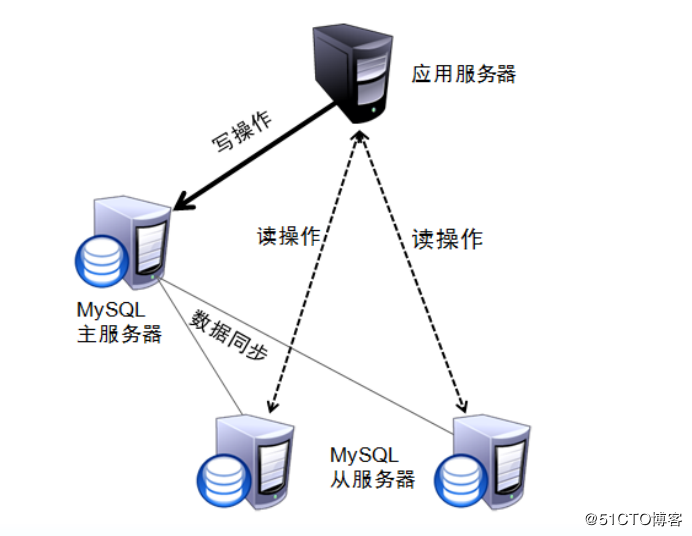
1. mysql主从复制与读写分离

[原创](javascript:;)[李佳良](http://blog.51cto.com/13555423)[2018-02-02 11:34:13](javascript:;)[评论(0)](http://blog.51cto.com/13555423/2068071#comment)[512人阅读](javascript:;)

**主从复制以及主从复制的作用：  
在实际的生产环境中，对数据库的读和写都在同一个数据库服务器中，是不能满足实际需求的，通过主从复制的方式来同步数据，再通过读写分离来提升数据库的并发负载能力**

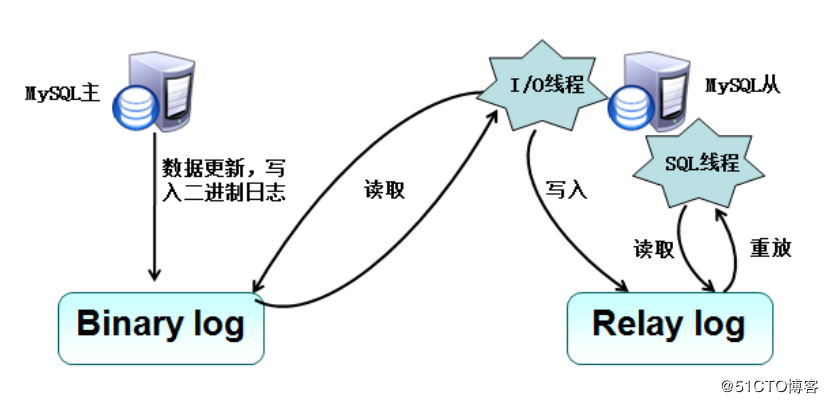
  
.  
**mysq支持的复制类型**

* 基于语句的复制。在服务器上执行sql语句，在从服务器上执行同样的语句，mysql默认采用基于语句的复制，执行效率高。
* 基于行的复制。把改变的内容复制过去，而不是把命令在从服务器上执行一遍。
* 混合类型的复制。默认采用基于语句的复制，一旦发现基于语句无法精确复制时，就会采用基于行的复制。  
  .  
  **复制的工作过程**  
  .
  + 在每个事务更新数据完成之前，master在二进制日志记录这些改变。写入二进制日志完成后，master通知存储引擎提交事务。
* Slave将master的binary log复制到其中继日志。首先slave开始一个工作线程（I/O），I/O线程在master上打开一个普通的连接，然后开始binlog dump process。binlog dump process从master的二进制日志中读取事件，如果已经跟上master，它会睡眠并等待master产生新的事件，I/O线程将这些事件写入中继日志。
* Sql slave thread（sql从线程）处理该过程的最后一步，sql线程从中继日志读取事件，并重放其中的事件而更新slave数据，使其与master中的数据一致，只要该线程与I/O线程保持一致，中继日志通常会位于os缓存中，所以中继日志的开销很小。  
  .  
  **前较为常见的Mysql读写分离分为以下两种：**  
  .
* **基于程序代码内部实现**

在代码中根据select 、insert进行路由分类，这类方法也是目前生产环境下应用最广泛的。优点是性能较好，因为程序在代码中实现，不需要增加额外的硬件开支，缺点是需要开发人员来实现，运维人员无从下手。  
.

