

**《高可用与负载均衡》**

**课程设计报告**

**题 目：** 基于MHA+Proxy实现

读写分离和负载均衡web集群

**班 级： 网络工程 二 班**

**学 号： 20192245068**

**姓 名： 连子蒙**

**指导教师： 王瑞江**

**完成时间：** 2021年12月13日

**河北经贸大学信息技术学院**

目录

[1、需求分析 2](#_Toc2624)

[1.1、功能需求 2](#_Toc19307)

[1.2、性能和安全需求 2](#_Toc1028)

[2、概要设计 2](#_Toc4893)

[2.1、系统结构设计 2](#_Toc20775)

[2.2、服务器选型 3](#_Toc25677)

[2.3、软件清单](#_Toc3200) 4

[3、系统实现 5](#_Toc21158)

[3.1、本系统所使用的MHA架构 5](#_Toc22754)

[3.2、配置镜像 6](#_Toc15700)

[3.3、搭建MHA相关环境 7](#_Toc26368)

3.4、安装docker-compose软件 --------------------------------------- 10

3.5、创建MHA容器 11

3.6、进入主数据库 11

3.7、配置ProxySQL环境 14

3.8、在容器ip136内为ProxySQL添加远程登录用户 15

3.9、开启ProxySQL的web监控功能 16

3.10、windows浏览器访问验证 17

3.11、配置被监控的数据库 17

[4、系统测试 20](#_Toc28578)

[4.1、故障切换 20](#_Toc25783)

[5、总结 24](#_Toc14523)

[5.1、遇到的困难 24](#_Toc16987)

[5.2、心得体会 24](#_Toc30245)

**1.需求分析**

网站建设是企业、组织、个人宣传和分享交流的重要的平台，尤其是在如今的信息化时代下，所有行业都在互联网的影响下实现数字化、在线化。在网站的建设中，网站的部署和发布是非常重要的一环，尤其是企业网站，随着企业的影响力和业务规模的扩大，企业的服务器需要承载的访问量和负载也日益增多，这就对企业提供网络服务提出了新的标准。因此，为使得企业网络可以承载更多的访问量和业务需求，我们就需要搭建web集群，实现企业集群负载均衡和高可用。

**1.1功能需求**

1.1.1网站搭建

网站开发测试完成后，需要正式发布上线，此时，我们需要配置web服务器，提供网络服务。当访问量较少时，我们可以部署单台节点的服务器站点，同时对服务器配置定期备份任务以及定期维护。

1.1.2负载均衡

当访问量越来越多时，单台节点的服务器无法承受超负载的HTTP（s）请求时，此时为满足外部的访问量，同时为了方便管理、提高扩展性和安全性以及减少企业开支，通常就需要采用增加服务器部署站点配置负载均衡集群的方法，来提高企业网站的网络服务能力。

1.1.3高可用

当负载均衡的服务器节点越来越多时，网站规模也越来越大，负载均衡分发器承担的网络流量最大，其在整个集群中的重要性也就不言而喻，一旦节点出现故障甚至宕机，整个集群将无法对外提供服务。因此，除了负载均衡之外，我们同时会部署负载均衡的高可用，即负载均衡器的主备部署。

**1.2性能和安全性需求**

1.2.1、可扩展性：随着访问的增加，系统具备良好的伸缩能力

1.2.2、可视性：系统、服务的状态处于一个实时的监控之下

1.2.3、高性能高可靠性：经过优化的体系结构及合理的备份策略

1.2.4、安全性：结构上的安全及主机的安全策略基本思路

（1）对于访问频繁，用户量大的对象（bbs,blog）采用某种合理的方式负载到多个服务器上。把数据库独立出来，准备3套mysql数据库，以实现主从复制，即减轻负载，又提高了可靠性。更近一步，使用mysqlproxy技术，实现主从服务器的读写分离，大大提高这个系统的性能和负载能力。

（2）数据库与外部网络隔离，只允许web服务器对数据库进行读写。

**2、概要设计**

**2.1、系统结构设计**

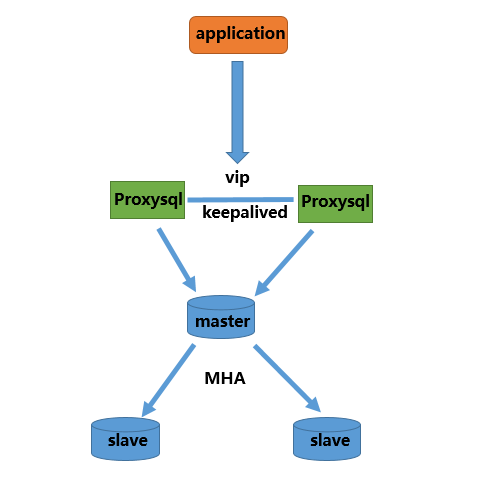
如图2-1所示。

图2-1 系统概要设计

**2.2、服务器选型**

2.2.1、负载均衡层的部件选择

　　本层需要服务器2台，并且不需要太高的配置即可胜任，使用市场上入门级的服务器即可满足要求，为了减少托管机柜的占用，使用1u尺寸的机架式服务器是最佳的选择。表2-1是某个生产环境负载均衡器的硬件配置。

2.2.2、应用层部件选择

　　应用层主要是运行web服务，它由apache整合php来完成。为了使用分布式共享文件系统，需要额外的软件来实现文件系统的挂接。要使apache发挥更好的性能，需要使用apache的worker模式（默认是prefork）。在worker模式下，通过查看进程数，可以发现开启的进程数远远少于默认的prefork模式。Worker模式需要4G以上的内存支持才能发挥较好的作用。

　　与负载均衡层所需的服务器硬件相比，所需的硬件要求差别不是太大。因此我们可以采用与负载均衡层相同的配置，增加内存到8G。从成本看，增加4G内存大概就是增加几百元的成本而已。前文描述过，应用层至少需要2个以上的服务器，建议初始配置使用3个服务器，以后根据业务需求再逐步扩充服务器的数量。反之也可以根据用户数量的降低减少服务器数量。表2-2是某个生产环境的服务器硬件配置。



　2.2.3、分布式文件系统及数据库层部件的选择

　　本层包含数据库和分布式文件系统。对于数据库来说，需要更快的处理能力，即频率更高的cpu和更大的内存；而对于分布式文件系统，则需要大容量的硬盘，以便可以存储更多的数据。

　　A、数据库硬件

　　主从复制最少要3个服务器，本方案初始设计使用3个服务器，一主二从。以后可以根据业务增长实现扩充，形成一主多从、速写分离的方式。这样既保证了数据安全，又能大幅提高负载。数据库服务器仍然使用1u机架式服务器，安装4核双cpu，16G内存，300G sas硬盘2个（做成raid 10）。数据库文件即可使用本地硬盘存储，也可以使用分布式文件系统。

　　B、分布式文件系统硬件

分布式文件系统由2个部分组成：元数据服务器master和数据存储服务器chunkserver。其中元数据服务器需要的配置要求不高，按负载均衡器那个标准配置就可以；数据存储服务器则尽可能地增加硬盘数量也获得最大的容量。初始配置时，使用一个元数据服务器，3个数据存储服务器。数据存储服务器采用2u机架式服务器，安装6个1T sata硬盘，做成RAID 5，再用一个单独的sata安装硬盘来安装操作系统，这样可以获得4.5T的可用磁盘空间来共享。3个服务器可提供总容量大概为13T存储空间。以后再随业务增长增加数据存储服务器。

**2.3、软件清单**

2.3.1、操作系统：CentOS7.9；

2.3.2、分布式文件系统：Moosefs；

2.3.3、web程序：APACHE；

2.3.4、数据库：MySQL, MySQL 是一款安全、跨平台、高效的，并与 PHP、Java 等主流编程语言紧密结合的数据库系统。

（1）使用 C 和 [C++](http://c.biancheng.net/cplus/) 编写，并使用多种编译器进行测试，保证源代码的可移植性。

（2）为多种编程语言提供了 API。这些编程语言包括 C、C++、[Python](http://c.biancheng.net/python/" \t "_blank)、[Java](http://c.biancheng.net/java/" \t "_blank)、Perl、PHP、Eiffel、Ruby 和 Tcl 等。

（3）支持多线程，充分利用 CPU 资源。

（4）优化的 SQL 查询算法，有效地提高查询速度。

（5）提供多语言支持，常见的编码如中文的 GB 2312、BIG 5，日文的 Shift\_JIS 等都可以用作数据表名和数据列名。

（6）提供 TCP/IP、ODBC 和 JDBC 等多种数据库连接途径。

（7）提供用于管理、检查、优化数据库操作的管理工具。

（8）支持大型的数据库。可以处理拥有上千万条记录的大型数据库。

（9）支持多种存储引擎。

2.3.5、高可用方案软件：MHA。

MHA（Master High Availability）目前在MySQL高可用方面是一个相对成熟的解决方案，是一套优秀的作为MySQL高可用性环境下故障切换和主从提升的高可用软件。在MySQL故障切换过程中，MHA能做到在10~30秒之内自动完成数据库的故障切换操作，并且在进行故障切换的过程中，MHA能在最大程度上保证数据的一致性，以达到真正意义上的高可用。

　　MHA还提供在线主库切换的功能，能够安全地切换当前运行的主库到一个新的主库中 (通过将从库提升为主库)，大概0.5-2秒内即可完成。

MHA优点总结

（1）自动故障转移快

（2）主库崩溃不存在数据一致性问题

（3）不需要对当前mysql环境做重大修改

（4）不需要添加额外的服务器(仅一台manager就可管理上百个replication)

（5）性能优秀，可工作在半同步复制和异步复制，当监控mysql状态时，仅需要每隔N秒向master发送ping包(默认3秒)，所以对性能无影响。你可以理解为MHA的性能和简单的主从复制框架性能一样。

