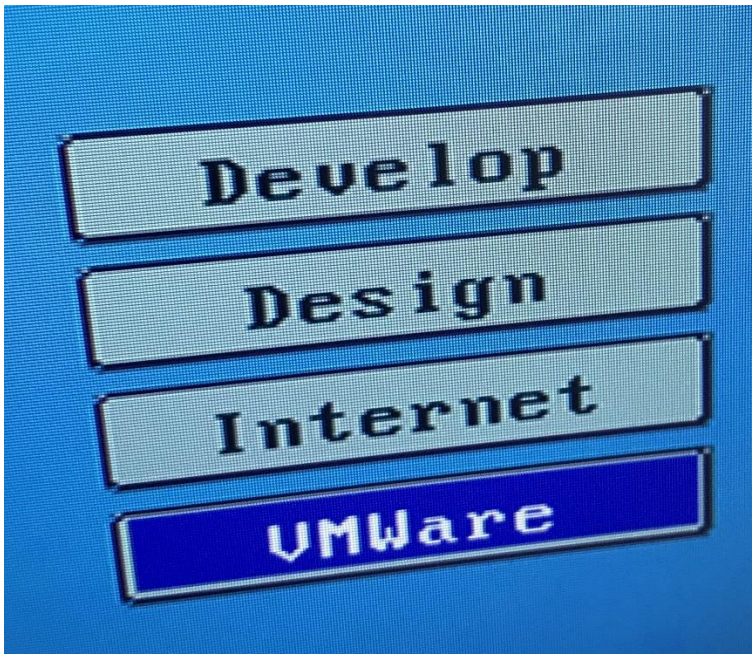
MySQL8 InnoDB Cluster 数据库安装教学指导书

【编写说明：聂耿青于 2022 年编写第一版（用于线上教学），2023 年修订为第二版，主要修订内容：改用计算中心 VMware 分区的虚拟机环境，Mysql 的版本更新为 8.0.33】

测试环境可以用 Docker 容器实例或 Sandbox 实例来搭建，但生成环境通常是多台物理机或虚拟机来运行 MySQL 数据库实例。

计算中心机房实验环境操作步骤（2023 年新增）：

1. 开机后要选择 VMware 分区：



2. 直接从校内 FTP 服务器下载有关安装文件：用 fileZilla 或文件资源管理器 ftp://202.204.120.72

在 D 盘或 E 盘创建目录 D:\ngq、D:\ngq\sharedsk

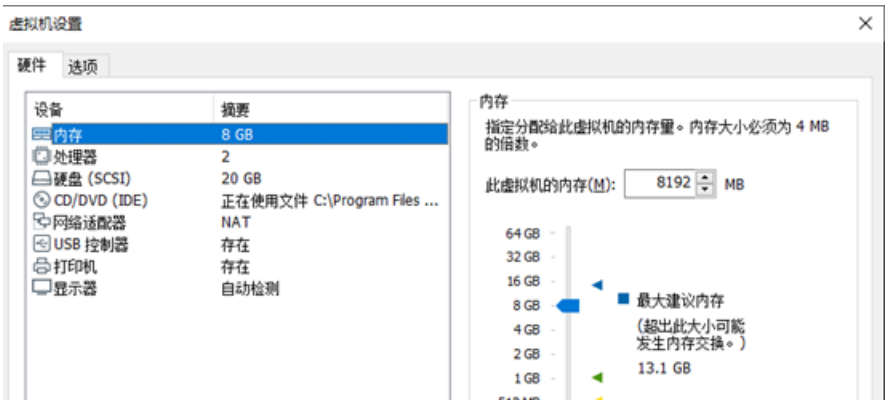
下载 CentOS7_DBPreinstall.zip 到 D:\ngq，并解压到当前目录

