### **Mycat的安装与配置文件分析**

**1.安装Mycat**

mycat是基于java的，所以需要java环境

[mycat下载地址]http://www.mycat.io/

１）准备java环境

]# yum -y install java-1.8.0-openjdk //安装JDK

[root@mycat56 ~]# which java //查看命令

/usr/bin/java

[root@mycat56 ~]# java –version //显示版本

openjdk version "1.8.0\_161"

OpenJDK Runtime Environment (build 1.8.0\_161-b14)

OpenJDK 64-Bit Server VM (build 25.161-b14, mixed mode)

 2）安装提供服务的软件包

[root@mycat56 ~]# tar -zxvf Mycat-server-1.6-RELEASE-20161028204710-linux.tar.gz //解压源码

[root@mycat56 ~]# mv mycat /usr/local/ //移动目录

[root@mycat56 ~]# ls /usr/local/mycat/ //查看文件列表

bin catlet conf lib logs version.txt

**2.Mycat配置文件**

　1)server.xml

]# vim /usr/local/mycat/conf/server.xml

<user name="root">        //连接mycat服务时使用的用户名

<property name="password">123456</property> //用户连接mycat用户时使用的密码

<property name="schemas">TESTDB</property> //逻辑库名

</user>

<user name="user">

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

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

<property name="readOnly">true</property>    //只读权限，连接mycat服务后只有读记录的权限,不写这一行则是可读可写

</user>

:wq

重点关注上面这段配置，其他默认即可。

=======================================

参数           说明

user          用户配置节点

name          登录的用户名，也就是连接Mycat的用户名。

password      登录的密码，也就是连接Mycat的密码

schemas       数据库名，这里会和schema.xml中的配置关联，多个用逗号分开，例如需要这个用户需要管理两个数据库db1,db2，则配置db1,dbs

privileges    配置用户针对表的增删改查的权限

readOnly      mycat逻辑库所具有的权限。true为只读，false为读写，默认为false。

=======================================

我这里配置了一个账号root，密码为123456,逻辑数据库为TESTDB（默认），这些信息都可以自己随意定义,读写权限都有，没有针对表做任何特殊的权限。

注意：

- server.xml文件里登录TESTDB的用户名和密码可以任意定义，这个账号和密码是为客户机登录TESTDB时使用的账号信息。

- 逻辑库名(逻辑库是对外展示的库，并非真实用于存储数据的库，逻辑库位于mycat代理服务器上，真实库位于数据库服务器上，如这里配置的逻辑库TESTDB，真实库需要在server.xml中配置，接下来会将）

- 这里只定义了一个标签，所以把多余的都注释了。如果定义多个标签，即设置多个连接mycat的用户名和密码，那么就需要在schema.xml文件中定义多个对应的库！

2)schema.xml

schema.xml是最主要的配置项，此文件关联mysql读写分离策略！读写分离、分库分表策略、分片节点都是在此文件中配置的！

MyCat作为中间件，它只是一个代理，本身并不进行数据存储，需要连接后端的MySQL物理服务器，此文件就是用来连接MySQL服务器的！

**schemaxml文件中配置的参数解释**

参数            说明

1.schema         数据库设置，此数据库为逻辑数据库，name与server.xml中schema对应

2.dataNode       分片信息，也就是分库相关配置

3.dataHost       物理数据库，真正存储数据的数据库

**配置说明**

1.name属性唯一标识dataHost标签，供上层的标签使用。

2.maxCon属性指定每个读写实例连接池的最大连接。也就是说，标签内嵌套的

3.writeHost、readHost标签都会使用这个属性的值来实例化出连接池的最大连接数。

4.minCon属性指定每个读写实例连接池的最小连接，初始化连接池的大小。

**每个节点的属性逐一说明**

schema:

属性             说明

1.name               逻辑数据库名，与server.xml中的schema对应

2.checkSQLschema     数据库前缀相关设置，建议看文档，这里暂时设为folse

3.sqlMaxLimit    select 时默认的limit，避免查询全表

Table

属性             说明

name               表名，物理数据库中表名

dataNode       表存储到哪些节点，多个节点用逗号分隔。节点为下文dataNode设置的name

primaryKey     主键字段名，自动生成主键时需要设置

autoIncrement      是否自增

rule               分片规则名，具体规则下文rule详细介绍

**dataNode**

属性             说明

name               节点名，与table中dataNode对应

datahost       物理数据库名，与datahost中name对应

database       物理数据库中数据库名

schema.xml文件dataHost中有三点需要注意：balance="1"，writeType="0" ,switchType="1"

schema.xml中的balance的取值决定了负载均衡对非事务内的读操作的处理。**balance 属性负载均衡类型，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,即读读写请求都发送到writeHost上。

