



链滴

# MyCAT+ 双主一从高可用读写分离

作者: [GeekBoyDqz](#)

原文链接: <https://ld246.com/article/1570977558594>

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MySQL双主一从架构

## 1、环境准备

### 1.1: 准备3台服务器进行搭建

主机 版本 IP地址 关系

MySQL-M1 MySQL5.7 192.168.66.37 M1-S1主从

MySQL-S1 MySQL5.7 192.168.66.39 M1-S1主从

MySQL-M2 MySQL5.7 192.168.66.38 M1-M2互为主从

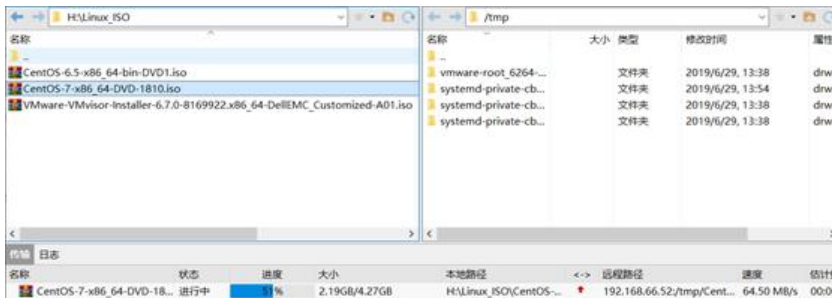
### 1.2: 系统环境配置

- 3台服务器进行以下操作
- 配置本地yum源

在内网环境下由于无法连接到互联网所有无法直接使用互联网yum源

#### 1、本地源

① 拷贝镜像到服务器/tmp目录下



② 创建挂载目录/mnt/yum-iso, 修改/etc/fstab文件

```
[root@yum-server ~]# mkdir /mnt/yum-iso  
[root@yum-server ~]# vim /etc/fstab
```

```
#  
# /etc/fstab  
# Created by anaconda on Sun Apr 21 01:15:17 2019  
#  
# Accessible filesystems, by reference, are maintained under '/dev/disk'  
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info  
#  
/dev/mapper/centos-root / xfs defaults 0 0  
UUID=ab1ca96d-18e1-4963-8ddb-6a7afeb2e80e /boot xfs defaults  
0 0  
/dev/mapper/centos-swap swap swap defaults 0 0  
/tmp/CentOS-7-x86_64-DVD-1810.iso /mnt/yum-iso iso9660 loop 0 0
```

```
[root@yum-server ~]# mount -a
```

mount: /dev/loop0 写保护, 将以只读方式挂载

```

[root@yum-server ~]# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                  8:0    0   20G  0 disk
├─sda1                8:1    0    1G  0 part /boot
├─sda2                8:2    0   19G  0 part
│   └─centos-root     253:0   0   17G  0 lvm  /
│   └─centos-swap     253:1   0    2G  0 lvm  [SWAP]
sr0                  11:0    1   4.3G  0 rom
loop0                7:0    0   4.3G  0 loop /mnt/yum-iso
[root@yum-server ~]# cd /mnt/yum-iso/
[root@yum-server yum-iso]# ls
CentOS_BuildTag  GPL          LiveOS        RPM-GPG-KEY-CentOS-7
EFI              images       Packages      RPM-GPG-KEY-CentOS-Testing-7
EULA             isolinux    repodata      TRANS.TBL

```

### ③ 创建yum源配置文件并进行配置

```

[root@yum-server ~]# cd /etc/yum.repos.d/
[root@yum-server yum.repos.d]# vim yum-iso.repo
[yum-iso]
name=yum-iso-local
baseurl=file:///mnt/yum-iso/
gpgcheck=0
enable=1

```

### ④ 备份其他默认源使其不进行使用

```

[root@yum-server yum.repos.d]# mkdir bak
[root@yum-server yum.repos.d]# mv CentOS-* bak/

```

### ⑤ 检查本地源是否正常

```

[root@yum-server ~]# yum repolist

```

## 2、网络源

### ① 通过ftp服务让其他客户端可以进行访问

```

[root@yum-server ~]# yum -y install vsftpd*

```

```

[root@yum-server ~]# systemctl enable vsftpd
[root@yum-server ~]# systemctl start vsftpd
[root@yum-server ~]# systemctl status vsftpd

```

### ② 将挂载本地源存储目录拷贝至/var/ftp/pub目录下

### ③ 通过浏览器访问

<ftp://192.168.66.52>

## 3、通过reposync命令将互联网源的包下载到本地

- 注意：需要能连接互联网
- 安装reposync命令，该命令包含在yum-utils包中

```

[root@yum-server ~]# yum install yum-utils
[root@yum-server ~]# reposync -r base -p /var/ftp/pub/
base为仓库标识 -p指定将下载的rpm包存储在本地的路径

```

```
# 局域网中的客户端配置本地yum源并指定访问路径为局域网yum服务器
[root@yum-server ~]# vim /etc/yum.repos.d/yum-c7.repo
[yum-c7]
name=yum-c7
baseurl=ftp://yum源服务器地址/
enable=1
gpgcache=0
```

- 安装常用工具已经依赖包

```
[root@localhost ~]# yum install vim wget lrzsz net-tools httpd-tools make gcc-c++ cmake bi
on-devel ncurses-devel
```

- 配置hosts

```
192.168.66.36 MyCAT
192.168.66.37 MySQL-M1
192.168.66.38 MySQL-M2
192.168.66.39 MySQL-S1
```

- 关闭防火墙

```
~]# systemctl stop firewalld
~]# systemctl disable firewalld
~]# sed -ri 's/SELINUX=enforcing/SELINUX=disabled/' /etc/selinux/config
```

- 时间同步

```
~]# yum -y install ntp
~]# ntpdate 0.asia.pool.ntp.org //内网环境需要与内网中时间服务器进行同步
~]# hwclock -w //将时间同步到硬件时间
```

## 1.3: 上传源码包

- 创建一个存储源码包的目录，3台数据库服务器都需要上传

```
[root@localhost ~]# mkdir /soft
[root@localhost soft]# ls
mysql-5.7.26.tar.gz
```

## 2、编译安装MySQL

- 三台MySQL服务器都需要执行以下的步骤

### 2.1: 解压源码包

```
[root@localhost soft]# tar -zxvf mysql-5.7.26.tar.gz
```

### 2.2: 安装依赖

- 建议先卸载自带的mariadb

```
[root@localhost ~]# rpm -qa | grep mariadb
mariadb-libs-5.5.60-1.el7_5.x86_64
```

```
[root@localhost ~]# rpm -e mariadb-libs-5.5.60-1.el7_5.x86_64 --nodeps
警告: /etc/my.cnf 已另存为 /etc/my.cnf.rpmsave #my.cnf另存为/etc/my.cnf.rpmsave
```