Xshell-7*.exe 到 D:\ngq

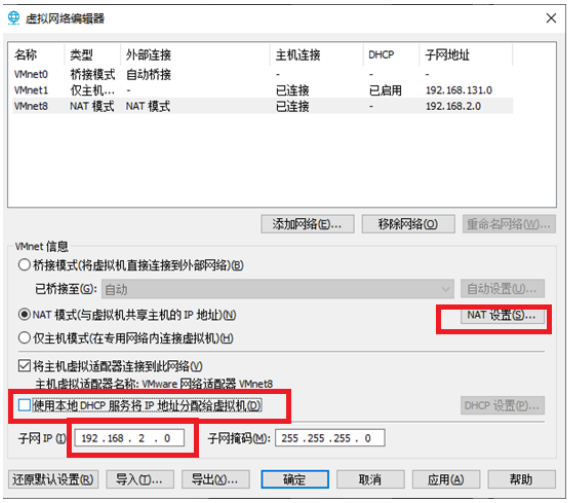
mysql*-8.0.33-1.el7.x86_64* 到 D:\ngq\sharedsk

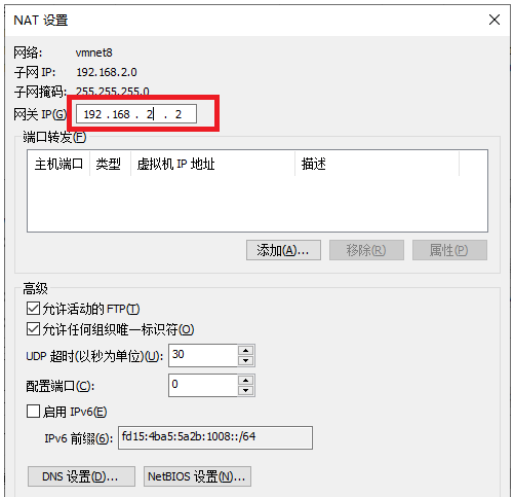
3. 用 VMware Worstation 打开 D:\ngq\CentOS7VM

