



MySQL读写分离技术

来源：来自网络 发布于： 2017-11-2

来自于[要资料](#) ★★☆☆ 57 次浏览 评价：好 中 差

阅读目录

- 1、简介
- 2、基本环境
- 3、配置主从复制
- 4、MySQL读写分离配置
 - 4.1、安装lua
 - 4.2、安装mysql-proxy
- 5、MySQL读写分离测试
 - 1)、修改rw-splitting.lua文件
 - 2)、修改完成后，启动mysql-proxy
 - 3)、创建用于读写分离的数据库连接用户
 - 4)、测试登陆账号proxy1@192.168.95.13进行添加数据
 - 5)、关闭12mysql的从复制
 - 6)、证明写分离
 - 7)、证明读分离
- 6、建议

1、简介

当今MySQL使用相当广泛，随着用户的增多以及数据量的增大，高并发随之而来。然而我们有很多办法可以缓解数据库的压力。分布式数据库、负载均衡、读写分离、增加缓存服务器等等。这里我们将采用读写分离技术进展缓解数据库的压力。

其中实现读写分离的技术有很多方法，这里我们将采用mysql-proxy这个中间软件来实现。这个软件中含有一个读写分离的lua文件，这也是我们使用mysql-proxy实现读写分离必用的文件，它需要lua解析器进行解析。因此我们还需要安装一个lua解析器。

2、基本环境

三台linux虚拟主机

Linux版本CentOS6.6、MySQL 5.5

mysql-proxy-0.8.5

lua-5.1.4

ip: 192.168.95.11（写）、192.168.95.12（读）、192.168.95.13（mysql-proxy）

3、配置主从复制

详细可以参考：[mysql主从复制与主主复制](#)

http://www.cnblogs.com/phpstudy2015-6/p/6485819.html#_label2

粗略介绍一下数据库的主从复制的配置：

第一步：

在192.168.95.11中创建一个192.168.95.12主机中可以登录的MySQL用户

用户：mysql12

密码：mysql12

```
mysql>GRANT REPLICATION SLAVE ON *.* TO 'mysql12' @'192.168.95.12' IDENTIFIED BY 'mysql12' ;
```

```
mysql>FLUSH PRIVILEGES;
```

第二步:

查看192.168.95.11MySQL服务器二进制文件名与位置

```
mysql>SHOW MASTER STATUS;
```

```
mysql> SHOW MASTER STATUS;
+-----+-----+-----+-----+
| File           | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+-----+-----+-----+-----+
| mysql-bin.000124 | 586      |              |                  |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

第三步:

告知二进制文件名与位置

在192.168.95.12中执行:

```
mysql> change master to
-> master_host='192.168.95.11',
-> master_user='mysql12',
-> master_password='mysql12',
-> master_log_file='mysql-bin.000124',
-> master_log_pos=586;
```

第四步:

在192.168.95.12中

```
mysql>SLAVE START; #开启复制
```

```
mysql>SHOW SLAVE STATUS\G #查看主从复制是否配置成功
```

```
mysql> show slave status\G
***** 1. row *****
Slave_IO_State: Waiting for master to send event
Master_Host: 192.168.95.11
Master_User: mysql12
Master_Port: 3306
Connect_Retry: 60
Master_Log_File: mysql-bin.000124
Read_Master_Log_Pos: 107
Relay_Log_File: localhost-relay-bin.000111
Relay_Log_Pos: 253
Relay_Master_Log_File: mysql-bin.000124
Slave_IO_Running: Yes
Slave_SQL_Running: Yes
Replicate_Do_DB: aa
Replicate_Ignore_DB:
```

主从复制配置成功!

(注意: 上面Relicate_Do_DB:aa表示主从复制只针对数据库aa【这是我之前设置的就没改了】, 这里就不讲这个了, 要想去了解学医这个的话可以参考文章http://www.cnblogs.com/phpstudy2015-6/p/6485819.html#_label7)

4、MySQL读写分离配置

百度云下载: 链接: <http://pan.baidu.com/s/1s1T118L> 密码: 9j0m

回到顶部

4.1、安装lua

官网下载: <http://www.lua.org/download.html>

Lua 是一个小巧的脚本语言。Lua由标准C编写而成, 代码简洁优美, 几乎在所有操作系统和平台上都可以编译, 运行。

一个完整的Lua解释器不过200k, 在目前所有脚本引擎中, Lua的速度是最快的。这一切都决定了Lua是作为嵌入式脚本的最佳选择。

1)、安装lua需要依赖很多软件包。

可以通过rpm -qa | grep name检查以下软件是否安装:

```
gcc*, gcc-c++, autoconf*, automake*, zlib*, libxml*, ncurses-devel*, libmcrypt*,
libtool*, flex*, pkgconfig*, libevent*, glib*
```

若缺少相关的软件包, 可通过yum -y install方式在线安装, 或直接从系统安装光盘中找到并通过rpm -ivh方式安装。(我的话一般是直接在系统光盘软件库中找到直接rpm安装的, 有些找不到, 则先在网上下载然后在ftp传给linux再进行安装)

2)、依赖软件安装完毕后则进行编译安装lua

MySQL-Proxy的读写分离主要是通过rw-splitting.lua脚本实现的，因此需要安装lua。

官网下载：<http://www.lua.org/download.html>（下载源码包）

```
# wget http://www.lua.org/ftp/lua-5.1.4.tar.gz
# tar zxvf lua-5.1.4.tar.gz
# cd lua-5.1.4
# make linux
# make install
# export LUA_FLAGS="-I/usr/local/include" LUA_LIBS="-L/usr/local/lib -llua -ldl"
LDFLAGS="-lm"
（我安装的时候是直接光盘软件库中找到，直接rpm安装）
```

4.2、安装mysql-proxy

1)、首先查看linux版本确认是32位还是64为系统

查看linux内核版本

```
# cat /etc/issue
```

查看linux版本

```
# cat /proc/version
```

2)、按系统位数下载（上面百度云链接64位的文件）

3)、安装

```
# tar -zxvf mysql-proxy-0.8.5-linux-rhel5-x86-64bit.tar.gz
# mkdir /usr/local/mysql-proxy
# cp ./mysql-proxy-0.8.5-linux-rhel5-x86-64bit/* /usr/local/mysql-proxy
# cd /usr/local/mysql-proxy
```

```
[root@localhost /]# cd /usr/local/mysql-proxy/
[root@localhost mysql-proxy]# ls
bin include init.d lib libexec licenses log share
```

安装成功

5、MySQL读写分离测试

1)、修改rw-splitting.lua文件

修改默认连接，进行快速测试，不修改的话要达到连接数为4时才启用读写分离

```
#cp /usr/local/mysql-proxy/share/doc/mysql-proxy/rw-splitting.lua ./
```

```
[root@localhost mysql-proxy]# cp /usr/local/mysql-proxy/share/doc/mysql-proxy/rw-splitting.lua ./
[root@localhost mysql-proxy]# ls
bin include init.d lib libexec licenses log rw-splitting.lua share
```

2)、修改完成后，启动mysql-proxy

```
--- config
---
--- connection pool
if not proxy.global.config.rwsplit then
    proxy.global.config.rwsplit = {
        min_idle_connections = 1, 默认为4
        max_idle_connections = 1, 默认为8
        is_debug = false
    }
end
```

```
# cd /usr/local/mysql/bin
# ./mysql-proxy --proxy-read-only-backend-addresses=192.168.95.12:3306 --proxy-backend-addresses=192.168.95.11:3306 --proxy-lua-script=/usr/local/mysql-proxy/rw-splitting.lua &
```

参数：

```
--proxy-read-only-backend-addresses    #只读服务器地址（ip）
--proxy-backend-addresses              #服务器地址（主服务器）
--proxy-lua-script                     #lua脚本路劲
```

&

#表示后台执行

3)、创建用于读写分离的数据库连接用户

用户名: proxy1

密 码: 321