注1: MySQL从5.5开始,源代码安装将原来的configure改为cmake,因此在安装MySQL5.5.x以上版本,需要先安装cmake,通过yum安装

注2: MySQL5.7.5以上版本源码安装必须要BOOST库,不然编译会出现如下图提示:且版本必须为1.9的

```
CMake Error at cmake/boost.cmake:81 (MESSAGE):
You can download it with -DDOWNLOAD_BOOST=1 -DWITH_BOOST=<directory>

This CMake script will look for boost in <directory>. If it is not there,
it will download and unpack it (in that directory) for you.

If you are inside a firewall, you may need to use an http proxy:

export http_proxy=http://example.com:80

Call Stack (most recent call first):
cmake/boost.cmake:238 (COULD_NOT_FIND_BOOST)
CMakeLists.txt:507 (INCLUDE)
```

- 安装cmake, 如果上面环境准备中已经安装了cmake则不需要执行下面cmake的安装

```
[root@localhost ~]# yum -y install cmake
```

- 安装boost库

1、创建boost库目录, 上传boost包, 然后安装

Boost库网站<https://sourceforge.net/projects/boost/files/boost/>

```
[root@localhost ~]# mkdir /usr/local/boost
```

2、下载并解压, 无法连接外网情况下需要下载至本地然后拷贝至服务器

```
[root@localhost ~]# cd /usr/local/boost/
```

```
[root@localhost boost]# wget http://www.sourceforge.net/projects/boost/files/boost/1.59.0/boost_1_59_0.tar.gz
```

3、将boost解压出来即可

```
[root@localhost boost]# tar -zxvf boost_1_59_0.tar.gz
```

## 2.3: 编译

- 创建安装目录与数据存储目录

```
[root@localhost ~]# mkdir -p /usr/local/mysql/data
```

```
[root@localhost ~]# mkdir /usr/local/mysql/logs #日志存储
```

- 进入mysql解压目录进行编译

```
[root@localhost ~]# cd /soft/mysql-5.7.26
```

```
[root@localhost mysql-5.7.26]# cmake . -DCMAKE_INSTALL_PREFIX=/usr/local/mysql -DMYSQL_DATADIR=/usr/local/mysql/data -DDOWNLOAD_BOOST=1 -DWITH_BOOST=/usr/local/boost -DSYSCONFDIR=/etc -DWITH_INNOBASE_STORAGE_ENGINE=1 -DWITH_PARTITION_STORAGE_ENGINE=1 -DWITH_FEDERATED_STORAGE_ENGINE=1 -DWITH_BLACKHOLE_STORAGE_E
```

```
GINE=1 -DWITH_MYISAM_STORAGE_ENGINE=1 -DENABLED_LOCAL_INFILE=1 -DENABLE_DT  
ACE=0 -DDEFAULT_CHARSET=utf8 -DDEFAULT_COLLATION=utf8_general_ci -DWITH_EMBE  
DED_SERVER=1
```

- 参数解释:

```
-DCMAKE_INSTALL_PREFIX=/usr/local/mysql # mysql安装路径  
-DMYSQL_DATADIR=/usr/local/mysql/data # mysql数据存储路径  
-DWITH_BOOST=/usr/local/boost # 使用本地boost库  
-DENABLED_LOCAL_INFILE=1 # 允许从本文件导入数据  
-DDEFAULT_CHARSET=utf8 \ # 指定字符集  
-DDEFAULT_COLLATION=utf8_general_ci \
```

- 注: 如果编译失败, 需要删除MySQL源码包解压路径下的CMakeCache.txt文件, 然后再次进行编译

## 2.4: 安装

```
[root@localhost mysql-5.7.26]# make && make install
```

- 安装完成

```
-- Installing: /usr/local/mysql/mysql-test/./t/wl6711_heap_to_disk.test  
-- Installing: /usr/local/mysql/mysql-test/./t/wl6978.test  
-- Installing: /usr/local/mysql/mysql-test/./t/xa.test  
-- Installing: /usr/local/mysql/mysql-test/./t/xa_debug.test  
-- Installing: /usr/local/mysql/mysql-test/./t/xa_gtid-master.opt  
-- Installing: /usr/local/mysql/mysql-test/./t/xa_gtid.test  
-- Installing: /usr/local/mysql/mysql-test/./t/xa_prepared_binlog_off-  
-- Installing: /usr/local/mysql/mysql-test/./t/xa_prepared_binlog_off-  
-- Installing: /usr/local/mysql/mysql-test/./t/xml.test  
-- Installing: /usr/local/mysql/mysql-test/./valgrind.supp  
-- Installing: /usr/local/mysql/mysql-test/./mtr  
-- Installing: /usr/local/mysql/mysql-test/./mysql-test-run  
-- Installing: /usr/local/mysql/mysql-test/./Makefile  
-- Installing: /usr/local/mysql/mysql-test/./cmake_install.cmake  
-- Installing: /usr/local/mysql/mysql-test/./CTestTestfile.cmake  
-- Installing: /usr/local/mysql/./COPYING-test  
-- Installing: /usr/local/mysql/./README-test  
-- Up-to-date: /usr/local/mysql/mysql-test/mtr  
-- Up-to-date: /usr/local/mysql/mysql-test/mysql-test-run  
-- Installing: /usr/local/mysql/mysql-test/lib/My/SafeProcess/my_safe-  
-- Up-to-date: /usr/local/mysql/mysql-test/lib/My/SafeProcess/my_safe-  
-- Installing: /usr/local/mysql/mysql-test/lib/My/SafeProcess/Base.pm  
-- Installing: /usr/local/mysql/support-files/mysqld_multi.server  
-- Installing: /usr/local/mysql/support-files/mysql-log-rotate  
-- Installing: /usr/local/mysql/support-files/magic  
-- Installing: /usr/local/mysql/share/aclocal/mysql.m4  
-- Installing: /usr/local/mysql/support-files/mysql.server
```

```
[root@localhost mysql-5.7.26]# make clean
```

## 2.5: 创建用户并进行授权

- 创建mysql用户

```
[root@mysql-m1 ~]# groupadd mysql  
[root@mysql-m1 ~]# useradd -s /sbin/nologin -g mysql -M -r mysql  
[root@mysql-m1 ~]# chown -R mysql. /usr/local/mysql/
```