在设置中修改内存到 6-8GB（如果物理内存大于 16GB）

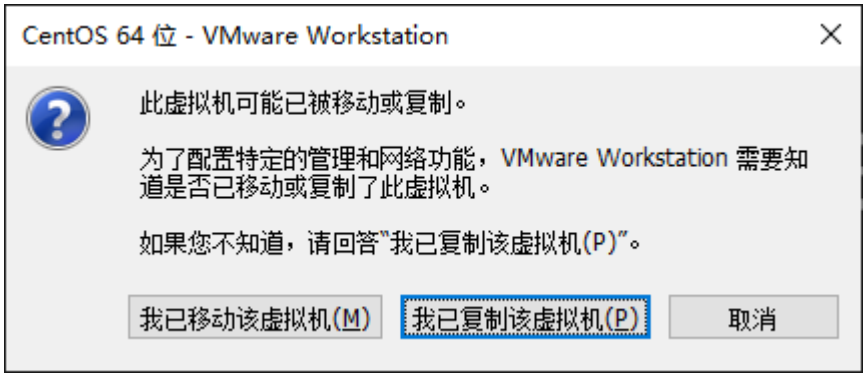


在编辑→虚拟网络编辑器



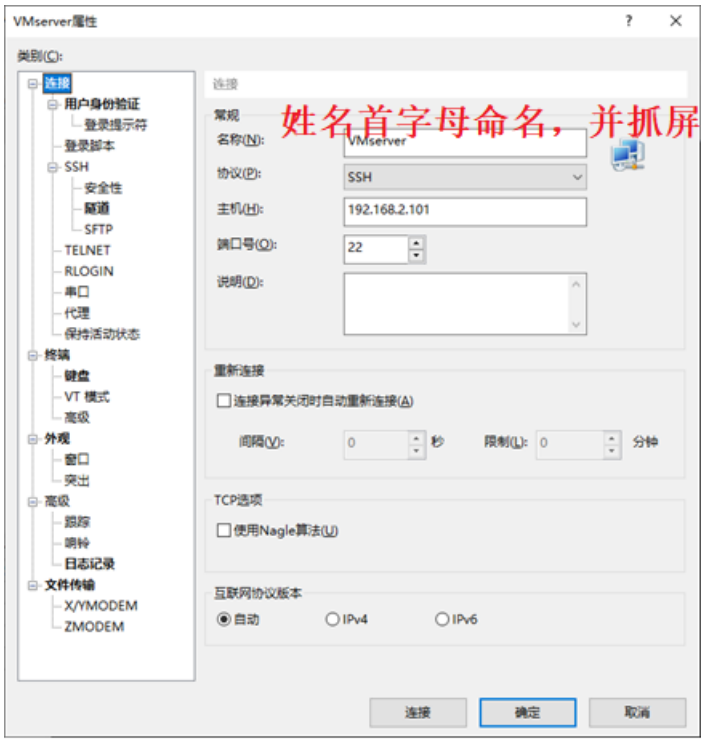


启动虚拟机：

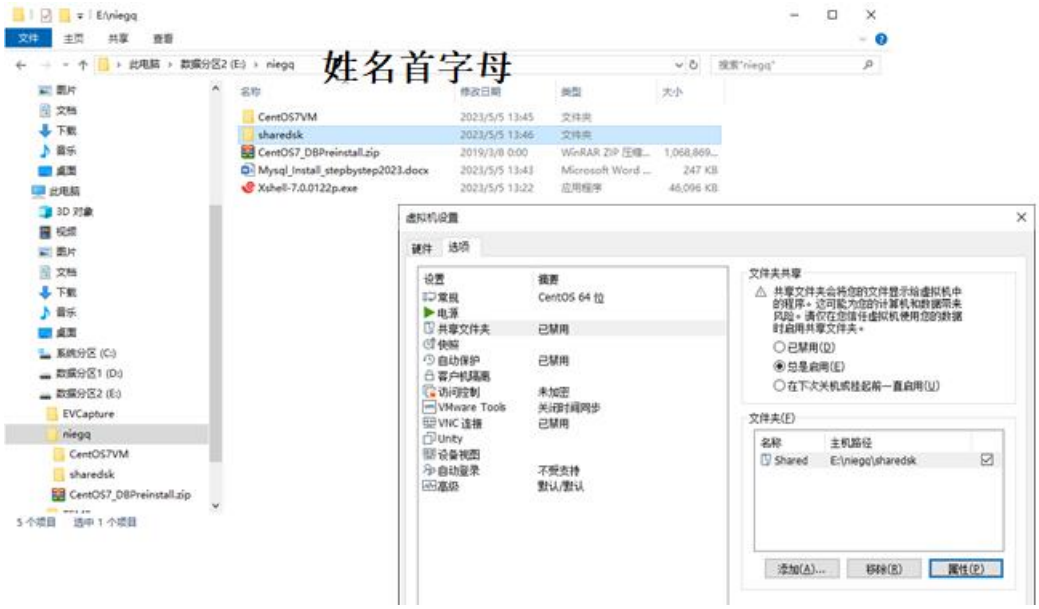


#ping www.baidu.com

4. 在 Windows 环境下安装 Xshell



5. 宿主机共享文件夹设置



#vmware-hgfsclient

#vmhgfs-fuse .host:/ /mnt/hgfs

#ls /mnt/hgfs

6. 通过 rpm 安装 MySQL8

#cd /mnt/hgfs/Shared

#tar xvf mysql-8.0.33-1.el7.x86_64.rpm-bundle.tar

#yum install mysql-community-{server,client,common,libs,icu}-*

#vi /etc/my.cnf

```
bind_address = 0.0.0.0
#port=3306
default-time-zone='+8:00'
```

```
#systemctl start mysqld
#systemctl status mysqld
```

