MySQL高可用

MySQL高可用离不开复制，复制是建立MySQL高可用集群的必要方式。可以结合第三方组件实现诸如：故障转移，自动切换，负载均衡，读写分离等业务需求。

业务需求驱动架构改造。使用何种方式是在业务容忍度与资源消耗，开发维护成本之间做衡量和取舍。

MySQL复制在MySQL 5.7中进行了各种改进。MySQL 5.7中突出的复制功能是True Multi Threaded Replication（MTS），多源复制和Group Replication。

# 1.复制方式：

异步复制：

主从复制，双主复制，级联复制，环形复制

延迟复制

多源复制

半同步复制

增强半同步复制

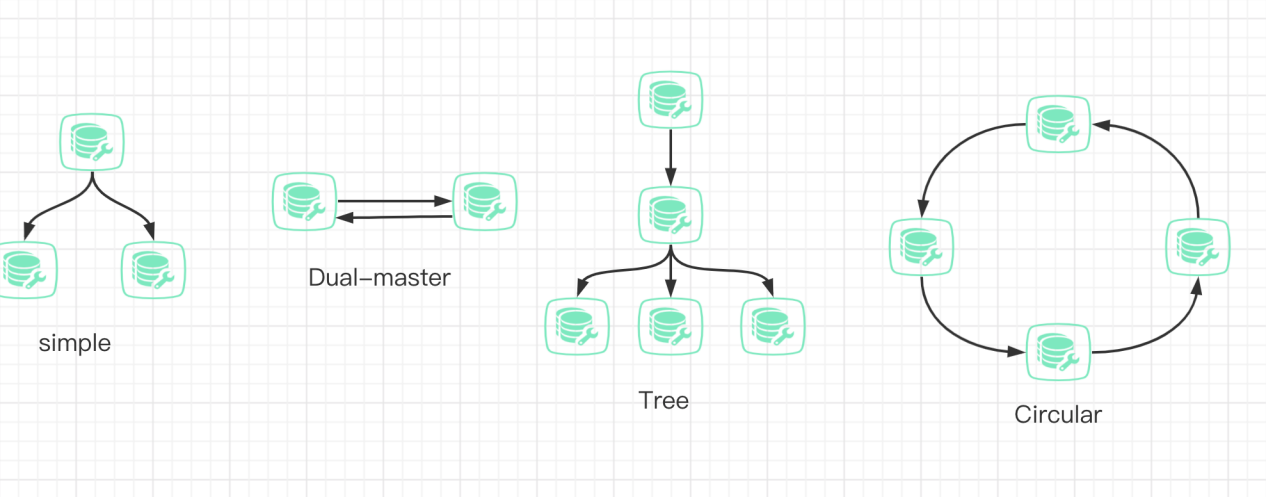
全同步复制

组复制

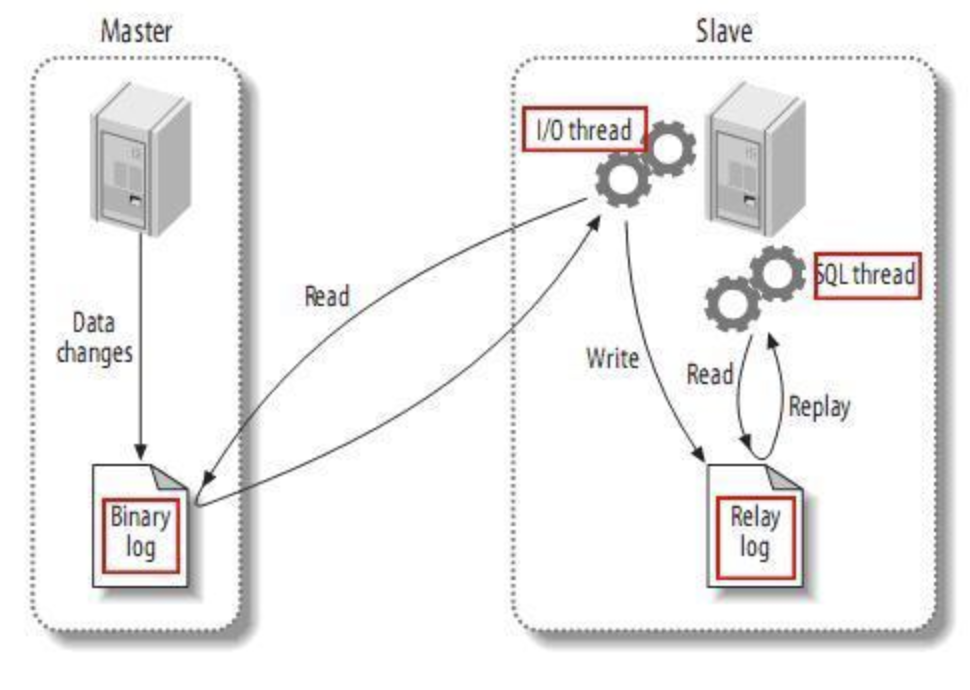
## 1.1异步复制

### 1.1.1复制拓扑

简单拓扑，双主拓扑，树形拓扑，环形拓扑



### 1.1.2简单异步复制流程



必要配置（my.cnf）

主库（实例） 从库（实例）

server-id = 1 server-id = 2

log-bin = mysql-bin log-bin = mysql-relay-bin

log-bin-index = mysql-bin.index log-bin-index = mysql-relay-bin.index

binlog\_format= row binlog\_format = row

### 1.1.3 异步复制手动搭建流程

1）两台机器分别部署单实例（文末有自动化搭建脚本）

2）master授权slave访问权限

grant replication slave,replication client on \*.\* to slave\_user@slave\_ip identified by ‘passwd’;

1. slave导入主实例数据。（如主服务上已运行并存在业务数据）

mysqldump -u -p -h -P --default-character-set=utf8 --single-transaction --master-data=2 > data.sql --主库导出数据

1. slave建立与master同步通信,并确认所在位置。

1) cat data.sql|grep -i 'change master' --查看备份文件位点信息

-- CHANGE MASTER TO MASTER\_LOG\_FILE='mysql-bin.000007', MASTER\_LOG\_POS=135752005;

1. change master to master\_host='',master\_port= ,master\_user='',master\_password='',master\_log\_file='',master\_log\_pos=;

或者change master to master\_host='',master\_port= ,master\_user='',master\_password='',master\_auto\_position=1;

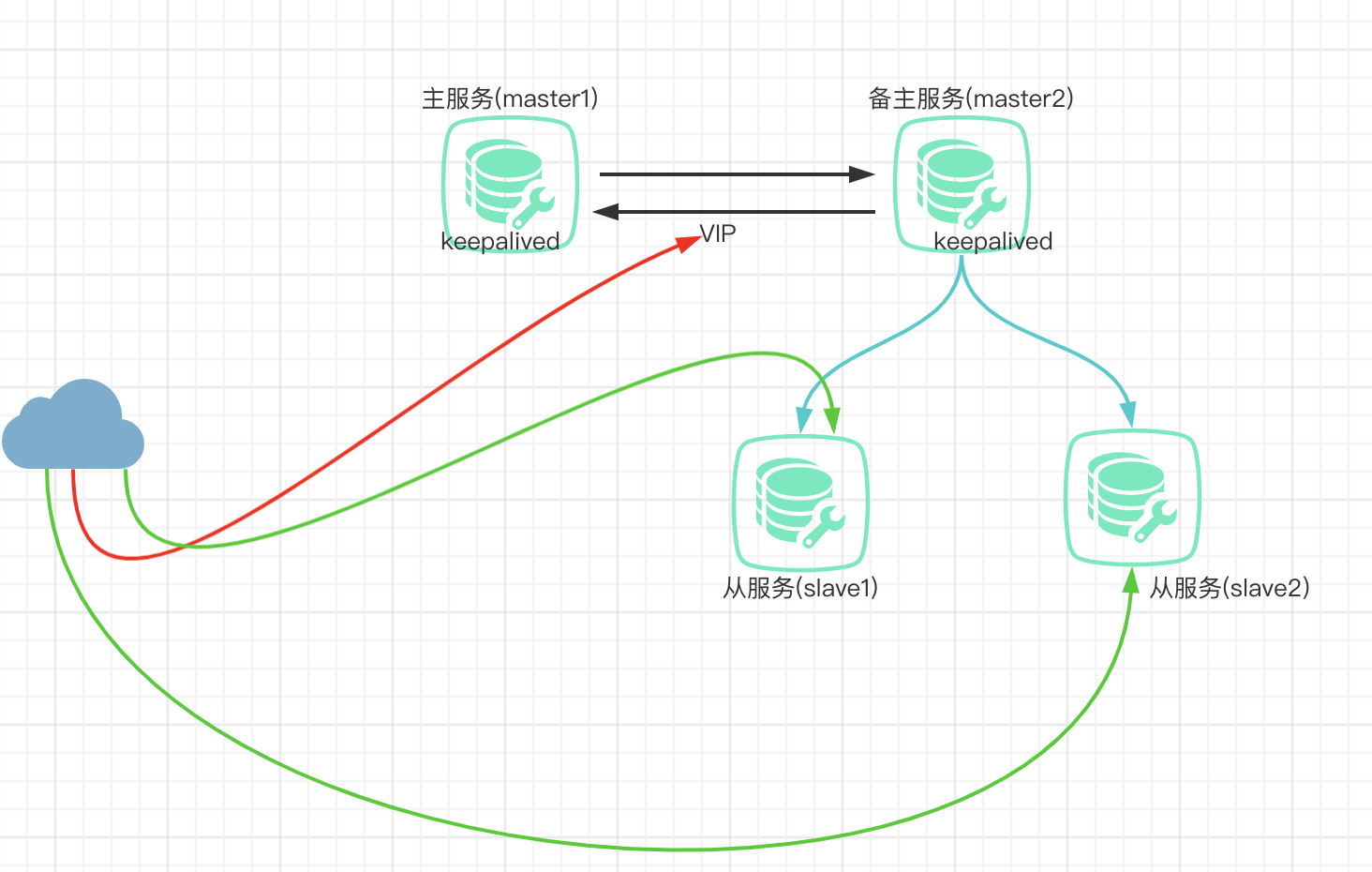
5) slave 开启同步线程,并观察同步状态是否正常

start slave; show slave status\G

### 1.1.4 异步复制”变形记“使用介绍

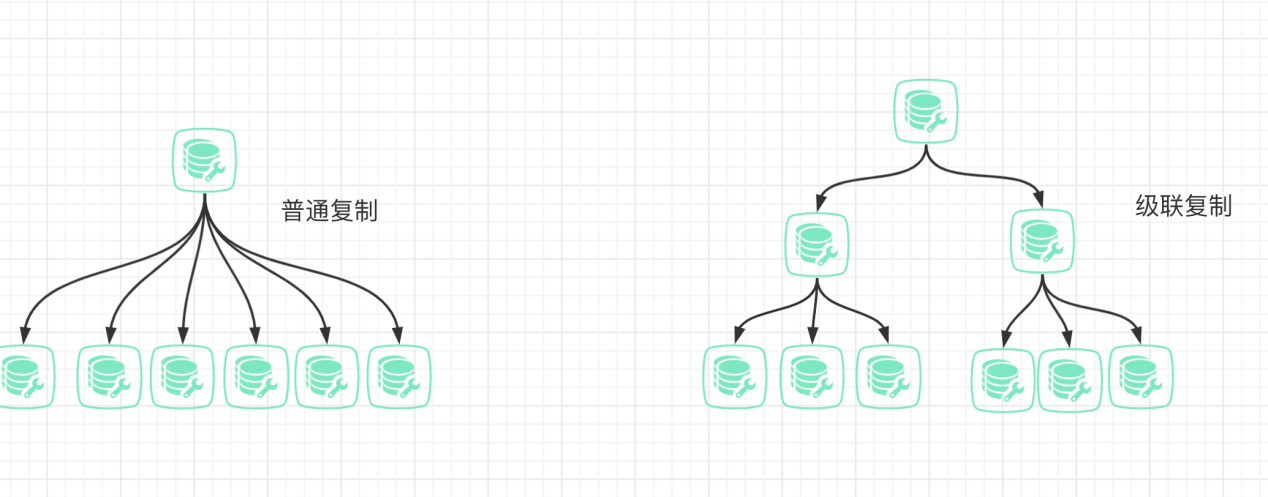
双主复制：

1. 场景



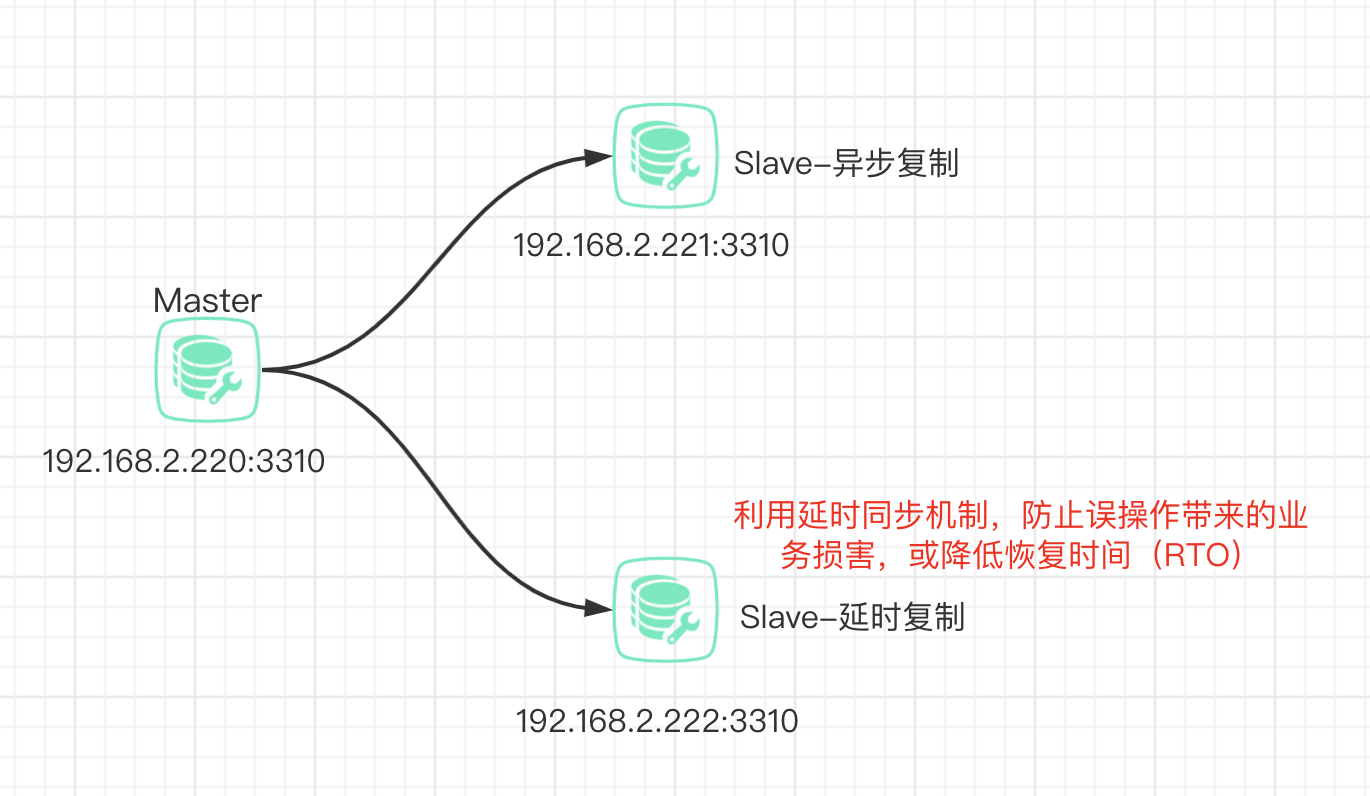
1. 级联复制

场景：



## 1.2 延时复制

### 1.2.1场景



### 1.2.2实验室：

step1:

-- **查看三台终端数据库，t1表数据，均为**

mysql> select \* from t1;

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

| id | name | age |

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

| 1 | v1 | 20 |

| 2 | v2 | 23 |

| 3 | v3 | 24 |

| 4 | v4 | 34 |

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

step 2:

-- **2.222:3310 设置延时复制7200秒**

stop slave;

change master to master\_delay=7200;

start slave;

mysql> show slave status\G

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

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

Master\_Host: 192.168.2.220

Master\_User: replic

Master\_Port: 3310

Connect\_Retry: 60

Master\_Log\_File: mysql3310-bin.000001

Read\_Master\_Log\_Pos: 2942

Relay\_Log\_File: mysql3310-relay-log.000002

Relay\_Log\_Pos: 426

Relay\_Master\_Log\_File: mysql3310-bin.000001

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

Replicate\_Do\_DB:

Replicate\_Ignore\_DB:

Replicate\_Do\_Table:

Replicate\_Ignore\_Table:

Replicate\_Wild\_Do\_Table:

Replicate\_Wild\_Ignore\_Table:

Last\_Errno: 0

Last\_Error:

Skip\_Counter: 0

Exec\_Master\_Log\_Pos: 2942

Relay\_Log\_Space: 637

Until\_Condition: None

Until\_Log\_File:

Until\_Log\_Pos: 0

Master\_SSL\_Allowed: No

Master\_SSL\_CA\_File:

Master\_SSL\_CA\_Path:

Master\_SSL\_Cert:

Master\_SSL\_Cipher:

Master\_SSL\_Key:

Seconds\_Behind\_Master: 0

Master\_SSL\_Verify\_Server\_Cert: No

Last\_IO\_Errno: 0

Last\_IO\_Error:

Last\_SQL\_Errno: 0

Last\_SQL\_Error:

Replicate\_Ignore\_Server\_Ids:

Master\_Server\_Id: 1

Master\_UUID: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc

Master\_Info\_File: mysql.slave\_master\_info

SQL\_Delay: 7200

SQL\_Remaining\_Delay: NULL

Slave\_SQL\_Running\_State: Slave has read all relay log; waiting for more updates

Master\_Retry\_Count: 86400

Master\_Bind:

Last\_IO\_Error\_Timestamp:

Last\_SQL\_Error\_Timestamp:

Master\_SSL\_Crl:

Master\_SSL\_Crlpath:

Retrieved\_Gtid\_Set:

Executed\_Gtid\_Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:1-9

Auto\_Position: 1

Replicate\_Rewrite\_DB:

Channel\_Name:

Master\_TLS\_Version:

1 row in set (0.00 sec)

step 3:

**-- 2.220:3310 主库模拟删除操作**

delete from t1;

step 4:

-- **确认三台服务数据**

终端 192.168.2.220:3310 --empty

终端 192.168.2.221:3310 --empty

终端 192.168.2.222:3310 -- 4 rows

step 5:

**-- 利用延时复制，追溯至误操作前一刻，跳过此操作后，切换应用服务。**

**-- 找出误操作前最近的binlog位点**

/usr/local/mysql57/bin/mysqlbinlog --base64-output=decode-rows -vv ./mysql3310-bin.000001 |grep -A10 -B 10 -i 'delete'

**-- 同步至此位置，切换应用连接的数据源**

start slave until master\_log\_file='mysql3310-bin.000001',master\_log\_pos=2942;

**-- 后续操作**

stop slave;

SET GTID\_NEXT='29acfba7-94b7-11ea-b6a7-e454e8ca20cc:11';

begin; commit;

SET GTID\_NEXT="AUTOMATIC";

change master to master\_delay=0;

start slave;

## 1.3 多源复制

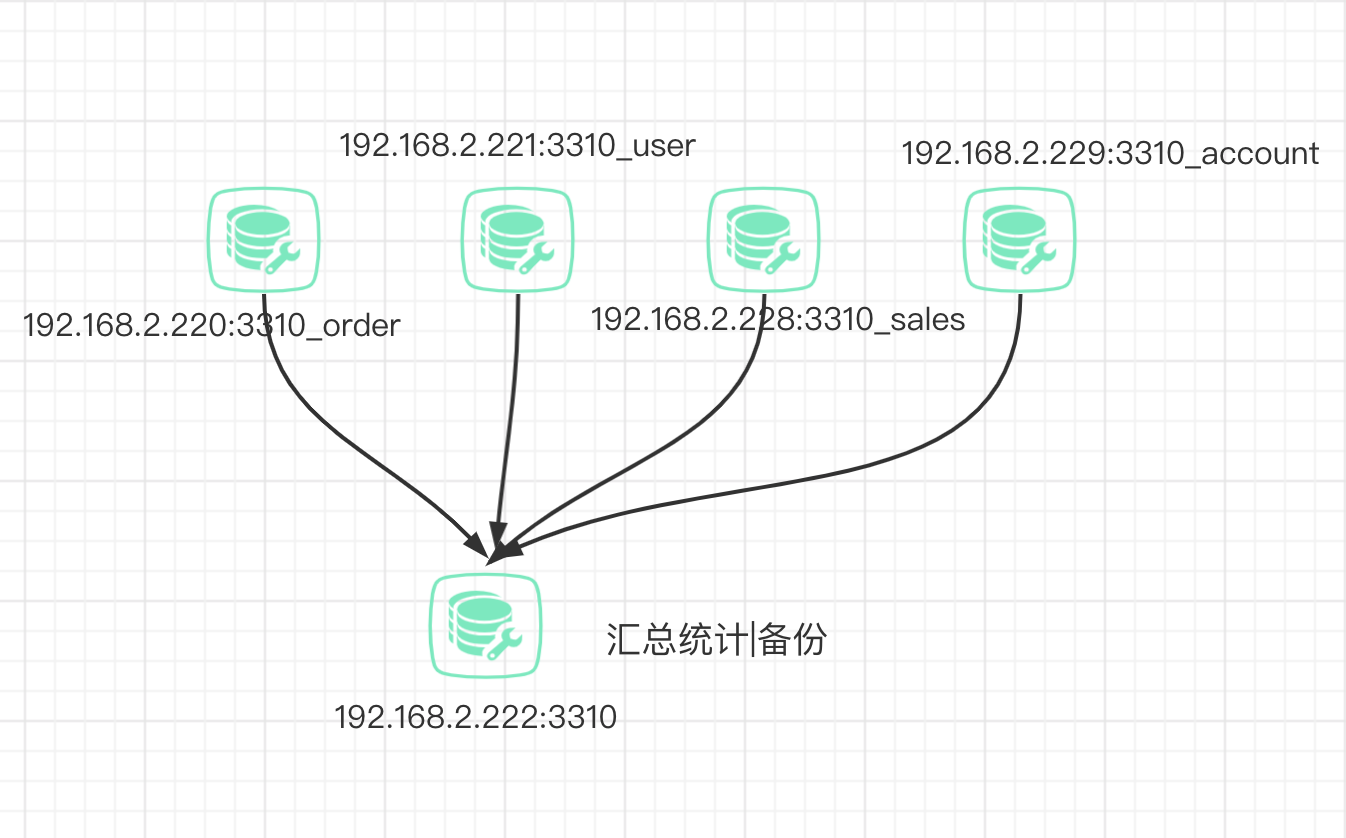
### 1.3.1概述

多源复制从属服务器能够从多个主服务器接收事务。

它通过在单个服务器下组合各种数据库来帮助Analytics。也有助于备份。

一个从服务器为每个主服务器创建一个复制通道。

### 1.3.2 场景



### 1.3.3 实验室

**-- 环境说明**

192.168.2.220:3310-master1

192.168.2.221:3310-master2

192.168.2.222:3310-slave

step 1:

-- muti-source **确认从服务2.222 必要配置参数.防止跨库更新造成更新失效，并且可防止新增schema导致需重启服务。**

master-info-repository = table

relay-log-info-repository = table

replicate\_wild\_ignore\_table=information\_schema.%

replicate\_wild\_ignore\_table=mysql.%

replicate\_wild\_ignore\_table=performance\_schema.%

replicate\_wild\_ignore\_table=sys.%

step 2:

-- 2.220，2.221分别授权

grant replication slave,replication client on \*.\* to multi@'192.168.2.%' identified by 'multi@qunje';

step 3:

-- **2.222 分别配置同步两台数据源**

CHANGE MASTER TO MASTER\_HOST='192.168.2.220',MASTER\_USER='multi', MASTER\_PASSWORD='multi@qunje',master\_port=3310,master\_auto\_position=1 FOR CHANNEL 'master\_1';

CHANGE MASTER TO MASTER\_HOST='192.168.2.221',MASTER\_USER='multi', MASTER\_PASSWORD='multi@qunje',master\_port=3310,master\_auto\_position=1 FOR CHANNEL 'master\_2';

step 4:

**-- 新建库表查看同步状态**

MySQL [jiale]> create database orders; --192.168.2.220

mysql> create database users; --192.168.2.221

mysql> show databases; --192.168.2.222

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

| Database |

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

| information\_schema |

| jiale |

| mysql |

| orders |

| performance\_schema |

| sys |

| users |

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

7 rows in set (0.00 sec)

mysql> show slave status for channel 'master\_1'\G

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

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

Master\_Host: 192.168.2.220

Master\_User: multi

Master\_Port: 3310

Connect\_Retry: 60

Master\_Log\_File: mysql3310-bin.000001

Read\_Master\_Log\_Pos: 4412

Relay\_Log\_File: mysql3310-relay-log-master\_1.000002

Relay\_Log\_Pos: 1239

Relay\_Master\_Log\_File: mysql3310-bin.000001

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

Replicate\_Do\_DB:

Replicate\_Ignore\_DB:

Replicate\_Do\_Table:

Replicate\_Ignore\_Table:

Replicate\_Wild\_Do\_Table:

Replicate\_Wild\_Ignore\_Table: information\_schema.%,mysql.%,performance\_schema.%,sys.%

Last\_Errno: 0

Last\_Error:

Skip\_Counter: 0

Exec\_Master\_Log\_Pos: 4412

Relay\_Log\_Space: 1459

Until\_Condition: None

Until\_Log\_File:

Until\_Log\_Pos: 0

Master\_SSL\_Allowed: No

Master\_SSL\_CA\_File:

Master\_SSL\_CA\_Path:

Master\_SSL\_Cert:

Master\_SSL\_Cipher:

Master\_SSL\_Key:

Seconds\_Behind\_Master: 0

Master\_SSL\_Verify\_Server\_Cert: No

Last\_IO\_Errno: 0

Last\_IO\_Error:

Last\_SQL\_Errno: 0

Last\_SQL\_Error:

Replicate\_Ignore\_Server\_Ids:

Master\_Server\_Id: 1

Master\_UUID: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc

Master\_Info\_File: mysql.slave\_master\_info

SQL\_Delay: 0

SQL\_Remaining\_Delay: NULL

Slave\_SQL\_Running\_State: Slave has read all relay log; waiting for more updates

Master\_Retry\_Count: 86400

Master\_Bind:

Last\_IO\_Error\_Timestamp:

Last\_SQL\_Error\_Timestamp:

Master\_SSL\_Crl:

Master\_SSL\_Crlpath:

Retrieved\_Gtid\_Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:12-15

Executed\_Gtid\_Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:1-15,

5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-3

Auto\_Position: 1

Replicate\_Rewrite\_DB:

Channel\_Name: master\_1

Master\_TLS\_Version:

1 row in set (0.00 sec)

mysql> show slave status for channel 'master\_2'\G

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

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

Master\_Host: 192.168.2.221

Master\_User: multi

Master\_Port: 3310

Connect\_Retry: 60

Master\_Log\_File: mysql3310-bin.000002

Read\_Master\_Log\_Pos: 994

Relay\_Log\_File: mysql3310-relay-log-master\_2.000002

Relay\_Log\_Pos: 1175

Relay\_Master\_Log\_File: mysql3310-bin.000002

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

Replicate\_Do\_DB:

Replicate\_Ignore\_DB:

Replicate\_Do\_Table:

Replicate\_Ignore\_Table:

Replicate\_Wild\_Do\_Table:

Replicate\_Wild\_Ignore\_Table: information\_schema.%,mysql.%,performance\_schema.%,sys.%

Last\_Errno: 0

Last\_Error:

Skip\_Counter: 0

Exec\_Master\_Log\_Pos: 994

Relay\_Log\_Space: 1395

Until\_Condition: None

Until\_Log\_File:

Until\_Log\_Pos: 0

Master\_SSL\_Allowed: No

Master\_SSL\_CA\_File:

Master\_SSL\_CA\_Path:

Master\_SSL\_Cert:

Master\_SSL\_Cipher:

Master\_SSL\_Key:

Seconds\_Behind\_Master: 0

Master\_SSL\_Verify\_Server\_Cert: No

Last\_IO\_Errno: 0

Last\_IO\_Error:

Last\_SQL\_Errno: 0

Last\_SQL\_Error:

Replicate\_Ignore\_Server\_Ids:

Master\_Server\_Id: 221

Master\_UUID: 5a755e01-94bf-11ea-b2fa-e454e8ca3c54

Master\_Info\_File: mysql.slave\_master\_info

SQL\_Delay: 0

SQL\_Remaining\_Delay: NULL

Slave\_SQL\_Running\_State: Slave has read all relay log; waiting for more updates

Master\_Retry\_Count: 86400

Master\_Bind:

Last\_IO\_Error\_Timestamp:

Last\_SQL\_Error\_Timestamp:

Master\_SSL\_Crl:

Master\_SSL\_Crlpath:

Retrieved\_Gtid\_Set: 5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-3

Executed\_Gtid\_Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:1-15,

5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-3

Auto\_Position: 1

Replicate\_Rewrite\_DB:

Channel\_Name: master\_2

Master\_TLS\_Version:

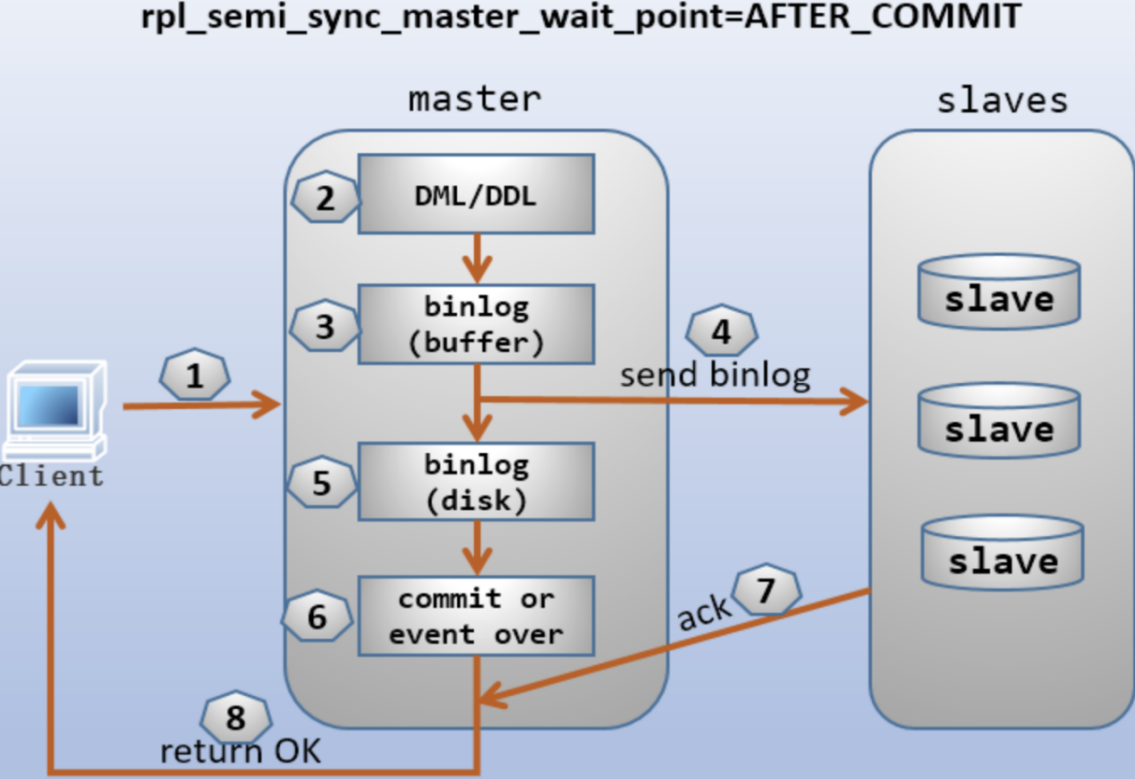
1 row in set (0.00 sec)

## 1.3 半同步复制

### 1.3.1概念

半同步复制(semi-synchronous replication)是指master在将新生成的binlog发送给各slave时，只需等待一个(默认)slave返回ack信息则返回成功。

### 1.3.2原理流程图



### 1.3.3 缺陷

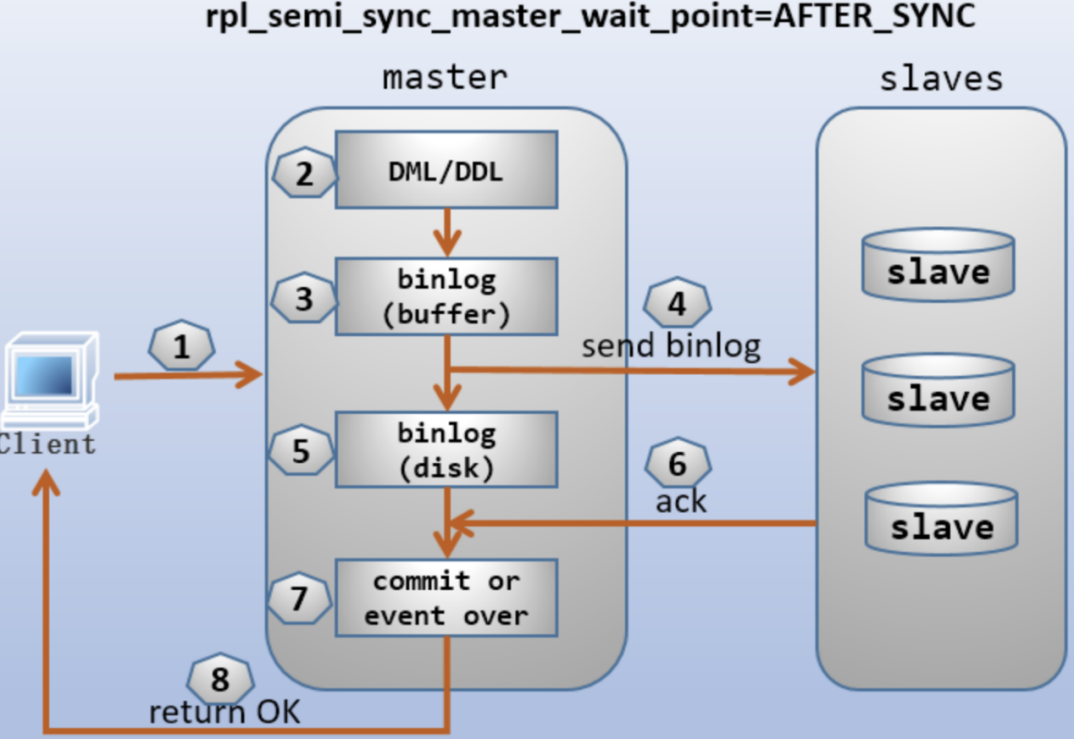
1.在MySQL 5.7以前，master上的binlog dump thread负责两件事：dump日志给slave的io\_thread。接收来自slave的ack消息。它们是串行方式工作的。

在MySQL 5.7中，新增了一个专门负责接受ack消息的线程ack collector thread。这样master上有两个线程独立工作，可以同时发送binlog到slave和接收slave的ack。

2.after\_commit极端情况下可能引擎层已经提交完成，事务对主库可见，但是从库还没有传输完成如果从库奔溃可能出现丢事务的情况。

## 1.4增强半同步复制

### 1.4.1原理流程图



### 1.4.2 参数配置

set rpl\_semi\_sync\_master\_wait\_point= after\_sync;

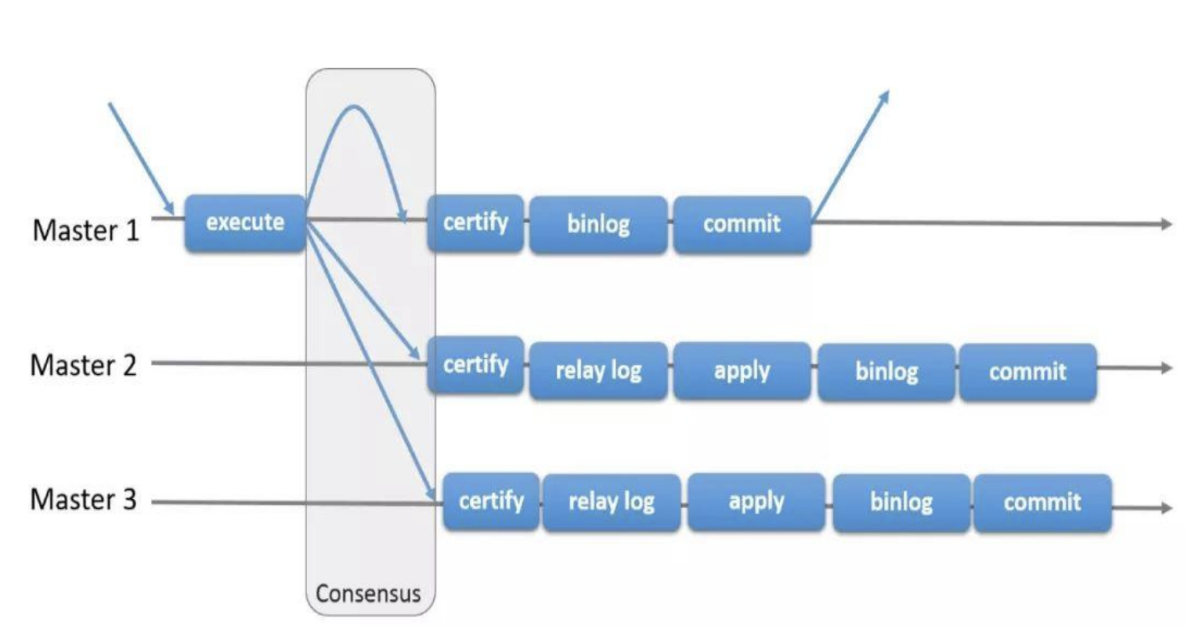
set global rpl\_semi\_sync\_master\_wait\_for\_slave\_count= N;

## 1.5 异步与半同步的总结

**介于异步复制和全同步复制之间，主库在执行完客户端提交的事务后不是立刻返回给客户端，而是等待至少一个从库接收到并写入relay log中才返回给客户端。相对于异步复制，半同步复制提高了数据的安全性，同时它也造成了一定程度的延迟，这个延迟最少是一个TCP/IP往返的时间。所以，半同步复制最好在低延时的网络中使用。**

## 1.6 组复制

### 1.6.1原理流程图



### 1.6.2 解读

MGR 相对于半同步复制，在 relay log 前增加了冲突检查协调，但是 binlog 回放仍然可能延时，也就是跟我们熟悉的半同步复制存在 io 线程的回放延迟情况类似。关于 IO线程回放慢的原因，跟半同步 也类似，比如大事务。

所以 MGR 并不是全同步方案，关于如何处理一致性读写的问题，MySQL 在 8.0.14 版本中加入了“读写一致性”特性，并引入了参数 group\_replication\_consistenc。

### 1.6.3实验室

-- 安装配置(所有节点执行，自动安装可参考脚本)



-- **环境信息**

version:8.0.17

server1:192.168.220.122

server2:192.168.220.128

server3:192.168.220.129

-- **配置大致流程**

**1.在220.122授权其他节点权限，清空binlog 记录**

SET SQL\_LOG\_BIN=0;

CREATE USER mgruser@'%' IDENTIFIED BY 'mgr@hot';

GRANT REPLICATION SLAVE,replication client ON \*.\* TO mgruser@'%';

FLUSH PRIVILEGES;

SET SQL\_LOG\_BIN=1;

**2.创建MGR依赖的复制通道(所有节点执行)**

CHANGE MASTER TO MASTER\_USER='mgruser', MASTER\_PASSWORD='mgr@hot' FOR CHANNEL 'group\_replication\_recovery';

**3. 在主节点上启动MGR(节点192.168.220.122上执行)**

SET GLOBAL group\_replication\_bootstrap\_group=ON;

START group\_replication;

SET GLOBAL group\_replication\_bootstrap\_group=off;

**4.在辅助节点上启动MGR(在节点192.168.220.128/129上执行)**

START group\_replication;

## 1.7异步与（全）同步的总结

异步复制比同步复制快的多。同步复制需要额外的机制保证一致性。一般通过两阶段提交协议实现。其保证了master与slave之间的一致性。但却需要额外的通信消息传递。大致过程：

1. 当执行提交语句时，事务被发送到slave,slave开始准备事务提交
2. 每个slave都要准备事务，然后向master发送ok(或abort) 消息，表面事务已经准备 好（或无法准备该事务）
3. master等待所有slave发送ok(或者abort)消息

-- 如果master收到所有slave的ok消息，它就会向slave发送提交消息，告诉slave可提交该事务

-- 如果master收到任何一个slave的abort消息，它就会向所有slave发送abort消息，告诉slave中止事务

1. 每个slave等待来自master的ok或者abort消息

-- 如果slave收到提交消息，它们就会提交事务，并向master发送事务已提交的确认

-- 如果slave收到取消消息，它们就会撤回所有改变并释放所占资源，中止事务，然后向master 发送事务已中止的确认

5.当master收到来自所有slave的确认后，就会报告该事务被提交（或中止），然后继续下一个事务处理。

# 高可用方案选型

## 2.1 MySQL Router

### 2.1.1 简介

MySQL Router是MySQL官方提供的一个轻量级中间件，可以在应用程序与MySQL服务器之间提供透明的路由方式。主要用于解决MySQL主从集群的高可用、负载均衡、易扩展问题。

Router作为流量转发层，位于应用与MySQL服务器之间。应用不再直连MySQLServers，而是与Router相连。根据Router的配置，应用程序的读写请求将被转发给下游的MySQL Servers。

### 2.1.2 功能

**高可用：实现故障转移和读写分离，负载均衡。**

当下游有多个MySQL Servers，Router可以对读写请求进行负载均衡。当下游某个Server失效时，Router可以将其从Active列表中移除，当其online后可再次加入Active列表。

**易扩展：**

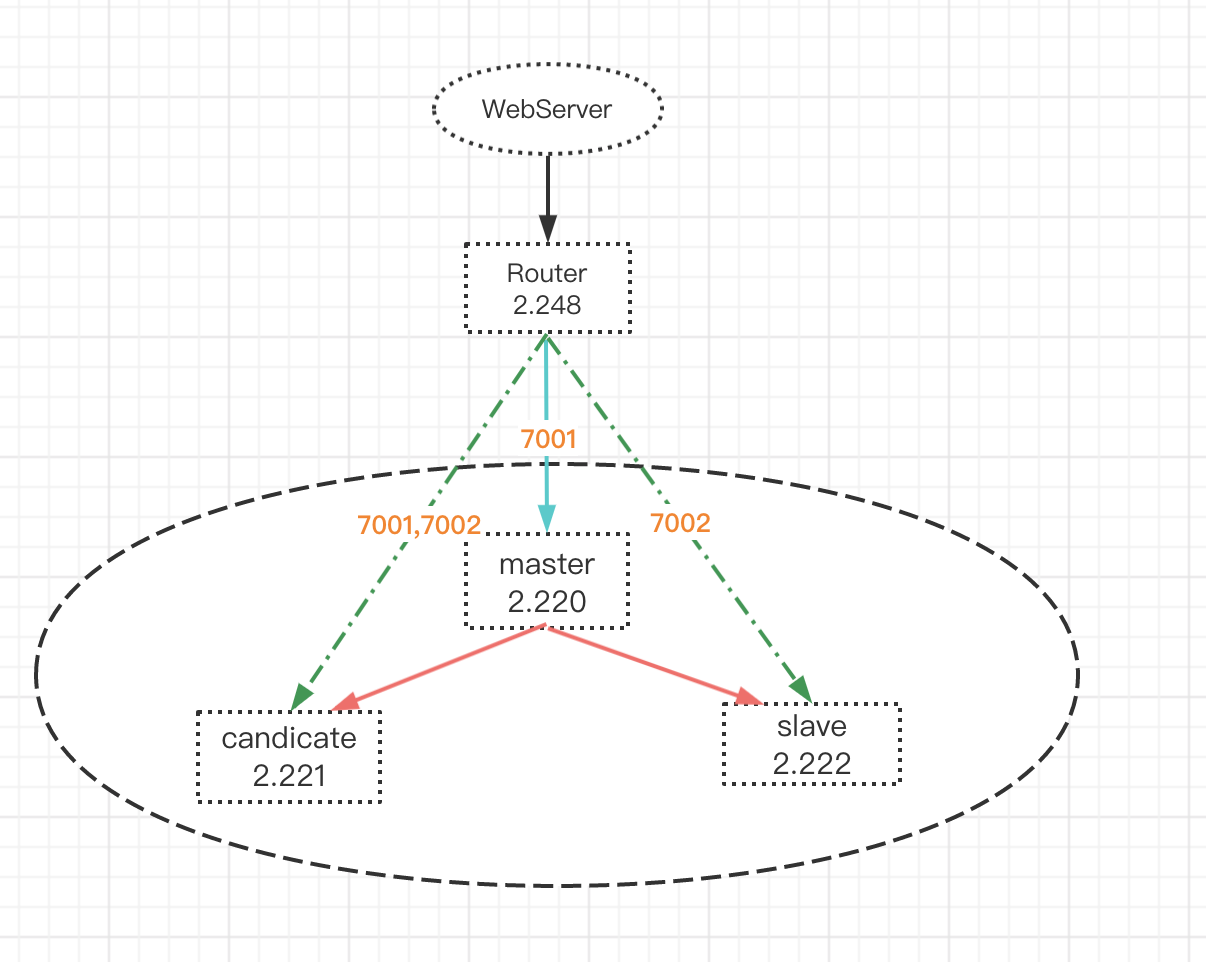
当MySQL Servers集群拓扑变更时，比如增减Slaves节点，只需要修改Router配置即可，无需修改应用中数据库连接配置，因为应用配置的是Router地址而非MySQL Servers地址，即数据库集群对应用来说是透明的。如果Master失效后，Cluster将会自动选举一个新的Master，此时Router不需要任何调整、可以自动发现此新Master进而继续为应用服务。