（6）只要replication支持的存储引擎，MHA都支持，不会局限于innodb

MySQL中间件：ProxySQL。

ProxySQL 是一款可以实际用于生产环境的 MySQL 中间件，它有官方版和 percona 版两种。percona版是在官方版的基础上修改的，添加了几个比较实用的工具。生产环境建议用官方版。

ProxySQL 是用 C++ 语言开发的，虽然也是一个轻量级产品，但性能很好(据测试，能处理千亿级的数据)，功能也足够，能满足中间件所需的绝大多数功能，包括：

（1）最基本的读/写分离，且方式有多种。

（2）可定制基于用户、基于schema、基于语句的规则对SQL语句进行路由。换句话说，规则很灵活。基于schema和与语句级的规则，可以实现简单的sharding(分库分表)。

（3）可缓存查询结果。虽然ProxySQL的缓存策略比较简陋，但实现了基本的缓存功能，绝大多数时候也够用了。

（4）监控后端节点。ProxySQL可以监控后端节点的多个指标，包括：ProxySQL和后端的心跳信息，后端节点的read-only/read-write，slave和master的数据同步延迟性(replication lag)。

**3、系统实现**

**3.1、本系统所使用的MHA架构如表3-1所示。**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| IP | 主机名 | 作用 | ServerID | Port | 类型 | 备注 |
| 192.168.68.131 | MHA-LHR-Master1-ip131 | Master.node | 573306131 | 3306 | 写入 | 对外提供写服务 |
| 192.168.68.132 | MHA-LHR-Slave1-ip132 | Slave.node1  (Candicate.Master) | 573306132 | 读 | 备选Master提供读服务 |
| 192.168.68.133 | MHA-LHR-Slave2-ip133 | Slave.node2 | 573306133 | 读 | 提供读服务 |
| 192.168.68.134 | MHA-LHR-Montior1-ip134 | Monitor.host |  |  |  | 监控其它机器，一旦Mater宕机，将会把备选Master提升为Master,而将Slave指向新的Master |
| 192.168.68.135 |  | VIP |  | 3306 |  | 在131和132之间进行浮动漂移 |

**3.2、配置镜像**

3.2.1、下载镜像

docker pull registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-master1-ip131

docker pull registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-slave1-ip132

docker pull registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-slave2-ip133

docker pull registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-monitor-ip134

3.2.2、重命名镜像

docker tag registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-master1-ip131 lhrbest/mha-lhr-master1-ip131

docker tag registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-slave1-ip132 lhrbest/mha-lhr-slave1-ip132

docker tag registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-slave2-ip133 lhrbest/mha-lhr-slave2-ip133

docker tag registry.cn-hangzhou.aliyuncs.com/lhrbest/mha-lhr-monitor-ip134 lhrbest/mha-lhr-monitor-ip134

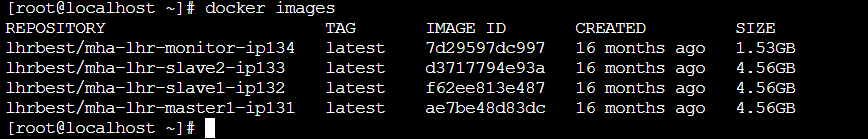
查看镜像，如图3-1所示。

图3-1 项目所需镜像

**3.3、搭建MHA相关环境**

准备工作

3.3.1、编辑docker-compose.yml文件

注意：yml文件的格式，对空格、缩进、对齐都有严格的要求。

3.3.2、创建存放docker-compose.yml文件的目录

# mkdir -p /root/mha

3.3.3、编辑文件 /root/mha/docker-compose.yml

（1）填入内容：

version: '3.8'

services:

MHA-LHR-Master1-ip131:

container\_name: "MHA-LHR-Master1-ip131"

restart: "always"

hostname: MHA-LHR-Master1-ip131

privileged: true

image: lhrbest/mha-lhr-master1-ip131

ports:

- "33131:3306"

- "2201:22"

networks:

mhalhr:

ipv4\_address: 192.168.68.131

MHA-LHR-Slave1-ip132:

container\_name: "MHA-LHR-Slave1-ip132"

restart: "always"

hostname: MHA-LHR-Slave1-ip132

privileged: true

image: lhrbest/mha-lhr-slave1-ip132

ports:

- "33132:3306"

- "2202:22"

networks:

mhalhr:

ipv4\_address: 192.168.68.132

MHA-LHR-Slave2-ip133:

container\_name: "MHA-LHR-Slave2-ip133"

restart: "always"

hostname: MHA-LHR-Slave2-ip133

privileged: true

image: lhrbest/mha-lhr-slave2-ip133

ports:

- "33133:3306"

- "2203:22"

networks:

mhalhr:

ipv4\_address: 192.168.68.133

MHA-LHR-Monitor-ip134:

container\_name: "MHA-LHR-Monitor-ip134"

restart: "always"

hostname: MHA-LHR-Monitor-ip134

privileged: true

image: lhrbest/mha-lhr-monitor-ip134

ports:

- "33134:3306"

- "2204:22"

networks:

mhalhr:

ipv4\_address: 192.168.68.134

networks:

mhalhr:

name: mhalhr

ipam:

config:

- subnet: "192.168.68.0/16"

docker-compose.yml文件具体配置如图3-2所示。

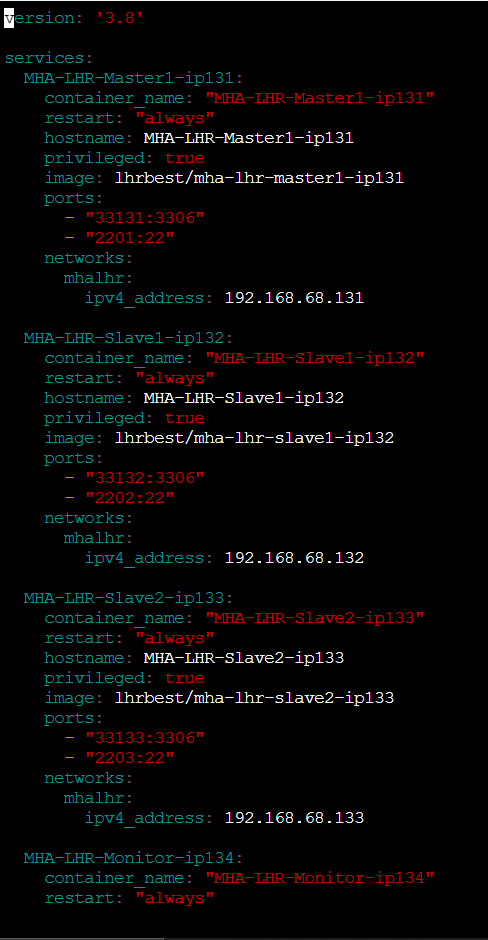
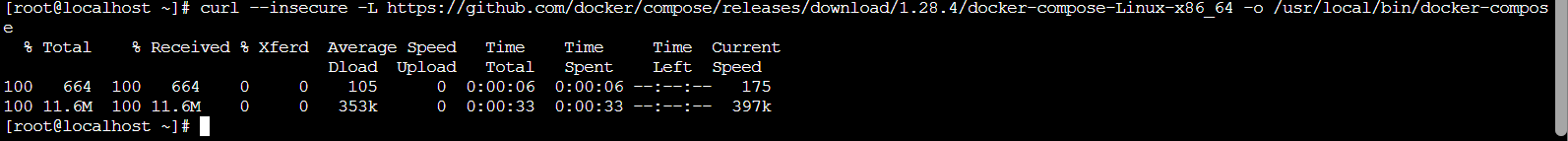


图3-2 docker-compose.yml配置文件

**3.4、安装docker-compose软件**

3.4.1、下载docker-compose，如图3-3所示。

# curl --insecure -L https://github.com/docker/compose/releases/download/1.28.4/docker-compose-Linux-x86\_64 -o /usr/local/bin/docker-compose