2.balance="1"：      读请求随机分发到当前writeHost对应的readHost和standby的writeHost上(standby是备选主服务器，以备当前主服务器宕机后充当主服务器)。即全部的readHost与stand by writeHost 参与select 语句的负载均衡，简单的说，当双主双从模式(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"：      读请求随机分发到当前dataHost内所有的writeHost和readHost上。即所有读操作都随机的在writeHost、 readhost 上分发。

4.balance="3"：      读请求随机分发到当前writeHost对应的readHost上。即所有读请求随机的分发到 wiriterHost 对应的 readhost 执行, writerHost 不负担读压力，注意 balance=3 只在 1.4 及其以后版本有，1.3 没有。

**writeType 属性，负载均衡类型，目前的取值有 3 种**

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，重新启动后已切换后的为准，切换记录在配置文件中:dnindex.properties.

2.writeType="1"   所有写操作都随机的发送到配置的 writeHost。

3.writeType="2"   没实现。

**对于事务内的SQL默认走写节点**

1.以 /\*balance\*/ 开头，可以指定SQL使用特定负载均衡方案。例如在大环境开启读写分离的情况下，特定强一致性的SQL查询需求；

2.slaveThreshold：近似的主从延迟时间（秒）Seconds\_Behind\_Master <slaveThreshold ，读请求才会分发到该Slave，确保读到的数据相对较新。

3.schema.xml中的writeType的取值决定了负载均衡对写操作的处理：

4.writeType="0"：所有的写操作都发送到配置文件中的第一个write host。（第一个write host故障切换到第二个后，即使之后修复了仍然维持第二个为写库）。推荐取0值，不建议修改.

**主从切换（双主failover）：**switchType 属性****

如果细心观察schem.xml文件的话，会发现有一个参数：switchType，如下配置：

 <dataHost name="237\_15" maxCon="1000" minCon="10" balance="1" writeType="0" dbType="mysql" dbDriver="native"switchType="1"  slaveThreshold="100">

参数解读

1.switchType="-1"：  不自动切换

2.switchType="1"：   默认值，自动切换

3.switchType="2"：   基于MySQL主从同步的状态来决定是否切换。需修改heartbeat语句（即心跳语句）：show slave status

4.switchType="3"：   基于Mysql Galera Cluster（集群多节点复制）的切换机制。需修改heartbeat语句（即心跳语句）：show status like 'wsrep%'

**dbType属性**

指定后端连接的数据库类型，目前支持二进制的mysql协议，还有其他使用JDBC连接的数据库。例如：mongodb、oracle、spark等。

**dbDriver属性指定连接后端数据库使用的**

Driver，目前可选的值有native和JDBC。

1.使用native的话，因为这个值执行的是二进制的mysql协议，所以可以使用mysql和maridb。

2.其他类型的数据库则需要使用JDBC驱动来支持。从1.6版本开始支持postgresql的native原始协议。

3.如果使用JDBC的话需要将符合JDBC 4标准的驱动JAR包放到MYCAT\lib目录下，并检查驱动JAR包中包括如下目录结构的文件：

META-INF\services\java.sql.Driver。在这个文件内写上具体的Driver类名，例如：com.mysql.jdbc.Driver。

**heartbeat标签**

1.这个标签内指明用于和后端数据库进行心跳检查的语句。例如,MYSQL可以使用select user()，Oracle可以使用select 1 from dual等。

2.这个标签还有一个connectionInitSql属性，主要是当使用Oracla数据库时，需要执行的初始化SQL语句就这个放到这里面来。例如：altersession set nls\_date\_format='yyyy-mm-dd hh24:mi:ss'

3.1.4主从切换的语句必须是：show slave status

**writeHost标签、readHost标签**

1.这两个标签都指定后端数据库的相关配置给mycat，用于实例化后端连接池。

2.足系统的要求。

3.在一个dataHost内可以定义多个writeHost和readHost。但是，如果writeHost指定的后端数据库宕机，那么这个writeHost绑定的所有readHost都将不可用。