* **基于中间代理层实现**

代理一般介于应用服务器和数据库服务器之间，代理数据库服务器接收到应用服务器的请求后根据判断后转发到，后端数据库，有以下代表性的程序。  
.  
**环境介绍；**  


安装mysql过程不做解释  
**mysql主从复制**  


**（mysql。主）**

[root@centos1 ]*# mount /dev/cdrom /media/*

[root@centos1 ]*# yum -y install ntp*

.

[root@centos1 ]*# vim /etc/ntp.conf*

*#restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap*

server 127.127.1.0 /添加

fudge 127.127.1.0 stratum 8 /添加

.

重启ntpd服务

[root@centos1 ]*# service ntpd restart*

.

防火墙开例外

[root@centos1 ]*# iptables -I INPUT -p udp --dport 123 -j ACCEPT*

[root@centos1 ]*# iptables -I INPUT -p udp --dport 3306 -j ACCEPT*

在(节点A)进行时间同步

[root@centos1 ]*# yum -y install ntpdate*

[root@centos1 ]*# /usr/sbin/ntpdate 192.168.1.30*

在(节点B)进行时间同步

[root@centos1 ]*# yum -y install ntpdate*

[root@centos1 ]*# /usr/sbin/ntpdate 192.168.1.30*

.  
**（mysql。主）**

修改/etc/my.cnf配置文件

[root@centos1 ]*# vim /etc/my.cnf*

server-id = 11 //mysql数据的唯一标示（不能重复）

log-slave-updates=true //允许连级复制 （增加）

log-bin=master-bin //二进制文件名（修改）

.

重启mysql服务

[root@centos1 ]*# service mysqld restart*

.

登陆mysql给从服务器授权

[root@centos1 ]# mysql -u root -p

.

**mysql>** GRANT REPLICATION SLAVE ON \*.\* TO 'lijialiang'@'192.168.1.%' IDENTIFIED BY '123456';

.

**mysql>** FLUSH PRIVILEGES;

.

mysql> show master status;

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

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

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

| master-bin.000001 | 558 | | |

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

1 row **in** set (0.03 sec)

.  
**(节点B，C)**

在/etc/my.cnf中修改以下内容

[root@centos3 ~]**# vim /etc/my.cnf**

server-id = 22 *//不能与其他实例重复*

log-bin=mysql-bin *//二进制日志文件名 修改*

relay-log=relay-log-bin *//复制过来的二进制文件名，增加*

relay-log-index=slave-relay-bin.index *//中继日志存放的文件名称，增加*

.

重启mysql服务

[root@centos1 ]*# service mysql restart*

.  
登录mysql 配置同步

[root@centos1 ]*# mysql -u root -p*

mysql> change master to

master\_host='192.168.1.30',master\_user='lijialiang',master\_password='123456',master\_log\_file='master-bin.000001',master\_log\_pos=558;

IP地址、用户、密码都master的数据库信息

.

启动同步

**mysql>** start slave;

.

查看slave状态确保以下两个值为YES

.

mysql> show slave status\G

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

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

Master\_Host: 192.168.1.30

Master\_User: lijialiang

Master\_Port: 3306

Connect\_Retry: 60

Master\_Log\_File: master-bin.000001

Read\_Master\_Log\_Pos: 558

Relay\_Log\_File: relay-log-bin.609531

Relay\_Log\_Pos: 254

Relay\_Master\_Log\_File: master-bin.000001

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

.  
**（mysql。主）**  
.

mysql> show databases;

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

| Database |

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

| information\_schema |

| mysql |

| performance\_schema |

| test |

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

5 rows **in** set (0.11 sec)

.

mysql> create datebase IT;

.

mysql> show databases;

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

| Database |

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

| information\_schema |

| IT |

| mysql |

| performance\_schema |

| test |

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

5 rows **in** set (0.11 sec)

.  
**（节点B，C）**

mysql> show databases;

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

| Database |

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

| information\_schema |

| IT |

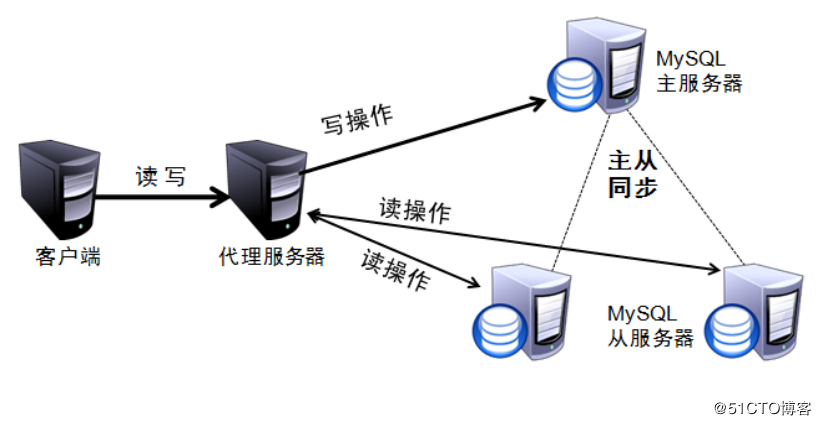
| mysql |

| performance\_schema |

| test |

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

5 rows **in** set (0.27 sec)