图3-3 docker-compose下载

3.4.2、为docker-compose添加执行权限，如图3-4所示。

# chmod +x /usr/local/bin/docker-compose

图3-4 docker-compose 添加执行权限

3.4.3、查看docker-compose版本，如图3-5所示。

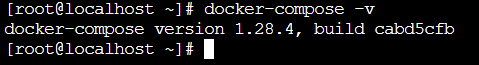
# docker-compose -v

图3-5 docker-compose版本

**3.5、创建MHA容器**

前提：进入/root/mha目录后进行相关操作

3.5.1、启动MHA环境的容器，如图3-6所示。

# cd /root/mha

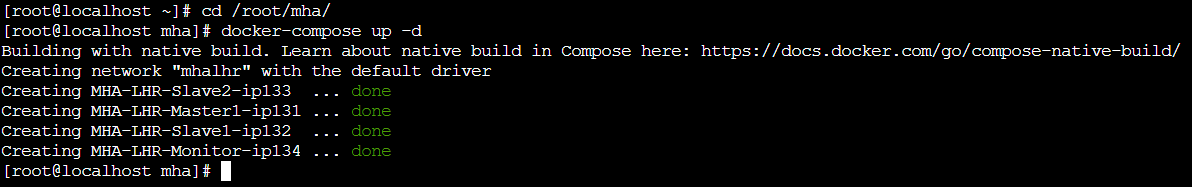
# docker-compose up -d

图3-5 启动MHA所需容器

4.5.2、查看容器，如图3-7所示。

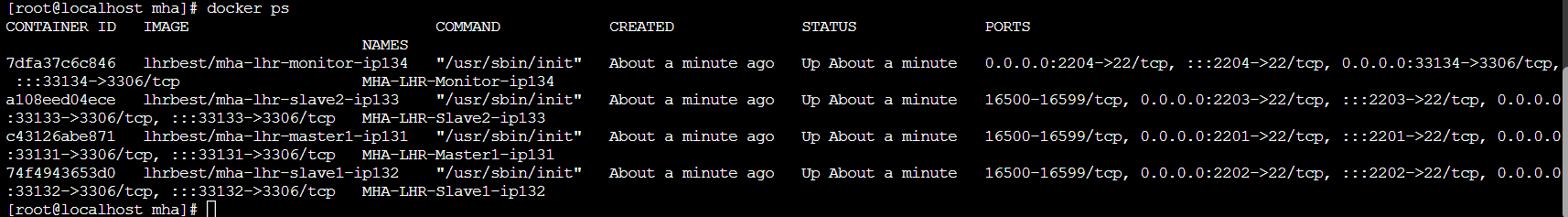
# docker ps

图3-7 MHA容器状态

**3.6、进入主数据库MHA-LHR-Master-ip131容器，添加VIP地址即192.168.68.135**

3.6.1、进入主数据库，如图3-8所示。

# docker exec -it MHA-LHR-Master1-ip131 bash



图3-8 进入主数据库MHA-LHR-Master1-ip131

3.6.2、添加VIP地址，如图3-9所示

# /sbin/ifconfig eth0:1 192.168.68.135/24

图3-9 添加VIP地址

3.6.3、查看VIP地址添加情况，如图3-10所示。

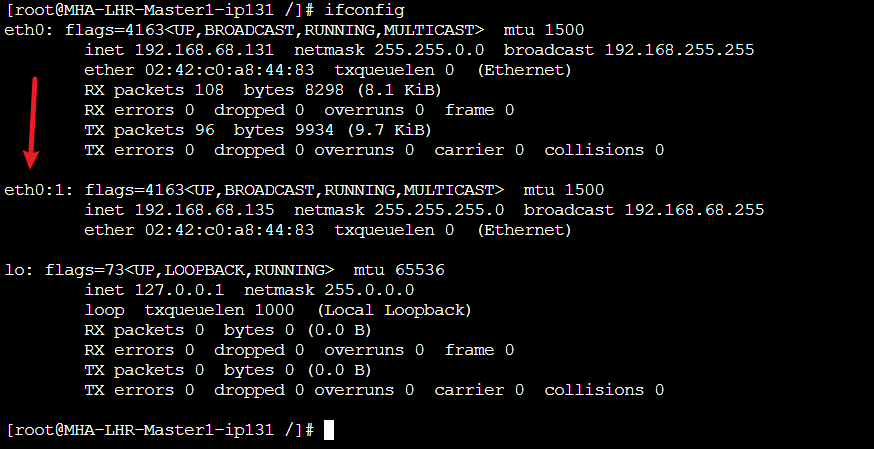
# ifconfig

图3-10 VIP地址添加情况

3.6.4、管理节点即MHA-LHR-Monitor-ip134容器与主数据库容器VIP地址联通测试，如图3-11所示。

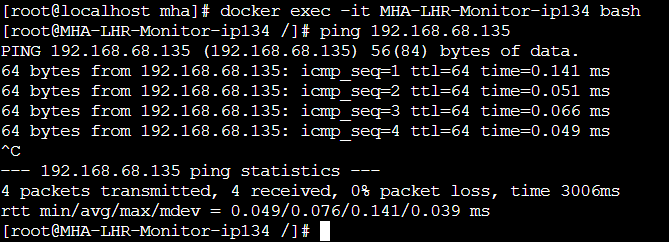
# ping 192.168.68.135

图3-11 联通测试

3.6.5、验证主从复制是否正确

（1）主数据库创建数据库、添加表、添加内容，如图3-12所示。

CREATE DATABASE huaqi;

CREATE TABLE user( \_id INT NOT NULL AUTO\_INCREMENT, name VARCHAR(40) NOT NULL, submission\_date DATE, PRIMARY KEY (\_id) );

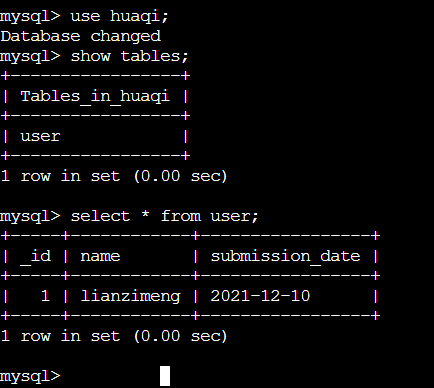
INSERT INTO user ( \_id, name, submission\_date ) VALUES ( 001, "lianzimeng", 20211210);

图3-12 主数据库添加内容

(2)从数据库MHA-LHR-Slave1-ip132查看数据库huaqi的表user内容，如图3-13。

docker exec -it MHA-LHR-Slave1-ip132 bash

show databases;

use huaqi;

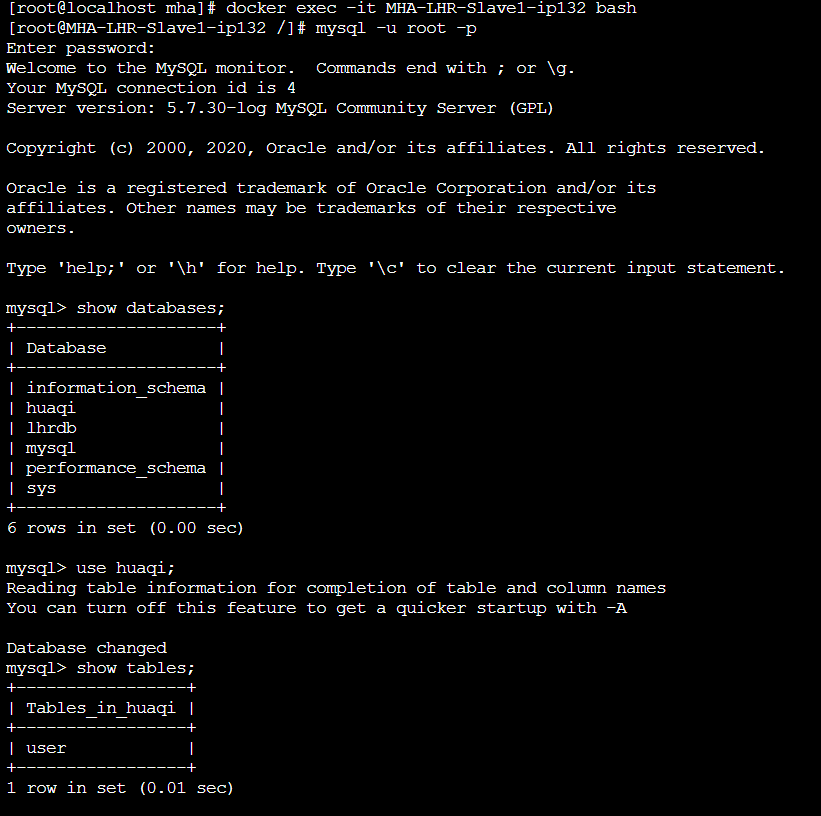
select \* from user;

图3-13 从数据库查看主从复制情况

**3.7、配置ProxySQL环境**

3.7.1、申请lhrcentos76容器

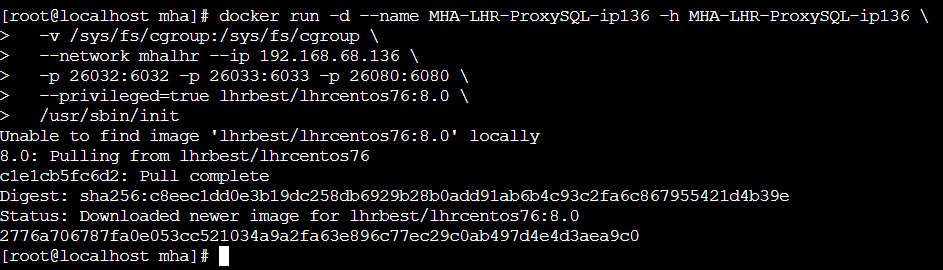
对其进行重命名、配置卷、设置ip地址、配置端口映射等配置并启动，如图3-14所示。