4.另一方面，由于这个writeHost宕机系统会自动的检测到，并切换到备用的writeHost上去。

dataHost

属性             说明

1.name               物理数据库名，与dataNode中dataHost对应

2.balance            均衡负载的方式

3.writeType      写入方式

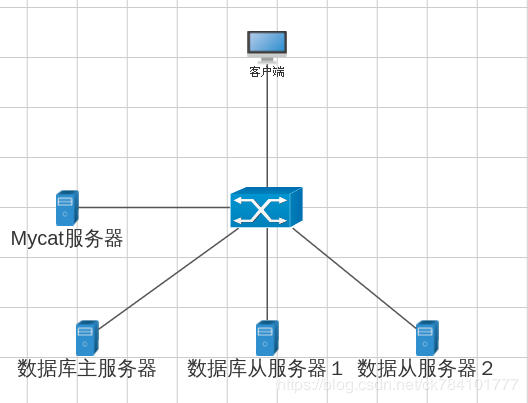
4.dbType             数据库类型

5.heartbeat      心跳检测语句，注意语句结尾的分号要加

## **二、主从同步配置与Mycat服务器配置**

拓扑结构如下，你需要准备５台虚拟机，ｉｐ配在同一网段即可

我们将为服务配置主从同步结构，为Mycat服务器安装mycat服务并修改配置文件以实现读写分离



1. ****配置主从服务器****

这里需要准备三台服务器，一台主服务器，二台从服务器准备好后保证三台主机能互相ping通，然后安装mysql服务。

主服务器配置：

1）启用binlog日志

]# vim /etc/my.cnf

[mysqld]

server\_id=51           　　 //server\_id

log-bin=master51       　 //日志名

:wq

]# systemctl restart mysqld

2）用户授权

用户名自定义、客户端地址使用% 或 只指定 从服务器的地址 都可以、只给复制数据的权限即可。

]# mysql -uroot -p密碼

mysql> grant replication slave on \*.\* to repluser@"%" identified by "123qqq...A";

mysql>quit;

3）查看binlog日志信息

查看日志文件名 和 偏移量位置。

mysql> show master status\G;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

File: master51.000001 　　//日志名

Position: 441　　　　　　 //偏移量

Binlog\_Do\_DB:

Binlog\_Ignore\_DB:

Executed\_Gtid\_Set:

1 row in set (0.00 sec)

从服务器配置（两台都需要配置）：

1）指定server\_id

Server\_id值可以自定义，但不可以与主服务器相同。（取值范围1-255)

]# vim /etc/my.cnf

[mysqld]

server\_id=52 //server\_id值

:wq

]# systemctl restart mysqld //重启服务

2）确保与主服务器数据一致（如果是使用2台新部署的数据库服务器配置主从同步，此操作可以忽略）

密码写你登录数据库的密码，导入的时候如果

]# mysqldump -uroot –p密码  -A > /allbak.sql //在主服务器上备份全部数据

]# scp /allbak.sql root@192.168.4.52:/root/ //将备份文件拷贝给从服务器

]# mysql -uroot –p密码 数据库名 < /root/allbak.sql 　//从服务器使用备份文件恢复数据

3）指定主服务器信息

数据库管理员root本机登录，指定主服务器信息，其中日志文件名和偏移量 写allbak.sql文件记录的。

指定主服务器信息后，会将主服务器信息保存在/var/lib/mysql/master.info文件下，若需要修改，则有两种途径，一种是在mysql下通过命令修改，而是修改上述文件．修改前必须停止slave线程，命令：stop slave 修改完后开启线程 : start slave

]# mysql -uroot –p密码 //管理员root 本机登录

mysql> show slave status; //查看状态信息，还不是从服务器

Empty set (0.00 sec)

mysql> change master to 　　//指定主服务器

-> master\_host=“192.168.4.51”, 　//主服务器ip地址

-> master\_user=“repluser”,　　 //主服务器授权用户

-> master\_password=“123qqq…A”, 　//主服务器授权用户密码

-> master\_log\_file=“master51.000001 ”,　//主服务器日志文件，以主服务器显示的为主

-> master\_log\_pos=441;         　 //主服务器日志偏移量，切忌不可加双引号

-> start slave;   //启动线程

mysql> show slave status\G;　　　 //查看状态信息，主要查看两个线程是否启动

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Slave\_IO\_State: Waiting for master to send event

Master\_Host: 192.168.4.51 //主服务器ip地址

Master\_User: repluser

Master\_Port: 3306

Connect\_Retry: 60

Master\_Log\_File: master51.000001

Read\_Master\_Log\_Pos: 437

Relay\_Log\_File: host52relay-bin.000002

Relay\_Log\_Pos: 604

Relay\_Master\_Log\_File: master51.000001

Slave\_IO\_Running: Yes        //IO线程yes状态

Slave\_SQL\_Running: Yes        //SQL线程yes状态

...........................