## 2.6: 实例初始化

三台数据库都需要进行初始化

- 注意

5.7.6之前初始化的方法是: bin/mysql\_install\_db

5.7.6之后的版本初始化数据库不再使用mysql\_install\_db, 而是使用: bin/mysqld --initialize

- 初始化完成后会生成一个随机密码用来登入数据库

```
[root@mysql-m1 ~]# cd /usr/local/mysql/
```

```
[root@mysql-m1 mysql]# ./bin/mysqld --initialize --user=mysql --basedir=/usr/local/mysql -  
datadir=/usr/local/mysql/data/
```

相关警告信息如下:

[警告]不推荐使用具有隐式DEFAULT值的TIMESTAMP。请使用--explicit\_defaults\_for\_timestamp  
务器选项 (有关详细信息, 请参阅文档)。

[警告] InnoDB: 创建新的日志文件, LSN = 45790

[警告] InnoDB: 创建外键约束系统表。

[警告]未找到现有的UUID, 因此我们假设这是第一次启动此服务器。生成新的UUID: 4c52a0d9-9b  
0-11e9-a75a-000c29dc5ef4。

[警告] Gtid表尚未准备好使用。表'mysql.gtid\_executed'无法打开。

[注意]为root @ localhost生成临时密码: u-jUegVt.1G<

## 2.7: 配置环境变量

```
[root@mysql-m1 mysql]# vim /etc/profile  
export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/usr/local  /mysql/bin
```

```
[root@mysql-m1 mysql]# source /etc/profile
```

## 2.8: 配置system管理服务

```
[root@mysql-m1 ~]# vim /etc/systemd/system/mysqld.service
```

```
[Unit]  
Description=MySQL Community Server  
After=network.target  
After=syslog.target
```

```
[Install]  
WantedBy=multi-user.target  
Alias=mysql.service
```

```
[Service]  
User=mysql  
Group=mysql
```

```
# Execute pre and post scripts as root
PermissionsStartOnly=true

# Start main service
ExecStart=/usr/local/mysql/bin/mysqld_safe

# Give up if ping don't get an answer
TimeoutSec=600

Restart=always
PrivateTmp=false
```

- 注意：ExecStart= 后面跟的是MySQL启动程序的安装路径，根据实际情况配置路径mysqld\_safe脚本会在启动MySQL服务器后继续监控其运行情况，并在其死机时重新启动它。

### 3、搭建主从

搭建思路

1. 系统时间需要同步
2. 关闭防火墙和selinux
3. hosts文件中两台服务器主机名和ip地址一一对应起来
4. master和slave的数据库版本保持一致
5. master端必须开启二进制日志；slave端必须开启relaylog日
6. master端和slave端的server\_id号必须不能一致
7. 告诉slave需要同步的master主机的IP、user、password等..

#### 3.1：修改my.cnf文件

- 由于上面将系统自带的mariadb卸载时，系统默认将/etc/my.cnf另存为/etc/my.cnf.rpmsave,所需需要复制一份出来

```
[root@mysql-m1 ~]# cp /etc/my.cnf.rpmsave /etc/my.cnf
```

- M1-M2互为主从，M1-S1为主从
- MySQL-M1配置如下：

```
[root@mysql-m1 ~]# vim /etc/my.cnf
```



```

1 [mysqld]
2 basedir=/usr/local/mysql      mysql安装路径
3 datadir=/usr/local/mysql/data/  mysql数据存储路径
4 socket=/usr/local/mysql/mysql.sock  mysql sock文件存储路径, mysql启动会自动生成
5 port=3306                    指定端口号
6 server_id=37                  master与slave的server_id必须不能一样, 可以设为本机的IP
7 log_bin=/usr/local/mysql/data/mysql-bin  开启binlog, 设置binlog存储路径
8 relay_log=/usr/local/mysql/data/relay.log  开启relay log
9 # Disabling symbolic-links is recommended to prevent assorted security risks
10 symbolic-links=0
11 # Settings user and group are ignored when systemd is used.
12 # If you need to run mysqld under a different user or group,
13 # customize your systemd unit file for mariadb according to the
14 # instructions in http://fedoraproject.org/wiki/Systemd
15 #skip-grant-tables
16 [mysqld_safe]
17 log-error=/usr/local/mysql/logs/mysql-error.log  mysql错误日志存储路径
18 pid-file=/usr/local/mysql/data/mysql.pid        mysql pid文件存储路径
19
20 #
21 # include all files from the config directory

```

● MySQL-M2配置如下:

```

1 [mysqld]
2 basedir=/usr/local/mysql
3 datadir=/usr/local/mysql/data/
4 socket=/usr/local/mysql/mysql.sock
5 port=3306
6 server_id=38
7 log_bin=/usr/local/mysql/data/mysql-bin
8 relay_log=/usr/local/mysql/data/relay.log
9 # Disabling symbolic-links is recommended to prevent assorted security risks
10 symbolic-links=0
11 # Settings user and group are ignored when systemd is used.
12 # If you need to run mysqld under a different user or group,
13 # customize your systemd unit file for mariadb according to the
14 # instructions in http://fedoraproject.org/wiki/Systemd
15
16 [mysqld_safe]
17 log-error=/usr/local/mysql/logs/mysql-error.log
18 pid-file=/usr/local/mysql/data/mysql.pid
19
20 #
21 # include all files from the config directory
22 #
23 !includedir /etc/my.cnf.d

```

● MySQL-S1配置如下

```

1 [mysqld]
2 basedir=/usr/local/mysql
3 datadir=/usr/local/mysql/data/
4 socket=/usr/local/mysql/mysql.sock
5 # Disabling symbolic-links is recommended to prevent assorted
6 symbolic-links=0
7 # Settings user and group are ignored when systemd is used.
8 # If you need to run mysqld under a different user or group,
9 # customize your systemd unit file for mariadb according to th
10 # instructions in http://fedoraproject.org/wiki/Systemd
11 server_id=39
12 relay_log=/usr/local/mysql/data/relay.log
13
14 [mysqld_safe]
15 log-error=/usr/local/mysql/logs/mysql-error.log
16 pid-file=/usr/local/mysql/data/mysql.pid
17 #
18 # include all files from the config directory
19 #
20 !includedir /etc/my.cnf.d

```

### 3.2: 数据同步

数据同步以当前数据库中存放最新数据的数据库为主

不初始化数据有可能slave端的Slave\_SQL\_Running:显示为NO

```
[root@mysql-m1 ~]# rsync -av /usr/local/mysql/data root@192.168.66.38:/usr/local/mysql/
[root@mysql-m1 ~]# rsync -av /usr/local/mysql/data root@192.168.66.39:/usr/local/mysql/
```