.  
**读写分离**  


**（amoeba）**  
读写分离安装java环境（amoeba软件基于java平台运行）  
运行jdk

[root@centos1 ]*# umount /dev/cdrom /media/*

[root@centos1 ]*# mount /dev/cdrom /media/*

[root@centos1 ]*# cp jdk-6u14-linux-x64.bin /usr/local/*

[root@centos1 ]*# cd /usr/local/*

[root@centos1 ]*# chmod +x jdk-6u14-linux-x64.bin*

[root@centos1 ]*# ./jdk-6u14-linux-x64.bin*

安装过程中提示（yes/no),我们要选择yes安装

.

修改/etc/profile配置文件，增加以下配置

[root@centos1 ]*# mv jdk1.6.0\_14/ /usr/local/jdk1.6*

[root@centos1 ]*# vim /etc/profile*

.

export JAVA\_HOME=/usr/local/jdk1.6 //设置jdk的根目录

export CLASSPATH=$CLASSPATH:$JAVA\_HOME/lib:$JAVA\_HOME/jre/lib //将jdk的程序文件赋予CLASSPATH变量

export PATH=$JAVA\_HOME/lib:$JAVA\_HOME/jre/bin:$PATH:$HOME/bin //将jdk的程序文件赋予PATH变量

export AMOEBA\_HOME=/usr/local/amoeba/ //定义AMOEBA的根目录

export PATH=$PATH:$AMOEBA\_HOME/bin //将amoeba的程序文件复制给PATH变量

.  
执行脚本

[root@centos1 ]*# source /etc/profile*

[root@centos1 ]*# java -version*

.  
安装并配置Amoeba软件  
安装

[root@centos1 ]*# mkdir /usr/local/amoeba*

[root@centos1 ]*# cd /media/*

[root@centos1 ]*# ls*

[root@centos1 ]*# tar zxf amoeba-mysql-binary-2.2.0.tar.gz -C /usr/local/amoeba/*

[root@centos1 ]*# chmod -R 755 /usr/local/amoeba/*

.  
出现以下内容说明安装成功了

[root@centos1 ]*# /usr/local/amoeba/bin/amoeba*

amoeba start|stop

.

配置**amoeba**读写分离，两个从节点读负载均衡

在主从服务器上开放权限给**amoeba**（三台服务器上都做相同设置，这里以一台为例）

**mysql**> **grant** **all** **on** \*.\* **to** **liang**@'**192**.**168**.**1**.%' identified by '123456';

.  
修改amoeba.xml文件  
注意：所有配置文件注释都是以 <!-- 内容 -->，再删除注释时请将内容也一并删除，最好是删除正行，但是有些时候只需要删除头和尾即可，里面的配置项是可以直接使用的。这个配置文件需要定义两个配置，第一是应用程序使用什么用户连接amoeba访问到后端的mysql数据库，第二个是定义默认写池以及读池。

[root@centos1 ]# cd /usr/local/amoeba/

[root@centos1 ]# vim conf/amoeba.xml

<bean class="com.meidusa.amoeba.mysql.server.MysqlClientAuthenticator">

<property

name="user">amoeba</property> //这里的帐户名和密码在后面链接amoeba使用

<property

name="password">123456</property>

<property name="filter">

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

<property name="LRUMapSize">1500</property>

<property name="defaultPool">master</property> //修改为master

//注意：这里原有注释，需要删除

<property name="writePool">master</property> //修改为master

<property name="readPool">slaves</property> //修改为slaves

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

.