图3-14 申请lhrcentos76

3.7.2、为容器配置网络，如图3-15所示。

# docker network connect bridge MHA-LHR-ProxySQL-ip136

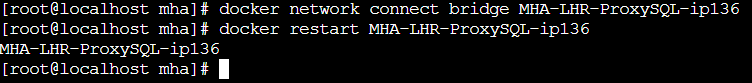
# docker restart MHA-LHR-ProxySQL-ip136

图3-15 配置容器lhrcentos76

3.7.3、宿主机下载proxySQL

（1）# yum -y localinstall proxysql-2.3.2-1-centos7.x86\_64.rpm

由于不知名原因，下载失败，遂于windows系统从github下载一份上传至宿主机。

（2）宿主机将proxysql.x86\_64 0:2.0.18-1复制给容器MHA-LHR-ProxySQL-ip136，如图3-16所示。

图3-16 下载proxySQL

3.7.4、在容器内配置proxySQL，如图3-17所示。

（1）进入容器MHA-LHR-ProxySQL-ip136

# docker exec -it MHA-LHR-ProxySQL-ip136 bash

（2）安装proxysql-2.3.2-1-centos7.x86\_64.rpm

# rpm -ivh proxysql-2.3.2-1-centos7.x86\_64.rpm

rpm参数解析：

-i：显示套件的相关信息；

-v：显示指令执行过程；

-h或--hash：套件安装时列出标记。

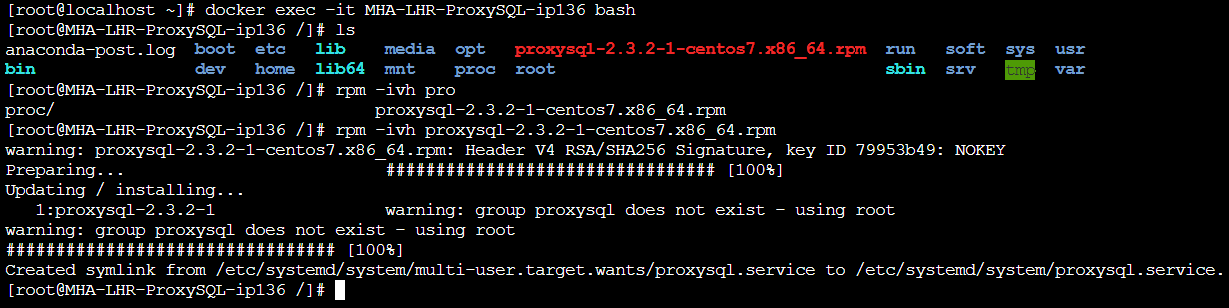


图3-17 容器内配置proxySQL

（3）启动proxysql

# systemctl start proxysql

（4）查看proxysql状态，如图3-18所示。

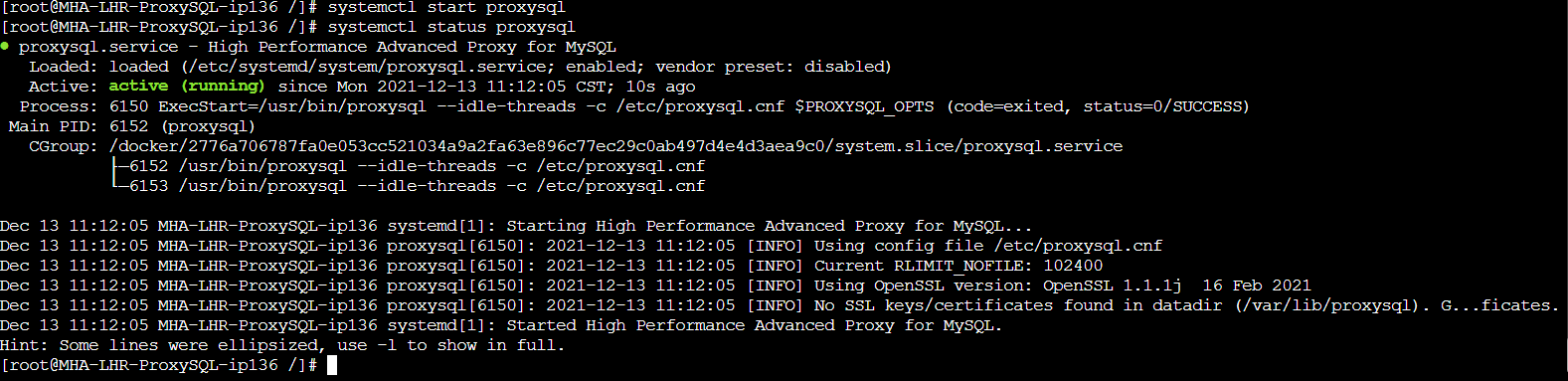
# systemctl status proxysql

图3-18 proxySQL状态

**3.8、在容器MHA-LHR-ProxySQL-ip136内为proxysql添加远程登录用户**

3.8.1、连接mysql数据库，如图3-19所示。

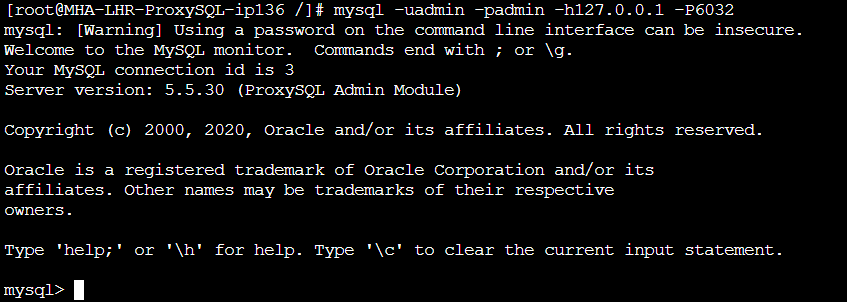
# mysql -uadmin -padmin -h127.0.0.1 -P6032

图3-19 MHA-LHR-ProxySQL-ip136容器内进入mysql

3.8.2、添加远程登录用户，如图3-20所示。

mysql> select @@admin-admin\_credentials;

mysql> set admin-admin\_credentials='admin:admin;root:lhr';

mysql> select @@admin-admin\_credentials;

mysql> load admin variables to runtime;

mysql> save admin variables to disk;

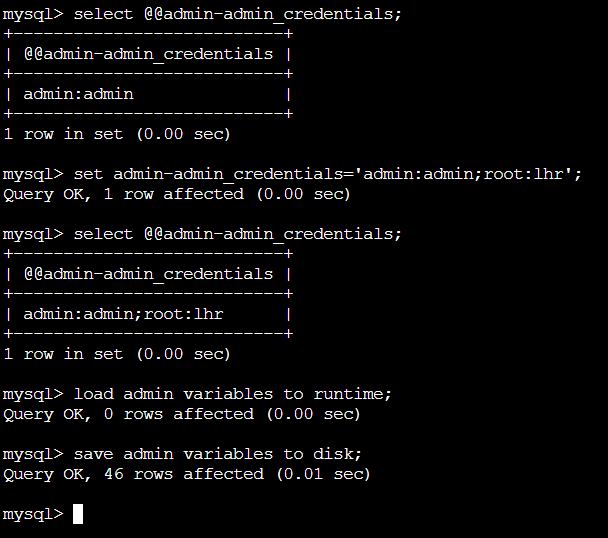


图3-20 添加远程登录用户

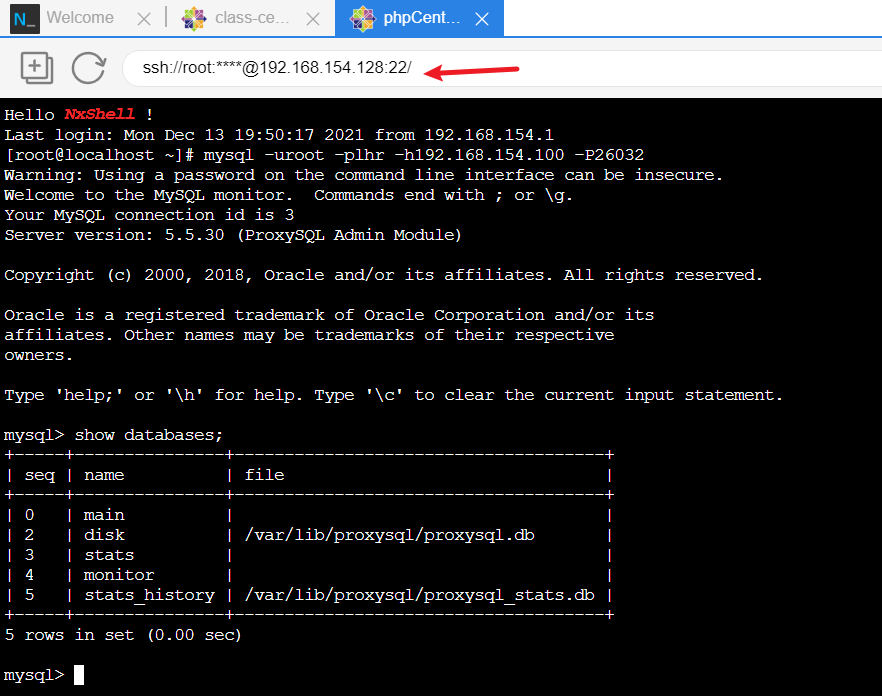
3.8.3、远程登录验证，如图3-21所示。

图3-21 远程登录验证

**3.9、开启ProxySQL的web监控功能，如图3-22所示。**

mysql> SET admin-web\_enabled='true';

mysql> LOAD ADMIN VARIABLES TO RUNTIME;

mysql> SAVE ADMIN VARIABLES TO DISK;

mysql> select \* from global\_variables where variable\_name LIKE 'admin-web\_enabled';

mysql> select @@admin-web\_enabled;

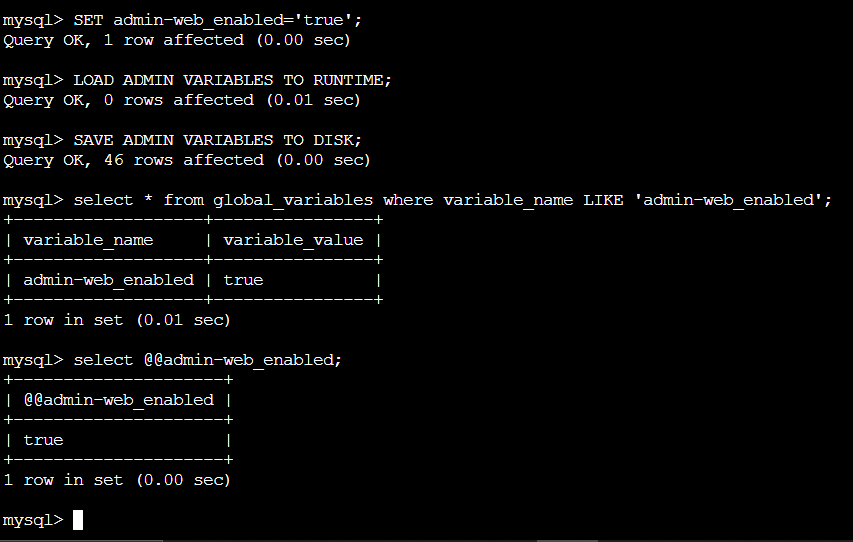


图3-22 开启proxySQLweb监控功能

**3.10、windows浏览器访问验证，如图3-23所示。**

<https://192.168.154.100:26080>

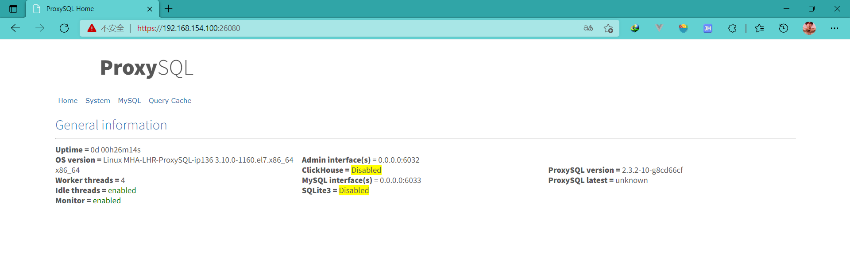
用户名：密码：stats:stats

图3-23 windows浏览器访问验证

**3.11、配置被监控的数据库**

3.11.1、向Proxy插入被监控数据库，如图3-24所示。

mysql> select \* from mysql\_servers;

insert into main.mysql\_servers(hostgroup\_id,hostname,port) values(10,'192.168.68.131',3306);

insert into main.mysql\_servers(hostgroup\_id,hostname,port) values(10,'192.168.68.132',3306);

insert into main.mysql\_servers(hostgroup\_id,hostname,port) values(10,'192.168.68.133',3306);

mysql> load mysql servers to runtime;

mysql> save mysql servers to disk;