```
mysql>grant all on *.* to 'proxy1'@'192.168.95.13' identified by '321';
mysql>use aa;
mysql>create table tab1(id int auto_increment,name varchar(32) not null,primary
key(id));
```

【因为已经开启了主从复制所以，11、12主机mysql中都创建了这个用户】

4)、测试登录账号proxy1@192.168.95.13进行添加数据

可以使用任意ip客户端登录这个账号

在192.168.95.13登陆:

```
# ./mysql -u proxy1 -P4040 -h192.168.95.13 -p
```

```
mysql> use aa
Database changed
mysql> insert into tab1 (name) values('proxy1111');
Query OK, 1 row affected (0.04 sec)

mysql> insert into tab1 (name) values('proxy2222');
Query OK, 1 row affected (0.02 sec)
```

在两个mysql中查看结果: 一致

```
mysql> select*from tab1;
+----+-----+
| id | name |
+----+-----+
| 1  | proxy1111 |
| 3  | proxy2222 |
+----+-----+
2 rows in set (0.02 sec)
```

结果表明: 账号使用

(ps: id是自增长, 之前高主主复制的时候更改了配置文件, 还没更改回来, 就就将用着先吧)

回到顶部

5)、关闭12mysql的从复制

```
mysql> stop slave;
```

```
mysql> stop slave;
Query OK, 0 rows affected (0.03 sec)

mysql> show slave status\G
***** 1. row *****
Slave_IO_State:
Master_Host: 192.168.95.11
Master_User: mysql12
Master_Port: 3306
Connect_Retry: 60
Master_Log_File: mysql-bin.000125
Read_Master_Log_Pos: 383
Relay_Log_File: localhost-relay-bin.000114
Relay_Log_Pos: 529
Relay_Master_Log_File: mysql-bin.000125
Slave_IO_Running: No
Slave_SQL_Running: No
```

6)、证明写分离

使用proxy1@192.168.95.13账号打开多个客户端进行插入数据

打开三个mysql客户端分别插入2条数据:

```
mysql> insert into tab1 (name) values('stop_slave11111');
...
mysql> insert into tab1 (name) values('stop_slave6666');
```

查看:

分别登陆11mysql与12mysql查看aa.tab1中的数据

主数据库：

```
mysql> select*from tab1;
+----+-----+
| id | name |
+----+-----+
| 1  | proxy1111 |
| 3  | proxy2222 |
| 5  | stop_slave11111 |
| 7  | stop_slave2222 |
| 9  | stop_slave33333 |
| 11 | stop_slave444 |
| 13 | stop_slave5555 |
| 15 | stop_slave6666 |
+----+-----+
8 rows in set (0.00 sec)
```

从数据库：

```
mysql> select*from tab1;
+----+-----+
| id | name |
+----+-----+
| 1  | proxy1111 |
| 3  | proxy2222 |
+----+-----+
2 rows in set (0.05 sec)
```

结果中显示插入的数据存在与主数据库，而从数据库没有，所以证明写能够分离。

7)、证明读分离

使用proxy1@192.168.95.13账号登陆mysql，查看aa.tab1中的数据

```
mysql>use aa;
mysql>select*from tab1;
```

结果中显示只有从数据库的数据，结合上面的测试，可以证明读分离。

```
mysql> select*from tab1;
+----+-----+
| id | name |
+----+-----+
| 1  | proxy1111 |
| 3  | proxy2222 |
+----+-----+
2 rows in set (0.05 sec)
```

6、建议

为了方便启动与管理mysql-proxy可以创建mysql-proxy服务管理脚本

下面这个管理脚本仅适合以上我给出的安装路径位置

【此管理脚本需要按照自己的安装路径做出相应的修改方可使用】

```
#!/bin/sh
2
3 #
4 # mysql-proxy This script starts and stops the mysql-proxy daemon
5 #
6 # chkconfig: - 78 30
7 # processname: mysql-proxy
8 # description: mysql-proxy is a proxy daemon to mysql
9
10 # Source function library.
11 . /etc/rc.d/init.d/functions
12
13 #PROXY_PATH=/usr/local/bin
14 PROXY_PATH=/usr/local/mysql-proxy/bin
15
16 prog="mysql-proxy"
17
18 # Source networking configuration.
19 . /etc/sysconfig/network
20
21 # Check that networking is up.
22 [ ${NETWORKING} = "no" ] && exit 0
23
24 # Set default mysql-proxy configuration.
```

