的标准配置，并且在主节点宕机的情况下，从节点还能读取数据，则需要在MyCat里
        配置为两个writeHost并设置balance="1" -->

    <!-- writeType="0"，所有写操作都发送到可用的writeHost上 -->
    <!-- writeType="1"，仅仅对于galera for mysql集群这种多主多节点都能写入的集群起效，此时Mycat会随机选择一个
        writeHost并写入数据，对于非galera for mysql集群，请不要配置writeType=1，会导致数据库不一致的严重问题 -->
</mycat:schema>
```

MyCat1.4 开始支持 MySQL 主从复制状态绑定的读写分离机制，让读更加安全可靠，配置如下：

MyCat 心跳检查语句配置为 show slave status，dataHost 上定义两个新属性：switchType="2" 与 slaveThreshold="100"，此时意味着开启 MySQL 主从复制状态绑定的读写分离与切换机制，MyCat 心跳机制通过检测 show slave status 中的 "Seconds_Behind_Master"，"Slave_IO_Running"，"Slave_SQL_Running" 三个字段来确定当前主从同步的状态以及 Seconds_Behind_Master 主从复制时延，当 Seconds_Behind_Master 大于 slaveThreshold 时，读写分离筛选器会过滤掉此 Slave 机器，防止读到很久之前的旧数据，而当主节点宕机后，切换逻辑会检查 Slave 上的 Seconds_Behind_Master 是否为 0，为 0 则表示主从同步，可以安全切换，否则不会切换。

3、配置 server.xml

server.xml 主要用于设置系统变量、管理用户、设置用户权限等。

[wusc@edu-mycat-01 conf]\$ vi server.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mycat:server SYSTEM "server.dtd">
```

```
<mycat:server xmlns:mycat="http://org.opencloudb/">

  <system>

    <property name="defaultSqlParser">druidparser</property>

    <property name="charset">utf8mb4</property>

    <!-- <property name="useCompression">1</property>-->

    <!--1为开启mysql压缩协议-->

    <!-- <property name="processorBufferChunk">40960</property> -->

    <!--

    <property name="processors">1</property>

    <property name="processorExecutor">32</property>

    -->

    <!--默认是65535 64K 用于sql解析时最大文本长度 -->

    <!--<property name="maxStringLiteralLength">65535</property>-->

    <!--<property name="sequenceHandlerType">0</property>-->

    <!--<property name="backSocketNoDelay">1</property>-->

    <!--<property name="frontSocketNoDelay">1</property>-->

    <!--<property name="processorExecutor">16</property>-->

    <!-- <property name="mutiNodeLimitType">1</property> 0: 开启小数量级（默认）； 1: 开启亿级数据排序

        <property name="mutiNodePatchSize">100</property> 亿级数量排序批量

        <property name="processors">32</property> <property name="processorExecutor">32</property>

        <property name="serverPort">8066</property> <property name="managerPort">9066</property>

        <property name="idleTimeout">300000</property> <property name="bindIp">0.0.0.0</property>

        <property name="frontWriteQueueSize">4096</property>

        <property name="processors">32</property>

    -->

  </system>

  <!-- 用户1，对应的MyCat逻辑库连接到的数据节点对应的主机为MySQL主从复制配置中的Master节点，没实现读写分离，读写都在该
  Master节点中进行 -->

  <!--

  <user name="user1">

    <property name="password">roncoo.1</property>

    <property name="schemas">rc_schema1,pay_schema1</property>

  </user>

  -->

  <!-- 用户2，对应的MyCat逻辑库连接到的数据节点对应的主机为主从复制集群，并通过MyCat实现了读写分离 -->

  <user name="user2">

    <property name="password">roncoo.2</property>

    <property name="schemas">rc_schema2,pay_schema2</property>

  </user>

  <!-- 用户3，只读权限-->

  <user name="user3">

    <property name="password">roncoo.3</property>

    <property name="schemas">rc_schema2,pay_schema2</property>

    <property name="readOnly">true</property>

  </user>
```

```
</user>
</mycat:server>
```

