Mycat配置方案

| 版本号 | 更新时间 | 修订人 | 修订内容 |
| --- | --- | --- | --- |
| v1.0 | 2017-02-28 | 赵继永 | 初版发行 |
|  |  |  |  |
|  |  |  |  |

# 1 MyCAT概述

## 1.1背景

随着传统的数据库技术日趋成熟、计算机网络技术的飞速发展和应用范围的扩充，数据库应用已经普遍建立于计算机网络之上。这时集中式数据库系统表现出它的不足：

（1）集中式处理，势必造成性能瓶颈；

（2）应用程序集中在一台计算机上运行，一旦该计算机发生故障，则整个系统受到影响，可靠性不高；

（3）集中式处理引起系统的规模和配置都不够灵活，系统的可扩充性差。

在这种形势下，集中式数据库将向分布式数据库发展。

## 1.2发展历程

MyCAT的诞生，要从其前身Amoeba和Cobar说起。

Amoeba（变形虫）项目，该开源框架于2008年开始发布一款 Amoeba for Mysql软件。这个软件致力于MySQL的分布式数据库前端代理层，它主要在应用层访问MySQL的时候充当SQL路由功能，专注于分布式数据库代理层（Database Proxy）开发。座落与 Client、DB Server(s)之间，对客户端透明。具有负载均衡、高可用性、SQL过滤、读写分离、可路由相关的到目标数据库、可并发请求多台数据库合并结果。 通过Amoeba你能够完成多数据源的高可用、负载均衡、数据切片的功能，目前Amoeba已在很多企业的生产线上面使用。

阿里巴巴于2012年6月19日，正式对外开源的数据库中间件Cobar，前身是早已经开源的Amoeba，不过其作者陈思儒离职去盛大之后，阿里巴巴内部考虑到Amoeba的稳定性、性能和功能支持，以及其他因素，重新设立了一个项目组并且更换名称为Cobar。[Cobar](https://github.com/alibaba/cobar) 是由 Alibaba 开源的 MySQL 分布式处理中间件，它可以在分布式的环境下看上去像传统数据库一样提供海量数据服务。

Cobar自诞生之日起， 就受到广大程序员的追捧，但是自2013年后，几乎没有后续更新。在此情况下，MyCAT应运而生，它基于阿里开源的Cobar产品而研发，Cobar的稳定性、可靠性、优秀的架构和性能，以及众多成熟的使用案例使得MyCAT一开始就拥有一个很好的起点，站在巨人的肩膀上，MyCAT能看到更远。目前MyCAT的最新发布版本为1.6版本。

## 1.3 关键特性

·支持 SQL 92标准；

·支持MySQL集群，可以作为Proxy使用；

·支持JDBC连接ORACLE、DB2、SQL Server，将其模拟为MySQL Server使用；

·支持galera for mysql集群，percona-cluster或者mariadb cluster，提供高可用性数据分片集群；

·自动故障切换，高可用性；

·支持读写分离，支持MySQL双主多从，以及一主多从的模式；

·支持全局表，数据自动分片到多个节点，用于高效表关联查询；

·支持独有的基于E-R 关系的分片策略，实现了高效的表关联查询；

·多平台支持，部署和实施简单。

## 1.4 优势

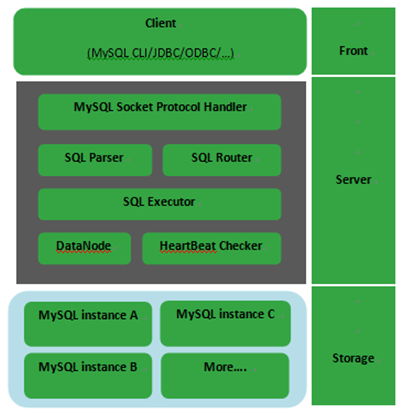
基于阿里开源的Cobar产品而研发，Cobar的稳定性、可靠性、优秀的架构和性能，以及众多成熟的使用案例使得MyCAT一开始就拥有一个很好的起点，站在巨人的肩膀上，能看到更远。