编辑dbServers.xml

[root@centos1 ]# vim conf/dbServers.xml

*<!-- mysql schema -->*

<property name="user">liang</property> //之前设置开放权限的用户名和密码

//注意删掉此位置的注释

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

//注意删掉此位置的注释

</factoryConfig>

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

<dbServer name="master" parent="abstractServer"> //修改为master

<factoryConfig>

*<!-- mysql ip -->*

<property name="ipAddress">192.168.1.30</property> //修改IP

</factoryConfig>

</dbServer>

<dbServer name="slave1" parent="abstractServer"> //修改为slave1

<factoryConfig>

*<!-- mysql ip -->*

<property name="ipAddress">192.168.1.40</property> //修改IP

</factoryConfig>

</dbServer>

<dbServer name="slave2" parent="abstractServer"> //复制一份，修改为slave2

<factoryConfig>

*<!-- mysql ip -->*

<property name="ipAddress">192.168.1.50</property> //修改IP

</factoryConfig>

</dbServer>

<dbServer name="slaves" virtual="true"> //修改为slaves

<poolConfig class="com.meidusa.amoeba.server.MultipleServerPool">

*<!-- Load balancing strategy: 1=ROUNDROBIN , 2=WEIGHTBASED , 3=HA-->*

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

*<!-- Separated by commas,such as: server1,server2,server1 -->*

<property name="poolNames">slave1,slave2</property> //修改为slave1，slave2

</poolConfig>

</dbServer>

</amoeba:dbServers>

.  
.  
启动amoeba软件

[root@centos1 ]*# bin/amoeba start&*

[root@centos1 ]*# netstat -anpt | grep java*

.

测试读写分离  
打开一台客户端192.168.1.10，也需要安装mysql，作为测试机，可以使用yum -y install mysql安装。  
.

建立防火墙规则

[root@centos1 /]*# iptables -I INPUT -p tcp --dport 8066 -j ACCEPT*

.

amoeba，主服务器和两个从节点，都需要开放3306端口入站

[root@centos1 /]*# iptables -I INPUT -p tcp --dport 3306 -j ACCEPT*

[root@centos1 /]*# service iptables save*

.  
也可修改amoeba的amoeba.xml配置文件的第一个8066改为3306之后建立一条防火墙规则为允许3306端口入站并且重新启动amoeba服务  
<property name="port">3306</property> //在11行  
这样一来测试机访问的时候后方就不用跟 P 8066 了  
.

重启服务

[root@centos1 ]*# bin/amoeba restart&*

**测试阶段**  
**（客户端）**

[root@centos1 ]# **mysql** **-u** **amoeba** **-p** 123456 **-h** 192.168.1.20 **P** 8066

密码123456，为：之前登陆**amoeba**设置的密码

在主服务器master上创建一个数据库WLZS，同步到各从服务器上，然后关掉从服务器的slave功能，再插入数据。  
**（主服务器）**

**mysql>** create database WLGCSZS;

Query OK, 1 row affected (0.84 sec)

mysql> **use** **WLGCSZS**;

Database changed

mysql> create table **student**(id **int**,name **char**(10));

Query OK, 0 rows **affected** (1.39 sec)

.  
**（从服务器查看同步）**

mysql> show databases;

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

| Database |

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

| information\_schema |

| WLGCSZS |

| IT |

| mysql |

| performance\_schema |

| test |

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

5 rows **in** set (0.27 sec)

.  
**（主服务器写入数据）**

mysql> insert **into** student **values**(1,'li');

Query OK, 1 row **affected** (0.48 sec)

.  
**(从服务器也同步了)**

mysql> use WLGCSZS;

Database changed

mysql> select \* from student;

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

| id | name |

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

| 1 | li |

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

1 row **in** set (0.45 sec)

.  
在两台从服务器执行stop slave  
分别在两台从服务器写入不同数据  
**（节点A）**

**mysql>** stop slave;

Query OK, 0 rows affected (0.05 sec)

**mysql>** use WLGCSZS;

Database changed

**mysql>** insert into student values(5,'jia');

Query OK, 1 row affected (0.07 sec)

.  
**（节点B）**

**mysql>** stop slave;

Query OK, 0 rows affected (0.05 sec)

**mysql>** use WLGCSZS;

Database changed

**mysql>** insert into student values(6,'liang');

Query OK, 1 row affected (0.07 sec)

**(客户端查询)**

mysql> use WLGCSZS;

Database changed

mysql> select \* from student; //第一次查询显示第一台从服务器的用户

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

| id | name |

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

| 1 | li |

| 4 | jia |

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

2 rows **in** set (0.00 sec)

mysql> select \* from student; //第一次查询显示第二台从服务器的用户，说明负载均衡成功

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

| id | name |

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

| 1 | li |

| 5 | liang |

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

2 rows **in** set (0.00 sec)

**（客户端写入一条语句）**

mysql> insert **into** student **values**(6,'asdf');

Query OK, 1 row **affected** (0.05 sec)

.  
查询不到刚写入的语句，只有在主服务器可以查到因为写操作只有master有另外两台负责读取数据

mysql> select \* from student;

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

| id | name |

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

| 1 | jia |

| 4 | abcd |

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

2 rows **in** set (0.00 sec)

**（客户端查询）**

mysql> use WLGCSZS;

mysql> select \* from student;

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

| id | name |

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

| 1 | li |

| 5 | jia |

| 6 | liang |

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

4 rows **in** set (0.00 sec)