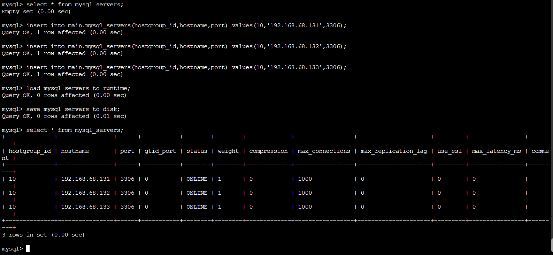
mysql> select \* from mysql\_servers;

图3-24 ProxySQL数据库插入被监控数据库

3.11.2、在全部被监控MySQL服务器上建立对外访问账户

（1）在主数据库中建立对外访问账户，如图3-25所示。

create user 'wr'@'%' IDENTIFIED BY 'lhr';

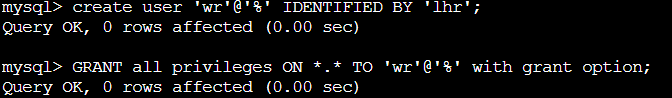
GRANT all privileges ON \*.\* TO 'wr'@'%' with grant option;

图3-25 在主数据库中建立对外访问账户

（2）在ProxySQL容器中间进行以下配置，如图3-26所示。

insert into mysql\_users(username,password,default\_hostgroup) values('wr','lhr',10);

update mysql\_users set transaction\_persistent=1 where username='wr';

load mysql users to runtime;

save mysql users to disk;

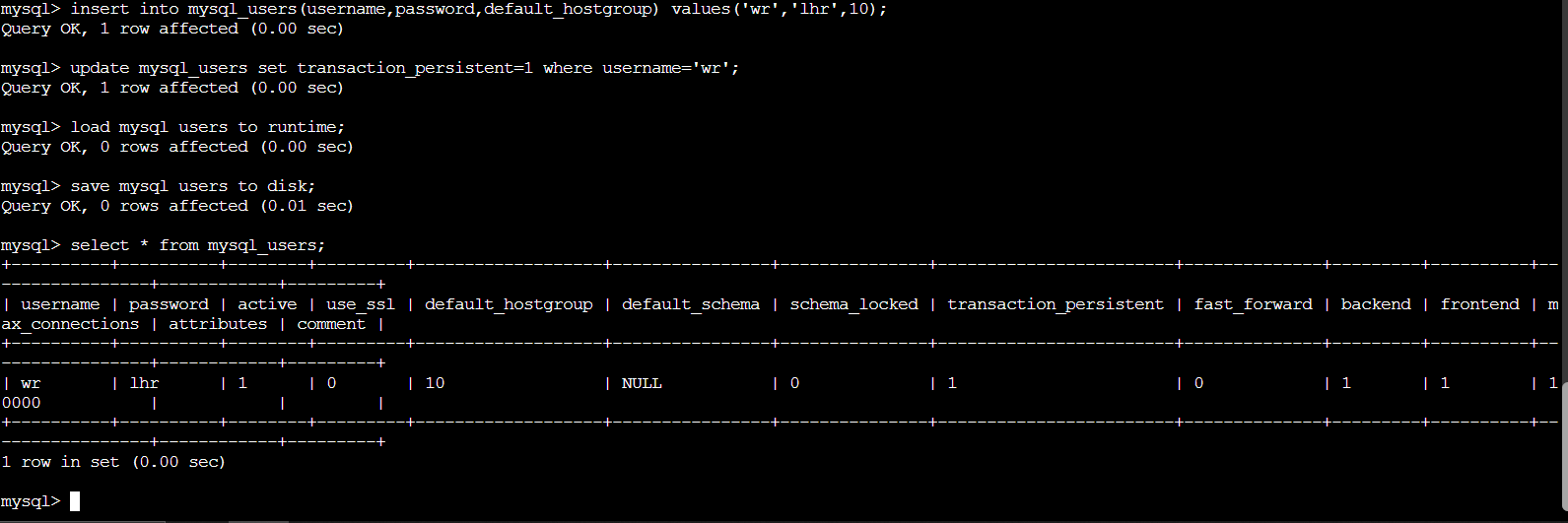
select \* from mysql\_users;

图3-26 ProxySQL容器进行MySQL配置

3.11.3、在ProxySQL配置监控，如图3-27所示。

set mysql-monitor\_username='monitor';

set mysql-monitor\_password='monitor';

load mysql servers to runtime;

save mysql servers to disk;

select \* from global\_variables where variable\_name in('mysql-monitor\_username','mysql-monitor\_password');

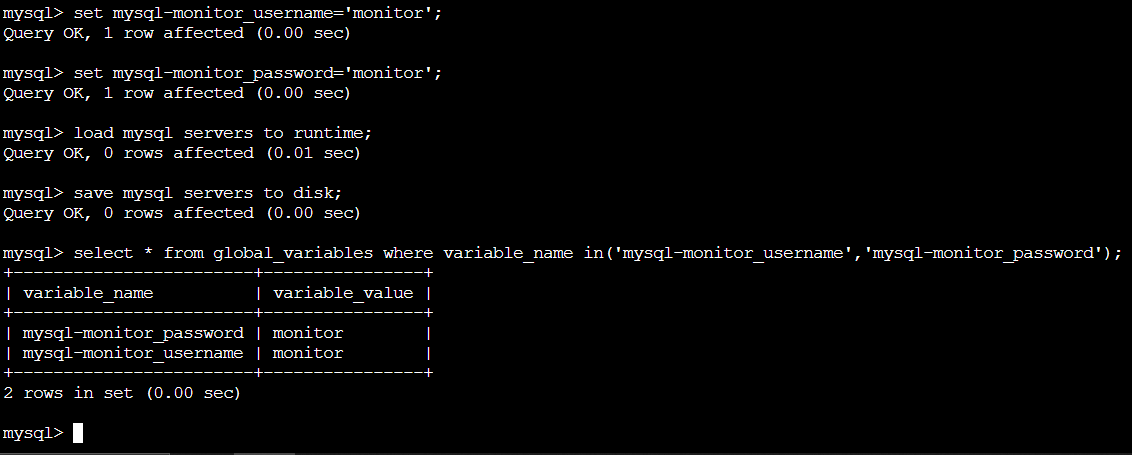


图3-27 ProxySQL容器中配置监控

3.11.4、检查链接到MySQL的日志，如图3-28所示。

select \* from monitor.mysql\_server\_ping\_log order by time\_start\_us desc limit 6;

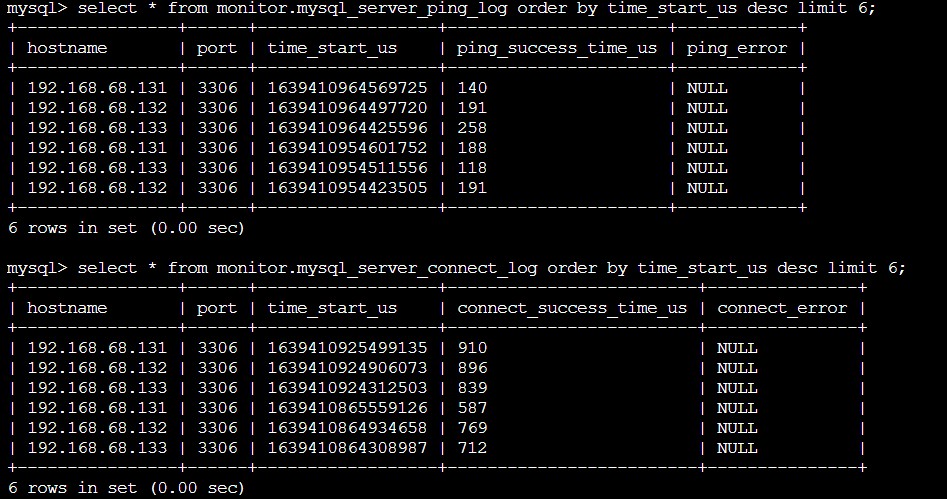
select \* from monitor.mysql\_server\_connect\_log order by time\_start\_us desc limit 6;

图3-28 检查链接到MySQL的日志

3.11.5、配置MySQL主机组，如图3-29所示。

show variables like 'read\_only';

set global read\_only=1;

insert into mysql\_replication\_hostgroups(writer\_hostgroup,reader\_hostgroup,comment) values(10,20,'proxy');

load mysql servers to runtime;

save mysql servers to disk;

select \* from mysql\_replication\_hostgroups;

select \* from mysql\_server\_read\_only\_log order by time\_start\_us desc limit 3;