广泛吸取业界优秀的开源项目和创新思路，将其融入到MyCAT的基因中，使得MyCAT在很多方面都领先于目前其他一些同类的开源项目，甚至超越某些商业产品。

MyCAT背后有一只强大的技术团队，其参与者都是5年以上资深软件工程师、架构师、DBA等，优秀的技术团队保证了MyCAT的产品质量。

MyCAT并不依托于任何一个商业公司，因此不像某些开源项目，将一些重要的特性封闭在其商业产品中，使得开源项目成了一个摆设。

## 1.5 总体架构

MyCAT的架构如下图所示：  


MyCAT使用MySQL的通讯协议模拟成一个MySQL服务器，并建立了完整的Schema（数据库）、Table （数据表）、User（用户）的逻辑模型，并将这套逻辑模型映射到后端的存储节点DataNode（MySQL Instance）上的真实物理库中，这样一来，所有能使用MySQL的客户端以及编程语言都能将MyCAT当成是MySQLServer来使用，不必开发新的客户端协议。

当MyCAT收到一个客户端发送的SQL请求时，会先对SQL进行语法分析和检查，分析的结果用于SQL路由，SQL路由策略支持传统的基于表格的分片字段方式进行分片，也支持独有的基于数据库E-R关系的分片策略，对于路由到多个数据节点（DataNode）的SQL，则会对收到的数据集进行“归并”然后输出到客户端。

SQL执行的过程，简单的说，就是把SQL通过网络协议发送给后端的真正的数据库上进行执行，对于MySQL Server来说，是通过MySQL网络协议发送报文，并解析返回的结果，若SQL不涉及到多个分片节点，则直接返回结果，写入客户端的SOCKET流中，这个过程是非阻塞模式（NIO）。

DataNode是MyCAT的逻辑数据节点，映射到后端的某一个物理数据库的一个Database，为了做到系统高可用，每个DataNode可以配置多个引用地址（DataSource），当主DataSource被检测为不可用时，系统会自动切换到下一个可用的DataSource上，这里的DataSource即可认为是Mysql的主从服务器的地址。

# 2安装部署

## 2.1 服务器信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ip | 192.168.172.201 | 192.168.172.202 | 192.168.172.203 | 192.168.172.205 |
| 操作系统 | Centos6.5 | Centos6.5 | Centos6.5 | Centos6.5 |
| JDK | 1.7.0\_67 | 1.7.0\_67 | 1.7.0\_67 | 1.7.0\_67 |
| mysql | 5.1.73(主库) | 5.1.73(从库) | 5.1.73(从库) |  |
| Mycat |  |  |  | 1.4 |

## 2.2安装步骤

**2.2.1 jdk安装**

**2.2.1.1下载**

官网下载JDK，建议Jdk7的版本且64位。

用Xfp上传到虚拟机上面 /usr/lib/jvm路径下面(切換到root用户)

**2.2.1.2 安装**

解压 tar –zxvf jdk-7u67-linux-x64.tar.gz

设置环境变量 vi ~/.bashrc

在最后添加

export JAVA\_HOME=/usr/lib/jvm/jdk1.7.0\_67

export JRE\_HOME=$JAVA\_HOME/jre

export PATH=$PATH:$JAVA\_HOME/bin

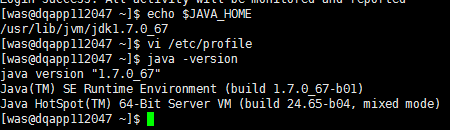
export CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar



:wq保存退出；并输入以下命令使之立即生效。  
source /etc/profile

**2.2.1.3 检查**

输入java –version检测安装是否成功：



**2.2.2 数据库同步配置**

Mysql的安装比较简单，暂不描述。

本次设计为，一主两从，主库mysql位于192.168.172.201机器(以下简称201机器)，主要负责数据的写入；从库mysql位于192.168.172.202、192.168.172.203两台机器(以下简称202、203机器)，主要负责数据的读取

**2.2.2.1主库文件设置**

主库mysql（201机器） my.cnf配置文件



**2.2.2.2 主库创建同步账号**

在mysql主库201机器上配置同步用到的账号密码

mysql> grant replication slave on \*.\* to root@192.168.172.202 identified by '123456'