**缺点或限制：**

Router独立部署可能引入“额外的部署成本”、“性能降级”、“连接数上限”等问题，通常建议基于“Agent”方式部署，即将Router与应用部署在一台机器上。

Router中间件本身不会对请求“拆包”（unpackage），无法在Router中间件上实现比如“SQL审计”、“隔离”、“限流”、“分库分表”等功能。

### 2.1.3架构图



### 2.1.4 实验室

-- 环境说明：

-- 192.168.2.248 mysql-router

-- 192.168.2.220:3310 master

-- 192.168.2.221:3310 slave1

-- 192.168.2.222:3310 slave2

-- 2.248 安装配置 mysql-router

wget https://downloads.mysql.com/archives/get/p/41/file/mysql-router-2.1.6-linux-glibc2.12-x86-64bit.tar.gz

tar xzf /opt/mysql-router-2.1.6-linux-glibc2.12-x86-64bit.tar.gz

mv /opt/mysql-router-2.1.6-linux-glibc2.12-x86-64bit /opt/mysql-router-2.1.6

ln -s /opt/mysql-router-2.1.6/bin/mysqlrouter /usr/bin/

cp /opt/mysql-router-2.1.6/share/doc/mysqlrouter/sample\_mysqlrouter.conf /etc/mysqlrouter.conf

mkdir /opt/mysql-router-2.1.6/log

cat <<EOF>/etc/mysqlrouter.conf

[DEFAULT]

logging\_folder = /opt/mysql-router-2.1.6/log

plugin\_folder = /opt/mysql-router-2.1.6/lib/mysqlrouter

config\_folder = /opt/mysql-router-2.1.6/config

runtime\_folder = /opt/mysql-router-2.1.6/run

data\_folder = /opt/mysql-router-2.1.6/data

[logger]

level = INFO

[routing:basic\_failover]

bind\_address = 192.168.2.248

bind\_port = 7001

mode = read-write

destinations = 192.168.2.220:3310,192.168.2.221:3310

[routing:load\_balance]

bind\_address = 192.168.2.248

bind\_port = 7002

mode = read-only

destinations = 192.168.2.221:3310,192.168.2.222:3310

[keepalive]

interval = 60

EOF

-- 启动服务

mysqlrouter -c /etc/mysqlrouter.conf &

-- 查看日志是否异常

[root@localhost opt]# tail -fn100 /opt/mysql-router-2.1.6/log/mysqlrouter.log

2020-05-14 10:56:22 INFO [7fdc33b9c700] keepalive started with interval 60

2020-05-14 10:56:22 INFO [7fdc33b9c700] keepalive

2020-05-14 10:56:22 INFO [7fdc32b9a700] [routing:load\_balance] started: listening on 192.168.2.248:7002; read-only

2020-05-14 10:56:22 INFO [7fdc3339b700] [routing:basic\_failover] started: listening on 192.168.2.248:7001; read-write

2020-05-14 10:57:14 INFO [7f5b3ba44700] keepalive started with interval 60

2020-05-14 10:57:14 INFO [7f5b3ba44700] keepalive

2020-05-14 10:57:14 INFO [7f5b3b243700] [routing:basic\_failover] started: listening on 192.168.2.248:7001; read-write

2020-05-14 10:57:14 INFO [7f5b3aa42700] [routing:load\_balance] started: listening on 192.168.2.248:7002; read-only

2020-05-14 10:58:14 INFO [7f5b3ba44700] keepalive

2020-05-14 10:59:14 INFO [7f5b3ba44700] keepalive

2020-05-14 11:00:14 INFO [7f5b3ba44700] keepalive

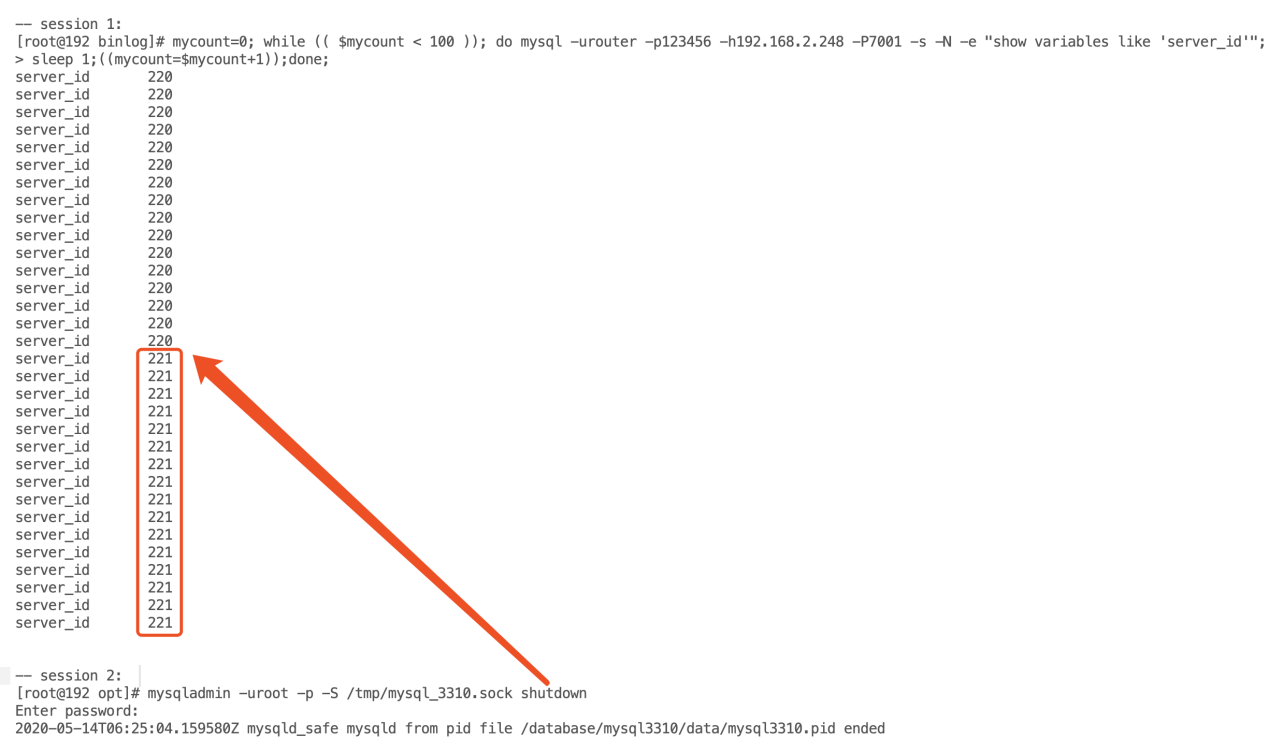
-- MySQL server集群授权mysqlrouter服务所在机器访问权限。

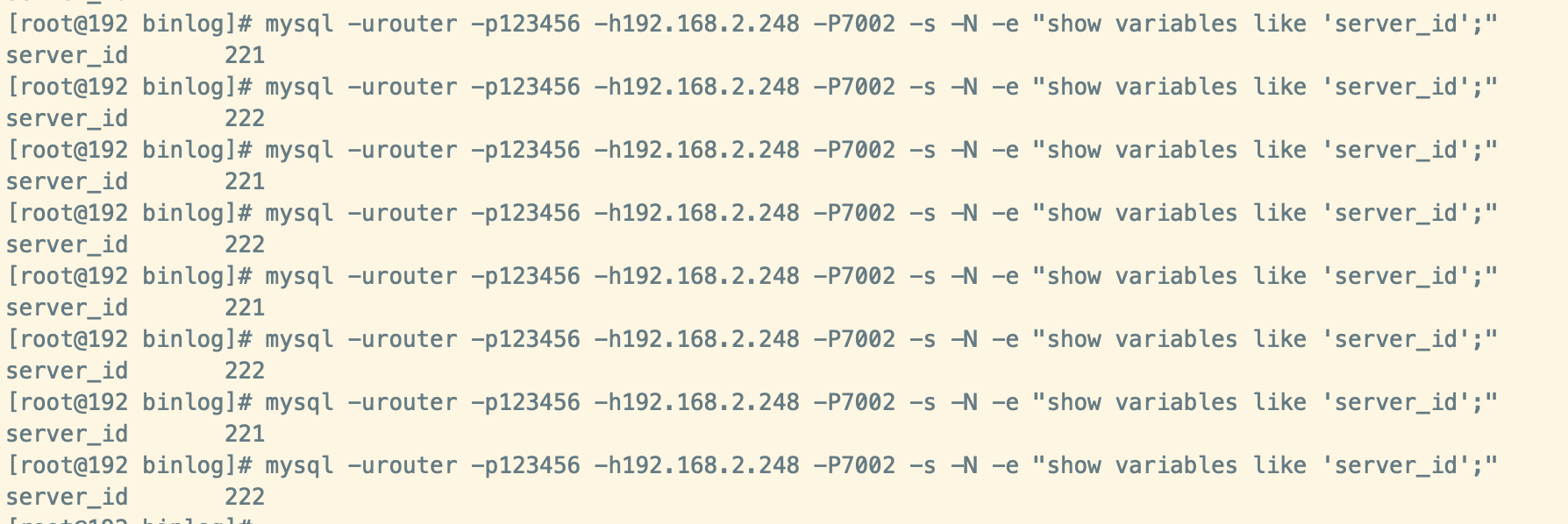
grant all on \*.\* to router@192.168.2.248 identified by '123456'; -- 2.220:3310

grant all on \*.\* to router@192.168.2.248 identified by '123456'; -- 2.221:3310

grant all on \*.\* to router@192.168.2.248 identified by '123456'; -- 2.222:3310

**-- 测试故障自动切换和读负载均衡情况**





## 2.2 Maxscale

### 2.2.1简介

**MaxScale 是 MariaDB 开发的一个数据库智能代理服务，允许根据数据库 SQL 语句将请求转向目标一个到多个服务器，可设定各种复杂程度的转向规则。MaxScale 用于透明的提供数据库的负载均衡和高可用性，同时提供高度可伸缩和灵活的架构，支持不同的协议和路由决策。MaxScale 使用 C 语言开发，利用 Linux 下的异步 I/O 功能。使用 epoll 作为事件驱动框架。**

### 2.2.2功能

1.带权重的读写分离(负载均衡)

2.SQL防火墙

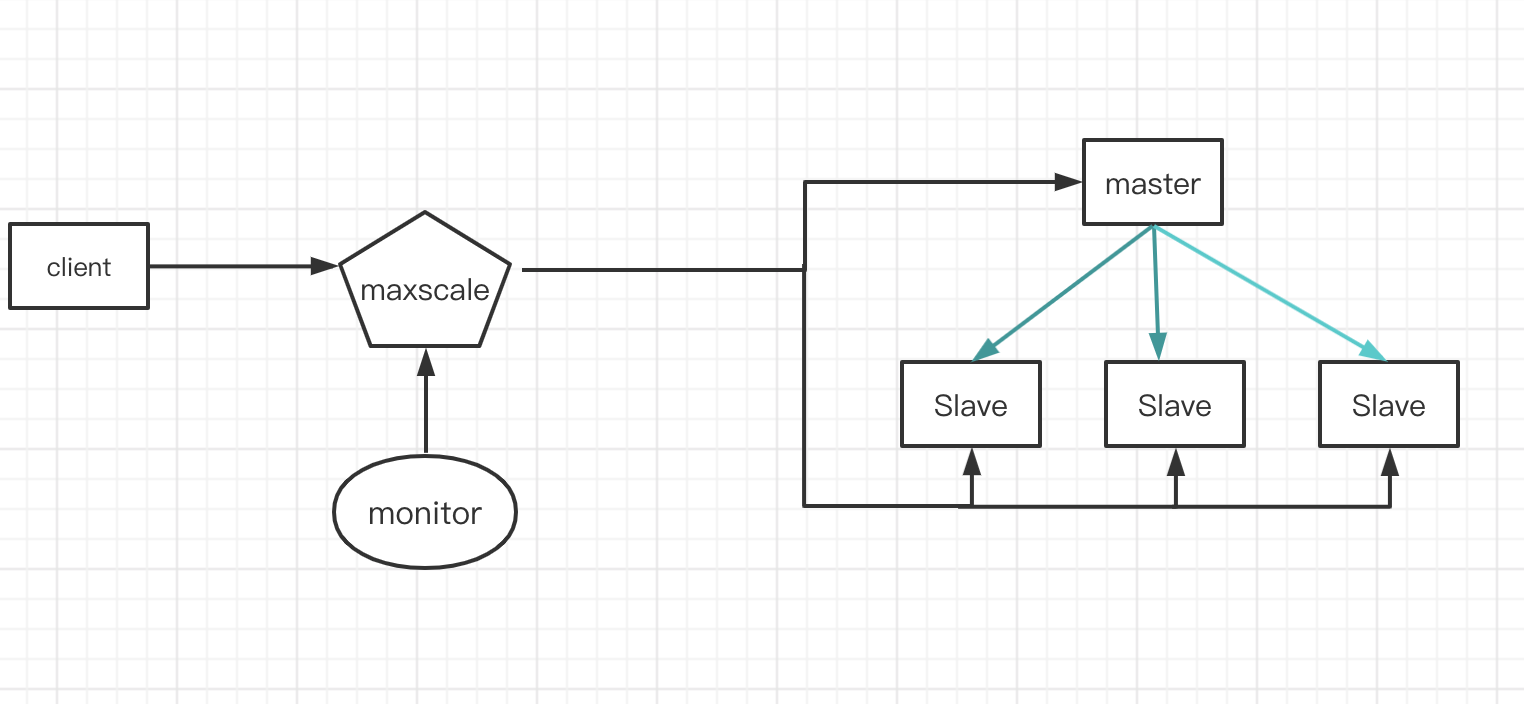
3.多种路由策略(Connection based， Statement based，Schema based)

4.自动检测MySQL master Failover

5.检测主从延时

6.多租户sharding架构

### 2.2.3架构图



### 2.2.4实验室

-- 192.168.2.220:3310 master

-- 192.168.2.221:3310 slave1

-- 192.168.2.222:3310 slave2

-- 192.168.2.248 maxscale

**-- MySQL Servers 创建监控账号**

grant replication slave, replication client on \*.\* to scalemon@'192.168.2.%' identified by 'scale123'; --220,221,222

**-- MySQL Servers 创建业务账号**

grant all privileges on \*.\* to maxscale@'192.168.2.%' identified by 'maxscale@qunje'; --220,221,222

**-- 简要安装步骤**

-- wget https://dlm.mariadb.com/612/MaxScale/2.0.6/centos/7/x86\_64/maxscale-2.0.6-1.centos.7.x86\_64.rpm

rpm -ivh maxscale-2.0.6-1.centos.7.x86\_64.rpm

mkdir -p /database/maxscale/{data,log,cache,pid}

groupadd maxscale

useradd maxscale -g maxscale

chown -R maxscale:maxscale /database/maxscale

maxkeys /database/maxscale/data/

maxpasswd /database/maxscale/data/.secrets scale123

80A661DDD0F2DF96C29811F702C74655

maxpasswd /database/maxscale/data/.secrets maxscale@qunje

E569C731C66626DFB3329FAB1D2D58F7

-- 新建maxscale配置文件

cat <<EOF >/etc/maxscale.cnf

[maxscale]

threads=1

maxlog=1

log\_to\_shm=0

log\_warning=1

log\_error=1

log\_notice=1

log\_info=1

log\_debug=0

#log\_augmentation=1

LimitNOFILE=65536

logdir=/database/maxscale/log/

datadir=/database/maxscale/data/

cachedir=/database/maxscale/cache/

piddir=/database/maxscale/pid/

libdir=/usr/lib64/maxscale/

execdir=/usr/bin/

[server1]

type=server

address=192.168.2.220

port=3310

protocol=MySQLBackend

router\_options=master

myweight=2

[server2]

type=server

address=192.168.2.221

port=3310

protocol=MySQLBackend

router\_options=slave

myweight=4

[server3]

type=server

address=192.168.2.222

port=3310

protocol=MySQLBackend

router\_options=slave

myweight=4

[MySQL Monitor]

type=monitor

module=mysqlmon

servers=server1,server2,server3

user=scalemon

passwd=80A661DDD0F2DF96C29811F702C74655

monitor\_interval=5000

backend\_connect\_timeout=300

backend\_write\_timeout=120

backend\_read\_timeout=120

detect\_stale\_master=true

#[Read-Only Service]

#type=service

#router=readconnroute

#servers=server2,server3

#user=maxscale

#passwd=E569C731C66626DFB3329FAB1D2D58F7

#router\_options=slave

[Read-Write Service]

type=service

router=readwritesplit

servers=server1,server2,server3

user=maxscale

passwd=E569C731C66626DFB3329FAB1D2D58F7

max\_slave\_replication\_lag=2

use\_sql\_variables\_in=all

enable\_root\_user=1

#router\_options=slave\_selection\_=LEAST\_GLOBAL\_CONNECTIONS

max\_slave\_connections=100%

weightby=myweight

[MaxAdmin Service]

type=service

router=cli

#[Read-Only Listener]

#type=listener

#service=Read-Only Service

#protocol=MySQLClient

#port=4001

[Read-Write Listener]

type=listener

service=Read-Write Service

protocol=MySQLClient

port=4007

[MaxAdmin Listener]

type=listener

service=MaxAdmin Service

protocol=maxscaled

socket=/database/maxscale/maxadmin.sock

port=6604

EOF

**-- 启动maxscale 服务**

maxscale -f /etc/maxscale.cnf

**-- 查看日志详情。（如有需要可开启log\_info=1）**

[root@localhost ~]# tail -fn30 /database/maxscale/log/maxscale1.log

2020-05-14 21:05:27 notice : Service cache: /database/maxscale/cache

2020-05-14 21:05:27 notice : The logging of warning messages has been enabled.

2020-05-14 21:05:27 notice : The logging of notice messages has been enabled.

2020-05-14 21:05:27 notice : The logging of informational messages has been disabled.

2020-05-14 21:05:27 notice : The logging of debug messages has been disabled.

2020-05-14 21:05:27 notice : Initialise CLI router module V1.0.0.

2020-05-14 21:05:27 notice : Loaded module cli: V1.0.0 from /usr/lib64/maxscale/libcli.so

2020-05-14 21:05:27 notice : Initializing statemend-based read/write split router module.

2020-05-14 21:05:27 notice : Loaded module readwritesplit: V1.1.0 from /usr/lib64/maxscale/libreadwritesplit.so

2020-05-14 21:05:27 notice : Initialise readconnroute router module V1.1.0.

2020-05-14 21:05:27 notice : Loaded module readconnroute: V1.1.0 from /usr/lib64/maxscale/libreadconnroute.so

2020-05-14 21:05:27 notice : Initialise the MySQL Monitor module V1.4.0.

2020-05-14 21:05:27 notice : Loaded module mysqlmon: V1.4.0 from /usr/lib64/maxscale/libmysqlmon.so

2020-05-14 21:05:27 notice : No query classifier specified, using default 'qc\_sqlite'.

2020-05-14 21:05:27 notice : Loaded module qc\_sqlite: V1.0.0 from /usr/lib64/maxscale/libqc\_sqlite.so

2020-05-14 21:05:27 notice : Using encrypted passwords. Encryption key: '/database/maxscale/data/.secrets'.

2020-05-14 21:05:27 notice : Loaded 8 MySQL Users for service [Read-Only Service].

2020-05-14 21:05:27 notice : Loaded module MySQLClient: V1.1.0 from /usr/lib64/maxscale/libMySQLClient.so

2020-05-14 21:05:27 notice : Listening connections at 0.0.0.0:4001 with protocol MySQL

2020-05-14 21:05:27 notice : Loaded 9 MySQL Users for service [Read-Write Service].

2020-05-14 21:05:27 notice : Listening connections at 0.0.0.0:4007 with protocol MySQL

2020-05-14 21:05:27 notice : Loaded module maxscaled: V2.0.0 from /usr/lib64/maxscale/libmaxscaled.so

2020-05-14 21:05:27 notice : Listening connections at 0.0.0.0:6604 with protocol MaxScale Admin

2020-05-14 21:05:27 notice : Listening connections at /database/maxscale/maxadmin.sock with protocol MaxScale Admin

2020-05-14 21:05:27 notice : MaxScale started with 1 server threads.

2020-05-14 21:05:27 notice : Started MaxScale log flusher.

2020-05-14 21:05:27 notice : Server changed state: server1[192.168.2.220:3310]: new\_master. [Running] -> [Master, Running]

2020-05-14 21:05:27 notice : Server changed state: server2[192.168.2.221:3310]: new\_slave. [Running] -> [Slave, Running]

2020-05-14 21:05:27 notice : Server changed state: server3[192.168.2.222:3310]: new\_slave. [Running] -> [Slave, Running]

**-- terminal 1**

[root@192 ~]# **mycount=0; while (( $mycount < 100 )); do mysql -umaxscale -h192.168.2.248 -pmaxscale@qunje -P4007 -s -N -e "show variables like 'server\_id'"; sleep 1;((mycount=$mycount+1));done;**

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

**terminal 2:**

mysql> stop slave; --192.168.2.221

**terminal 1: -- output**

[root@192 ~]# **mycount=0; while (( $mycount < 100 )); do mysql -umaxscale -h192.168.2.248 -pmaxscale@qunje -P4007 -s -N -e "show variables like 'server\_id'"; sleep 1;((mycount=$mycount+1));done;**

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

server\_id 222

-- 备注:其他功能不做演示。

## 2.3 LVS + keepalived

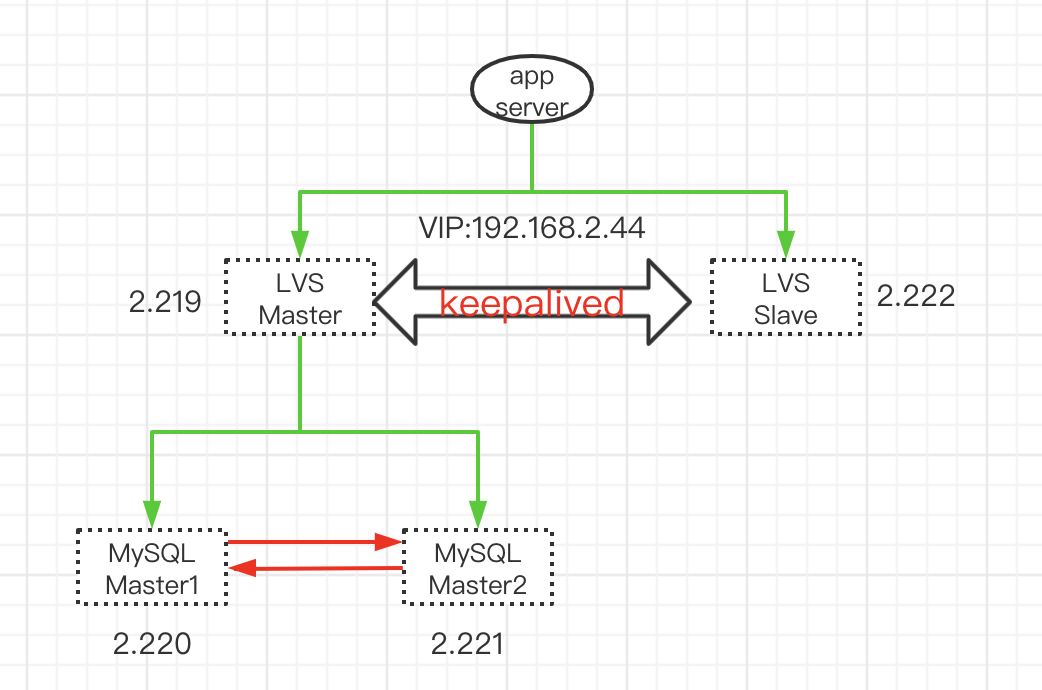
### 2.3.1简介

**LVS是 Linux Virtual Server 的简称，也就是Linux虚拟服务器。使用 LVS 可以达到的技术目标：通过 LVS 达到负载均衡，实现一个高性能高可用的Linux 服务器集群，它具有良好的可靠性、可扩展性和可操作性。**

### 2.3.2 功能

**LVS可以实现负载均衡，但是不能够进行健康检查，比如一个rs出现故障，LVS 仍然会把请求转发给故障的rs服务器，这样就会导致请求无效。keepalived 软件可以进行健康检查，而且能同时实现 LVS 的高可用性，解决LVS 单点故障的问题。**

### 2.3.3 架构图



### 2.3.4 实验室

**-- 219,222 Direct Servers 安装ipvs并加载ipvs模块**

yum -y install ipvsadm

modprobe ip\_vs

**-- 220,221 MySQL realservers 配置并启动**

vi /etc/init.d/realserver

#!/bin/sh

VIP=192.168.2.44

. /etc/rc.d/init.d/functions

case "$1" in

start)