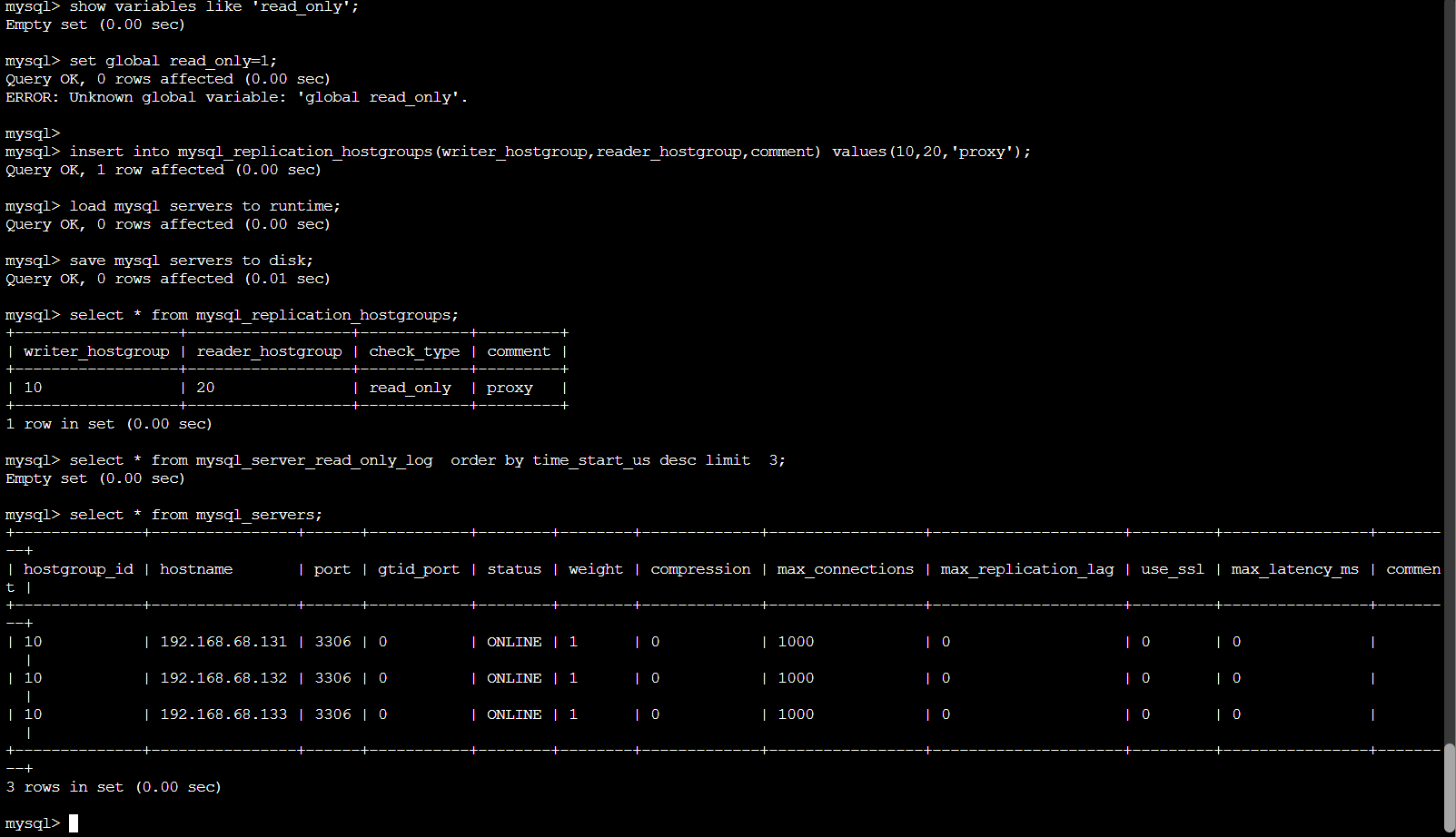
select \* from mysql\_servers;

图3-29 配置MySQL主机组

3.11.6、配置读写分离策略，如图3-30所示。

insert into mysql\_query\_rules(active,match\_pattern,destination\_hostgroup,apply) values(1,'^SELECT.\*FOR UPDATE$',10,1);

insert into mysql\_query\_rules(active,match\_pattern,destination\_hostgroup,apply) values(1,'^SELECT',20,1);

LOAD MYSQL QUERY RULES TO RUNTIME;

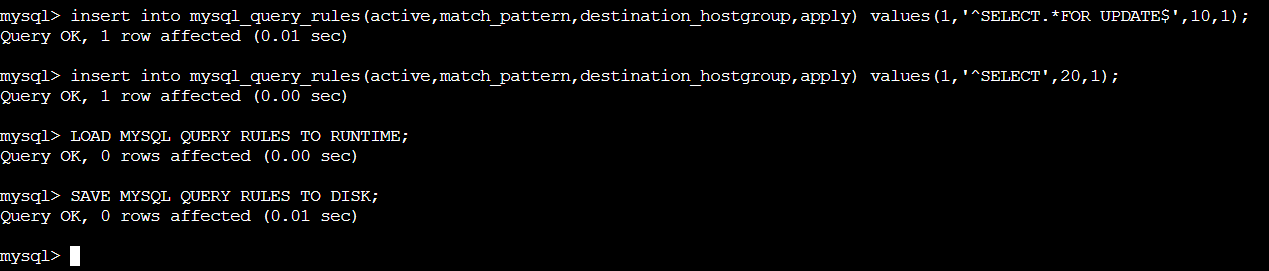
SAVE MYSQL QUERY RULES TO DISK;