检查

mysql> flush privileges;

mysql> show master status;+------------------+----------+--------------+------------------+

| File | Position | Binlog\_Do\_DB | Binlog\_Ignore\_DB |

+------------------+----------+--------------+------------------+

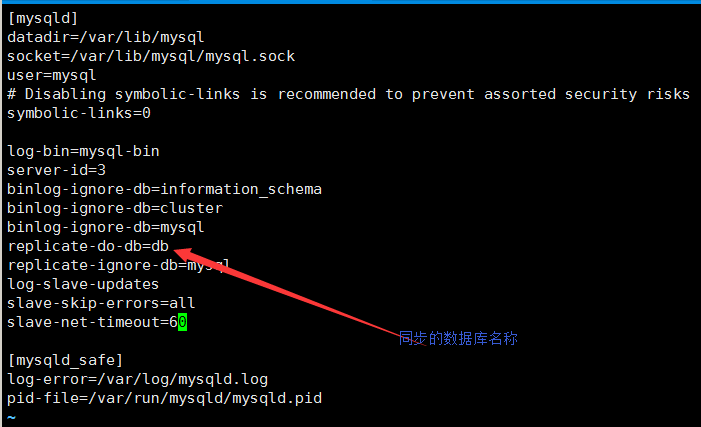
| mysql-bin.000008 | 337 | | |

+------------------+----------+--------------+------------------+

**记录下二进制日志文件名和位置**

**2.2.2.3从库文件设置**

从库mysql（202、203机器） my.cnf配置文件



**2.2.2.4从库账号配置**

202、203机器上的mysql从库做同步账号设置

mysql> change master to master\_host='192.168.172.201',master\_user='root',master\_password='123456', master\_log\_file='mysql-bin.000008',master\_log\_pos=337;

**2.2.2.5 从库启动同步**

202、203机器启动mysql同步

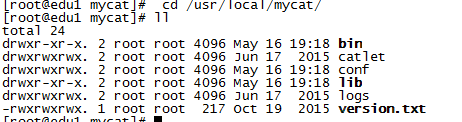
mysql> start slave;

mysql> show slave status\G;

**2.2.3 mycat安装**

**2.2.3.1解压**

解压mycat的安装包到/user/local/下



**2.2.3.2 设置环境变量**

vi /etc/profile

IMG_259

source /etc/profile

**2.2.3.3 主要文件配置**

vi /usr/local/mycat/conf/schema.xml

<?xml version="1.0"?>

<!DOCTYPE mycat:schema SYSTEM "schema.dtd">

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

<schema name="TESTDB" checkSQLschema="false" sqlMaxLimit="100" dataNode='dn1'>

</schema>

<dataNode name="dn1" dataHost="localhost1" database="db" />

<dataHost name="localhost1" maxCon="1000" minCon="10" balance="3" writeType="0" dbType="mysql" dbDriver="native" switchType="1" slaveThreshold="100">

<heartbeat>select user()</heartbeat>

<writeHost host="hostM1" url="192.168.172.201:3306" user="root" password="123456">

<readHost host="hostS2" url="192.168.172.202:3306" user="root" password="123456" />

<readHost host="hostS3" url="192.168.172.203:3306" user="root" password="123456" />

</writeHost>

</dataHost>

</mycat:schema>

1. vi /usr/local/mycat/conf/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>