/sbin/ifconfig lo down

/sbin/ifconfig lo up

echo "1" >/proc/sys/net/ipv4/conf/lo/arp\_ignore

echo "2" >/proc/sys/net/ipv4/conf/lo/arp\_announce

echo "1" >/proc/sys/net/ipv4/conf/all/arp\_ignore

echo "2" >/proc/sys/net/ipv4/conf/all/arp\_announce

/sbin/sysctl -p >/dev/null 2>&1

/sbin/ifconfig lo:0 $VIP netmask 255.255.255.255 up

/sbin/route add -host $VIP dev lo:0

echo "LVS-DR real server starts successfully.\n"

;;

stop)

/sbin/ifconfig lo:0 down

/sbin/route del $VIP >/dev/null 2>&1

echo "1" >/proc/sys/net/ipv4/conf/lo/arp\_ignore

echo "2" >/proc/sys/net/ipv4/conf/lo/arp\_announce

echo "1" >/proc/sys/net/ipv4/conf/all/arp\_ignore

echo "2" >/proc/sys/net/ipv4/conf/all/arp\_announce

echo "LVS-DR real server stopped.\n"

;;

status)

isLoOn=`/sbin/ifconfig lo:0 | grep "$VIP"`

isRoOn=`/bin/netstat -rn | grep "$VIP"`

if [ "$isLoON" == "" -a "$isRoOn" == "" ]; then

echo "LVS-DR real server has run yet."

else

echo "LVS-DR real server is running."

fi

exit 3

;;

\*)

echo "Usage: $0 {start|stop|status}"

exit 1

esac

exit 0

**-- 赋权并启动**

chmod +x /etc/init.d/realserver

/etc/init.d/realserver start

echo "/etc/init.d/realserver" >> /etc/rc.d/rc.local --加入自启动

**-- 219，222**

**-- 安装配置keepalived 实现directserver故障转移**

wget https://www.keepalived.org/software/keepalived-2.0.8.tar.gz

tar xzf keepalived-2.0.8.tar.gz

cd keepalived-2.0.8/

yum -y install gcc\* e2fsprogs-devel keyutils-libs-devel libsepol-devel libselinux-devel krb5-devel zlib-devel openssl-devel popt-devel libnfnetlink-devel

./configure --prefix=/usr/local/keepalived

make && make install

mkdir /etc/keepalived/

cp /usr/local/keepalived/etc/keepalived/keepalived.conf /etc/keepalived/

cp /usr/local/keepalived-2.0.8/keepalived/etc/init.d/keepalived /etc/init.d/

**-- 219 LVS-Master**

cat<<EOF>/etc/keepalived/keepalived.conf

! Configuration File for keepalived

global\_defs {

router\_id LVS\_DEVEL

}

vrrp\_instance VI\_1 {

state MASTER --配置为主

interface p8p1 --根据实际网卡设备

virtual\_router\_id 51

priority 100 -- 优先级

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

192.168.2.44 -- 虚拟IP地址

}

}

virtual\_server 192.168.2.44 3310 {

delay\_loop 6

lb\_algo wrr -- 加权轮询

lb\_kind DR

persistence\_timeout 10 -- 连接复用超时时长

protocol TCP

real\_server 192.168.2.220 3310 {

weight 3 -- 权重

TCP\_CHECK {

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

connect\_port 3310

}

}

real\_server 192.168.2.221 3310 {

weight 3

TCP\_CHECK {

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

connect\_port 3310

}

}

}

EOF

**-- 启动keepalive服务**

/etc/init.d/keepalived start

**-- 222 LVS-Slave 配置文件**

cat<<EOF>/etc/keepalived/keepalived.conf

! Configuration File for keepalived

global\_defs {

router\_id LVS\_DEVEL

}

vrrp\_instance VI\_1 {

state BACKUP

interface em1 --根据实际网卡设备

virtual\_router\_id 51

priority 90 -- 优先级

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

192.168.2.44

}

}

virtual\_server 192.168.2.44 3310 {

delay\_loop 6

lb\_algo wrr -- 加权重轮循算法

lb\_kind DR

persistence\_timeout 10

protocol TCP

real\_server 192.168.2.220 3310 {

weight 3

TCP\_CHECK {

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

connect\_port 3310

}

}

real\_server 192.168.2.221 3310 {

weight 3

TCP\_CHECK {

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

connect\_port 3310

}

}

}

EOF

**-- 启动keepalive服务**

/etc/init.d/keepalived start

**-- 219/222 LVS server 查看状态,调度算法,权重等信息**

[root@qj-mysql-node-219 ~]# ipvsadm -Ln

IP Virtual Server version 1.2.1 (size=4096)

Prot LocalAddress:Port Scheduler Flags

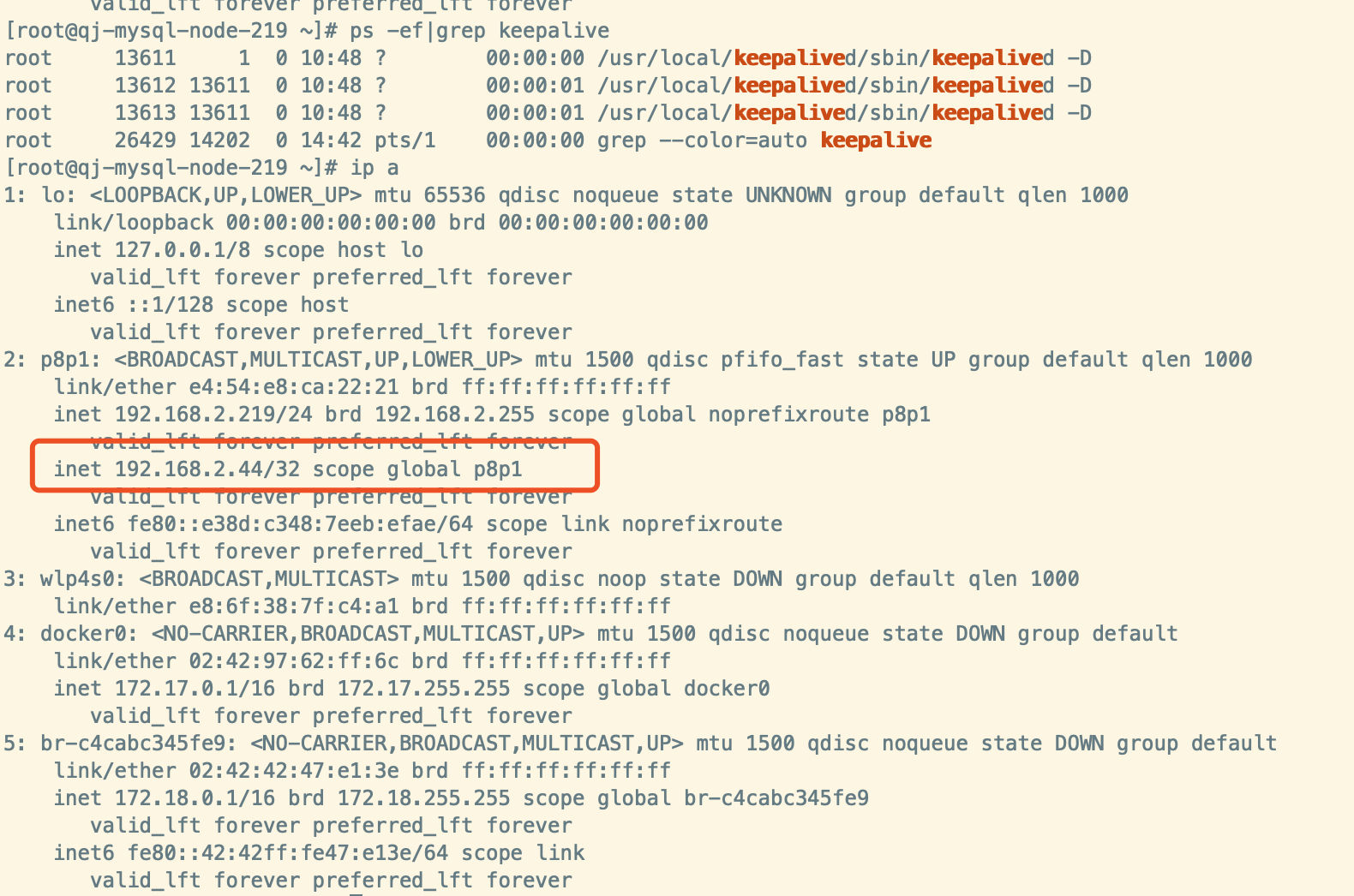
-> RemoteAddress:Port Forward Weight ActiveConn InActConn

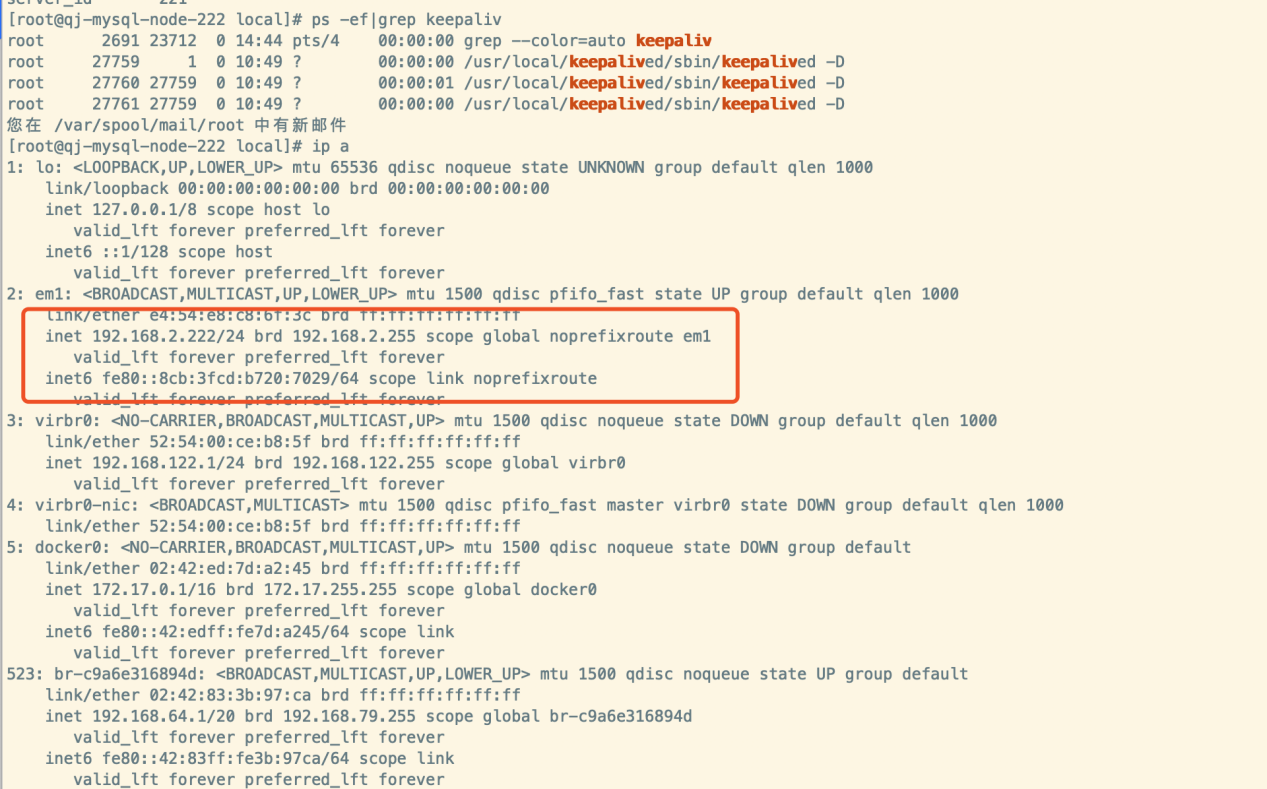
TCP 192.168.2.44:3310 wrr persistent 10

-> 192.168.2.220:3310 Route 3 0 0

-> 192.168.2.221:3310 Route 3 0 0

-- 查看目前VIP所在服务器





**-- 模拟LVS服务宕机**

-- terminal 1

[root@qj-mysql-node-248 ~]# mycount=0; while (( $mycount < 100 )); do mysql -ulvs\_user -plvs123 -h192.168.2.44 -P3310 -s -N -e "show variables like 'server\_id'";

> sleep 1;((mycount=$mycount+1));done;

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

server\_id 220

termial 2:

[root@qj-mysql-node-219 ~]# /etc/init.d/keepalived stop

Stopping keepalived (via systemctl): [ 确定 ]

[root@qj-mysql-node-219 ~]# ip a

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: p8p1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether e4:54:e8:ca:22:21 brd ff:ff:ff:ff:ff:ff

inet 192.168.2.219/24 brd 192.168.2.255 scope global noprefixroute p8p1

valid\_lft forever preferred\_lft forever

inet6 fe80::e38d:c348:7eeb:efae/64 scope link noprefixroute

valid\_lft forever preferred\_lft forever

-- termail 3

[root@qj-mysql-node-222 ~]# ip a

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: em1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether e4:54:e8:c8:6f:3c brd ff:ff:ff:ff:ff:ff

inet 192.168.2.222/24 brd 192.168.2.255 scope global noprefixroute em1

valid\_lft forever preferred\_lft forever

inet 192.168.2.44/32 scope global em1

valid\_lft forever preferred\_lft forever

inet6 fe80::8cb:3fcd:b720:7029/64 scope link noprefixroute

valid\_lft forever preferred\_lft forever

3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default qlen 1000

link/ether 52:54:00:ce:b8:5f brd ff:ff:ff:ff:ff:ff

inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0

valid\_lft forever preferred\_lft forever

**2.模拟MySQL-Master当机**

-- terminal 1

[root@localhost opt]# mycount=0; while (( $mycount < 100 )); do mysql -ulvs\_user -plvs123 -h192.168.2.44 -P3310 -s -N -e "show variables like 'server\_id'"; sleep 5;((mycount=$mycount+1));done;

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

server\_id 221

mysql: [Warning] Using a password on the command line interface can be insecure.

ERROR 2003 (HY000): Can't connect to MySQL server on '192.168.2.44' (111)

server\_id 220

server\_id 220

server\_id 220

-- termial 2

[root@qj-mysql-node-221 ~]# mysqladmin -uroot -p -S /tmp/mysql\_3310.sock shutdown

-- termial 3

[root@qj-mysql-node-219 opt]# ipvsadm -Ln

IP Virtual Server version 1.2.1 (size=4096)

Prot LocalAddress:Port Scheduler Flags

-> RemoteAddress:Port Forward Weight ActiveConn InActConn

TCP 192.168.2.44:3310 wrr persistent 3

-> 192.168.2.220:3310 Route 7 0 24

### 2.3.5总结

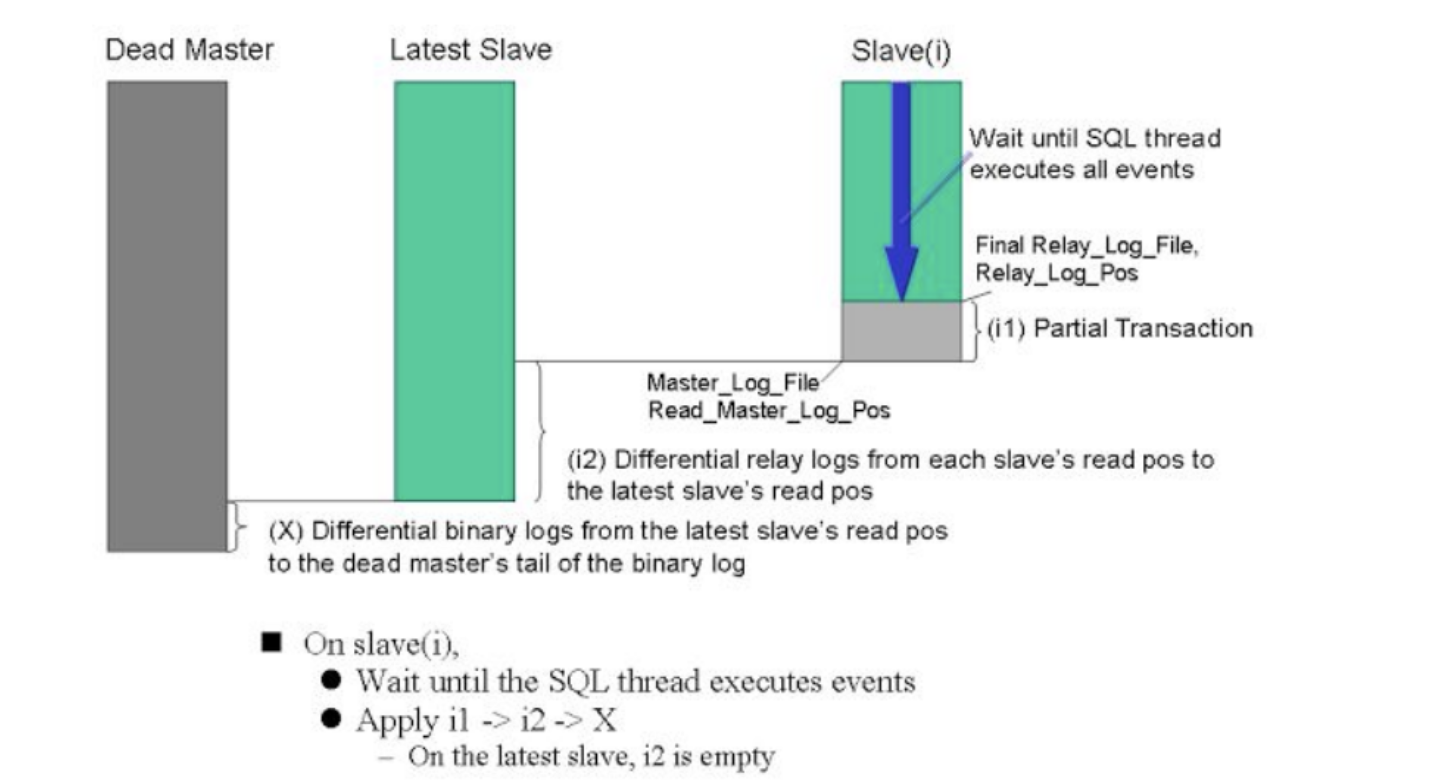
1. **MySQL server集群在双主模式下可实现读写负载均衡（即两台主机均可读写）。单主模式下必须使用多VIP网卡，配置读写和只读服务节点才可。那么应用层需要添加多个读写数据源。开发维护成本高。**
2. **单纯的主从架构，主节点宕机，从节点接管服务没问题。（可通过keepalived➕mysql\_check验证mysql服务是否正常）但是原主恢复后需要重新建立主从一致性环境。**
3. **目前我们的环境改造还没完成，表主键非自增无法避免双写模式下的主键冲突。**

## 2.4 MHA

### 2.4.1简介

MHA（Master High Availability）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0~30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。

### 2.4.2原理



1.从宕机崩溃的master保存二进制日志事件（binlog events）;

2.识别含有最新更新的slave;

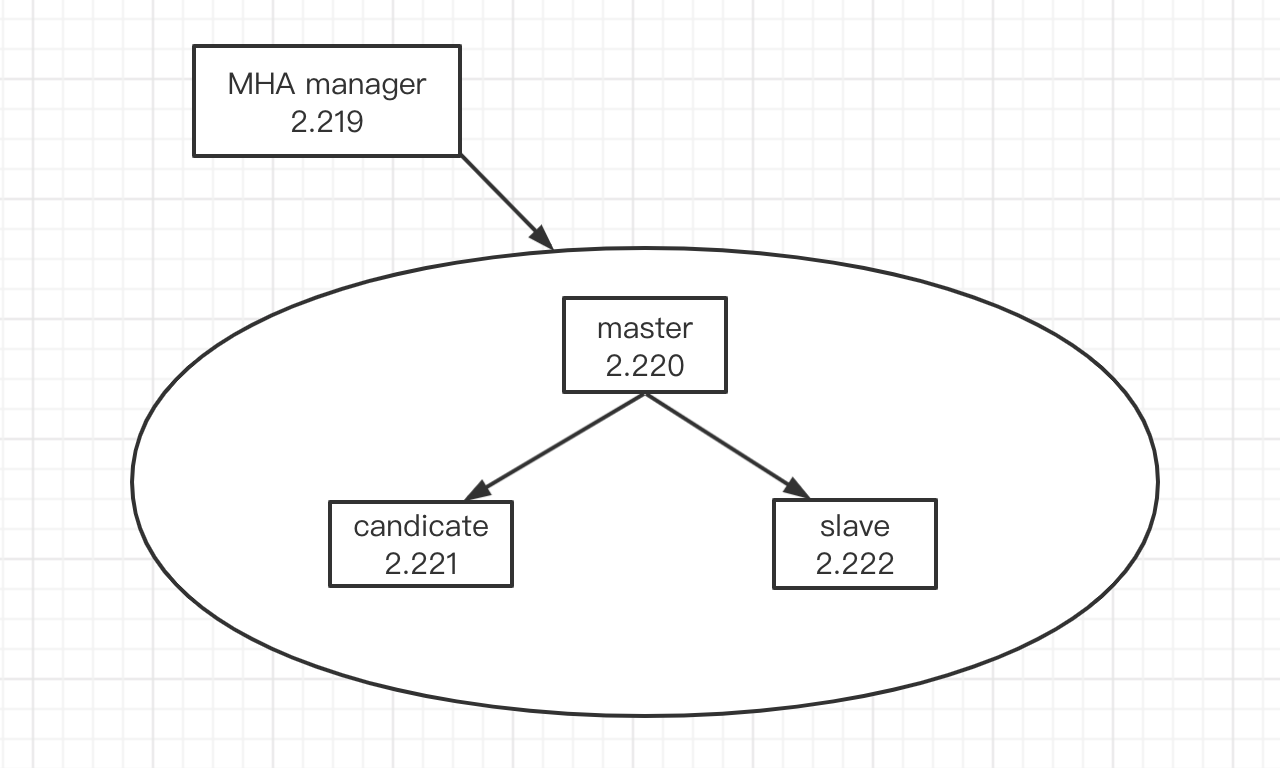
3.应用差异的中继日志（relay log）到其他的slave;

4.应用从master保存的二进制日志事件（binlog events);

5.提升一个slave为新的master;

6.使其他的slave连接新的master进行复制;

### 2.4.3架构图



### 2.4.4实验室

**-- MHA 集群角色信息**

192.168.2.220:3310 master

192.168.2.221:3310 candicate master

192.168.2.222:3310 slave

192.168.2.219 masterha monitor

-- download地址

https://github.com/yoshinorim/mha4mysql-manager/wiki/Downloads

**-- 所有节点安装perl模块**

yum -y install perl-DBD-MySQL perl-CPAN

-- mv /etc/yum.repos.d/CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo.backup

-- wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

**-- 所有节点均需安装 MHA Node**

tar xzf /opt/mha4mysql-node-0.56.tar.gz

cd /opt/mha4mysql-node-0.56/

perl Makefile.PL

make && make install

**-- 2.219 安装 MHA Manager**

yum -y install perl-DBD-MySQL perl-Config-Tiny perl-Log-Dispatch perl-Parallel-ForkManager perl-Time-HiRes

tar xzf mha4mysql-manager-0.56.tar.gz

cd mha4mysql-manager-0.56

perl Makefile.PL

make && make install

**-- 复制包文件至系统路径**

[root@qj-mysql-node-219 scripts]# cp /opt/mha4mysql-manager-0.56/samples/scripts/\* /usr/local/bin/

[root@qj-mysql-node-219 scripts]# ls -l /usr/local/bin/

-r-xr-xr-x. 1 root root 15977 5月 26 15:12 apply\_diff\_relay\_logs -- 应用差异中继日志工具

-r-xr-xr-x. 1 root root 7138 5月 26 15:39 config\_data

-r-xr-xr-x. 1 root root 15080 5月 26 15:38 corelist

-r-xr-xr-x. 1 root root 4807 5月 26 15:12 filter\_mysqlbinlog --过滤binlog日志

-r-xr-xr-x. 1 root root 4981 5月 26 15:36 json\_pp

-r-xr-xr-x. 1 root root 1995 5月 26 15:44 masterha\_check\_repl --检测复制状态

-r-xr-xr-x. 1 root root 1779 5月 26 15:44 masterha\_check\_ssh --检测环境通信

-r-xr-xr-x. 1 root root 1865 5月 26 15:44 masterha\_check\_status --检测服务状态

-r-xr-xr-x. 1 root root 3201 5月 26 15:44 masterha\_conf\_host -- 添加或删除配置文件节点相关信息

-r-xr-xr-x. 1 root root 2517 5月 26 15:44 masterha\_manager -- MHA主程序

-r-xr-xr-x. 1 root root 2165 5月 26 15:44 masterha\_master\_monitor -- 检测节点可用性

-r-xr-xr-x. 1 root root 2373 5月 26 15:44 masterha\_master\_switch -- 节点切换

-r-xr-xr-x. 1 root root 3879 5月 26 15:44 masterha\_secondary\_check -- 二次检验

-r-xr-xr-x. 1 root root 1739 5月 26 15:44 masterha\_stop

-rwxr-xr-x. 1 root root 3648 5月 26 15:57 master\_ip\_failover -- 故障转移

-rwxr-xr-x. 1 root root 9559 5月 26 15:57 master\_ip\_online\_change -- 在线切换

-r-xr-xr-x. 1 root root 1136 5月 26 15:42 package-stash-conflicts

-rwxr-xr-x. 1 root root 11867 5月 26 15:57 power\_manager

-r-xr-xr-x. 1 root root 13649 5月 26 15:39 prove

-r-xr-xr-x. 1 root root 7401 5月 26 15:12 purge\_relay\_logs -- 清理中继日志工具

-r-xr-xr-x. 1 root root 7263 5月 26 15:12 save\_binary\_logs -- 保存并复制主节点二进制日志

-rwxr-xr-x. 1 root root 1360 5月 26 15:57 send\_report

-- **所有节点配置互信机制，免密登陆。**

[root@qj-mysql-node-220 mha4mysql-node-0.56]# ssh-keygen -t rsa

Generating public/private rsa key pair.