7. 防火墙

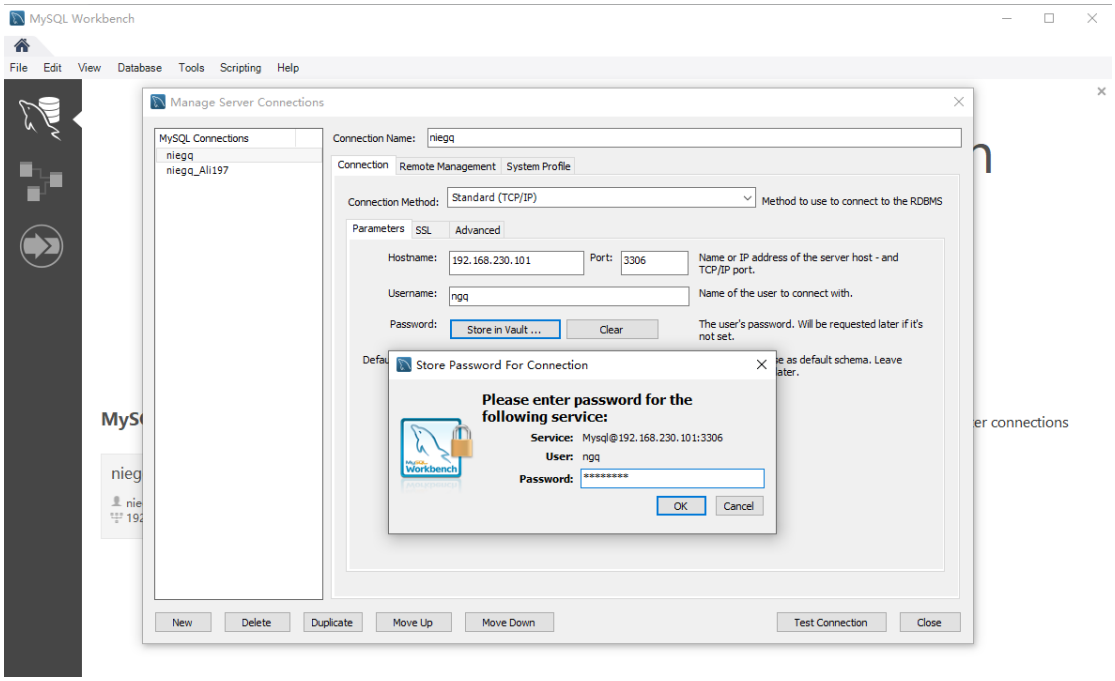
```
firewall-cmd --zone=public --permanent --add-port=3306/tcp
firewall-cmd --reload
```

8. 客户端连接 MySql Server

```
#grep password /var/log/mysqld.log
#mysql -uroot -p
ALTER USER 'root'@'localhost' IDENTIFIED BY ' abc123!Test';

create user ngq@"%" identified by 'ngqpwd';
GRANT ALL privileges ON *.* TO "ngq"@"%";
FLUSH PRIVILEGES;
use mysql
select host, user from user;
```

MySQL Workbench 客户端



在客户端写点 SQL 看看：

create database ngqdb;

use ngqdb;

show databases;

show tables;