</system>

<user name="root">

<property name="password">123456</property>

<property name="schemas">TESTDB</property>

</user>

</mycat:server>

**2.2.3.4防火墙配置**

修改防火墙,允许mycat的端口被外界访问

vi /etc/sysconfig/iptables

-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

service iptables restart

**2.2.3.5启动mycat服务**

/usr/local/mycat/bin/mycat start

# 3 mycat测试

我们可以使用mysql客户端连接或者navicat来连接mycat

Mycat的使用与mysql基本一致，只是在数据连接用到host及端口(默认链接端口为8066)略有不同

## 3.1配置mycat的连接

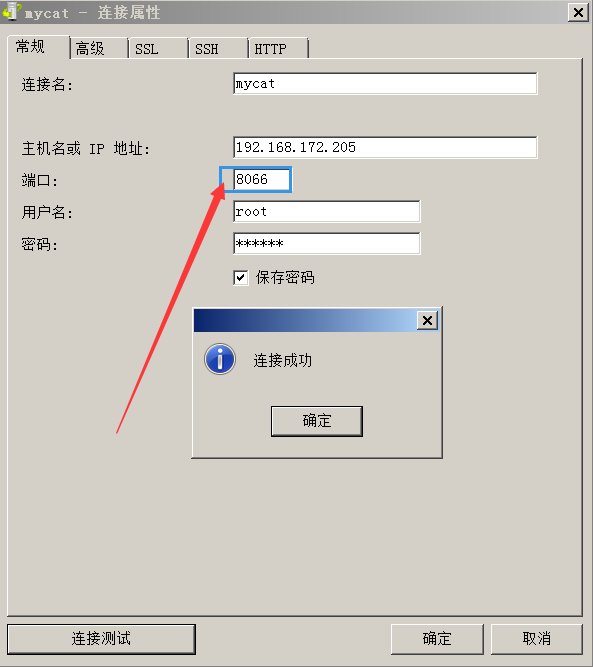
使用navicat配制方法如下

Ip:192.168.172.205

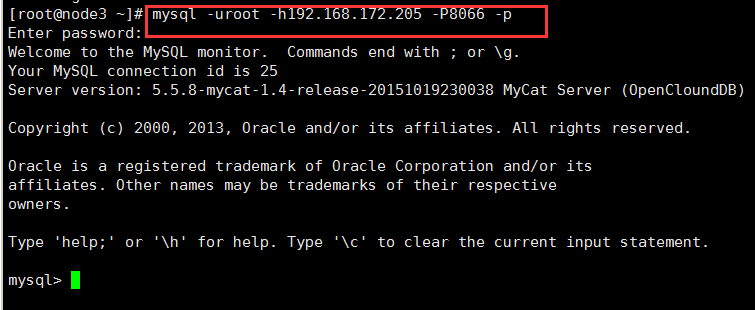
端口:8066

帐号/密码:root/123456

具体配置信息如下图



命令行方式如下



## 3.2创建测试表

使用如下命令创建表

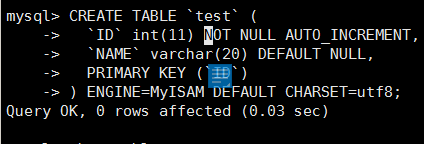
CREATE TABLE `test` (

`ID` int(11) NOT NULL AUTO\_INCREMENT,

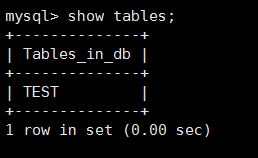
`NAME` varchar(20) DEFAULT NULL,

PRIMARY KEY (`ID`)

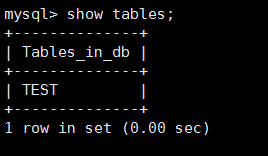
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

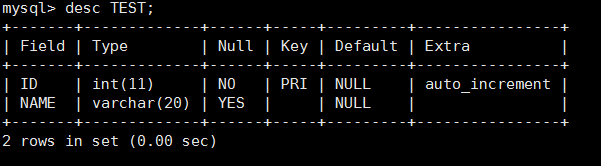


检查表创建



在201、202、203三台机器上检查，表格是否已经创建以及创建的表结构





以上显示，通过mycat在主库（201机器mysql库）中可以正确创建表，数据库主库（201机器mysql）可以及时同步到数据库从库（202、203机器上的mysql）

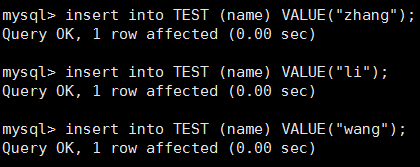
## 3.3添加记录

在mycat中执行以下命令

insert into TEST (name) VALUE("zhang");

insert into TEST (name) VALUE("li");

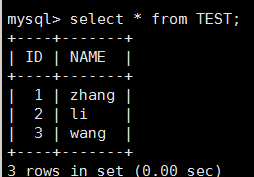
insert into TEST (name) VALUE("wang");



## 3.4查询记录

在mycat中执行以下语句

select \* from TEST;



查看mycat日志（位于$MYCAT\_HOME/logs/mycat.log）



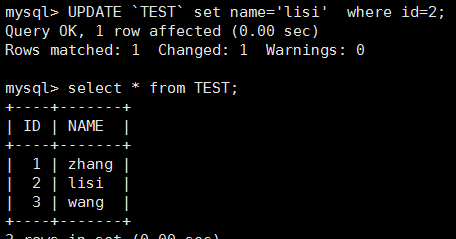
查询请求被分配到从库中。

## 3.5修改记录

执行以下语句

UPDATE `TEST` set name='lisi' where id=2;

select \* from TEST;

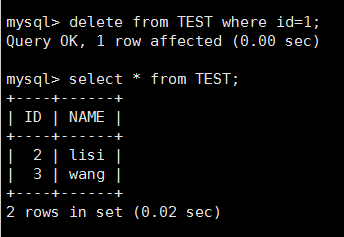


## 3.6 删除记录

执行以下语句

delete from TEST where id=1;

select \* from TEST;



## 3.7删除表

执行以下语句

Drop table TEST ;

show table TEST;