Enter file in which to save the key (/root/.ssh/id\_rsa):

/root/.ssh/id\_rsa already exists.

Overwrite (y/n)? y

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /root/.ssh/id\_rsa.

Your public key has been saved in /root/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:nr1+K46VAaT7CunGQmf5FpMoyuafAs8w7CFHYPbdFcQ root@192.168.2.220

The key's randomart image is:

+---[RSA 2048]----+

| +o. |

|.o o E |

|o.. . o o |

| .. . o . |

|.. o..S . |

|=o+ =.+o o o |

|=O.=o. o+ + |

|.=+.+.o. o... |

|o.o=... .o+o.. |

+----[SHA256]-----+

[root@qj-mysql-node-220 .ssh]# cat /root/.ssh/id\_rsa.pub >> /root/.ssh/authorized\_keys

[root@qj-mysql-node-220 .ssh]# chmod 600 /root/.ssh/authorized\_keys

[root@qj-mysql-node-220 .ssh]# cat /root/.ssh/authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDN/YHOqxoZ+zYbI0TfdQWP0675vtlIVImgamwLAhmqYvQY1LViGoNUFk+lDvRHrHhaxWGPqi6zIt/JomxZcehMjbRscAlcSNPYRcew8spXzpEedvDbrgohmtR1pOXb72/+61XHMt2FHs94HMkqikX4VEwvHdRrDyTovPxDABDixVG1H+rvn4107N1GeboCE//n3xnLdW24oX3xnsRYJKyBdF5gLOYHeWeFvhX7fleO1jsT2XWBQc5nb7GNFnSSxoG4IltwghOm9lwL7FIY/VUPVKMQHBMMD84BrDTUk95DnpEnI3RIHIAZml307FAe1jr32NlTfc10m6CmCJuEvOY9 root@192.168.2.220

-- 将id\_rsa.pub的内容粘贴到其他所有服务器的/root/.ssh/authorized\_keys里（保证机器互相信任）。

-- **最终集群内所有节点**

[root@qj-mysql-node-220 ~]# cat /root/.ssh/authorized\_keys

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDgV9jP8/1qnOhtaNkxynXe4p7Zs0OEHGZKsi5JF9HO+ItzIOacMS/hTTtqiWQQC/qE/VtUiLywE6NhkcmzV61trwzc1FUbk2PnYt/I3trh3G6V+gNmix9qhoN8FgNAxc2Wog9s9enk5fH1oubtIKTfKonad2PrV5GT89t8ak/q55kRzD/1aFmW8UL5axMoz/3XzmuZlVmFmmN9AX2l8j1Bc12Ndv+8bYwNixQU+UcU0F1n7mob4Ts+Qn0hYherDA/mCABfKC7XYIcwozMdo2BOvVIq3j5xXuZS+J7sQ7nPVjk3KSXvwMce+wwS4LarAAaNUC63gaJ430ybhDxrJMdj root@qj-mysql-node-220

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDSedz6BqDEoxrfwox+iplIJzG0hoZwQJtVq1YE0rV1LUH2vWjRdaVkjCLYZ3X4tf+R28k/Qq9nP3LL8fA01BARlsDKwBsT0LNiGuC8mrYoIE3L4QfL6dt8uuuLxIEXoo103vAMpo9wCzRt73epde2BMRbjctB6qs3T3n5Odapf5VQJsBWmGv/H1cDZiZLG9sSoozx9y7FhCICR3v2YOmeyOaSzdmjCJDsT1C+P9U6CUKr0oyjvdJ3QiwnGx5RJQ49+bY0vdw0jjsBUEyoc0eFOBfLrZ81eJdBj21ZhMYnFBvkifdxy/IKAn5xw7Ep3jaCpxEz8S3RDpLCWQ/q/4xmP root@qj-mysql-node-221

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDrefVsJ6eQokH988XfwNYaYLE4ClVjS0Td+/sLXASqqo73ndJFzsr99dK0tOR2PHIdecwqKUHXsAa5eimCjuzeyOr65UDuJPut1HN7snzTTSGeCxieCa26wx8sl5YRw/6nGFLX3g63PktjI3RuLMtYiUx8JO7EjyYJXtVqYweXDlYTqbJlhWeWy98g/IcxrVgt7+KeH9PV1BA5QuPSHgDw9Yk0oU4/1uVBvMFBXRdEXx7ql5Xp7SmyFUnO/ACKcXvw3nUaeBTK7Lt1zkGFLJZgIX5z2+yf5bRZ9+p/tGQx45Qk5myVFacmiEaq+LDhVGrHnIe+D0ZIoKx7bNBJyKEr root@qj-mysql-node-222

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCqmKq7BWmGJJr4TQTEF9gJDKQ2K9b7/ZftgmldAfjdvf4Lzn/gOqvnmueKv30GzRkix3Sja2oBwW/vGRbTCQhnvt2ZNFkTCi1Okk5esB2X+KN2qxg1NYaxIZLAdIJFPvFVetVQzH52/SFM7TbNekebSpXF0915vggiDqDAnEeYgOaWv4Stowj0LcYT/CwSy/hgWGiWZwiTGoaLG3pDg7gfRiiynTerD1n0nlZWDbTinxsvmzCsERBAKYMuup8ZbSqt0zz7MN3HdATRPlWnKXvOMkA3T54CYM1Q+0U7y4J9ekX3eGWCr2HvVsUQTeJv7r+ONy1fAQr4hQviNDffeeqF root@qj-mysql-node-219

-- **创建必要文件**

[root@qj-mysql-node-219 ~]# mkdir -p /etc/masterha/log

[root@qj-mysql-node-219 ~]# cp /opt/mha4mysql-manager-0.56/samples/conf/app1.cnf /etc/masterha/

[root@qj-mysql-node-219 ~]# cp /etc/masterha/app1.cnf /etc/masterha/app1.cnf.bak

**-- 创建高可用配置文件**

[root@qj-mysql-node-219 ~]# cat <<EOF >/etc/masterha/app1.cnf

[server default]

user = manager

password = 123456

ssh\_user = root

repl\_user = replic

repl\_password = replic@qunje

ping\_interval = 1

ping\_type = SELECT

manager\_workdir=/etc/masterha

manager\_log=/etc/masterha/log/manager.log

remote\_workdir=/etc/masterha

master\_binlog\_dir="/database/mysql3310/binlog/"

master\_ip\_failover\_script="/usr/local/bin/master\_ip\_failover"

master\_ip\_online\_change\_script="/usr/local/bin/master\_ip\_online\_change"

secondary\_check\_script= /usr/local/bin/masterha\_secondary\_check -s qj-mysql-node-221 -s qj-mysql-node-220

shutdown\_script=""

report\_script=""

#check\_repl\_delay=0

[server1]

ip=192.168.2.220

hostname=qj-mysql-node-220

port=3310

master\_binlog\_dir="/database/mysql3310/binlog/"

candidate\_master=1

ignore\_fail=1

[server2]

ip=192.168.2.221

hostname=qj-mysql-node-221

port=3310

master\_binlog\_dir="/database/mysql3310/binlog/"

candidate\_master=1

ignore\_fail=1

[server3]

ip=192.168.2.222

hostname=qj-mysql-node-220

port=3310

master\_binlog\_dir="/database/mysql3310/binlog/"

no\_master=1

ignore\_fail=1

EOF

-- **三台 node （220，221，222）均授权manager节点（219）访问权限**

MySQL [(none)]> grant all on \*.\* to manager@'192.168.2.119' identified by '123456';

-- 重新编写故障转移脚本 ，脚本通过down 老服务>>>up 新服务实现VIP漂移。（未引入keepalived软件，防止脑裂情况）

[root@qj-mysql-node-219 bin]# cat /usr/local/bin/master\_ip\_failover

#!/usr/bin/env perl

use strict;

use warnings FATAL => 'all';

use Getopt::Long;

my (

$command, $ssh\_user, $orig\_master\_host, $orig\_master\_ip,

$orig\_master\_port, $new\_master\_host, $new\_master\_ip, $new\_master\_port

);

my $vip = '192.168.2.55/24'; # Virtual IP

my $key = "1";

my $int = "p8p1";

my $ssh\_start\_vip = "/sbin/ifconfig $int:$key $vip";

my $ssh\_stop\_vip = "/sbin/ifconfig $int:$key down";

my $arp\_effect = "/sbin/arping -Uq -s192.168.2.55 -I $int 192.168.2.1 -c 3";

#my $test = "echo successfull >/tmp/test.txt";

$ssh\_user = "root";

GetOptions(

'command=s' => \$command,

'ssh\_user=s' => \$ssh\_user,

'orig\_master\_host=s' => \$orig\_master\_host,

'orig\_master\_ip=s' => \$orig\_master\_ip,

'orig\_master\_port=i' => \$orig\_master\_port,

'new\_master\_host=s' => \$new\_master\_host,

'new\_master\_ip=s' => \$new\_master\_ip,

'new\_master\_port=i' => \$new\_master\_port,

);

exit &main();

sub main {

print "\n\nIN SCRIPT TEST====$ssh\_stop\_vip==$ssh\_start\_vip===\n\n";

if ( $command eq "stop" || $command eq "stopssh" ) {

my $exit\_code = 1;

eval {

print "Disabling the VIP on old master: $orig\_master\_host \n";

&stop\_vip();

$exit\_code = 0;

};

if ($@) {

warn "Got Error: $@\n";

exit $exit\_code;

}

exit $exit\_code;

}

elsif ( $command eq "start" ) {

my $exit\_code = 10;

eval {

print "Enabling the VIP - $vip on the new master - $new\_master\_host \n";

&start\_vip();

$exit\_code = 0;

};

if ($@) {

warn $@;

exit $exit\_code;

}

exit $exit\_code;

}

elsif ( $command eq "status" ) {

print "Checking the Status of the script.. OK \n";

#`ssh $ssh\_user\@cluster1 \" $ssh\_start\_vip \"`;

&status();

exit 0;

}

else {

&usage();

exit 1;

}

}

sub start\_vip() {

`ssh $ssh\_user\@$new\_master\_host \" $ssh\_start\_vip \"`;

`ssh $ssh\_user\@$new\_master\_host \" $arp\_effect \"`;

# `ssh $ssh\_user\@$new\_master\_host \" $test \"`;

}

sub stop\_vip() {

`ssh $ssh\_user\@$orig\_master\_host \" $ssh\_stop\_vip \"`;

}

sub status() {

print `ssh $ssh\_user\@$orig\_master\_host \" ip add show $int \"`;

}

sub usage {

print

"Usage: master\_ip\_failover --command=start|stop|stopssh|status --orig\_master\_host=host --orig\_master\_ip=ip --orig\_master\_port=port --new\_master\_host=host --new\_master\_ip=ip --new\_master\_port=port\n";

}

**至此MHA基础配置搭建完成。如下开始检测服务是否可用，并测试故障转移和主从自动建立等功能是否正常**

**-- 检测通信**

[root@qj-mysql-node-219 bin]# masterha\_check\_ssh --conf=/etc/masterha/app1.cnf

Tue May 26 19:27:07 2020 - [warning] Global configuration file /etc/masterha\_default.cnf not found. Skipping.

Tue May 26 19:27:07 2020 - [info] Reading application default configurations from /etc/masterha/app1.cnf..

Tue May 26 19:27:07 2020 - [info] Reading server configurations from /etc/masterha/app1.cnf..

Tue May 26 19:27:07 2020 - [info] Starting SSH connection tests..

Tue May 26 19:27:08 2020 - [debug]

Tue May 26 19:27:08 2020 - [debug] Connecting via SSH from root@qj-mysql-node-220(192.168.2.222:22) to root@qj-mysql-node-220(192.168.2.220:22)..

Tue May 26 19:27:08 2020 - [debug] ok.

Tue May 26 19:27:08 2020 - [debug] Connecting via SSH from root@qj-mysql-node-220(192.168.2.222:22) to root@qj-mysql-node-221(192.168.2.221:22)..

Tue May 26 19:27:09 2020 - [debug]

Tue May 26 19:27:07 2020 - [debug] Connecting via SSH from root@qj-mysql-node-221(192.168.2.221:22) to root@qj-mysql-node-220(192.168.2.220:22)..

Tue May 26 19:27:08 2020 - [debug] ok.

Tue May 26 19:27:08 2020 - [debug] Connecting via SSH from root@qj-mysql-node-221(192.168.2.221:22) to root@qj-mysql-node-220(192.168.2.222:22)..

Tue May 26 19:27:08 2020 - [debug] ok.

Tue May 26 19:27:09 2020 - [info] All SSH connection tests passed successfully.

-- **检测复制相关状态**

[root@qj-mysql-node-219 bin]# masterha\_check\_repl --conf=/etc/masterha/app1.cnf

Tue May 26 19:26:15 2020 - [warning] Global configuration file /etc/masterha\_default.cnf not found. Skipping.

Tue May 26 19:26:15 2020 - [info] Reading application default configurations from /etc/masterha/app1.cnf..

Tue May 26 19:26:15 2020 - [info] Reading server configurations from /etc/masterha/app1.cnf..

Tue May 26 19:26:15 2020 - [info] MHA::MasterMonitor version 0.56.

Tue May 26 19:26:16 2020 - [info] Dead Servers:

Tue May 26 19:26:16 2020 - [info] Alive Servers:

Tue May 26 19:26:16 2020 - [info] qj-mysql-node-220(192.168.2.220:3310)

Tue May 26 19:26:16 2020 - [info] qj-mysql-node-221(192.168.2.221:3310)

Tue May 26 19:26:16 2020 - [info] qj-mysql-node-220(192.168.2.222:3310)

Tue May 26 19:26:16 2020 - [info] Alive Slaves:

Tue May 26 19:26:16 2020 - [info] qj-mysql-node-221(192.168.2.221:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Tue May 26 19:26:16 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Tue May 26 19:26:16 2020 - [info] Primary candidate for the new Master (candidate\_master is set)

Tue May 26 19:26:16 2020 - [info] qj-mysql-node-220(192.168.2.222:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Tue May 26 19:26:16 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Tue May 26 19:26:16 2020 - [info] Not candidate for the new Master (no\_master is set)

Tue May 26 19:26:16 2020 - [info] Current Alive Master: qj-mysql-node-220(192.168.2.220:3310)

Tue May 26 19:26:16 2020 - [info] Checking slave configurations..

Tue May 26 19:26:16 2020 - [info] read\_only=1 is not set on slave qj-mysql-node-221(192.168.2.221:3310).

Tue May 26 19:26:16 2020 - [warning] relay\_log\_purge=0 is not set on slave qj-mysql-node-221(192.168.2.221:3310).

Tue May 26 19:26:16 2020 - [info] read\_only=1 is not set on slave qj-mysql-node-220(192.168.2.222:3310).

Tue May 26 19:26:16 2020 - [warning] relay\_log\_purge=0 is not set on slave qj-mysql-node-220(192.168.2.222:3310).

Tue May 26 19:26:16 2020 - [info] Checking replication filtering settings..

Tue May 26 19:26:16 2020 - [info] binlog\_do\_db= , binlog\_ignore\_db=

Tue May 26 19:26:16 2020 - [info] Replication filtering check ok.

Tue May 26 19:26:16 2020 - [info] Starting SSH connection tests..

Tue May 26 19:26:19 2020 - [info] All SSH connection tests passed successfully.

Tue May 26 19:26:19 2020 - [info] Checking MHA Node version..

Tue May 26 19:26:20 2020 - [info] Version check ok.

Tue May 26 19:26:20 2020 - [info] Checking SSH publickey authentication settings on the current master..

Tue May 26 19:26:20 2020 - [info] HealthCheck: SSH to qj-mysql-node-220 is reachable.

Tue May 26 19:26:21 2020 - [info] Master MHA Node version is 0.56.

Tue May 26 19:26:21 2020 - [info] Checking recovery script configurations on the current master..

Tue May 26 19:26:21 2020 - [info] Executing command: save\_binary\_logs --command=test --start\_pos=4 --binlog\_dir=/database/mysql3310/binlog/ --output\_file=/etc/masterha/save\_binary\_logs\_test --manager\_version=0.56 --start\_file=mysql3310-bin.000007

Tue May 26 19:26:21 2020 - [info] Connecting to root@qj-mysql-node-220(qj-mysql-node-220)..

Warning: Permanently added 'qj-mysql-node-220' (ECDSA) to the list of known hosts.

Creating /etc/masterha if not exists.. Creating directory /etc/masterha.. done.

ok.

Checking output directory is accessible or not..

ok.

Binlog found at /database/mysql3310/binlog/, up to mysql3310-bin.000007

Tue May 26 19:26:21 2020 - [info] Master setting check done.

Tue May 26 19:26:21 2020 - [info] Checking SSH publickey authentication and checking recovery script configurations on all alive slave servers..

Tue May 26 19:26:21 2020 - [info] Executing command : apply\_diff\_relay\_logs --command=test --slave\_user='manager' --slave\_host=qj-mysql-node-221 --slave\_ip=192.168.2.221 --slave\_port=3310 --workdir=/etc/masterha --target\_version=5.7.28-log --manager\_version=0.56 --relay\_log\_info=/database/mysql3310/relaylog/mysql3310-relay-bin --relay\_dir=/database/mysql3310/data/ --slave\_pass=xxx

Tue May 26 19:26:21 2020 - [info] Connecting to root@192.168.2.221(qj-mysql-node-221:22)..

Creating directory /etc/masterha.. done.

Checking slave recovery environment settings..

Opening /database/mysql3310/relaylog/mysql3310-relay-bin ... ok.

Relay log found at /database/mysql3310/relaylog, up to mysql3310-relay-log.000006

Temporary relay log file is /database/mysql3310/relaylog/mysql3310-relay-log.000006

Testing mysql connection and privileges..mysql: [Warning] Using a password on the command line interface can be insecure.

done.

Testing mysqlbinlog output.. done.

Cleaning up test file(s).. done.

Tue May 26 19:26:23 2020 - [info] Executing command : apply\_diff\_relay\_logs --command=test --slave\_user='manager' --slave\_host=qj-mysql-node-220 --slave\_ip=192.168.2.222 --slave\_port=3310 --workdir=/etc/masterha --target\_version=5.7.28-log --manager\_version=0.56 --relay\_log\_info=/database/mysql3310/relaylog/mysql3310-relay-bin --relay\_dir=/database/mysql3310/data/ --slave\_pass=xxx

Tue May 26 19:26:23 2020 - [info] Connecting to root@192.168.2.222(qj-mysql-node-220:22)..

Creating directory /etc/masterha.. done.

Checking slave recovery environment settings..

Opening /database/mysql3310/relaylog/mysql3310-relay-bin ... ok.

Relay log found at /database/mysql3310/relaylog, up to mysql3310-relay-log.000015

Temporary relay log file is /database/mysql3310/relaylog/mysql3310-relay-log.000015

Testing mysql connection and privileges..mysql: [Warning] Using a password on the command line interface can be insecure.

done.

Testing mysqlbinlog output.. done.

Cleaning up test file(s).. done.

Tue May 26 19:26:24 2020 - [info] Slaves settings check done.

Tue May 26 19:26:24 2020 - [info]

qj-mysql-node-220 (current master)

+--qj-mysql-node-221

+--qj-mysql-node-220

Tue May 26 19:26:24 2020 - [info] Checking replication health on qj-mysql-node-221..

Tue May 26 19:26:24 2020 - [info] ok.

Tue May 26 19:26:24 2020 - [info] Checking replication health on qj-mysql-node-220..

Tue May 26 19:26:24 2020 - [info] ok.

Tue May 26 19:26:24 2020 - [info] Checking master\_ip\_failover\_script status:

Tue May 26 19:26:24 2020 - [info] /usr/local/bin/master\_ip\_failover --command=status --ssh\_user=root --orig\_master\_host=qj-mysql-node-220 --orig\_master\_ip=192.168.2.220 --orig\_master\_port=3310

IN SCRIPT TEST====/sbin/ifconfig p8p1:1 down==/sbin/ifconfig p8p1:1 192.168.2.55/24===

Checking the Status of the script.. OK

2: p8p1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc fq\_codel state UP group default qlen 1000

link/ether e4:54:e8:ca:20:cc brd ff:ff:ff:ff:ff:ff

inet 192.168.2.147/24 brd 192.168.2.255 scope global noprefixroute dynamic p8p1

valid\_lft 72656sec preferred\_lft 72656sec

inet 192.168.2.220/24 brd 192.168.2.255 scope global secondary noprefixroute p8p1

valid\_lft forever preferred\_lft forever

inet6 fe80::c735:e45f:491f:a8c2/64 scope link noprefixroute

valid\_lft forever preferred\_lft forever

Tue May 26 19:26:24 2020 - [info] OK.

Tue May 26 19:26:24 2020 - [warning] shutdown\_script is not defined.

Tue May 26 19:26:24 2020 - [info] Got exit code 0 (Not master dead).

MySQL Replication Health is OK.