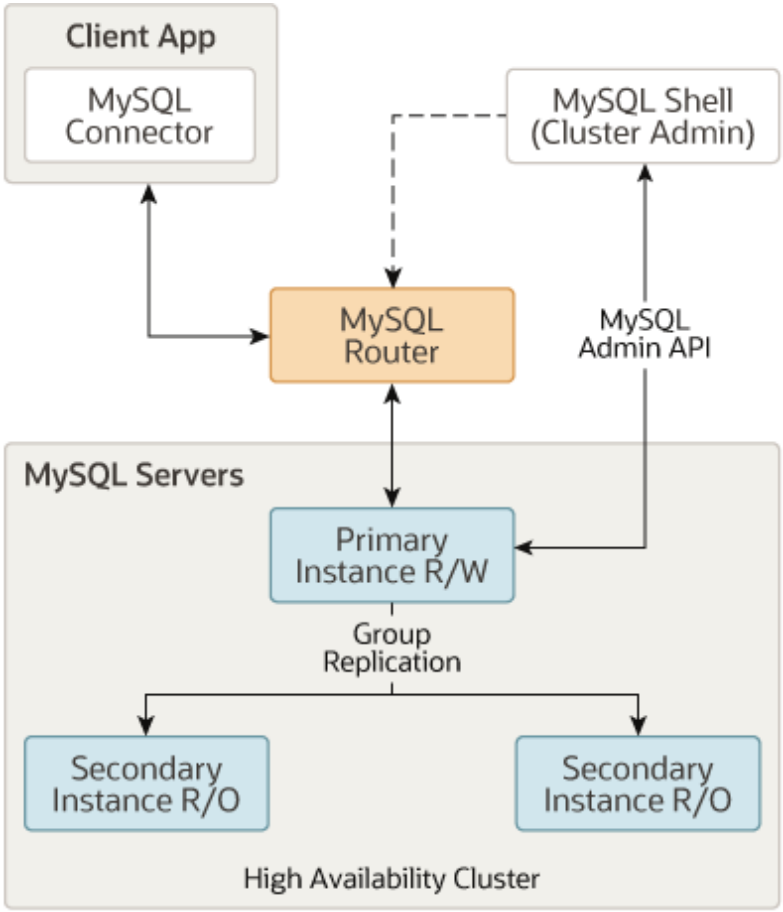
create table nietest (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, name varchar(100), create_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP);

insert into…

MySQL 单机服务器安装完毕!

下面为 MySQL 集群的安装实验:

MySQL InnoDB Cluster 集群服务器拓扑结构如下图:



9. 安装 shell 和 router

#yum localinstall mysql-shell-8.0.33-1.el7.x86_64.rpm

yum localinstall mysql-router-community-8.0.33-1.el7.x86_64.rpm

10. 创建三个沙漏实例并构建一个实验集群

```
mysqlsh -uroot -p
dba.deploySandboxInstance(3307,{password:'ngqpwd'});
dba.deploySandboxInstance(3308,{password:'ngqpwd'});
dba.deploySandboxInstance(3309,{password:'ngqpwd'});

firewall-cmd --zone=public --permanent --add-port=3307/tcp
firewall-cmd --zone=public --permanent --add-port=3308/tcp
firewall-cmd --zone=public --permanent --add-port=3309/tcp
firewall-cmd --reload
```

创建 InnoDB Cluster 并添加集群实例

```
mysqlsh -uroot -p -h127.0.0.1 -P3307
cluster=dba.createCluster("ngqcluster");
cluster.addInstance("root@127.0.0.1:3308",{password:'ngqpwd'});
cluster.addInstance("root@127.0.0.1:3309",{password:'ngqpwd'});
mysqlsh -uroot -p -h127.0.0.1 -P3309
cluster = dba.getCluster();
cluster.status();
#显示 status:ONLINE is OK
```

11. 在当前目录生成路由配置文件

```
mysqlrouter --bootstrap localhost:3307 -d ngqrouter --user=root
firewall-cmd --zone=public --permanent --add-port=6446/tcp
firewall-cmd --zone=public --permanent --add-port=6447/tcp
firewall-cmd --reload
mysqlrouter -c /root/ngqrouter/mysqlrouter.conf &
```