- 注意：拷贝的是data整个目录，data后面千万不要加/，不然拷贝过去的就是data目录下的文件

slave端需要删除数据(data)目录下的auto.cnf文件

- M1数据库

```
[root@mysql-m1 ~]# cd /usr/local/mysql/data/
[root@mysql-m1 data]# ls
auto.cnf      ibdata1      ib_logfile1  mysql-m1.err  sys
ib_buffer_pool  ib_logfile0  mysql        performance_schema
[root@mysql-m1 data]# rm -f auto.cnf
```

- M2数据库:

```
[root@MySQL-M2 ~]# cd /usr/local/mysql/data/
[root@MySQL-M2 data]# rm -f auto.cnf
```

- S1数据库:

```
[root@MySQL-S1 ~]# cd /usr/local/mysql/data/
[root@MySQL-S1 data]# rm -f auto.cnf
```

### 3.3: 启动数据库

- 三台服务器都进行启动

```
[root@mysql-m1 ~]# systemctl enable mysqld
[root@mysql-m1 ~]# systemctl start mysqld
[root@mysql-m1 ~]# systemctl status mysqld
```

启动报错:

```
mysqld_safe: my_print_defaults: Can't read dir of '/etc/my.cnf.d' (Errcode: 2 - No such file or directory)
```

原因:

mysqld\_safe脚本进行启动时没有找到/etc/my.cnf.d目录;默认系统中是存在的

解决方案:

创建/etc/my.cnf.d

```
[root@mysql-m1 ~]# mkdir /etc/my.cnf.d
```

再次启动数据库

```
[root@mysql-m1 ~]# systemctl start mysqld
```

### 3.4: 登入数据库

- 三台数据登入后修改密码

```
[root@mysql-m1 ~]# mysql -uroot -p
Enter password: #输入初始化时生成的随机密码
```

ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/tmp/mysql.sock' (2)

- 解决方案:  
设置软链接

```
[root@mysql-m1 ~]# ln -s /usr/local/mysql/mysql.sock /tmp/
```

- 如果随机密码不记得可以通过以下方式进行登入

```
[root@8-mysql-m1 ~]# vim /etc/my.cnf  
在[mysqld]段下添加skip-grant-tables参数, 重置密码后将该参数去除
```

- 重启数据库

```
[root@8-mysql-m1 ~]# systemctl restart mysqld  
[root@8-mysql-m1 ~]# mysql #直接输入mysql即可登入
```

- 重置密码

```
mysql> update mysql.user set authentication_string=PASSWORD('123456') where User='root'
```

```
[root@mysql-m1 ~]# vim /etc/my.cnf
```

- 重启数据库

```
[root@mysql-m1 ~]# systemctl restart mysqld
```

```
[root@mysql-m1 ~]# mysql  
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)  
[root@mysql-m1 ~]# mysql -uroot -p #输入密码登录
```

#### 【问题】

通过密码登入执行任何语句都报如下错误

```
mysql> show databases;
```

```
ERROR 1820 (HY000): You must reset your password using ALTER USER statement before executing this statement.
```

#### 【解决方法】

```
mysql> SET PASSWORD = PASSWORD('你的新密码');  
mysql> ALTER USER 'root'@'localhost' PASSWORD EXPIRE NEVER;  
mysql> flush privileges;
```

#### 【问题解决】

```
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| sys |  
+-----+  
4 rows in set (0.00 sec)
```

## 3.5: Master创建复制授权账户

### 3.5.1: M1和M2上都需要创建授权账户

- Master1:

```
mysql> grant replication slave on *.* to 'slave'@'192.168.66.%' identified by '123456';
```

- Master2:

```
mysql> grant replication slave on *.* to 'slave'@'192.168.66.%' identified by '123456';
```

### 3.5.2: 查看Master中binlog文件名和pos点位

- ① Master1

#### 1、添加只读锁

mysql> flush tables with read lock; 先加锁，防止两边数据不一致;如果业务还未上线，这个就没有要了

```
mysql> create database t;  
ERROR 1223 (HY000): Can't execute the query because you have a conflicting read lock
```

#### 2、查看binlog信息

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

- ② M2和S1同步M1信息

Master2和Slave1上配置复制信息

```
mysql> change master to master_host='192.168.66.37',master_user='slave',master_password=  
123456',master_port=3306,master_log_file='mysql-bin.000004',master_log_pos=449;
```

```
mysql> change master to  
-> master_host='192.168.66.37',      master ip  
-> master_user='slave',             同步用户  
-> master_password='123456',        密码  
-> master_port=3306,                端口  
-> master_log_file='mybinlog.000004', master上查到到二进制日志名  
-> master_log_pos=449;              master上面查到的位置号
```

- ③ Master2

#### 1、添加只读锁

mysql> flush tables with read lock; 先加锁，防止两边数据不一致;如果业务还未上线，这个就没有要了

```
mysql> create database t;  
ERROR 1223 (HY000): Can't execute the query because you have a conflicting read lock
```

#### 2、查看binlog信息

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

- ④ M1同步M2的信息

```
mysql> change master to master_host='192.168.66.38',master_user='slave',master_password='123456',master_port=3306,master_log_file='mysql-bin.000005',master_log_pos=449;
```

- ⑤ M1\M2\S1启动复制线程

```
mysql> start slave;  
mysql> show slave status \G; //以下2项全部为Yes可正常使用
```

Slave\_IO\_Running: Yes 代表成功连接到master并且下载日志  
Slave\_SQL\_Running: Yes 代表成功执行日志中的SQL语句

- ⑥ M1\M2端解锁

```
mysql> unlock tables; 解锁
```

- ⑦ 测试验证

M1写数据, M2/S1上可以看到

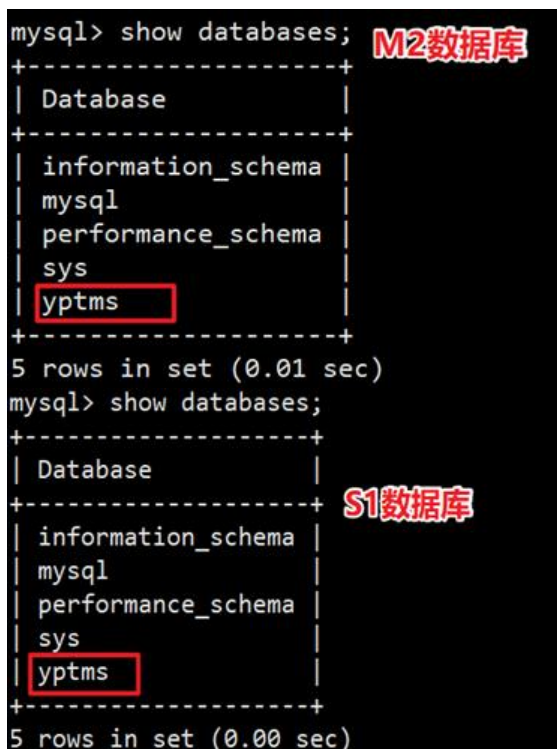
M2写数据, M1上可以看到

S1上写数据, M1/M2上无法看到

- 测试1: M1上创建一个库

```
mysql> create database yptms; ##创建了一个yptms的数据库  
Query OK, 1 row affected (0.00 sec)
```