图3-30 配置读写分离策略

**4、系统测试**

**4.1、故障切换**

4.1.1、在Manager节点检查SSH、复制及MHA的状态。

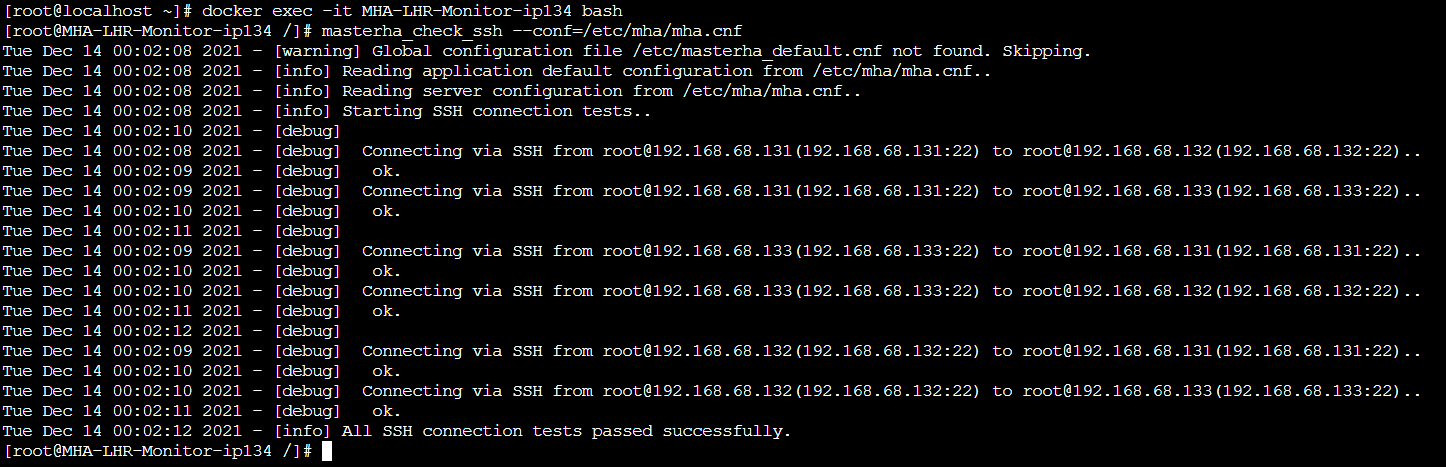
（1）在Manager节点检查SSH，如图4-1所示。

图4-1 在Manager节点检查SSH

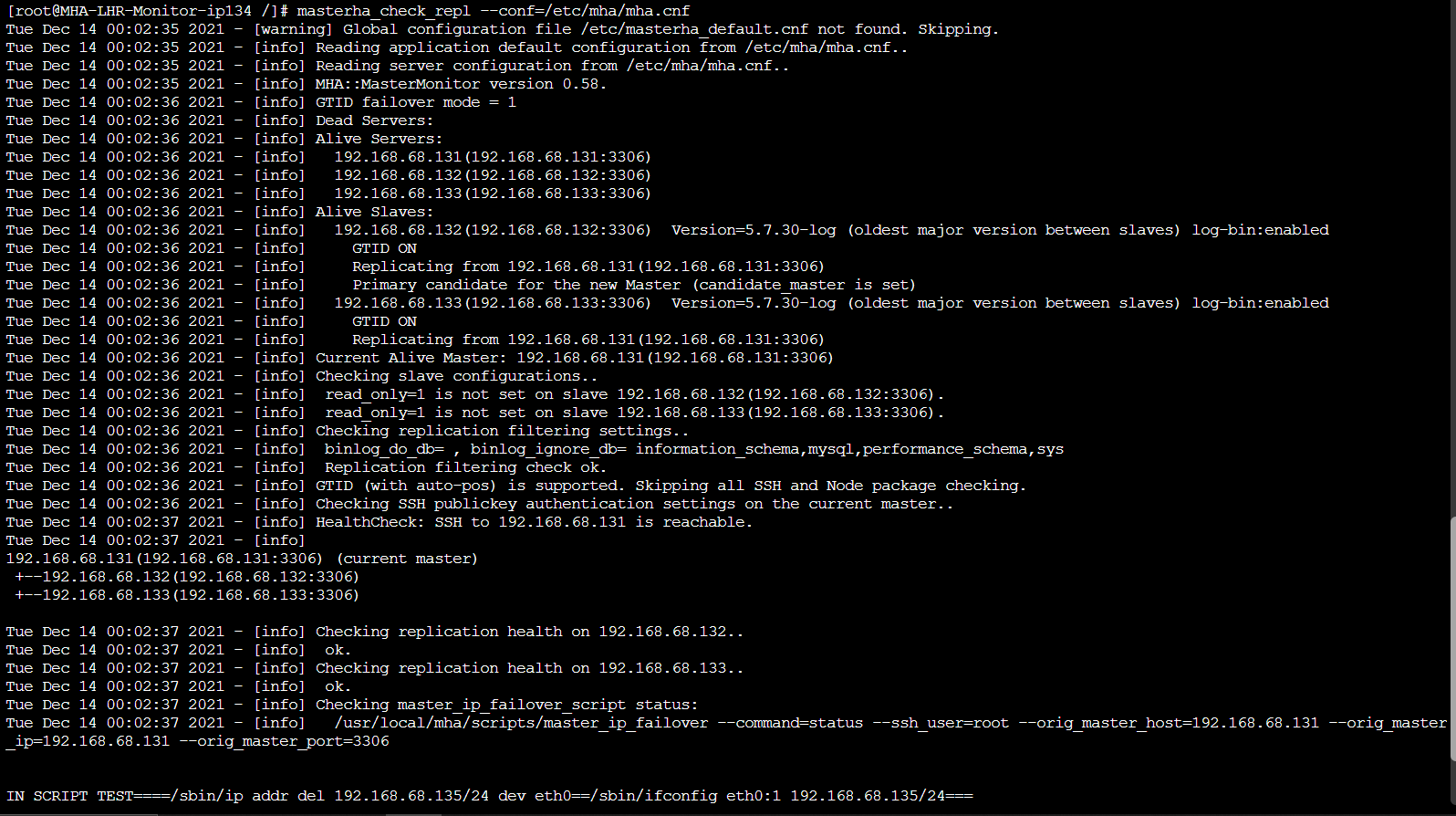
（2）在Manager节点检查复制，如图4-2所示

图4-2 在Manager节点检查复制

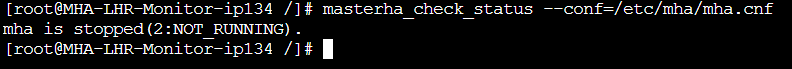
（3）在Manager节点检查MHA状态，如图4-3所示。

图4-3 在Manager节点检查MHA状态

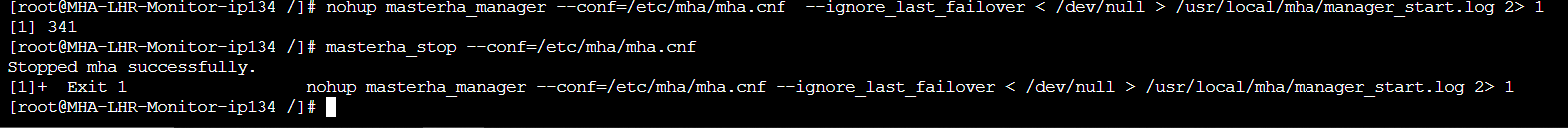
4.1.2、启动MHA监控进程，如图4-4所示。

图4-4 启动MHA监控进程

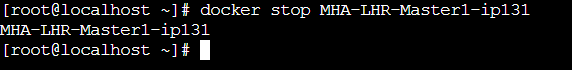
4.1.3、宕掉主库MHA-LHR-Master1-ip131，如图4-5所示。

图4-5 宕掉主库

4.1.4、查看ProxySQL的状况

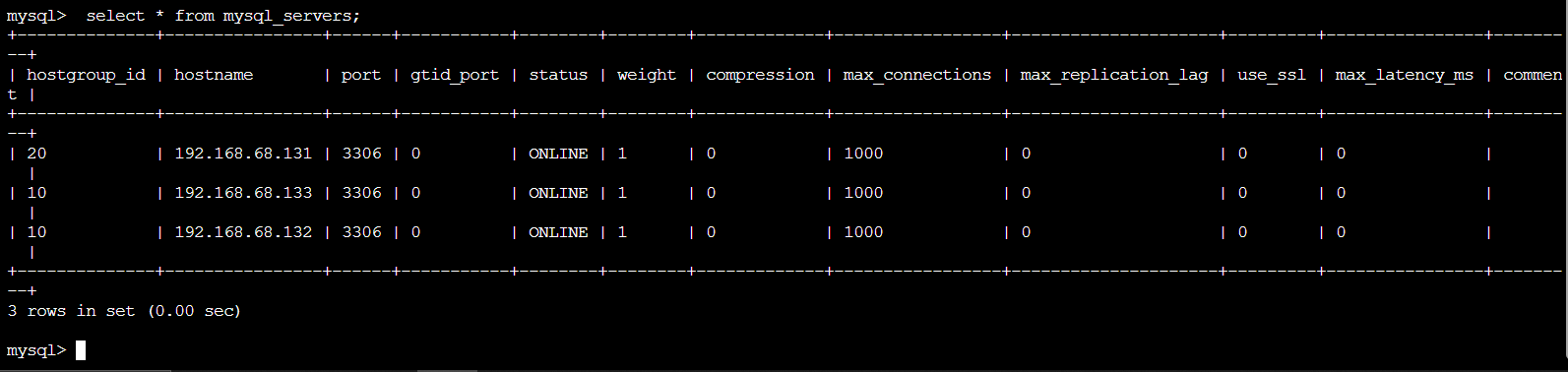
（1）查看数据库服务器状态，如图4-6所示。

图4-6 数据库服务器状态

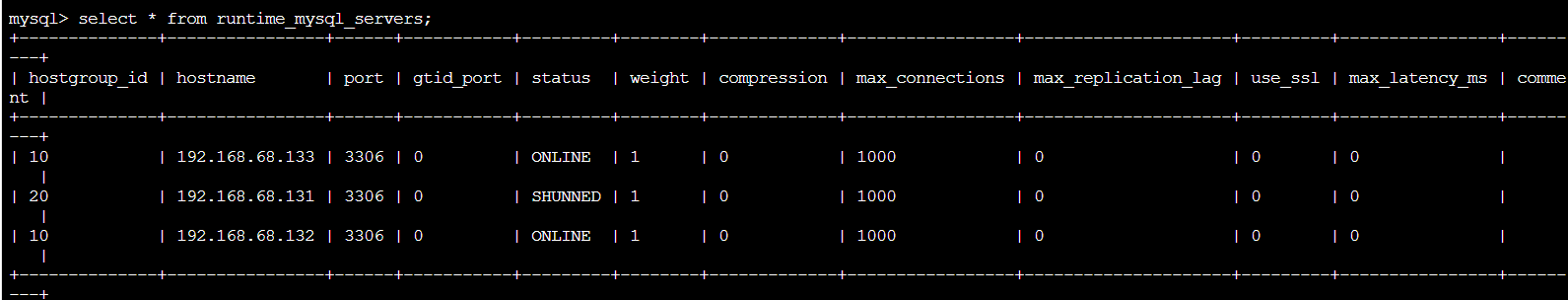
（2）查看运行状态的数据库服务器，如图4-7所示。

图4-7 运行状态的数据库服务器

此时MHA-LHR-Master1-ip131数据库服务器变为SHUNNED状态，ProxySQL会避开这个主机。

4.1.5、启动主库MHA-LHR-Master1-ip131，如图4-8所示。

图4-8 启动主库

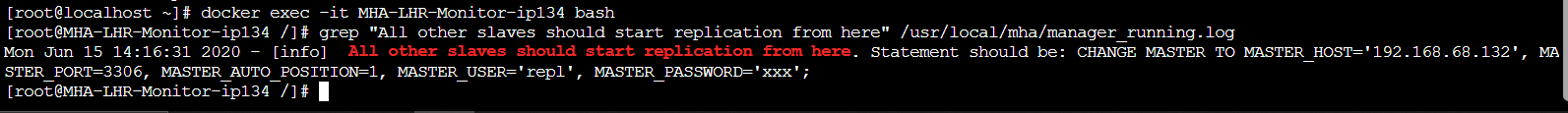
4.1.6、在MHA-LHR-Monitor-ip134的日志文件中找到恢复语句，如图4-9所示。

图4-9 从日志文件中找到恢复语句

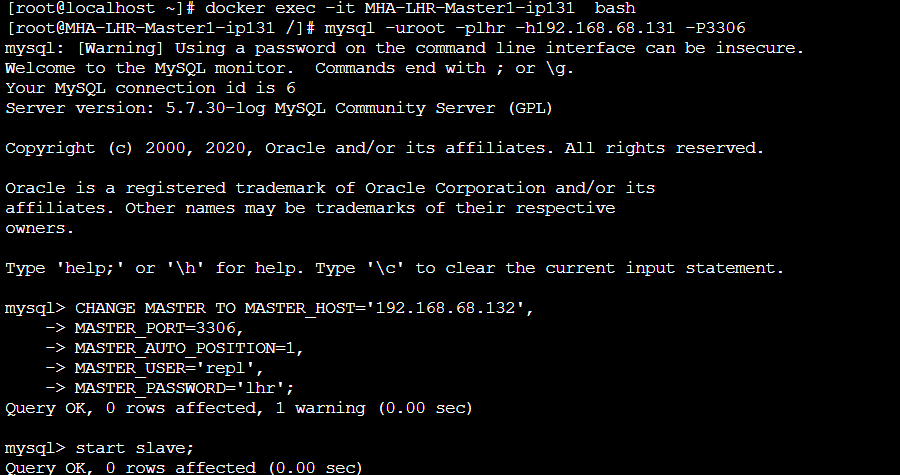
4.1.7、在主库MHA-LHR-Master1-ip131执行恢复，如图4-10所示。