4、防火墙中打开 8066 和 9066 端口

MyCat 的默认数据端口为 8066, mycat 通过这个端口接收数据库客户端的访问请求。
管理端口为 9066, 用来接收 mycat 监控命令、查询 mycat 运行状况、重新加载配置文件等。

```
[root@edu-mycat-01 mycat]# vi /etc/sysconfig/iptables
```

增加:

```
## MyCat
```

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 8066 -j ACCEPT
```

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 9066 -j ACCEPT
```

重启防火墙:

```
[root@edu-mycat-01 mycat]# service iptables restart
```

5、修改 log 日志级别为 debug, 以便通过日志确认基于 MyCat 的 MySQL 数据库集群读写分离的数据操作状态 (可以在正式上生产前改成 info 级别)

```
[mycat@edu-mycat-01 conf]$ vi /usr/local/mycat/conf/log4j.xml
```

```
<root>
  <level value="debug" />
  <appender-ref ref="FILE" />
  <!--<appender-ref ref="FILE" />-->
</root>
```

龙果学院 <http://www.roncoo.com>

6、启动 MyCat

```
[mycat@edu-mycat-01 bin]$ cd /usr/local/mycat/bin/
```

(1) 控制台启动, 这种启动方式在控制台关闭后, MyCat 服务也将关闭, 适合调试使用:

```
[mycat@edu-mycat-01 bin]$ ./mycat console
```

(2) 可以采用以下后台启动的方式:

```
[mycat@edu-mycat-01 bin]$ ./mycat start
```

Starting Mycat-server...

(对应的, 重启: `mycat restart` , 关闭: `mycat stop`)

7、MyCat 连接测试

(1) 如果本地 Windows 安装有 MySQL, 可以使用已有的 mysql 客户端远程操作 MyCat



```
管理员: 命令提示符 - mysql -uuser2 -proncoo.2 -h192.168.1.203 -P8066
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>E:
E:\>cd MySQL-5.6.17-winx64\bin
E:\MySQL-5.6.17-winx64\bin>mysql -uuser2 -proncoo.2 -h192.168.1.203 -P8066
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: 5.5.8-mycat-1.4-release-20151019230038 MyCat Server (OpenCloudDB)

Copyright (c) 2000, 2014, 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>
mysql> show databases;
+-----+
| DATABASE |
+-----+
| pay_schema2 |
| rc_schema2 |
+-----+
2 rows in set (0.00 sec)

mysql> use rc_schema2
Database changed
mysql> show tables;
+-----+
| Tables_in_rcschema2 |
+-----+
| edu_user |
+-----+
1 row in set (0.03 sec)

mysql>
```

使用user2这个用户登录MyCat

查看user2用户拥有的MyCat逻辑库

显示rc_schema2 逻辑库下的表

龙果学院 <http://www.roncoo.com>

```
管理员: 命令提示符 - mysql -uuser2 -proncoo.2 -h192.168.1.203 -P8066
mysql> select * from edu_user;
+-----+
| Id | userName | pwd |
+-----+
| 1 | 吴水成 | 123456 |
| 2 | 清风 | 123456 |
| 3 | 龙果 | roncoo.com |
+-----+
3 rows in set (0.01 sec)

mysql>
```

(2) 如果为了方便，需要在 MyCat 主机上对 MyCat 进行操作（把 MyCat 当是本地 MySQL 来操作），可以在 MyCat 节点主机上安装 MySQL 客户端：

```
[mycat@edu-mycat-01 bin]$ su root
```

```
[root@edu-mycat-01 bin]# yum install mysql
```