12. Cluster 路由测试及故障转移(failover)测试：

```
mysqlsh -uroot -p -h127.0.0.1 -P6446
\sql
create user ngq@"%" identified by 'ngqpwd';
GRANT ALL privileges ON *.* TO ngq@"%" WITH GRANT OPTION;
FLUSH PRIVILEGES;
mysql -ungq -pngqpwd -h127.0.0.1 -P6446
workbench:
create database ngqdb;
use ngqdb;
create table ngq_tab (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY
, name varchar(100) not null
, type varchar(20)
, create_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
```

```
, last_update timestamp default current_timestamp on update current_timestamp);
insert into ngq_tab (name) values (concat('test:port',@@port));
select * from ngq_tab;

mysql -ungq -pngqpwd -h127.0.0.1 -P6447
SELECT @@PORT;

mysql -ungq -pngqpwd -h127.0.0.1 -P6447 -se"select @@port\G"

pgrep mysqld -fla
kill -9
SELECT @@PORT;  --连续多次，几秒后恢复

恢复就主实例
mysqlsh -uroot -p -hlocalhost -P6446
 dba.startSandboxInstance(3307);
cluster = dba.getCluster();
 cluster.status()
```

13. 虚拟机重启（或三个沙漏实例全部关闭，相当于集群所有节点全部断电），要做如下处理才能重启集群：
    (1) 先启动所有沙漏节点的数据库实例：
    mysqlsh –uroot -p
    dba.startSandboxInstance(3307)
    dba.startSandboxInstance(3308)
    dba.startSandboxInstance(3309)
    (2) 连接主节点，重启集群：
    mysqlsh -uroot -p -h127.0.0.1 -P3307
    cluster = dba.rebootClusterFromCompleteOutage()

14. 其它实验（比如删除集群等操作）可以参见 2022 年的内容：

以下为 2022 年的内容（适合线上教学，在笔记本电脑操作）：

预备、服务器安装环境准备

首先确认磁盘剩余空间，如果少于 3GB，就建议先按如下指导书作磁盘扩容：

https://www.niepub.com/static/docs/exadmin/CentOS7_DiskExtend_Guide.pdf

一、采用MySQL数据库服务器的Sandbox沙漏实例来搭建集群(InnoDB Cluster)

(1) 安装数据库服务器：install mysql8.0.28 & mysqlsh

```
mkdir & cd Software
wget https://mirrors.ustc.edu.cn/mysql-ftp/Downloads/MySQL-8.0/mysql-8.0.28-1.el7.x86_64.rpm-bundle.tar --no-check-certificate
tar xvf mysql-8.0.28-1.el7.x86_64.rpm-bundle.tar
yum install mysql-community-{server,client,common,libs,icu-data-files}-*
systemctl start mysqld
grep password /var/log/mysqld.log
mysql -uroot -p
ALTER USER 'root'@'localhost' IDENTIFIED BY 'ngqpwd';
update mysql.user set host = '%' where user = 'root';
create user ngq@"%" identified by 'ngqpwd';
GRANT ALL privileges ON *.* TO ngq@"%" WITH GRANT OPTION;
FLUSH PRIVILEGES;
wget https://mirrors.ustc.edu.cn/mysql-ftp/Downloads/MySQL-Shell/mysql-shell-8.0.28-1.el7.x86_64.rpm --no-check-certificate
yum localinstall mysql-shell-8.0.28-1.el7.x86_64.rpm
wget https://mirrors.ustc.edu.cn/mysql-ftp/Downloads/MySQL-Router/mysql-router-community-8.0.28-1.el7.x86_64.rpm --no-check-certificate
yum localinstall mysql-router-community-8.0.28-1.el7.x86_64.rpm

firewall-cmd --zone=public --permanent --add-port=3306/tcp
firewall-cmd --zone=public --permanent --add-port=3307/tcp
firewall-cmd --zone=public --permanent --add-port=3308/tcp
firewall-cmd --zone=public --permanent --add-port=3309/tcp
firewall-cmd --reload
```

(2) 配置集群：Config InnoDB Cluster

创建三个沙漏实例(SandboxInstance)

```
mysqlsh -uroot -p
dba.deploySandboxInstance(3307,{password:'ngqpwd'});
dba.deploySandboxInstance(3308,{password:'ngqpwd'});
dba.deploySandboxInstance(3309,{password:'ngqpwd'});
```