图4-10 在主库中执行恢复

4.1.8、查看从数据库状态、并设置全局只读。

（1）、查看从数据库状态，如图4-11所示。

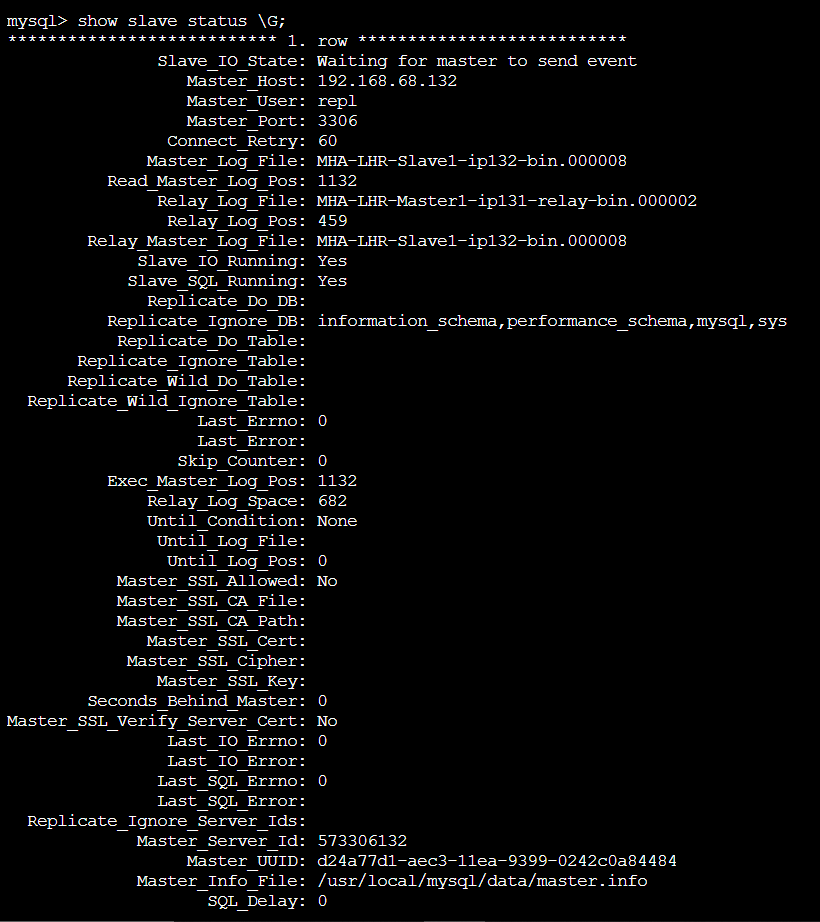


图4-11 从数据库状态

（2）设置全局只读，如图4-12所示。

图4-12 设置全局只读

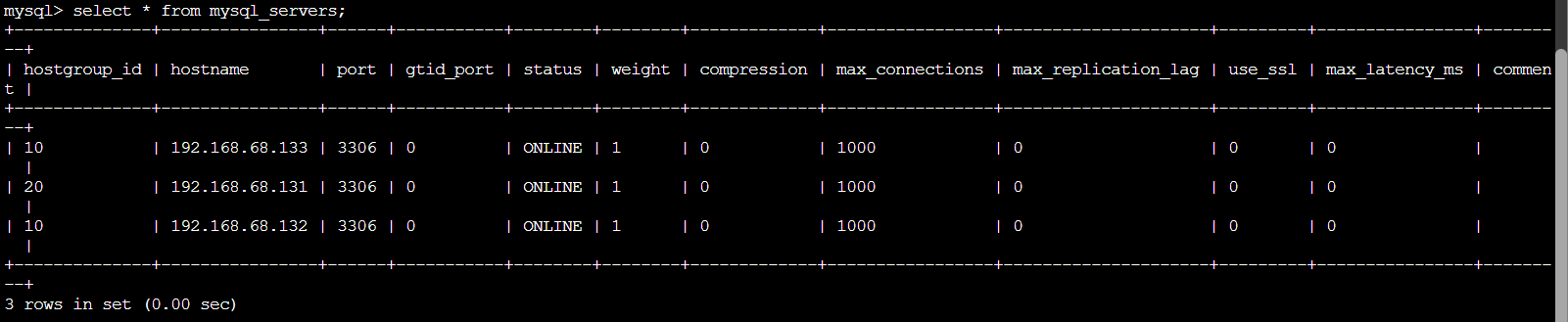
4.1.9、查询ProxySQL，如图4-13所示。

图4-13 查询ProxySQL

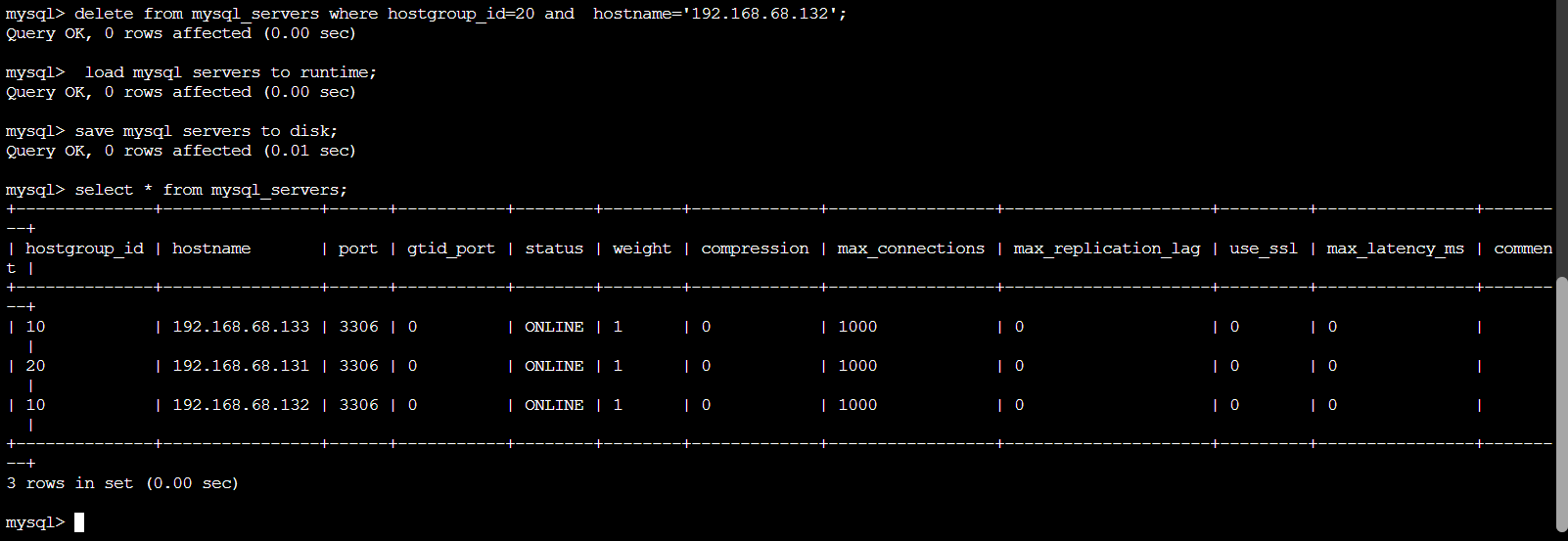
4.1.10、设置MHA-LHR-Slave1-ip132只写，并删除相关记录，如图4-14所示。

图4-15 操作从库Slave1

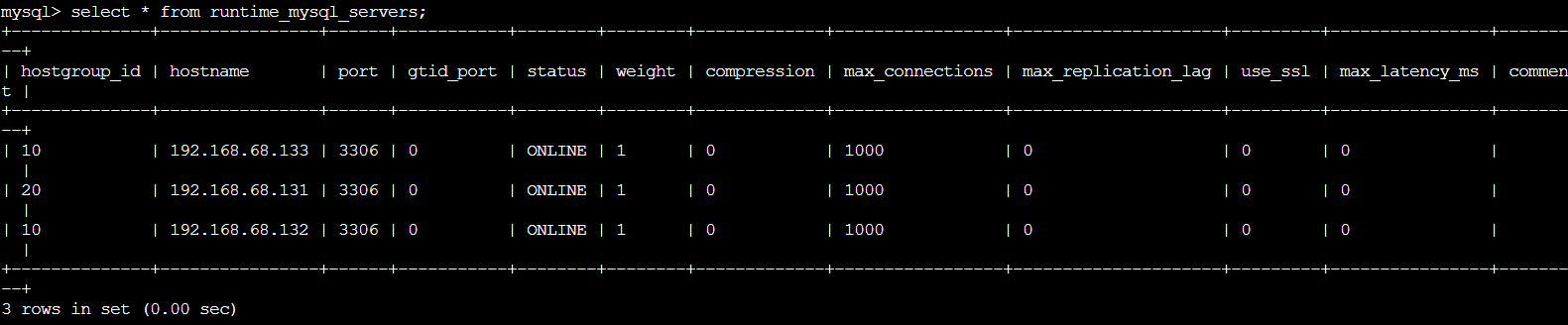
4.1.11、查询数据库服务器状态，如图4-16所示。

图4-16查询数据库服务器状态。

**5、总结**

**5.1、遇到的困难**

5.1.1、在实验过程中，容器与宿主机之间突然不能通信，在尝试例如新建网桥的方法之后，仍是不成功，最后选择新建宿主虚拟机重新开始实验。

5.1.2、在实验的最终阶段，配置MySQL主机组相关操作失败，导致后续检测阶段出现问题，具体原因由于时间原因还未找到。

**5.2、心得体会**

在学习、搭建MHA+ProxySQL的实验过程中，我了解到一些企业在搭建高可用与负载均衡服务器所使用的一些方法，使我进一步认识到在生产实践过程中，高可用、负载均衡、高性能等服务器指标对业务的必要性。

最后，十分感谢老师的指导，祝老师身体健康。