- M2和S1上进行查看



```
mysql> show databases; M2数据库  
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| sys |  
| yptms |  
+-----+  
5 rows in set (0.01 sec)  
mysql> show databases;  
+-----+  
| Database | S1数据库  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| sys |  
| yptms |  
+-----+  
5 rows in set (0.00 sec)
```

- 测试2: M2上创建一个数据库

```
mysql> create database test; ##创建了一个test数据库  
Query OK, 1 row affected (0.01 sec)
```

- M1/S1上查看数据库

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| test |
| yptms |
+-----+
6 rows in set (0.00 sec)
```

**M1数据库**

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| yptms |
+-----+
5 rows in set (0.00 sec)
```

**S1数据库**

**没有test数据库**

- 测试3: S1上创建一个数据库

```
mysql> create database web;
Query OK, 1 row affected (0.00 sec)
```

- M1/M2上查看数据库

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| test |
| yptms |
+-----+
6 rows in set (0.00 sec)
```

**M1数据库**

**没有web数据库**

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| test |
| yptms |
+-----+
6 rows in set (0.00 sec)
```

**M2数据库**

**没有web数据库**

## 二、安装MyCAT实现读写分离

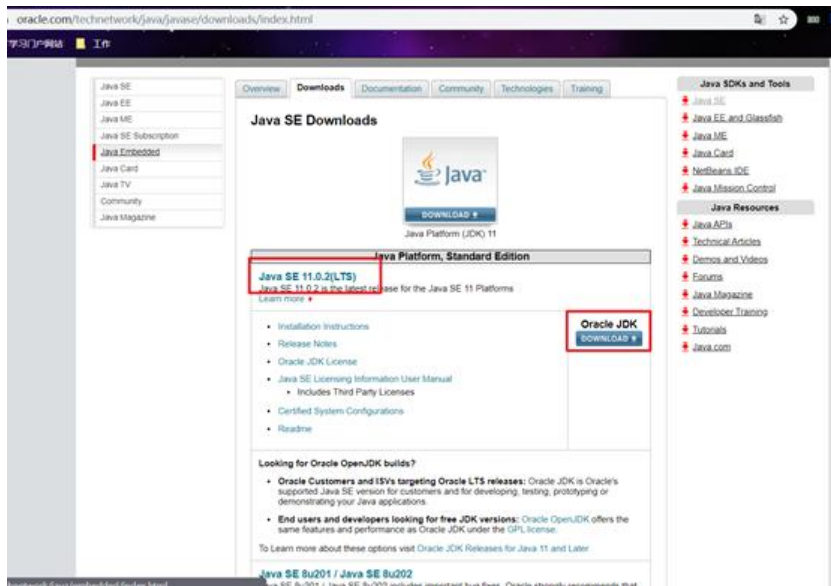
- 注意：由于MyCAT是由JAVA开发的，所以在使用MyCAT时需要安装JDK

### 1、安装配置JDK环境

#### 1.1: 下载JDK

<https://www.oracle.com/technetwork/java/javase/downloads/index.html>

- 选择所需版本进行下载



- 选择Linux平台的源码包

With JDK 11 Oracle has updated the license terms on which we offer the Oracle JDK. The new Oracle Technology Network License Agreement for Oracle Java SE is substantially different from the licenses under which previous versions of the JDK were offered. Please review the new terms carefully before downloading and using this product.

Oracle also offers this software under the GPL License on [jdk.java.net/11](http://jdk.java.net/11)

See also:

- [Java Developer Newsletter](#): From your Oracle account, select **Subscriptions**, expand **Technology**, and subscribe to **Java**.
- [Java Developer Day hands-on workshops \(free\)](#) and other events
- [Java Magazine](#)

JDK 11.0.2 checksum

**Java SE Development Kit 11.0.2**

You must accept the [Oracle Technology Network License Agreement for Oracle Java SE to download this software.](#)

1、勾选  Accept License Agreement  Decline License Agreement

Product / File Description	File Size	Download
Linux	147.28 MB	<a href="#">jdk-11.0.2_linux-x64_bin.de</a>
Linux	154.01 MB	<a href="#">jdk-11.0.2_linux-x64_bin.rpm</a>
Linux	171.32 MB	<a href="#">jdk-11.0.2_linux-x64_bin.tar.gz</a>
macOS	166.13 MB	<a href="#">jdk-11.0.2_osx-x64_bin.dmg</a>
macOS	166.49 MB	<a href="#">jdk-11.0.2_osx-x64_bin.tar.gz</a>
Solaris SPARC	186.78 MB	<a href="#">jdk-11.0.2_solaris-sparcv9_bin.tar.gz</a>
Windows	150.94 MB	<a href="#">jdk-11.0.2_windows-x64_bin.exe</a>
Windows	170.96 MB	<a href="#">jdk-11.0.2_windows-x64_bin.zip</a>

2、点击下载

## 1.2: 解压安装

- 上传jdk包至服务器/soft目录中，可通过ftp的方式进行上传

```
[root@mycat ~]# mkdir /soft
[root@mycat ~]# cd /soft
[root@mycat soft]# ls
jdk-12.0.1_linux-x64_bin.tar.gz
```

```
[root@mycat soft]# tar -zxvf jdk-12.0.1_linux-x64_bin.tar.gz -C /usr/local/
```

## 1.3: 配置jdk环境

```
[root@mycat soft]# vim /etc/profile
export JAVA_HOME=/usr/local/jdk-12.0.1
export PATH=$PATH:$JAVA_HOME/bin:$JAVA_HOME/jre/bin:$PATH
export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/jre/lib
export PATH=$PATH:/usr/local/mycat/bin
export JAVA_HOME JAVA_BIN PATH CLASSPATH MYCAT_HOME
```

```
[root@mycat soft]# source /etc/profile
```

```
[root@mycat soft]# java -version #检查是否安装成功
java version "12.0.1" 2019-04-16
Java(TM) SE Runtime Environment (build 12.0.1+12)
Java HotSpot(TM) 64-Bit Server VM (build 12.0.1+12, mixed mode, sharing)
```