**-- 启动服务**

[root@qj-mysql-node-219 masterha]# nohup masterha\_manager --conf=/etc/masterha/app1.cnf --remove\_dead\_master\_conf --ignore\_last\_failover < /dev/null > /etc/masterha/log/manager.log 2>&1 &

[1] 21601

[root@qj-mysql-node-219 masterha]# tail -fn100 /etc/masterha/log/manager.log

Tue May 26 20:02:54 2020 - [warning] Global configuration file /etc/masterha\_default.cnf not found. Skipping.

Tue May 26 20:02:54 2020 - [info] Reading application default configurations from /etc/masterha/app1.cnf..

Tue May 26 20:02:54 2020 - [info] Reading server configurations from /etc/masterha/app1.cnf..

Tue May 26 20:02:54 2020 - [info] MHA::MasterMonitor version 0.56.

Tue May 26 20:02:56 2020 - [info] Dead Servers:

Tue May 26 20:02:56 2020 - [info] Alive Servers:

Tue May 26 20:02:56 2020 - [info] qj-mysql-node-220(192.168.2.220:3310)

Tue May 26 20:02:56 2020 - [info] qj-mysql-node-221(192.168.2.221:3310)

Tue May 26 20:02:56 2020 - [info] qj-mysql-node-220(192.168.2.222:3310)

Tue May 26 20:02:56 2020 - [info] Alive Slaves:

Tue May 26 20:02:56 2020 - [info] qj-mysql-node-221(192.168.2.221:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Tue May 26 20:02:56 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Tue May 26 20:02:56 2020 - [info] Primary candidate for the new Master (candidate\_master is set)

Tue May 26 20:02:56 2020 - [info] qj-mysql-node-220(192.168.2.222:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Tue May 26 20:02:56 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Tue May 26 20:02:56 2020 - [info] Not candidate for the new Master (no\_master is set)

Tue May 26 20:02:56 2020 - [info] Current Alive Master: qj-mysql-node-220(192.168.2.220:3310)

Tue May 26 20:02:56 2020 - [info] Checking slave configurations..

Tue May 26 20:02:56 2020 - [info] read\_only=1 is not set on slave qj-mysql-node-221(192.168.2.221:3310).

Tue May 26 20:02:56 2020 - [warning] relay\_log\_purge=0 is not set on slave qj-mysql-node-221(192.168.2.221:3310).

Tue May 26 20:02:56 2020 - [info] read\_only=1 is not set on slave qj-mysql-node-220(192.168.2.222:3310).

Tue May 26 20:02:56 2020 - [warning] relay\_log\_purge=0 is not set on slave qj-mysql-node-220(192.168.2.222:3310).

Tue May 26 20:02:56 2020 - [info] Checking replication filtering settings..

Tue May 26 20:02:56 2020 - [info] binlog\_do\_db= , binlog\_ignore\_db=

Tue May 26 20:02:56 2020 - [info] Replication filtering check ok.

Tue May 26 20:02:56 2020 - [info] Starting SSH connection tests..

Tue May 26 20:02:58 2020 - [info] All SSH connection tests passed successfully.

Tue May 26 20:02:58 2020 - [info] Checking MHA Node version..

Tue May 26 20:02:59 2020 - [info] Version check ok.

Tue May 26 20:02:59 2020 - [info] Checking SSH publickey authentication settings on the current master..

Tue May 26 20:02:59 2020 - [info] HealthCheck: SSH to qj-mysql-node-220 is reachable.

Tue May 26 20:02:59 2020 - [info] Master MHA Node version is 0.56.

Tue May 26 20:02:59 2020 - [info] Checking recovery script configurations on the current master..

Tue May 26 20:02:59 2020 - [info] Executing command: save\_binary\_logs --command=test --start\_pos=4 --binlog\_dir=/database/mysql3310/binlog/ --output\_file=/etc/masterha/save\_binary\_logs\_test --manager\_version=0.56 --start\_file=mysql3310-bin.000007

Tue May 26 20:02:59 2020 - [info] Connecting to root@qj-mysql-node-220(qj-mysql-node-220)..

Creating /etc/masterha if not exists.. ok.

Checking output directory is accessible or not..

ok.

Binlog found at /database/mysql3310/binlog/, up to mysql3310-bin.000007

Tue May 26 20:03:00 2020 - [info] Master setting check done.

Tue May 26 20:03:00 2020 - [info] Checking SSH publickey authentication and checking recovery script configurations on all alive slave servers..

Tue May 26 20:03:00 2020 - [info] Executing command : apply\_diff\_relay\_logs --command=test --slave\_user='manager' --slave\_host=qj-mysql-node-221 --slave\_ip=192.168.2.221 --slave\_port=3310 --workdir=/etc/masterha --target\_version=5.7.28-log --manager\_version=0.56 --relay\_log\_info=/database/mysql3310/relaylog/mysql3310-relay-bin --relay\_dir=/database/mysql3310/data/ --slave\_pass=xxx

Tue May 26 20:03:00 2020 - [info] Connecting to root@192.168.2.221(qj-mysql-node-221:22)..

Checking slave recovery environment settings..

Opening /database/mysql3310/relaylog/mysql3310-relay-bin ... ok.

Relay log found at /database/mysql3310/relaylog, up to mysql3310-relay-log.000006

Temporary relay log file is /database/mysql3310/relaylog/mysql3310-relay-log.000006

Testing mysql connection and privileges..mysql: [Warning] Using a password on the command line interface can be insecure.

done.

Testing mysqlbinlog output.. done.

Cleaning up test file(s).. done.

Tue May 26 20:03:01 2020 - [info] Executing command : apply\_diff\_relay\_logs --command=test --slave\_user='manager' --slave\_host=qj-mysql-node-220 --slave\_ip=192.168.2.222 --slave\_port=3310 --workdir=/etc/masterha --target\_version=5.7.28-log --manager\_version=0.56 --relay\_log\_info=/database/mysql3310/relaylog/mysql3310-relay-bin --relay\_dir=/database/mysql3310/data/ --slave\_pass=xxx

Tue May 26 20:03:01 2020 - [info] Connecting to root@192.168.2.222(qj-mysql-node-220:22)..

Checking slave recovery environment settings..

Opening /database/mysql3310/relaylog/mysql3310-relay-bin ... ok.

Relay log found at /database/mysql3310/relaylog, up to mysql3310-relay-log.000015

Temporary relay log file is /database/mysql3310/relaylog/mysql3310-relay-log.000015

Testing mysql connection and privileges..mysql: [Warning] Using a password on the command line interface can be insecure.

done.

Testing mysqlbinlog output.. done.

Cleaning up test file(s).. done.

Tue May 26 20:03:02 2020 - [info] Slaves settings check done.

Tue May 26 20:03:02 2020 - [info]

qj-mysql-node-220 (current master)

+--qj-mysql-node-221

+--qj-mysql-node-220

Tue May 26 20:03:02 2020 - [info] Checking master\_ip\_failover\_script status:

Tue May 26 20:03:02 2020 - [info] /usr/local/bin/master\_ip\_failover --command=status --ssh\_user=root --orig\_master\_host=qj-mysql-node-220 --orig\_master\_ip=192.168.2.220 --orig\_master\_port=3310

IN SCRIPT TEST====/sbin/ifconfig p8p1:1 down==/sbin/ifconfig p8p1:1 192.168.2.55/24===

Checking the Status of the script.. OK

2: p8p1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc fq\_codel state UP group default qlen 1000

link/ether e4:54:e8:ca:20:cc brd ff:ff:ff:ff:ff:ff

inet 192.168.2.147/24 brd 192.168.2.255 scope global noprefixroute dynamic p8p1

valid\_lft 70458sec preferred\_lft 70458sec

inet 192.168.2.220/24 brd 192.168.2.255 scope global secondary noprefixroute p8p1

valid\_lft forever preferred\_lft forever

inet6 fe80::c735:e45f:491f:a8c2/64 scope link noprefixroute

valid\_lft forever preferred\_lft forever

Tue May 26 20:03:02 2020 - [info] OK.

Tue May 26 20:03:02 2020 - [warning] shutdown\_script is not defined.

Tue May 26 20:03:02 2020 - [info] Set master ping interval 1 seconds.

Tue May 26 20:03:02 2020 - [info] Set secondary check script: /usr/local/bin/masterha\_secondary\_check -s qj-mysql-node-221 -s qj-mysql-node-220

Tue May 26 20:03:02 2020 - [info] Starting ping health check on qj-mysql-node-220(192.168.2.220:3310)..

Tue May 26 20:03:02 2020 - [info] Ping(SELECT) succeeded, waiting until MySQL doesn't respond..

**-- 检查高可用服务状态**

[root@qj-mysql-node-219 bin]# masterha\_check\_status --conf=/etc/masterha/app1.cnf

app1 (pid:21601) is running(0:PING\_OK), master:qj-mysql-node-220

-- **在2.220 手动启动 VIP**

[root@qj-mysql-node-220 bin]# /sbin/ifconfig p8p1:1 192.168.2.55/24

**--模拟主库宕机后切换流程**

**-- kill主库实例进程**

[root@qj-mysql-node-220 bin]# kill -9 18180 16768

**-- 观察切换过程如下：**

[root@qj-mysql-node-219 masterha]# tail -fn100 /etc/masterha/log/manager.log

Wed May 27 10:51:26 2020 - [warning] Got error on MySQL select ping: 2006 (MySQL server has gone away)

Wed May 27 10:51:26 2020 - [info] Executing secondary network check script: /usr/local/bin/masterha\_secondary\_check -s qj-mysql-node-221 -s qj-mysql-node-220 --user=root --master\_host=qj-mysql-node-220 --master\_ip=192.168.2.220 --master\_port=3310 --master\_user=manager --master\_password=123456 --ping\_type=SELECT

Wed May 27 10:51:26 2020 - [info] Executing SSH check script: exit 0

Wed May 27 10:51:27 2020 - [info] HealthCheck: SSH to qj-mysql-node-220 is reachable.

Monitoring server qj-mysql-node-221 is reachable, Master is not reachable from qj-mysql-node-221. OK.

Monitoring server qj-mysql-node-220 is reachable, Master is not reachable from qj-mysql-node-220. OK.

Wed May 27 10:51:27 2020 - [info] Master is not reachable from all other monitoring servers. Failover should start.

Wed May 27 10:51:27 2020 - [warning] Got error on MySQL connect: 2003 (Can't connect to MySQL server on '192.168.2.220' (111))

Wed May 27 10:51:27 2020 - [warning] Connection failed 2 time(s)..

Wed May 27 10:51:28 2020 - [warning] Got error on MySQL connect: 2003 (Can't connect to MySQL server on '192.168.2.220' (111))

Wed May 27 10:51:28 2020 - [warning] Connection failed 3 time(s)..

Wed May 27 10:51:29 2020 - [warning] Got error on MySQL connect: 2003 (Can't connect to MySQL server on '192.168.2.220' (111))

Wed May 27 10:51:29 2020 - [warning] Connection failed 4 time(s)..

Wed May 27 10:51:29 2020 - [warning] Master is not reachable from health checker!

Wed May 27 10:51:29 2020 - [warning] Master qj-mysql-node-220(192.168.2.220:3310) is not reachable!

Wed May 27 10:51:29 2020 - [warning] SSH is reachable.

Wed May 27 10:51:29 2020 - [info] Connecting to a master server failed. Reading configuration file /etc/masterha\_default.cnf and /etc/masterha/app1.cnf again, and trying to connect to all servers to check server status..

Wed May 27 10:51:29 2020 - [warning] Global configuration file /etc/masterha\_default.cnf not found. Skipping.

Wed May 27 10:51:29 2020 - [info] Reading application default configuration from /etc/masterha/app1.cnf..

Wed May 27 10:51:29 2020 - [info] Reading server configuration from /etc/masterha/app1.cnf..

Wed May 27 10:51:30 2020 - [info] GTID failover mode = 1

Wed May 27 10:51:30 2020 - [info] Dead Servers:

Wed May 27 10:51:30 2020 - [info] qj-mysql-node-220(192.168.2.220:3310)

Wed May 27 10:51:30 2020 - [info] Alive Servers:

Wed May 27 10:51:30 2020 - [info] qj-mysql-node-221(192.168.2.221:3310)

Wed May 27 10:51:30 2020 - [info] qj-mysql-node-220(192.168.2.222:3310)

Wed May 27 10:51:30 2020 - [info] Alive Slaves:

Wed May 27 10:51:30 2020 - [info] qj-mysql-node-221(192.168.2.221:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:30 2020 - [info] GTID ON

Wed May 27 10:51:30 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:30 2020 - [info] Primary candidate for the new Master (candidate\_master is set)

Wed May 27 10:51:30 2020 - [info] qj-mysql-node-220(192.168.2.222:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:30 2020 - [info] GTID ON

Wed May 27 10:51:30 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:30 2020 - [info] Not candidate for the new Master (no\_master is set)

Wed May 27 10:51:30 2020 - [info] Checking slave configurations..

Wed May 27 10:51:30 2020 - [info] read\_only=1 is not set on slave qj-mysql-node-221(192.168.2.221:3310).

Wed May 27 10:51:30 2020 - [info] read\_only=1 is not set on slave qj-mysql-node-220(192.168.2.222:3310).

Wed May 27 10:51:30 2020 - [info] Checking replication filtering settings..

Wed May 27 10:51:30 2020 - [info] Replication filtering check ok.

Wed May 27 10:51:30 2020 - [info] Master is down!

Wed May 27 10:51:30 2020 - [info] Terminating monitoring script.

Wed May 27 10:51:30 2020 - [info] Got exit code 20 (Master dead).

Wed May 27 10:51:30 2020 - [info] MHA::MasterFailover version 0.56.

Wed May 27 10:51:30 2020 - [info] Starting master failover.

Wed May 27 10:51:30 2020 - [info]

Wed May 27 10:51:30 2020 - [info] \* Phase 1: Configuration Check Phase..

Wed May 27 10:51:30 2020 - [info]

Wed May 27 10:51:32 2020 - [info] GTID failover mode = 1

Wed May 27 10:51:32 2020 - [info] Dead Servers:

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Checking master reachability via MySQL(double check)...

Wed May 27 10:51:32 2020 - [info] ok.

Wed May 27 10:51:32 2020 - [info] Alive Servers:

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-221(192.168.2.221:3310)

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-220(192.168.2.222:3310)

Wed May 27 10:51:32 2020 - [info] Alive Slaves:

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-221(192.168.2.221:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Primary candidate for the new Master (candidate\_master is set)

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-220(192.168.2.222:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Not candidate for the new Master (no\_master is set)

Wed May 27 10:51:32 2020 - [info] Starting GTID based failover.

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] \*\* Phase 1: Configuration Check Phase completed.

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] \* Phase 2: Dead Master Shutdown Phase..

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] Forcing shutdown so that applications never connect to the current master..

Wed May 27 10:51:32 2020 - [info] Executing master IP deactivation script:

Wed May 27 10:51:32 2020 - [info] /usr/local/bin/master\_ip\_failover --orig\_master\_host=qj-mysql-node-220 --orig\_master\_ip=192.168.2.220 --orig\_master\_port=3310 --command=stopssh --ssh\_user=root

IN SCRIPT TEST====/sbin/ifconfig p8p1:1 down==/sbin/ifconfig p8p1:1 192.168.2.55/24===

Disabling the VIP on old master: qj-mysql-node-220

Wed May 27 10:51:32 2020 - [info] done.

Wed May 27 10:51:32 2020 - [warning] shutdown\_script is not set. Skipping explicit shutting down of the dead master.

Wed May 27 10:51:32 2020 - [info] \* Phase 2: Dead Master Shutdown Phase completed.

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] \* Phase 3: Master Recovery Phase..

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] \* Phase 3.1: Getting Latest Slaves Phase..

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] The latest binary log file/position on all slaves is mysql3310-bin.000008:194

Wed May 27 10:51:32 2020 - [info] Retrieved Gtid Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:40

Wed May 27 10:51:32 2020 - [info] Latest slaves (Slaves that received relay log files to the latest):

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-221(192.168.2.221:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Primary candidate for the new Master (candidate\_master is set)

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-220(192.168.2.222:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Not candidate for the new Master (no\_master is set)

Wed May 27 10:51:32 2020 - [info] The oldest binary log file/position on all slaves is mysql3310-bin.000008:194

Wed May 27 10:51:32 2020 - [info] Retrieved Gtid Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:40

Wed May 27 10:51:32 2020 - [info] Oldest slaves:

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-221(192.168.2.221:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Primary candidate for the new Master (candidate\_master is set)

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-220(192.168.2.222:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Not candidate for the new Master (no\_master is set)

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] \* Phase 3.3: Determining New Master Phase..

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] Searching new master from slaves..

Wed May 27 10:51:32 2020 - [info] Candidate masters from the configuration file:

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-221(192.168.2.221:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Primary candidate for the new Master (candidate\_master is set)

Wed May 27 10:51:32 2020 - [info] Non-candidate masters:

Wed May 27 10:51:32 2020 - [info] qj-mysql-node-220(192.168.2.222:3310) Version=5.7.28-log (oldest major version between slaves) log-bin:enabled

Wed May 27 10:51:32 2020 - [info] GTID ON

Wed May 27 10:51:32 2020 - [info] Replicating from 192.168.2.220(192.168.2.220:3310)

Wed May 27 10:51:32 2020 - [info] Not candidate for the new Master (no\_master is set)

Wed May 27 10:51:32 2020 - [info] Searching from candidate\_master slaves which have received the latest relay log events..

Wed May 27 10:51:32 2020 - [info] New master is qj-mysql-node-221(192.168.2.221:3310)

Wed May 27 10:51:32 2020 - [info] Starting master failover..

Wed May 27 10:51:32 2020 - [info]

From:

qj-mysql-node-220(192.168.2.220:3310) (current master)

+--qj-mysql-node-221(192.168.2.221:3310)

+--qj-mysql-node-220(192.168.2.222:3310)

To:

qj-mysql-node-221(192.168.2.221:3310) (new master)

+--qj-mysql-node-220(192.168.2.222:3310)

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] \* Phase 3.3: New Master Recovery Phase..

Wed May 27 10:51:32 2020 - [info]

Wed May 27 10:51:32 2020 - [info] Waiting all logs to be applied..

Wed May 27 10:51:32 2020 - [info] done.

Wed May 27 10:51:32 2020 - [info] Getting new master's binlog name and position..

Wed May 27 10:51:32 2020 - [info] mysql3310-bin.000007:234

Wed May 27 10:51:32 2020 - [info] All other slaves should start replication from here. Statement should be: CHANGE MASTER TO MASTER\_HOST='qj-mysql-node-221 or 192.168.2.221', MASTER\_PORT=3310, MASTER\_AUTO\_POSITION=1, MASTER\_USER='replic', MASTER\_PASSWORD='xxx';

Wed May 27 10:51:32 2020 - [info] Master Recovery succeeded. File:Pos:Exec\_Gtid\_Set: mysql3310-bin.000007, 234, 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:1-40,

5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-11

Wed May 27 10:51:32 2020 - [info] Executing master IP activate script:

Wed May 27 10:51:32 2020 - [info] /usr/local/bin/master\_ip\_failover --command=start --ssh\_user=root --orig\_master\_host=qj-mysql-node-220 --orig\_master\_ip=192.168.2.220 --orig\_master\_port=3310 --new\_master\_host=qj-mysql-node-221 --new\_master\_ip=192.168.2.221 --new\_master\_port=3310 --new\_master\_user='manager' --new\_master\_password='123456'

Unknown option: new\_master\_user

Unknown option: new\_master\_password

IN SCRIPT TEST====/sbin/ifconfig p8p1:1 down==/sbin/ifconfig p8p1:1 192.168.2.55/24===

Enabling the VIP - 192.168.2.55/24 on the new master - qj-mysql-node-221

Wed May 27 10:51:35 2020 - [info] OK.

Wed May 27 10:51:35 2020 - [info] \*\* Finished master recovery successfully.

Wed May 27 10:51:35 2020 - [info] \* Phase 3: Master Recovery Phase completed.

Wed May 27 10:51:35 2020 - [info]

Wed May 27 10:51:35 2020 - [info] \* Phase 4: Slaves Recovery Phase..

Wed May 27 10:51:35 2020 - [info]

Wed May 27 10:51:35 2020 - [info]

Wed May 27 10:51:35 2020 - [info] \* Phase 4.1: Starting Slaves in parallel..

Wed May 27 10:51:35 2020 - [info]

Wed May 27 10:51:35 2020 - [info] -- Slave recovery on host qj-mysql-node-220(192.168.2.222:3310) started, pid: 2400. Check tmp log /etc/masterha/qj-mysql-node-220\_3310\_20200527105130.log if it takes time..

Wed May 27 10:51:40 2020 - [info]

Wed May 27 10:51:40 2020 - [info] Log messages from qj-mysql-node-220 ...

Wed May 27 10:51:40 2020 - [info]

Wed May 27 10:51:35 2020 - [info] Resetting slave qj-mysql-node-220(192.168.2.222:3310) and starting replication from the new master qj-mysql-node-221(192.168.2.221:3310)..

Wed May 27 10:51:35 2020 - [info] Executed CHANGE MASTER.

Wed May 27 10:51:36 2020 - [info] Slave started.

Wed May 27 10:51:39 2020 - [info] gtid\_wait(29acfba7-94b7-11ea-b6a7-e454e8ca20cc:1-40,

5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-11) completed on qj-mysql-node-220(192.168.2.222:3310). Executed 11 events.

Wed May 27 10:51:40 2020 - [info] End of log messages from qj-mysql-node-220.

Wed May 27 10:51:40 2020 - [info] -- Slave on host qj-mysql-node-220(192.168.2.222:3310) started.

Wed May 27 10:51:40 2020 - [info] All new slave servers recovered successfully.

Wed May 27 10:51:40 2020 - [info]

Wed May 27 10:51:40 2020 - [info] \* Phase 5: New master cleanup phase..

Wed May 27 10:51:40 2020 - [info]

Wed May 27 10:51:40 2020 - [info] Resetting slave info on the new master..

Wed May 27 10:51:40 2020 - [info] qj-mysql-node-221: Resetting slave info succeeded.

Wed May 27 10:51:40 2020 - [info] Master failover to qj-mysql-node-221(192.168.2.221:3310) completed successfully.

Wed May 27 10:51:40 2020 - [info] Deleted server1 entry from /etc/masterha/app1.cnf .

Wed May 27 10:51:40 2020 - [info]

----- Failover Report -----

app1: MySQL Master failover qj-mysql-node-220(192.168.2.220:3310) to qj-mysql-node-221(192.168.2.221:3310) succeeded

Master qj-mysql-node-220(192.168.2.220:3310) is down!

Check MHA Manager logs at qj-mysql-node-219:/etc/masterha/log/manager.log for details.

Started automated(non-interactive) failover.

Invalidated master IP address on qj-mysql-node-220(192.168.2.220:3310)

Selected qj-mysql-node-221(192.168.2.221:3310) as a new master.

qj-mysql-node-221(192.168.2.221:3310): OK: Applying all logs succeeded.

qj-mysql-node-221(192.168.2.221:3310): OK: Activated master IP address.

qj-mysql-node-220(192.168.2.222:3310): OK: Slave started, replicating from qj-mysql-node-221(192.168.2.221:3310)

qj-mysql-node-221(192.168.2.221:3310): Resetting slave info succeeded.