使用安装好的 mysql 客户端登录 MyCat

```
[mycat@edu-mycat-01 bin]$ mysql -uuser2 -proncoo.2 -h192.168.1.203 -P8066
```



```
[mysql@edu-mycat-01 bin]$ mysql -uuser2 -pproncoo.2 -h192.168.1.203 -P8066
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.5.8-mycat-1.4-release-20151019230038 MyCat Server (OpenCloudDB)

Copyright (c) 2000, 2013, 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> show databases;
+-----+
| DATABASE |
+-----+
| pay_schema2 |
| rc_schema2 |
+-----+
2 rows in set (0.00 sec)

mysql>
mysql> use rc_schema2
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>
mysql> show tables;
+-----+
| Tables_in_rc_schema2 |
+-----+
| edu_user |
+-----+
1 row in set (0.00 sec)

mysql>
```

龙果学院 <http://www.roncoo.com>

```
mysql> select * from edu_user;
+----+-----+-----+
| Id | userName | pwd |
+----+-----+-----+
| 1  | ???      | 123456 |
| 2  | ??       | 123456 |
| 3  | ??       | roncoo.com |
+----+-----+-----+
3 rows in set (0.00 sec)
```

如果使用MyCat主机上安装了MySQL客户端进行查询出现乱码，则需要设置客户端的编码，在 /etc/my.cnf 中的[client]中设置客户端默认编码为utf8

龙果学院 <http://www.roncoo.com>

```
[root@edu-mycat-01 mycat]# vi /etc/my.cnf
```

增加:

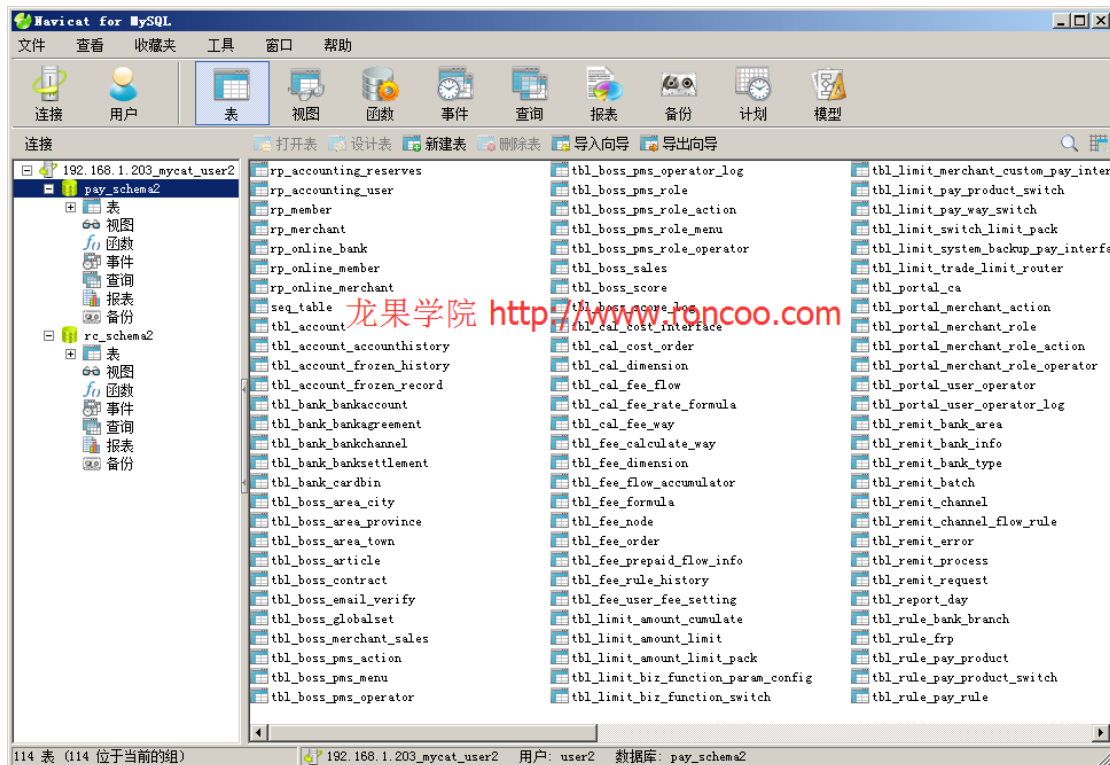
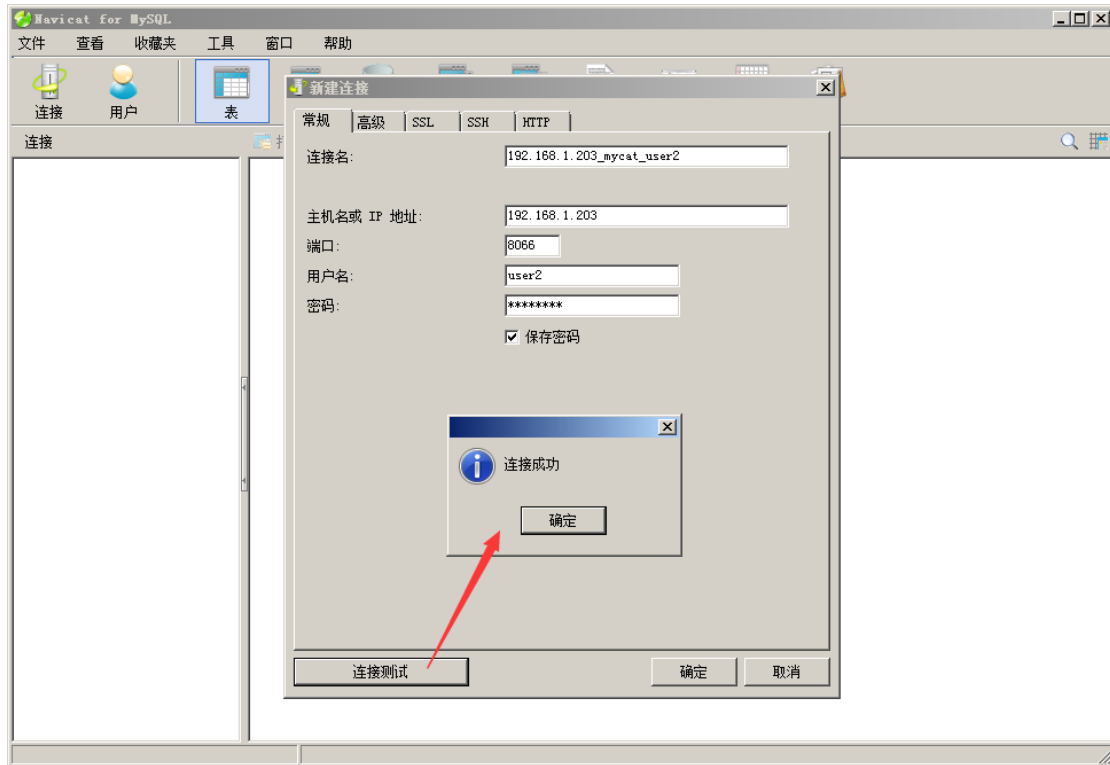
```
[client]
```

```
default-character-set=utf8
```

保存后再查询，乱码问题解决，如下:

```
mysql> select * from edu_user;
+----+-----+-----+
| Id | userName | pwd |
+----+-----+-----+
| 1  | 吴水成   | 123456 |
| 2  | 清风     | 123456 |
| 3  | 龙果     | roncoo.com |
+----+-----+-----+
3 rows in set (0.00 sec)
```

(3) 使用第三方 MySQL 管理客户端连接 MyCat 测试 (navicat 支持, MySQL-Front 兼容性不太好), 以 navicat 为例:



8、读写分离测试

(1) 监听 MyCat 日志

```
[mycat@edu-mycat-01 ~]$ cd /usr/local/mycat/logs/
```

```
[mycat@edu-mycat-01 logs]$ tail -f mycat.log
```

(2) 读测试

```
$ mysql -uuser2 -proncoo.2 -h192.168.1.203 -P8066
```

```
mysql> show databases;
```

```
mysql> show databases;
+-----+
| DATABASE |
+-----+
| pay_schema2 |
| rc_schema2 |
+-----+
2 rows in set (0.01 sec)
```

```
mysql> use rc_schema2;
```

```
mysql> use rc_schema2;
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>
```