## 2、安装MyCAT

### 2.1: 下载MyCAT

官网: <http://www.mycat.io/>

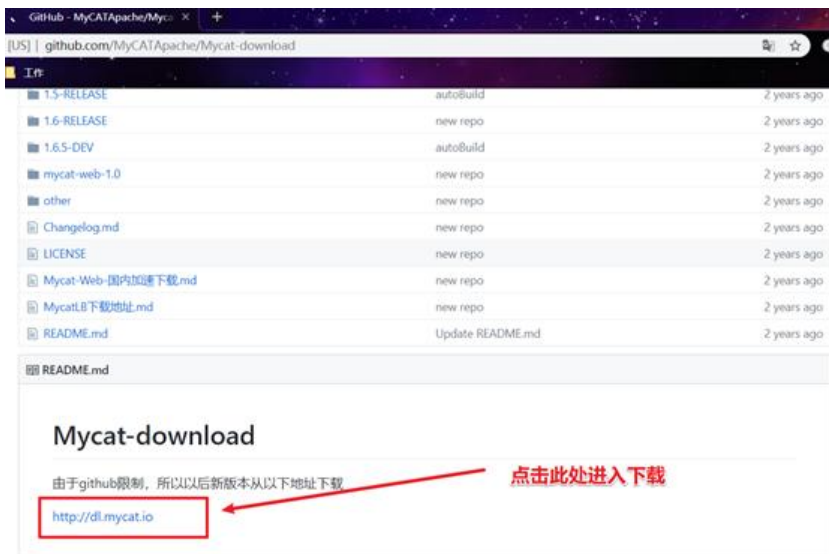
软件下载地址: <http://dl.mycat.io/>

- 点击下载



- 跳转至MyCAT GitHub界面 <http://dl.mycat.io/>





- 选择版本, 此处选择1.6版本

## Index of /

..			
<a href="#">1.6-RELEASE/</a>	28-Oct-2016 12:56	-	
<a href="#">1.6.5/</a>	22-Jan-2018 14:07	-	
<a href="#">1.6.5-BETA/</a>	08-Oct-2017 09:06	-	
<a href="#">1.6.6/</a>	30-Jul-2018 21:55	-	
<a href="#">1.6.6.1/</a>	07-Nov-2018 05:02	-	
<a href="#">1.6.7.1/</a>	22-Feb-2019 07:25	-	
<a href="#">1.7-BETA/</a>	16-Apr-2017 05:54	-	
<a href="#">2.0-dev/</a>	02-Jan-2017 07:24	-	
<a href="#">mycat-web-1.0/</a>	02-Jan-2017 07:40	-	
<a href="#">yum/</a>	18-May-2016 02:51	-	
<a href="#">Mycat-server-1.4-beta-20150604171601-linux.tar.gz</a>	27-Jun-2015 10:09	7663894	
<a href="#">apache-maven-3.3.3-bin.tar.gz</a>	27-Jun-2015 10:09	8042383	
<a href="#">apache-tomcat-7.0.62.tar.gz</a>	27-Jun-2015 10:09	8824528	
<a href="#">jdk-7u79-linux-x64.tar.gz</a>	27-Jun-2015 10:09	153512879	
<a href="#">jdk-8u20-linux-x64.tar.gz</a>	27-Jun-2015 10:09	160872342	
<a href="#">phpMyAdmin-4.4.9-all-languages.tar.gz</a>	27-Jun-2015 10:09	9352049	
<a href="#">probe-2.3.3.zip</a>	27-Jun-2015 10:09	7957290	
<a href="#">toolset.sh</a>	26-Oct-2015 05:03	16015	
<a href="#">zookeeper-3.4.6.tar.gz</a>	27-Jun-2015 10:09	17699306	

- 选择Linux版本下载

## Index of /1.6.7.1/

Linux版本最新的包

..			
<a href="#">Mycat-server-1.6.7.1-release-20190213150257-lin...&gt;</a>	22-Feb-2019 08:20	17558357	
<a href="#">Mycat-server-1.6.7.1-release-20190213150257-mac...&gt;</a>	22-Feb-2019 08:20	17634998	
<a href="#">Mycat-server-1.6.7.1-release-20190213150257-sol...&gt;</a>	22-Feb-2019 08:20	17582175	
<a href="#">Mycat-server-1.6.7.1-release-20190213150257-tes...&gt;</a>	22-Feb-2019 08:20	1065286	
<a href="#">Mycat-server-1.6.7.1-release-20190213150257-uni...&gt;</a>	22-Feb-2019 08:20	17683642	
<a href="#">Mycat-server-1.6.7.1-release-20190213150257-win...&gt;</a>	22-Feb-2019 08:21	17677314	
<a href="#">Mycat-server-1.6.7.1-release-20190627191042-lin...&gt;</a>	27-Jun-2019 10:04	17567422	
<a href="#">Mycat-server-1.6.7.1-release-20190627191042-mac...&gt;</a>	27-Jun-2019 10:04	17645940	
<a href="#">Mycat-server-1.6.7.1-release-20190627191042-sol...&gt;</a>	27-Jun-2019 10:04	17591615	
<a href="#">Mycat-server-1.6.7.1-release-20190627191042-tes...&gt;</a>	27-Jun-2019 10:04	1065329	
<a href="#">Mycat-server-1.6.7.1-release-20190627191042-uni...&gt;</a>	27-Jun-2019 10:04	17692265	
<a href="#">Mycat-server-1.6.7.1-release-20190627191042-win...&gt;</a>	27-Jun-2019 10:04	17688258	

## 2.2: 解压安装

- 上传软件包至服务器/soft目录下, 并解压至/usr/local目录下
- 注意: mycat无需进行三部曲, 直接解压到指定目录即可完成安装

```
[root@mycat soft]# ls
jdk-12.0.1_linux-x64_bin.tar.gz
Mycat-server-1.6.7.1-release-20190627191042-linux.tar.gz
```

```
[root@mycat soft]# tar -zxvf Mycat-server-1.6.7.1-release-20190627191042-linux.tar.gz -C /usr/local/
```

```
[root@mycat soft]# cd /usr/local/mycat/
```