Master failover to qj-mysql-node-221(192.168.2.221:3310) completed successfully.

mysql> system ip a

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: em1: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000

link/ether e4:54:e8:c8:6f:3c brd ff:ff:ff:ff:ff:ff

inet 192.168.2.222/24 brd 192.168.2.255 scope global noprefixroute em1

valid\_lft forever preferred\_lft forever

inet6 fe80::8cb:3fcd:b720:7029/64 scope link noprefixroute

valid\_lft forever preferred\_lft forever

mysql> show slave status\G

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

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

Master\_Host: 192.168.2.221

Master\_User: replic

Master\_Port: 3310

Connect\_Retry: 60

Master\_Log\_File: mysql3310-bin.000007

Read\_Master\_Log\_Pos: 234

Relay\_Log\_File: mysql3310-relay-log.000010

Relay\_Log\_Pos: 415

Relay\_Master\_Log\_File: mysql3310-bin.000007

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

Replicate\_Do\_DB:

Replicate\_Ignore\_DB:

Replicate\_Do\_Table:

Replicate\_Ignore\_Table:

Replicate\_Wild\_Do\_Table:

Replicate\_Wild\_Ignore\_Table:

Last\_Errno: 0

Last\_Error:

Skip\_Counter: 0

Exec\_Master\_Log\_Pos: 234

Relay\_Log\_Space: 717

Until\_Condition: None

Until\_Log\_File:

Until\_Log\_Pos: 0

Master\_SSL\_Allowed: No

Master\_SSL\_CA\_File:

Master\_SSL\_CA\_Path:

Master\_SSL\_Cert:

Master\_SSL\_Cipher:

Master\_SSL\_Key:

Seconds\_Behind\_Master: 0

Master\_SSL\_Verify\_Server\_Cert: No

Last\_IO\_Errno: 0

Last\_IO\_Error:

Last\_SQL\_Errno: 0

Last\_SQL\_Error:

Replicate\_Ignore\_Server\_Ids:

Master\_Server\_Id: 221

Master\_UUID: 5a755e01-94bf-11ea-b2fa-e454e8ca3c54

Master\_Info\_File: /database/mysql3310/data/master.info

SQL\_Delay: 0

SQL\_Remaining\_Delay: NULL

Slave\_SQL\_Running\_State: Slave has read all relay log; waiting for more updates

Master\_Retry\_Count: 86400

Master\_Bind:

Last\_IO\_Error\_Timestamp:

Last\_SQL\_Error\_Timestamp:

Master\_SSL\_Crl:

Master\_SSL\_Crlpath:

Retrieved\_Gtid\_Set: 5a755e01-94bf-11ea-b2fa-e454e8ca3c54:4-11

Executed\_Gtid\_Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:1-40,

5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-11,

844d0e6d-94c0-11ea-a085-e454e8c86f3c:1-4

Auto\_Position: 1

Replicate\_Rewrite\_DB:

Channel\_Name:

Master\_TLS\_Version:

1 row in set (0.00 sec)

-- 重启宕机老主库，根据log记录位点重新搭建主从关系

[root@192 ~]# mysqld\_safe --defaults-file=/etc/mysql\_3310.cnf --user=mysql &

MySQL [(none)]> CHANGE MASTER TO MASTER\_HOST='192.168.2.221', MASTER\_PORT=3310, MASTER\_AUTO\_POSITION=1, MASTER\_USER='replic', MASTER\_PASSWORD='replic@qunje';

Query OK, 0 rows affected, 2 warnings (0.41 sec)

MySQL [(none)]> start slave;

Query OK, 0 rows affected (0.29 sec)

MySQL [(none)]> show slave status\G

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

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

Master\_Host: 192.168.2.221

Master\_User: replic

Master\_Port: 3310

Connect\_Retry: 60

Master\_Log\_File: mysql3310-bin.000007

Read\_Master\_Log\_Pos: 234

Relay\_Log\_File: mysql3310-relay-log.000010

Relay\_Log\_Pos: 415

Relay\_Master\_Log\_File: mysql3310-bin.000007

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

Replicate\_Do\_DB:

Replicate\_Ignore\_DB:

Replicate\_Do\_Table:

Replicate\_Ignore\_Table:

Replicate\_Wild\_Do\_Table:

Replicate\_Wild\_Ignore\_Table:

Last\_Errno: 0

Last\_Error:

Skip\_Counter: 0

Exec\_Master\_Log\_Pos: 234

Relay\_Log\_Space: 717

Until\_Condition: None

Until\_Log\_File:

Until\_Log\_Pos: 0

Master\_SSL\_Allowed: No

Master\_SSL\_CA\_File:

Master\_SSL\_CA\_Path:

Master\_SSL\_Cert:

Master\_SSL\_Cipher:

Master\_SSL\_Key:

Seconds\_Behind\_Master: 0

Master\_SSL\_Verify\_Server\_Cert: No

Last\_IO\_Errno: 0

Last\_IO\_Error:

Last\_SQL\_Errno: 0

Last\_SQL\_Error:

Replicate\_Ignore\_Server\_Ids:

Master\_Server\_Id: 221

Master\_UUID: 5a755e01-94bf-11ea-b2fa-e454e8ca3c54

Master\_Info\_File: mysql.slave\_master\_info

SQL\_Delay: 0

SQL\_Remaining\_Delay: NULL

Slave\_SQL\_Running\_State: Slave has read all relay log; waiting for more updates

Master\_Retry\_Count: 86400

Master\_Bind:

Last\_IO\_Error\_Timestamp:

Last\_SQL\_Error\_Timestamp:

Master\_SSL\_Crl:

Master\_SSL\_Crlpath:

Retrieved\_Gtid\_Set: 5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-11

Executed\_Gtid\_Set: 29acfba7-94b7-11ea-b6a7-e454e8ca20cc:1-40,

5a755e01-94bf-11ea-b2fa-e454e8ca3c54:1-11

Auto\_Position: 1

Replicate\_Rewrite\_DB:

Channel\_Name:

Master\_TLS\_Version:

1 row in set (0.00 sec)

### 2.4.5 总结

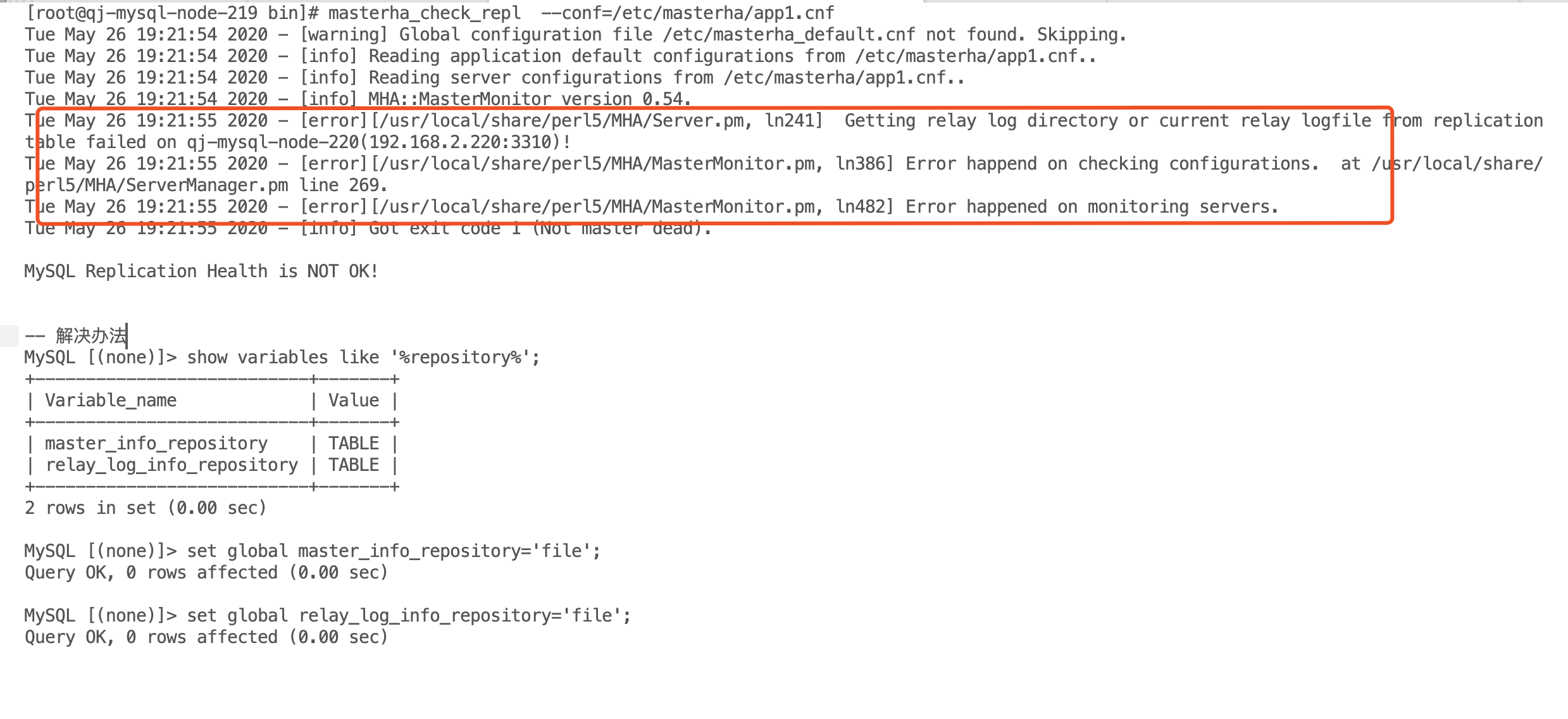
1. MHA tarball安装依赖perl模块，老版本存在较多限制和bug。需注意数据库参数配置是否满足版本的要求。维护成本高。

master\_info\_repository

relay\_log\_info\_repository

read\_only

pruge\_relay\_log



1. MHA结合半同步复制可最大程度保证数据完整性，一致性。（主库所在服务器宕机无法完成文件传输）
2. 发生故障切换后，原配置文件会被“破坏”（masterha\_conf\_host工具会删除故障节点信息），需提前备份配置文件。

## 2.5 ProxySQL

### 2.5.1简介

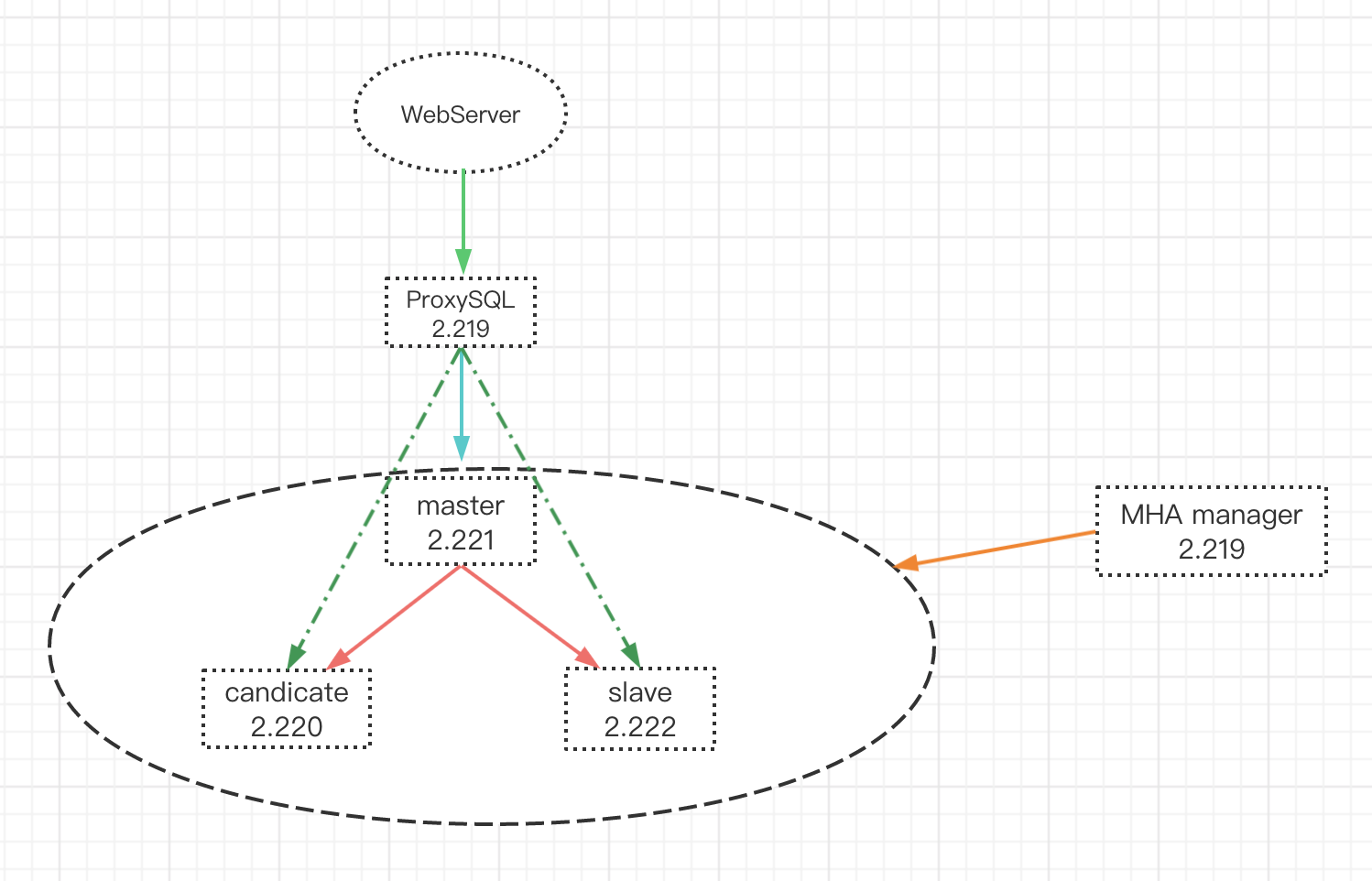
ProxySQL是MySQL的一款中间件产品，是灵活强大的MySQL代理层，可以实现读写分离，可自定义基于用户、基于schema、基于语句的规则对SQL语句进行路由，支持动态指定某个SQL进行缓存，支持动态加载配置、故障转移和一些SQL的过滤功能，可以实现简单的sharding。

3

### 2.5.2 功能亮点

-  连接池，而且是 multiplexing；  
-  主机和用户的最大连接数限制；  
-  自动下线后端DB  
    -  延迟超过阀值  
    -  ping 延迟超过阀值  
    -  网络不通或宕机  
-  强大的规则路由引擎；  
   -  读写分离  
   -  查询重写  
   -  sql流量镜像  
-  支持prepared statement；  
-  支持Query Cache；  
-  支持负载均衡，与gelera结合自动failover  
-  支持动态加载配置

### 2.5.3 架构图



### 2.5.4 实验室

-- 下载安装proxysql最新稳定版本。具体功能细节可见官网release\_node

wget https://github.com/sysown/proxysql/releases/download/v2.0.12/proxysql-2.0.12-1-centos7.x86\_64.rpm

yum -y install gnutls

rpm -ivh proxysql-2.0.12-1-centos7.x86\_64.rpm

[root@qj-mysql-node-219 opt]# rpm -ql proxysql

/etc/logrotate.d/proxysql

/etc/proxysql.cnf

/etc/systemd/system/proxysql-initial.service

/etc/systemd/system/proxysql.service

/usr/bin/proxysql

/usr/share/proxysql/tools/proxysql\_galera\_checker.sh

/usr/share/proxysql/tools/proxysql\_galera\_writer.pl

-- 启动服务

[root@qj-mysql-node-219 opt]# systemctl start proxysql.service

[root@qj-mysql-node-219 opt]# ps -ef|grep proxy|grep -v grep

proxysql 19965 1 0 17:07 ? 00:00:00 /usr/bin/proxysql --idle-threads -c /etc/proxysql.cnf

proxysql 19966 19965 2 17:07 ? 00:00:00 /usr/bin/proxysql --idle-threads -c /etc/proxysql.cnf

[root@qj-mysql-node-219 opt]# systemctl status proxysql.service

● proxysql.service - High Performance Advanced Proxy for MySQL

Loaded: loaded (/etc/systemd/system/proxysql.service; enabled; vendor preset: disabled)

Active: active (running) since 三 2020-05-27 17:07:44 CST; 43s ago

Process: 19963 ExecStart=/usr/bin/proxysql --idle-threads -c /etc/proxysql.cnf $PROXYSQL\_OPTS (code=exited, status=0/SUCCESS)

Main PID: 19965 (proxysql)

Tasks: 24

Memory: 102.6M

CGroup: /system.slice/proxysql.service

├─19965 /usr/bin/proxysql --idle-threads -c /etc/proxysql.cnf

└─19966 /usr/bin/proxysql --idle-threads -c /etc/proxysql.cnf

5月 27 17:07:44 qj-mysql-node-219 systemd[1]: Starting High Performance Advanced Proxy for MySQL...

5月 27 17:07:44 qj-mysql-node-219 proxysql[19963]: 2020-05-27 17:07:44 [INFO] Using config file /etc/proxysql.cnf

5月 27 17:07:44 qj-mysql-node-219 proxysql[19963]: 2020-05-27 17:07:44 [INFO] Using OpenSSL version: OpenSSL 1.1.1d 10 Sep 2019

5月 27 17:07:44 qj-mysql-node-219 proxysql[19963]: 2020-05-27 17:07:44 [INFO] No SSL keys/certificates found in datadir (/var/lib/proxysql). Generating new keys/certificates.

5月 27 17:07:44 qj-mysql-node-219 systemd[1]: Started High Performance Advanced Proxy for MySQL.

[root@qj-mysql-node-219 opt]# netstat -ntpl |grep proxysql

tcp 0 0 0.0.0.0:6032 0.0.0.0:\* LISTEN 19966/proxysql

tcp 0 0 0.0.0.0:6033 0.0.0.0:\* LISTEN 19966/proxysql

[root@qj-mysql-node-219 opt]# mysql -uadmin -padmin -P6032 -h127.0.0.1 --prompt='<proxysql>'

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 2

Server version: 5.5.30 (ProxySQL Admin Module)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

<proxysql>

<proxysql>insert into mysql\_servers(hostgroup\_id,hostname,port,weight,comment) values(10,'192.168.2.221',3310,1,'master'),(20,'192.168.2.220',3310,1,'slave1'),(20,'192.168.2.222',3310,1,'slave2');

Query OK, 3 rows affected (0.00 sec)

<proxysql> load mysql servers to runtime;

Query OK, 0 rows affected (0.01 sec)

<proxysql> save mysql servers to disk;

Query OK, 0 rows affected (0.73 sec)

<proxysql>select \* from mysql\_servers;

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

| hostgroup\_id | hostname | port | gtid\_port | status | weight | compression | max\_connections | max\_replication\_lag | use\_ssl | max\_latency\_ms | comment |

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

| 10 | 192.168.2.221 | 3310 | 0 | ONLINE | 1 | 0 | 1000 | 0 | 0 | 0 | master |

| 20 | 192.168.2.222 | 3310 | 0 | ONLINE | 1 | 0 | 1000 | 0 | 0 | 0 | slave2 |

| 20 | 192.168.2.220 | 3310 | 0 | ONLINE | 1 | 0 | 1000 | 0 | 0 | 0 | slave1 |

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

3 rows in set (0.00 sec)

-- 2.221 MySQL servers授权ProxySQL 节点访问权限。（如果从库未配置replicate\_ignore\_db，则主库会自动同步权限至所有从库。）

mysql> grant all on \*.\* to monitor@'192.168.2.219' identified by 'monitor';

Query OK, 0 rows affected, 1 warning (0.06 sec)

<proxysql>select \* from mysql\_server\_connect\_log;

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

| hostname | port | time\_start\_us | connect\_success\_time\_us | connect\_error |

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

| 192.168.2.222 | 3310 | 1590644420327383 | 3376 | NULL |

| 192.168.2.221 | 3310 | 1590644420820282 | 3366 | NULL |

| 192.168.2.220 | 3310 | 1590644421313738 | 5346 | NULL |

| 192.168.2.222 | 3310 | 1590644480327627 | 2999 | NULL |

| 192.168.2.221 | 3310 | 1590644480948540 | 5170 | NULL |

| 192.168.2.220 | 3310 | 1590644481569607 | 5456 | NULL |

| 192.168.2.221 | 3310 | 1590644540327866 | 4611 | NULL |

| 192.168.2.220 | 3310 | 1590644540818269 | 5320 | NULL |

| 192.168.2.222 | 3310 | 1590644541308728 | 3715 | NULL |

| 192.168.2.221 | 3310 | 1590644600328135 | 5526 | NULL |

| 192.168.2.220 | 3310 | 1590644601123660 | 5271 | NULL |

| 192.168.2.222 | 3310 | 1590644601919152 | 3645 | NULL |

| 192.168.2.222 | 3310 | 1590644660328493 | 3428 | NULL |

| 192.168.2.220 | 3310 | 1590644660986907 | 4858 | NULL |

| 192.168.2.221 | 3310 | 1590644661645492 | 5442 | NULL |

| 192.168.2.221 | 3310 | 1590644720328706 | 5276 | NULL |

| 192.168.2.220 | 3310 | 1590644721066221 | 5292 | NULL |

| 192.168.2.222 | 3310 | 1590644721803782 | 3404 | NULL |

| 192.168.2.222 | 3310 | 1590644780328869 | 3201 | NULL |

| 192.168.2.220 | 3310 | 1590644780857485 | 5465 | NULL |

| 192.168.2.221 | 3310 | 1590644781386051 | 5544 | NULL |

| 192.168.2.221 | 3310 | 1590644840329147 | 5474 | NULL |

| 192.168.2.220 | 3310 | 1590644840959953 | 4916 | NULL |

| 192.168.2.222 | 3310 | 1590644841590923 | 3428 | NULL |

| 192.168.2.222 | 3310 | 1590644900328797 | 2816 | NULL |

| 192.168.2.221 | 3310 | 1590644900902221 | 5555 | NULL |

| 192.168.2.220 | 3310 | 1590644901475546 | 5556 | NULL |

| 192.168.2.222 | 3310 | 1590644960329555 | 3654 | NULL |

| 192.168.2.220 | 3310 | 1590644960955043 | 4655 | NULL |

| 192.168.2.221 | 3310 | 1590644961581058 | 5490 | NULL |

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

30 rows in set (0.00 sec)

<proxysql>insert into mysql\_replication\_hostgroups values(10,20,'read\_only','read-write split');

Query OK, 1 row affected (0.00 sec)

-- 从库 220，222 需要配置只读状态。切记！

MySQL [(220)]> set global read\_only=on;

Query OK, 0 rows affected (0.00 sec)

MySQL [(222)]> set global read\_only=on;

Query OK, 0 rows affected (0.00 sec)

-- 通过proxySQL查看

<proxysql>select \* from mysql\_server\_read\_only\_log order by time\_start\_us desc limit 20;

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

| hostname | port | time\_start\_us | success\_time\_us | read\_only | error |

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

| 192.168.2.220 | 3310 | 1590645197795343 | 1935 | 1 | NULL |

| 192.168.2.221 | 3310 | 1590645197777932 | 2110 | 0 | NULL |

| 192.168.2.222 | 3310 | 1590645197760507 | 1556 | 1 | NULL |

| 192.168.2.222 | 3310 | 1590645196294470 | 1519 | 1 | NULL |

| 192.168.2.220 | 3310 | 1590645196277394 | 2150 | 1 | NULL |

| 192.168.2.221 | 3310 | 1590645196260238 | 2111 | 0 | NULL |

| 192.168.2.220 | 3310 | 1590645194787683 | 2124 | 1 | NULL |

| 192.168.2.222 | 3310 | 1590645194773795 | 872 | 1 | NULL |

| 192.168.2.221 | 3310 | 1590645194759934 | 2116 | 0 | NULL |

| 192.168.2.220 | 3310 | 1590645193287287 | 2174 | 1 | NULL |

| 192.168.2.222 | 3310 | 1590645193273531 | 1511 | 1 | NULL |

| 192.168.2.221 | 3310 | 1590645193259686 | 2125 | 0 | NULL |

| 192.168.2.222 | 3310 | 1590645191791949 | 1571 | 1 | NULL |

| 192.168.2.221 | 3310 | 1590645191775683 | 2129 | 0 | NULL |

| 192.168.2.220 | 3310 | 1590645191759398 | 2143 | 1 | NULL |

| 192.168.2.221 | 3310 | 1590645190282099 | 2096 | 0 | NULL |

| 192.168.2.222 | 3310 | 1590645190270651 | 1483 | 1 | NULL |

| 192.168.2.220 | 3310 | 1590645190259168 | 2055 | 1 | NULL |

| 192.168.2.221 | 3310 | 1590645188792327 | 2114 | 0 | NULL |

| 192.168.2.220 | 3310 | 1590645188775726 | 2115 | 1 | NULL |

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

20 rows in set (0.00 sec)