龙果学院 <http://www.roncoo.com>

```
mysql> show tables;
```

```
mysql> show tables;
+-----+
| Tables_in_rc_schema2 |
+-----+
| edu_user |
+-----+
1 row in set (0.00 sec)

mysql>
```

```
mysql> select * from edu_user;
```

```
mysql> select * from edu_user;
+----+-----+-----+
| Id | userName | pwd |
+----+-----+-----+
| 1 | 吴水成 | 123456 |
| 2 | 清风 | 123456 |
| 3 | 龙果 | roncoo.com |
+----+-----+-----+
3 rows in set (0.00 sec)
```

执行上面的查询语句, 此时对应的 MyCat 日志信息如下:

```
03/02 05:19:20.426 DEBUG [$NIOREACTOR-0-RW] (ServerQueryHandler.java:56) -ServerConnection [id=3, schema=rc_schema2, host=192.168.1.203,
user=user2, txIsolation=3, autocommit=true, schema=rc_schema2]select * from edu_user
03/02 05:19:20.426 DEBUG [$NIOREACTOR-0-RW] (ZooKeeperCache.java:96) -ZooKeeperCache miss cache, key:rc_schema2select * from edu_user
03/02 05:19:20.426 DEBUG [$NIOREACTOR-0-RW] (NonBlockingSession.java:113) -ServerConnection [id=3, schema=rc_schema2, host=192.168.1.203,
user=user2, txIsolation=3, autocommit=true, schema=rc_schema2]select * from edu_user, route={
  1 -> rc_dn2(select * from edu_user)
} rrs
03/02 05:19:20.426 DEBUG [$NIOREACTOR-0-RW] (PhysicalDBPool.java:431) -select read source hostS2 for dataHost:dtHost2
03/02 05:19:20.427 DEBUG [$NIOREACTOR-0-RW] (NonBlockingSession.java:230) -release connection MySQLConnection [id=4, lastTime=14568671604
10, schema=roncoo, old schema=roncoo, borrowed=true, fromSlaveDB=false, threadId=63, charset=utf8, txIsolation=3, autocommit=true, attachmen
t=rc_dn2(select * from edu_user), respHandler=SingleNodeHandler [node=rc_dn2(select * from edu_user), packetId=8], host=192.168.1.206, port
=3306, statusSync=null, writeQueue=0, modifiedSQLExecuted=false]
03/02 05:19:20.427 DEBUG [$NIOREACTOR-0-RW] (PhysicalDatasource.java:403) -release channel MySQLConnection [id=4, lastTime=1456867160410,
schema=roncoo, old schema=roncoo, borrowed=true, fromSlaveDB=false, threadId=63, charset=utf8, txIsolation=3, autocommit=true, attachment=n
ull, respHandler=null, host=192.168.1.206, port=3306, statusSync=null, writeQueue=0, modifiedSQLExecuted=false]
```

多次执行 `select * from edu_user` 语句, MyCat 打印出来的日志信息显示读操作请求都是路由到 Slave 节点 (192.168.1.206)。

(2) 写测试

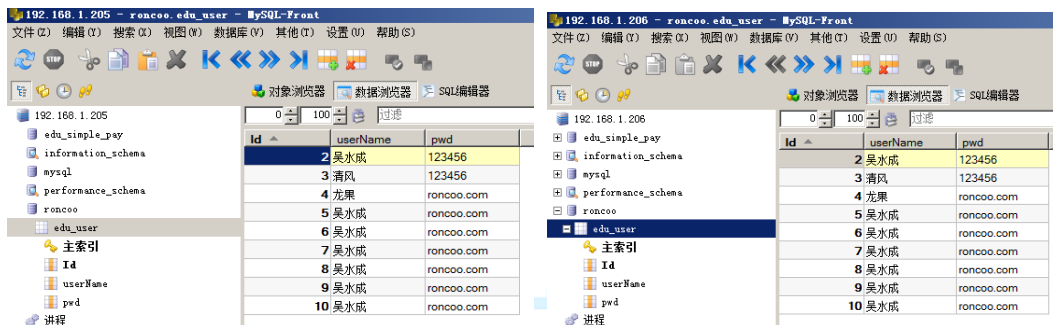
```
mysql> insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com');
```

```
mysql> insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com');  
Query OK, 1 row affected (0.01 sec)
```