```

25 #PROXY_OPTIONS="--daemon"
26
27 PROXY_OPTIONS="--proxy-read-only-backend-addresses=192.168.95.12:3306 --proxy-
backend-addresses=192.168.95.11:3306 --proxy-lua-script=/usr/local/mysql-proxy/rw-
splitting.lua"
28
29 PROXY_PID=/usr/local/mysql-proxy/run/mysql-proxy.pid
30
31 # Source mysql-proxy configuration.
32 if [ -f /etc/sysconfig/mysql-proxy ]; then
33 . /etc/sysconfig/mysql-proxy
34 fi
35
36 PATH=$PATH:/usr/bin:/usr/local/bin:$PROXY_PATH
37 # By default it's all good
38 RETVAL=0
39
40 # See how we were called.
41 case "$1" in
42 start)
43 # Start daemon.
44 echo -n "$Starting $prog: "
45 $NICELEVEL $PROXY_PATH/mysql-proxy $PROXY_OPTIONS --daemon --pid-
file=$PROXY_PID --user=root --log-level=debug --log-file=/usr/local/mysql-
proxy/log/mysql-proxy.log
46 RETVAL=$?
47 echo
48 if [ $RETVAL = 0 ]; then
49 touch /var/lock/subsys/mysql-proxy]
50 echo "ok"
51 fi
52 ;;
53 stop)
54 # Stop daemons.
55 echo -n "$Stopping $prog: "
56 killproc $prog
57 RETVAL=$?
58 echo
59 if [ $RETVAL = 0 ]; then
60 rm -f /var/lock/subsys/mysql-proxy
61 rm -f $PROXY_PID
62 fi
63 ;;
64 restart)
65 $0 stop
66 sleep 3
67 $0 start
68 ;;
69 condrestart)
70 [ -e /var/lock/subsys/mysql-proxy ] && $0 restart
71 ;;
72 status)
73 status mysql-proxy
74 RETVAL=$?
75 ;;
76 *)
77 echo "Usage: $0 {start|stop|restart|status|condrestart}"
78 RETVAL=1
79 ;;
80 esac
81 exit $RETVAL

#--我将mysql-proxy服务管理脚本放在了/usr/local/mysql-proxy/init.d/文件夹里
#--给执行权限，建立相应目录
#chmod +x /usr/local/mysql-proxy/init.d/mysql-proxy
#mkdir /usr/local/mysql-proxy/run
#mkdir /usr/local/mysql-proxy/log

#cd /usr/local/mysql-proxy/init.d/
#--启动mysql-proxy
#./mysql-proxy start
#--停止mysql-proxy
#./mysql-proxy stop
#--重启mysql-proxy
#./mysql-proxy restart

```

一些相关参数:

```
PROXY_PATH=/usr/local/mysql-proxy/bin //定义mysql-proxy服务二进制文件路径

--proxy-read-only-backend-addresses=192.168.95.12:3306 //定义后端只读从服务器地址

--proxy-backend-addresses=192.168.95.11:3306 //定义后端主服务器地址

--proxy-lua-script=/usr/local/mysql-proxy/rw-splitting.lua //定义lua读写分离脚本路径

PROXY_PID=/usr/local/mysql-proxy/run/mysql-proxy.pid //定义mysql-proxy PID文件路径

--daemon //定义以守护进程模式启动

--keepalive //使进程在异常关闭后能够自动恢复【上面的管理脚本没有加上此参数】

--user=root //以root用户身份启动服务

--log-level=debug //定义log日志级别，由高到低分别有(error|warning|info|message|debug)

--log-file=/usr/local/mysql-proxy/log/mysql-proxy.log //定义log日志文件路径
```



相关文章

[我们该如何设计数据库](#)

[数据库设计经验谈](#)

[数据库设计过程](#)

[数据库编程总结](#)