-- 221主库授权应用

MySQL [(221)]> grant all on \*.\* to qjweb@'192.168.2.219' identified by '123456';

Query OK, 0 rows affected, 1 warning (0.07 sec)

MySQL [(221)]> grant all on \*.\* to qjweb\_read@'192.168.2.219' identified by '654321';

Query OK, 0 rows affected, 1 warning (0.04 sec)

-- 配置mysql\_users数据

<proxysql>insert into mysql\_users(username,password,default\_hostgroup) values('qjweb','123456',10);

Query OK, 1 row affected (0.00 sec)

<proxysql>insert into mysql\_users(username,password,default\_hostgroup) values('qjweb\_read','654321',20);

Query OK, 1 row affected (0.00 sec)

<proxysql>load mysql users to runtime;

Query OK, 0 rows affected (0.00 sec)

<proxysql>save mysql users to disk;

Query OK, 0 rows affected (0.21 sec)

<proxysql>select \* from mysql\_users;

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

| username | password | active | use\_ssl | default\_hostgroup | default\_schema | schema\_locked | transaction\_persistent | fast\_forward | backend | frontend | max\_connections | comment |

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

| qjweb | 123456 | 1 | 0 | 10 | NULL | 0 | 1 | 0 | 1 | 1 | 10000 | |

| qjweb\_read | 654321 | 1 | 0 | 20 | NULL | 0 | 1 | 0 | 1 | 1 | 10000 | |

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

2 rows in set (0.00 sec)

**-- 测试proxySQL中间件读写路由情况**

[root@qj-mysql-node-219 ~]# mysql -uqjweb -p123456 -P6033 -h192.168.2.219 -e "select @@server\_id"

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

| @@server\_id |

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

| 221 |

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

[root@qj-mysql-node-219 ~]# mysql -uqjweb -p123456 -P6033 -h192.168.2.219 -e "create database if not exists orders"

[root@qj-mysql-node-219 ~]# mysql -uqjweb\_read -p654321 -P6033 -h192.168.2.219 -e "select @@server\_id"

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

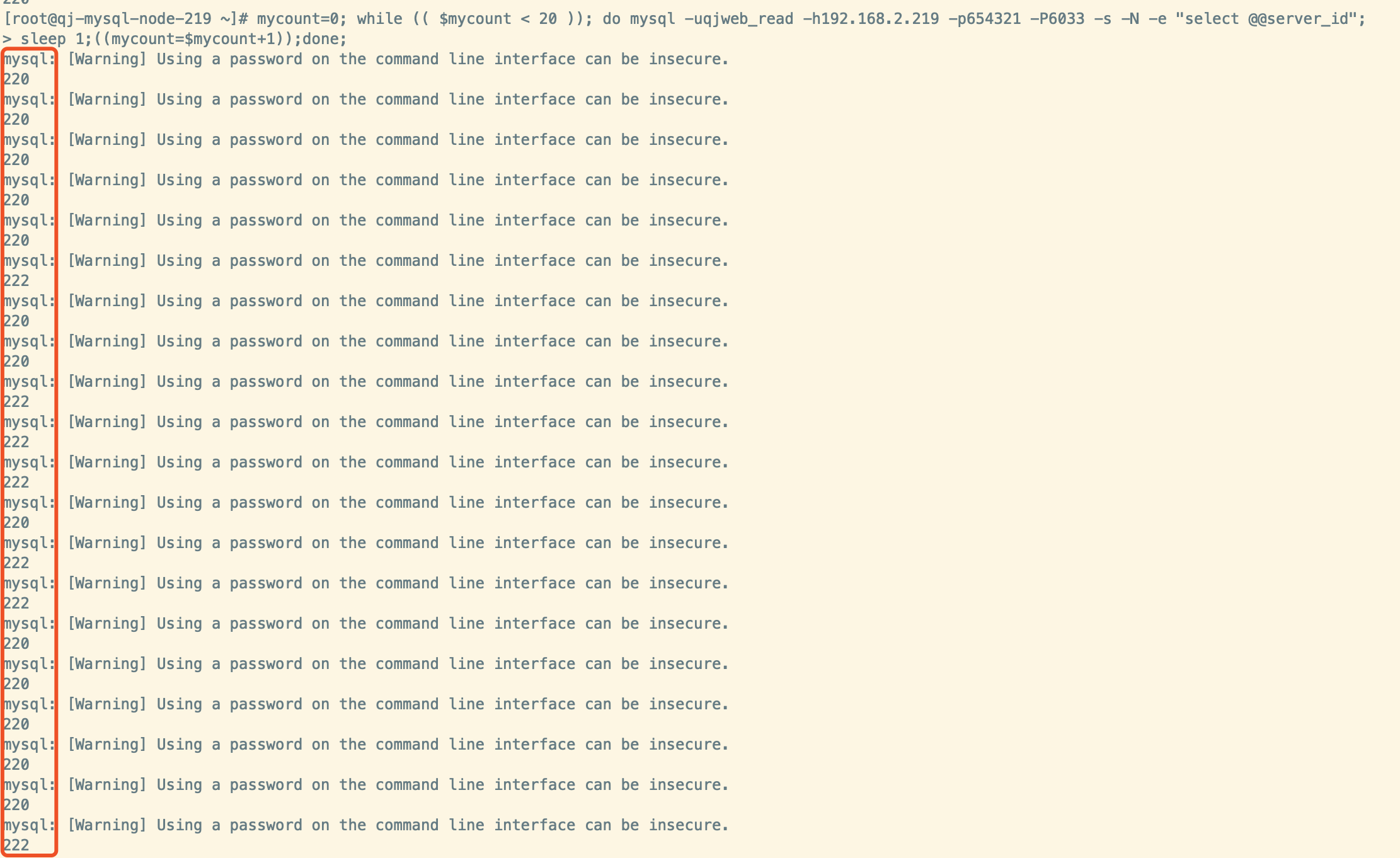
| @@server\_id |

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

| 220 |

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

mycount=0; while (( $mycount < 20 )); do mysql -uqjweb\_read -h192.168.2.219 -p654321 -P6033 -s -N -e "select @@server\_id"; sleep 1;((mycount=$mycount+1));done;



**-- 此时通过配置不同的读写用户可实现读写分离。但实际应用连接驱动中需要添加不同用户（组），重写类代码。**

**-- 上面是通过基于用户的机制实现读写分离。思考能否通过SQL代码，即基于语句的机制实现读写分离？**

<proxysql>insert into mysql\_query\_rules(rule\_id,active,username,match\_digest,destination\_hostgroup,apply) VALUES (1,1,'qjweb','^SELECT.\* FOR UPDATE$',10,1), (2,1,'qjweb','^SELECT',20,1);

Query OK, 2 rows affected (0.00 sec)

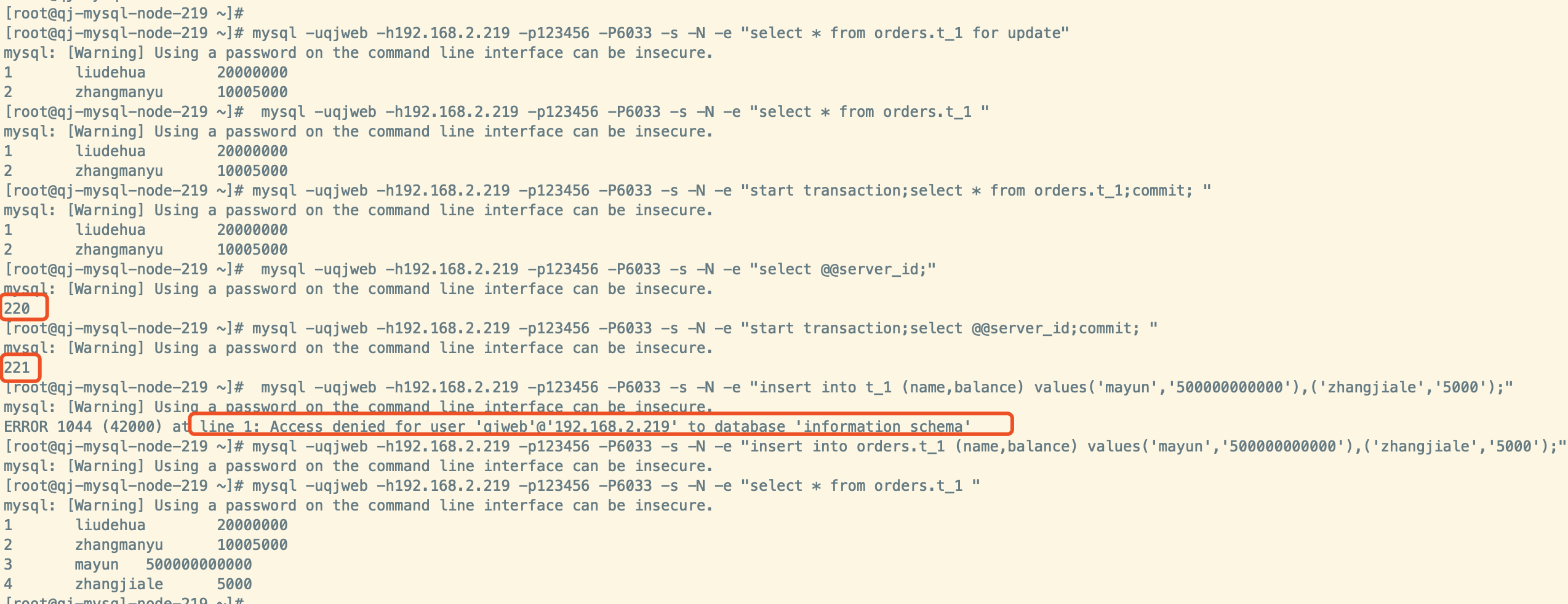
<proxysql> load mysql query rules to runtime;

Query OK, 0 rows affected (0.00 sec)

<proxysql>save mysql query rules to disk;

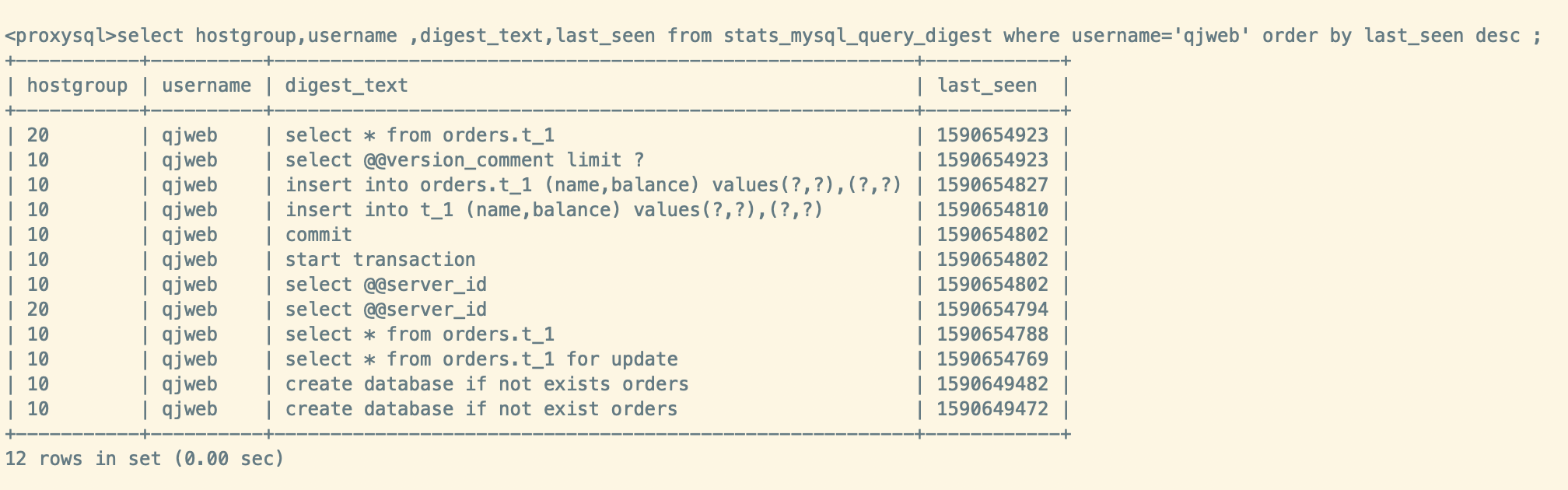
Query OK, 0 rows affected (0.35 sec)

**需要注意：select ... for update规则的rule\_id必须要小于普通的select规则的rule\_id，因为ProxySQL是根据rule\_id的顺序进行规则匹配的.**



-- **查看路由具体情况**

<proxysql>select hostgroup,username ,digest\_text,last\_seen from stats\_mysql\_query\_digest where username='qjweb' order by last\_seen desc ;



**-- 针对截图报错处理**





-- 负载均衡测试

-- 脚本如下

[root@qj-mysql-node-219 ~]# cat proxysql\_lb.sh

#!/bin/bash

case "$1" in

-h|--help|?)

echo "Usage: 1st arg:connects"

exit 0

;;

esac

if [ ! -n "$1" ]; then

echo "pls input first paramater"

echo "example:sh ./script.sh 100"

exit

fi

user=qjweb

passwd=123456

file="/tmp/proxysql\_lb.txt"

if [ -f "$file" ]; then

rm -f "$file"

fi

i=0

while(($i<$1))

do

/usr/bin/mysql -u$user -p$passwd -P6033 -h192.168.2.219 -e "select @@hostname;" >> $file 2>/dev/null

let "i++"

sleep 0.1

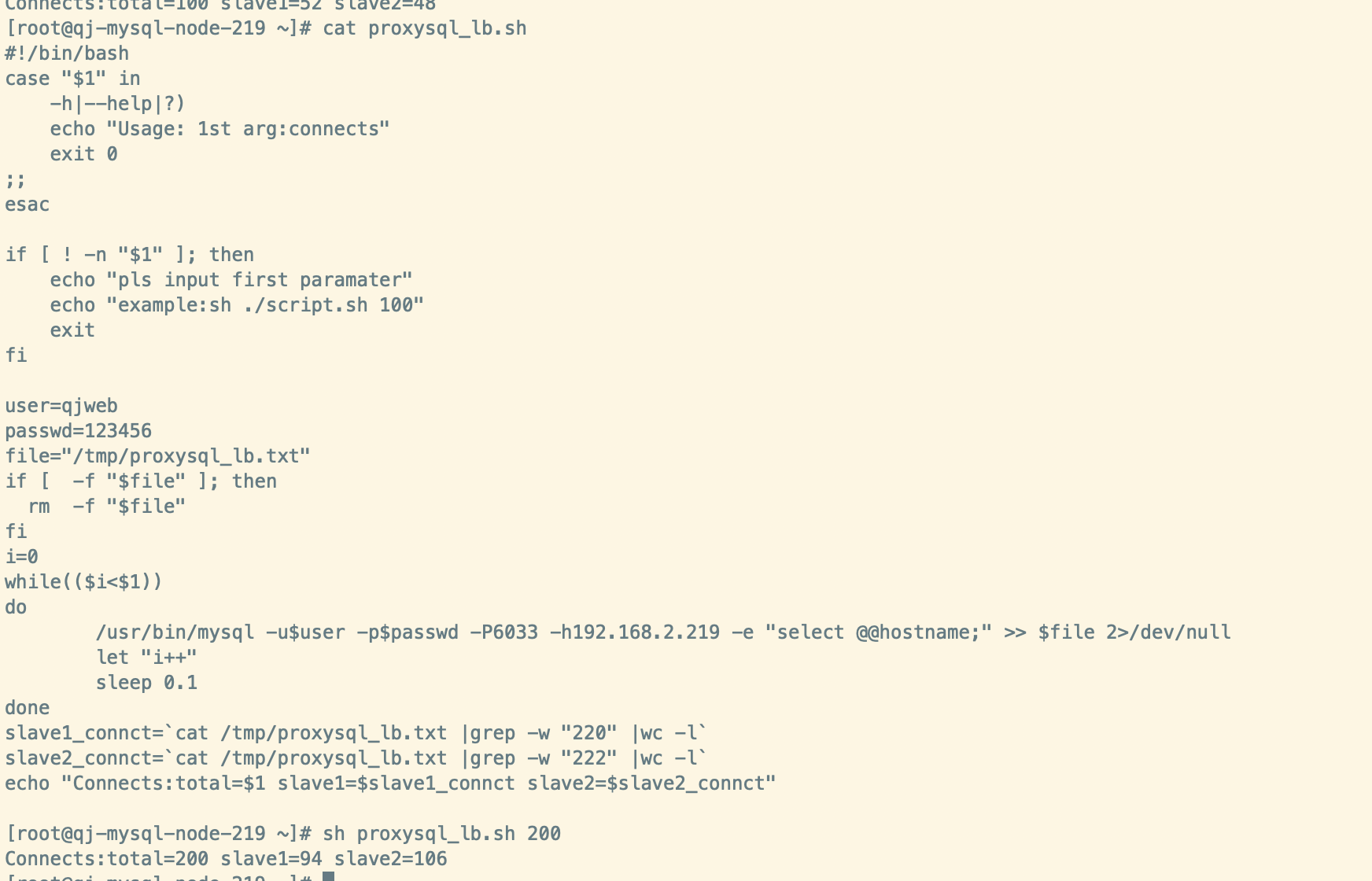
done

slave1\_connct=`cat /tmp/proxysql\_lb.txt |grep -w "220" |wc -l`

slave2\_connct=`cat /tmp/proxysql\_lb.txt |grep -w "222" |wc -l`

echo "Connects:total=$1 slave1=$slave1\_connct slave2=$slave2\_connct"

-- 运行测试结果：



**-- Web访问**

<proxysql>select \* from global\_variables where variable\_name='admin-web\_enabled';

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

| variable\_name | variable\_value |

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

| admin-web\_enabled | false |

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

1 row in set (0.00 sec)

<proxysql>update global\_variables set variable\_value='true' where variable\_name='admin-web\_enabled';

Query OK, 1 row affected (0.01 sec)

<proxysql>LOAD ADMIN VARIABLES TO RUNTIME;

Query OK, 0 rows affected (0.04 sec)

<proxysql>SAVE ADMIN VARIABLES TO DISK;

Query OK, 35 rows affected (0.11 sec)

<proxysql>select \* from global\_variables where variable\_name LIKE 'admin-web%' or variable\_name LIKE 'admin-stats%';

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

| variable\_name | variable\_value |

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

| admin-stats\_credentials | stats:stats |

| admin-stats\_mysql\_connections | 60 |

| admin-stats\_mysql\_connection\_pool | 60 |

| admin-stats\_mysql\_query\_cache | 60 |

| admin-stats\_mysql\_query\_digest\_to\_disk | 0 |

| admin-stats\_system\_cpu | 60 |

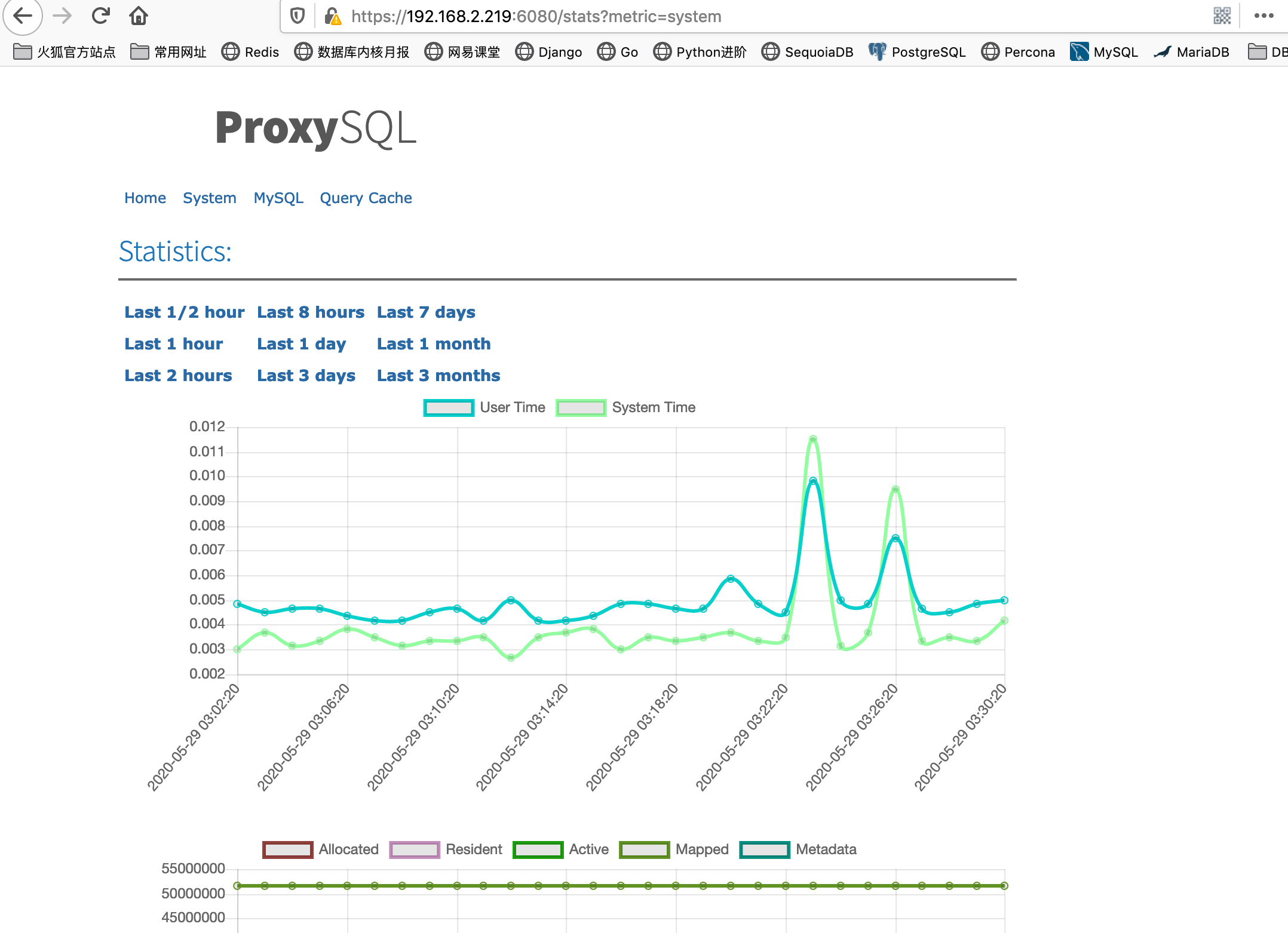
| admin-stats\_system\_memory | 60 |

| admin-web\_enabled | true |

| admin-web\_port | 6080 |

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

9 rows in set (0.00 sec)



## 2.6 MGR + consul

### 2.6.1简介

### 2.6.2 功能

### 2.6.3 架构图

### 2.6.4 实验室

## 2.7 Innodb cluster + MySQL router + MySQL shell

### 2.7.1简介

### 2.7.2 功能

### 2.7.3 架构图

### 2.7.4 实验室