４）在主数据库上添加授权用户

只需要在主服务器上插入即可，因为你已经配置了主从同步

mysql> grant all on \*.\* to adminplj@"%" identified by "123qqq...A" ;

2.配置Mycat服务器

准备一台主机做Mycat服务器

修改mycat的schema.xml：

balance为1：让全部的readHost及备用的writeHost参与select的负载均衡。

switchType为2：基于MySQL主从同步的状态决定是否切换。

heartbeat：主从切换的心跳语句必须为show slave status。

１）修改server.xml

]# vim /usr/local/mycat/conf/server.xml

<user name="root">

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

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

</user>

<user name="user">

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

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

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

</user>

２）修改 schema.xml

如果不知道配置标签的含义请往上翻

我这里只配置了一个数据节点，一个数据主机（一主writeHost二从readHost）

<?xml version="1.0"?>

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

<mycat:schema xmlns:mycat="http://io.mycat/">

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

        </schema>

        <dataNode name="dn1" dataHost="localhost1" database="db1" />

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

                <heartbeat>select user()</heartbeat>

                <writeHost host="hostM1" url="192.168.4.100:3306" user="adminplj" password="123qqq...A">

                <readHost host="hostS1" url="192.168.4.110:3306" user="adminplj" password="123qqq...A" > </readHost>

　　　　<readHost host="hostS1" url="192.168.4.120:3306" user="adminplj" password="123qqq...A" > </readHost>

                </writeHost>

        </dataHost>

</mycat:schema>

上面配置中，balance改为1，表示读写分离。

以上配置达到的效果就是192.168.4.100为主库，192.168.4.110和192.168.4.120为从库。

MyCat支持双主多从，如果有N个主，那么就配置N个writeHost兄弟节点；如果有M个从节点，那么就配置M个readHost节点即可。

也可以有多台MySQL服务器，或者SQL Server、Oracle等，配置多个dataHost节点就可以。

需要注意的是：

Mycat主从分离只是在读的时候做了处理，写入数据的时候，只会写入到writehost，需要通过mycat的主从复制将数据复制到readhost！这个问题需要弄明白！！

如果没有提前做mysql主从复制，会发现Mycat读写分离配置后，数据写入writehost后，readhost一直没有数据！因为Mycat就没有实现主从复制的功能，毕竟数据库本身自带的这个功能才是最高效稳定的。

3)开启mycat服务

[root@mycat56 ~]/usr/local/mycat/bin/mycat　--help   //查看一下可以用的命令

Usage /usr/local/mycat/bin/mycat {console | start | stop | restart | status | dump}

[root@mycat56 ~]# /usr/local/mycat/bin/mycat start

Starting Mycat-server...

查看服务状态

[root@mycat56 ~]# netstat -utnlp | grep :8066 //查看端口

tcp6 0 0 :::8066 :::\* LISTEN 2924/java

[root@mycat56 ~]# ps -C java //查看进程

PID TTY TIME CMD

2924 ? 00:00:01 java

3.Mycat连通性测试(重要)

测试授权用户：在mycat主机，使用授权用户分别连接3台数据库服务器，若连接失败，请检查数据库服务器是否有对应的授权用户。确认三台数据库服务器可以使用

[root@mycat56 ~]# which mysql || yum -y install mariadb //安装提供mysql命令的软件包

//连接数据库服务器192.168.4.110

[root@mycat56 ~]# mysql -h192.168.4.110 -uadminplj -p123qqq...A

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 54

Server version: 5.7.17 MySQL Community Server (GPL)

mysql> exit; //连接成功 断开连接

Bye

//连接数据库服务器192.168.4.120

[root@mycat56 ~]# mysql -h192.168.4.120 -uadminplj -p123qqq...A

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 47

Server version: 5.7.17 MySQL Community Server (GPL)

mysql> exit; //连接成功 断开连接

Bye

//连接数据库服务器192.168.4.130

[root@mycat56 ~]# mysql -h192.168.4.130 -uadminplj -p123qqq...A

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 47

Server version: 5.7.17 MySQL Community Server (GPL)

mysql> exit; //连接成功 断开连接

Bye

4.读写分离测试

讲解一下测试流程：想确定读写是否分离，在从服务器上插入一条数据，如果在mycat上可以读到，则说明读写分离成功．

你在从服务器上插入数据在主服务器是读不到的，但是mycat读到了，说明读请求确实交给了从服务器

1)在从服务器上插入数据

原数据库上只有一条数据，再插入一条数据

[root@mysql2 opt]# mysql -uroot -p123qqq...A

mysql> use db1

mysql> insert into test value(2);

Query OK, 1 row affected (0.05 sec)

mysql> select \*from test;

+------+

| id   |

+------+

|    1 |

|    2 |

+------+

2 rows in set (0.00 sec)

2)连接mycat服务器读取

读的到证明读写分离成功，如果觉得还不确定就在数据库主服务器上查看是否记录

[root@client opt]# mysql -h192.168.4.140 -P8066 -p123456

MySQL [TESTDB]> select \*from test;　　//在插入之前读

+------+

| id   |

+------+

|    1 |

+------+

1 row in set (0.01 sec)

MySQL [TESTDB]> select \*from test;　//插入之后读

+------+

| id   |

+------+

|    1 |

|    2 |

+------+

2 rows in set (0.01 sec)

MySQL [TESTDB]> exit

Bye