执行上面的新增插入语句后，此时对应的 MyCat 日志信息如下：

```
03/02 06:17:43.685 DEBUG [$NIOREACTOR-2-RW] (ServerQueryHandler.java:56) -ServerConnection [id=1, schema=rc_schema2, host=192.168.1.203, user=user2, txIsolation=3, autocommit=true, schema=rc_schema2]insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com')  
03/02 06:17:43.685 DEBUG [$NIOREACTOR-2-RW] (NonBlockingSession.java:113) -ServerConnection [id=1, schema=rc_schema2, host=192.168.1.203, user=user2, txIsolation=3, autocommit=true, schema=rc_schema2]insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com'), route={  
  1 -> rc_dn2(insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com'))  
}  
} rrs  
03/02 06:17:43.685 DEBUG [$NIOREACTOR-2-RW] (MySQLConnection.java:442) -con need syn ,total syn cmd 1 commands SET SESSION TRANSACTION ISOLATION LEVEL REPEATABLE READ;schema change:false con:MySQLConnection [id=9, lastTime=1456870663685, schema=roncoo, old schema=roncoo, borrowed=true, fromSlaveDB=false, threadId=112, charset=utf8, txIsolation=0, autocommit=true, attachment=rc_dn2(insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com'))], respHandler=SingleNodeHandler [node=rc_dn2(insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com'))], packetId=0], host=192.168.1.205, port=3306, statusSync=null, writeQueue=0, modifiedSQLExecuted=true]  
03/02 06:17:43.694 DEBUG [$NIOREACTOR-1-RW] (NonBlockingSession.java:230) -release connection MySQLConnection [id=9, lastTime=1456870663674, schema=roncoo, old schema=roncoo, borrowed=true, fromSlaveDB=false, threadId=112, charset=utf8, txIsolation=3, autocommit=true, attachment=rc_dn2(insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com'))], respHandler=SingleNodeHandler [node=rc_dn2(insert into edu_user (userName, pwd) values('吴水成', 'roncoo.com'))], packetId=0], host=192.168.1.205, port=3306, statusSync=null, writeQueue=0, modifiedSQLExecuted=true]  
03/02 06:17:43.694 DEBUG [$NIOREACTOR-1-RW] (PhysicalDataSource.java:403) -release channel MySQLConnection [id=9, lastTime=1456870663674, schema=roncoo, old schema=roncoo, borrowed=true, fromSlaveDB=false, threadId=112, charset=utf8, txIsolation=3, autocommit=true, attachment=null, respHandler=null, host=192.168.1.205, port=3306, statusSync=null, writeQueue=0, modifiedSQLExecuted=false]
```

多次执行以上插入语句，发现新增数据都是从 Master 节点（192.168.1.205）插进去的，并且 Slave 节点通过 Binlog 同步了 Master 节点中的数据。



Id	userName	pwd
2	吴水成	123456
3	清风	123456
4	龙果	roncoo.com
5	吴水成	roncoo.com
6	吴水成	roncoo.com
7	吴水成	roncoo.com
8	吴水成	roncoo.com
9	吴水成	roncoo.com
10	吴水成	roncoo.com

综上，基于 MyCat 的读写分离集群配置成功。

接下来计划课程：

MyCat 读写分离集群的主从容错（切换）、恢复；

MyCat 的高可用集群 HAProxy + Keepalived + MyCat;

敬请关注！

龙果学院 <http://www.roncoo.com>