- 目录结构

```
[root@mycat mycat]# ls
```



### 3、后端数据库创建授权用户

#### 3.1: Master数据库创建全权限用户

- 在Master1上创建即可

```
mysql> create user root@'%' IDENTIFIED BY '123456';
```

```
mysql> GRANT ALL ON solo.* TO root@'%' IDENTIFIED BY '123456';
```

```
mysql> FLUSH PRIVILEGES;
```

#### 3.2: Slave数据库创建只读权限用户

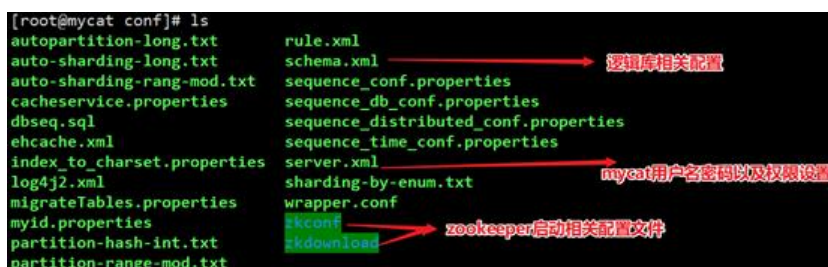
```
mysql> CREATE USER readonly@'%' IDENTIFIED BY '123456';
```

```
mysql> GRANT SELECT ON solo.* TO readonly@'%' IDENTIFIED BY '123456';
```

```
mysql> FLUSH PRIVILEGES;
```

### 4: 配置文件修改

```
[root@mycat ~]# cd /usr/local/mycat/conf/
```



配置文件说明:

server.xml 默认可不用修改, 此文件是配置中间件登入时用到的用户名和密码, 对应映射虚拟库的

称

schema.xml 配置中间件真实数据库的数据库用户名、数据库登入密码、IP、端口

### 4.1: 配置文件schema.xml

[root@mycat conf]# vim schema.xml

拷贝一份默认配置然后再进行修改

[root@mycat conf]# cp schema.xml{,.bak}

[root@mycat conf]# vim schema.xml

```

1 <?xml version="1.0"?>
2 <!DOCTYPE mycat:schema SYSTEM "schema.dtd">
3 <mycat:schema xmlns:mycat="http://io.mycat/">
4   <schema name="solo" checkSQLschema="false" sqlMaxLimit="100">
5     <!-- auto sharding by id (lonel) -->
6     <table name="travelrecord" dataNode="dn1,dn2,dn3" rule="auto-sharding-long" />
7     <!-- global table is auto cloned to all defined data nodes ,so can join
8         with any table whose sharding node is in the same data node -->
9     <table name="company" primaryKey="ID" type="global" dataNode="dn1,dn2,dn3" />
10    <table name="goods" primaryKey="ID" type="global" dataNode="dn1,dn2" />
11    <!-- random sharding using mod sharind rule -->
12    <table name="hotnews" primaryKey="ID" autoIncrement="true" dataNode="dn1,dn1,dn3"
13          rule="mod-long" />
14    <!-- <table name="dual" primaryKey="ID" dataNode="dnx,dnoracle2" type="global"

```

注意上面的datanode对应下面的datanode name, 所以如果不对应则mycat无法启动

```

34 <!-- <dataNode name="dn10-743" dataHost="localhost1" database="db10-743"
35 /> -->
36 <dataNode name="dn1" dataHost="localhost1" database="solo" />
37 <!-- <dataNode name="dn4" dataHost="sequoiadb1" database="SAMPLE" />
38 <dataNode name="jdbc_dn1" dataHost="jdbchost" database="db1" />
39 <dataNode name="jdbc_dn2" dataHost="jdbchost" database="db2" />
40 <dataNode name="jdbc_dn3" dataHost="jdbchost" database="db3" />
41 <dataHost name="hostM1" maxCon="1000" minCon="10"
42           writeType="0" dbType="mysql" dbDriver="native" switchType="1" slaveThreshold="100">
43   <!-- 所有写操作发送
44       <writeHost host="hostM1" url="192.168.66.37:3306" user="root"
45             password="123456">
46     <!-- 只有第一个挂掉后
47         才会切换到第二个
48     <!-- can have multi read hosts -->
49     <readHost host="hostM1" url="192.168.66.37:3306" user="readonly" password="123456" />
50     </writeHost>
51     <writeHost host="hostM2" url="192.168.66.38:3306" user="root"
52             password="123456" />
53     <!-- <writeHost host="hostM2" url="localhost:3316" user="root" password="123456" /> -->
54 </dataHost>
55 <!--
56 <dataHost name="localhost1" maxCon="1000" minCon="10" balance="3"
57       writeType="0" dbType="mysql" dbDriver="native" switchType="1" slaveThreshold="100">
58   <heartbeat>select user()</heartbeat>
59   <!-- can have multi write hosts -->
60   <writeHost host="hostM1" url="192.168.66.37:3306" user="root"
61         password="123456">
62     <!-- can have multi read hosts -->
63     <readHost host="hostM1" url="192.168.66.37:3306" user="readonly" password="123456" />
64     <writeHost host="hostM2" url="192.168.66.38:3306" user="root"
65             password="123456" />
66     <!-- <writeHost host="hostM2" url="localhost:3316" user="root" password="123456" /> -->
67 </dataHost>

```

### 4.2: 配置文件server.xml

[root@mycat conf]# vim server.xml

```

99 <user name="root" defaultAccount="true">
100   <property name="password">123456</property>
101   <property name="schemas">solo</property>
102 </user>
103 <!-- 表级 DML 权限设置 -->
104 <!--
105 <privileges check="false">
106   <schema name="TESTDB" dml="0110">
107     <table name="tb01" dml="0000"></table>
108     <table name="tb02" dml="1111"></table>
109   </schema>
110 </privileges>
111 </user>
112 <!--
113 <user name="readonly">
114   <property name="password">123456</property>
115   <property name="schemas">solo</property>
116   <property name="readOnly">true</property>
117 </user>
118 </mycat:server>

```

## 4.3: 主要参数说明

**balance**指的负载均衡类型，目前的取值有4种：

1. balance=" 0" ，不开启读写分离机制，所有读操作都发送到当前可用的writeHost上。
2. balance=" 1" ，全部的readHost与stand bywriteHost参与select语句的负载均衡，简单的说，双主双从模式(M1->S1, M2->S2, 并且M1与 M2互为主备)，正常情况下，M2,S1,S2都参与select语句的负载均衡。
3. balance=" 2" ，所有读操作都随机的在writeHost、readhost上分发。
4. balance=" 3" ，所有读请求随机的分发到writerHost对应的readhost执行，writerHost不承担压力

注意：balance=3只在1.4及其以后版本有，1.3没有。

**writeType**属性：

1. writeType=" 0" ,所有写操作发送到配置的第一个writeHost,第一个挂了切到还生存的第二个writeHost,重新启动后以切换后的为准，切换记录在配置文件中:dnindex.properties.
2. writeType=" 1" ,所有写操作都随机地发送到配置的writeHost,1.5以后废弃不推荐。

**switchType**指的是切换的模式，目前的取值也有4种

1. switchType=' -1' 表示不自动切换
2. switchType=' 1' 默认值，表示自动切换
3. switchType=' 2' 基于MySQL主从同步的状态决定是否切换,心跳语句为 show slave status
4. switchType=' 3' 基于MySQLgalary cluster的切换机制（适合集群）（1.4.1），心跳语句为 show slave status like 'wsrep%' 。

● 注意：估计Mycat1.4才开始支持switchType。1.3版本配置该属性的话，日志里会报错：org.xml.sax.SAXParseException;lineNumber: 61; columnNumber: 86; Attribute "switchType" must be declared for element type "dataHost" 。

MyCAT心跳检查查询配置为 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，Myat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及Seconds\_Behind\_Master主从复制延，当Seconds\_Behind\_Master>slaveThreshold时，读写分离筛选器会过滤掉此Slave机器，防止到很久以前的旧数据，当主节点宕机后，切换逻辑会检查Slave上的Seconds\_Behind\_Master是否为，为0时则表示主仅同步，可安全切换，否则不会切换。

## 5、启动MyCAT

### 5.1: 通过system管理服务

```
[root@mycat ~]# vim /etc/systemd/system/mycat.service
```

```
[Unit]
```

Description=mycat

After=syslog.target network.target remote-fs.target nss-lookup.target

[Service]

Type=forking

PIDFile=/usr/local/mycat/logs/mycat.pid

ExecStart=/usr/local/mycat/bin/mycat start

ExecReload=/usr/local/mycat/bin/mycat restart

ExecStop=/usr/local/mycat/bin/mycat stop

PrivateTmp=true

[Install]

WantedBy=multi-user.target

## 5.2: 启动MyCAT

```
[root@mycat ~]# systemctl daemon-reload
```

```
[root@mycat ~]# systemctl start mycat
```

```
[root@mycat ~]# systemctl status mycat
```

```
[root@mycat ~]# systemctl status mycat.service
● mycat.service - mycat
   Loaded: loaded (/usr/lib/systemd/system/mycat.service; enabled; Vendor preset)
   Active: active (running) since 日 2019-06-30 19:09:20 CST; 1s ago
   Process: 25226 ExecStart=/usr/local/mycat/bin/mycat start (code=exited, status=0/SUCCESS)
   Main PID: 25275 (wrapper-linux-x)
   CGroup: /system.slice/mycat.service
           └─25275 /usr/local/mycat/bin/./wrapper-linux-x86-64 /usr/local/mycat/bin/mycat start

[root@mycat ~]# ss -tnlp
State      Recv-Q Send-Q           Local Address:Port
LISTEN    0      50                *:3306
users: (("mysqld",pid=35611,fd=13))
LISTEN    0      128                *:22
users: (("sshd",pid=6896,fd=3))
LISTEN    0      100             127.0.0.1:25
users: (("master",pid=6986,fd=13))
LISTEN    0      1                127.0.0.1:32000
users: (("java",pid=40098,fd=4))
LISTEN    0      100                :::9066
users: (("java",pid=40098,fd=69))
LISTEN    0      128                :::22
users: (("sshd",pid=6896,fd=4))
LISTEN    0      100                :::1:25
users: (("master",pid=6986,fd=14))
LISTEN    0      50                :::43259
users: (("java",pid=40098,fd=52))
LISTEN    0      50                :::40573
users: (("java",pid=40098,fd=54))
LISTEN    0      50                :::1984
users: (("java",pid=40098,fd=53))
LISTEN    0      100                :::8066
users: (("java",pid=40098,fd=73))
[root@mycat ~]#
```

```
[root@mycat conf]# ss -tnlp
State      Recv-Q Send-Q                               Local Address:Port
LISTEN     0      50                                  *:3306
users: (("mysqld",pid=35611,fd=13))
LISTEN     0      128                                 *:22
users: (("sshd",pid=6896,fd=3))
LISTEN     0      100                                127.0.0.1:25
users: (("master",pid=6986,fd=13))
LISTEN     0      1                                  127.0.0.1:32000
users: (("java",pid=40098,fd=4))
LISTEN     0      100                                 :::9066
users: (("java",pid=40098,fd=69))
LISTEN     0      128                                 :::22
users: (("sshd",pid=6896,fd=4))
LISTEN     0      100                                 :::1:25
users: (("master",pid=6986,fd=14))
LISTEN     0      50                                  :::43259
users: (("java",pid=40098,fd=52))
LISTEN     0      50                                  :::40573
users: (("java",pid=40098,fd=54))
LISTEN     0      50                                  :::1984
users: (("java",pid=40098,fd=53))
LISTEN     0      100                                 :::8066
users: (("java",pid=40098,fd=73))
[root@mycat conf]#
```

### 5.3: 启动问题汇总

【问题1】如果出现以下报错

```
FATAL | wrapper | 2019/06/30 19:12:14 | ERROR: Could not write pid file /usr/local/mycat/logs/mycat.pid: No such file or directory
```

【解决方案】

直接在提示的目录/usr/local/mycat/下创建logs目录

【问题2】

```
ERROR | wrapper | 2019/06/30 19:12:14 | Unable to start JVM: No such file or directory (2)
ERROR | wrapper | 2019/06/30 19:12:14 | JVM exited while loading the application.
```

查看报错日志

```
[root@mycat ~]# cat /usr/local/mycat/logs/wrapper.log | grep ERROR
```

【解决方案1】修改配置文件

```
[root@mycat ~]# vim /usr/local/mycat/conf/wrapper.conf
```

增加以下参数:

```
wrapper.java.command=/usr/local/jdk-12.0.1/bin/java
```

【解决方案2】检查profile文件中java路径是否正确

【问题3】

```
INFO | WrapperSimpleApp: Encountered an error running main: java.lang.ExceptionInInitializerError
```

```
INFO | java.lang.ExceptionInInitializerError
```

【原因】

以上报错提示为，数据库连接不上的问题

【解决方案】

1、 排查配置文件是否正确server.xml,schema.xml

2、 检查schema.xml中配置的数据库用户密码是否正确，是否进行了访问授权

## 6、测试访问

- 通过navicat连接测试