创建 InnoDB Cluster 并添加集群实例

```
mysqlsh -uroot -p -h127.0.0.1 -P3307
cluster=dba.createCluster("ngqcluster");
#cluster=dba.getCluster("ngqcluster");
cluster.addInstance("root@127.0.0.1:3308",{password:'ngqpwd'});
```

```
cluster.addInstance("root@127.0.0.1:3309",{password:'ngqpwd'});
print (cluster.status());
mysqlsh -uroot -p -h127.0.0.1 -P3309
cluster = dba.getCluster();
cluster.status();
#显示 status:ONLINE is OK
```

（3）启动路由：Start MySQL Router

```
mysqlrouter --bootstrap localhost:3307 -d ngqrouter --user=root
#mysqlrouter --bootstrap localhost:3307 -d ngqrouter --user=root --force
firewall-cmd --zone=public --permanent --add-port=6446/tcp
firewall-cmd --zone=public --permanent --add-port=6447/tcp
firewall-cmd --reload
mysqlrouter -c /root/Software/ngqrouter/mysqlrouter.conf &
or ./ngqrouter/start.sh ./ngqrouter/stop.sh
lsof -i :6446
mysql -uroot -p -P6446 -h127.0.0.1
select @@group_replication_local_address\G
SELECT @@PORT;
create database ngqdb;
create table ngq_tab (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY
, name varchar(100) not null
, type    varchar(20)
, create_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
, last_update timestamp default current_timestamp on update current_timestamp);
insert into ngq_tab (name) values (concat('test:port',@@port));
select * from ngq_tab;
select user();
mysql -uroot -p -P6447 -h127.0.0.1
SELECT @@PORT;
mysql -uroot -p -h127.0.0.1 -P6447 -se"select @@port\G"
连续连接 6447 三次看看
mysql -uroot -p -h127.0.0.1 -P3308 -se"select @@port\G select count(*) from ngq_tab;"
```

（4）．故障转移测试

```
mysqlsh -uroot -p -h127.0.0.1 -P6446
cluster = dba.getCluster();
cluster.status()
\sql
select * from performance_schema.replication_group_members\G
\js
#pgrep mysqld -fla
#kill -9    primaryInstancePID
\sql
```

select * from performance_schema.replication_group_members\G
\js
cluster = dba.getCluster();
cluster.status()
恢复就主实例
mysqlsh -uroot -p -h127.0.0.1 -P6446
dba.startSandboxInstance(3307);
cluster = dba.getCluster();
cluster.status()

（5）. 多活与单活的转换测试

cluster.switchToMultiPrimaryMode()　多主模式:所有实例都成为主要实例。
mysql -uroot -p -h127.0.0.1 -P3309|3307|3308
cluster.switchToSinglePrimaryMode('root@127.0.0.1:3309')切换到单主模式
重启 router：./stop.sh ./start.sh
mysql -uroot -p -h127.0.0.1 -P6446|6447

（6）集群重启：所有数据库实例全部停止（或停电）后重启整个集群服务器

模拟所有节点全部断电或 shutdown
dba.stopSandboxInstance(3307)
【或者
#mysql -uroot -p -h127.0.0.1 -P3309
Mysql>**shutdown**;
】
方法一：
先启动所有节点实例 mysqlsh
dba.startSandboxInstance(3307)

mysqlsh -uroot -p -h127.0.0.1 -P3307
cluster = dba.rebootClusterFromCompleteOutage()

方法二：
Mysqlsh 先启动主节点，并运行：
#mysql -uroot -p -h127.0.0.1 -P3307
SET GLOBAL group_replication_bootstrap_group=ON;
start group_replication;
SET GLOBAL group_replication_bootstrap_group=OFF;
SELECT * FROM performance_schema.replication_group_members;
然后再依次启动所有其它节点(GTID 由大到小):

(7) （可选）删除集群及路由
mysqlsh -uroot -proot -h127.0.0.1 -P3309

```
dba.stopSandboxInstance(3309,{password:'ngqpwd'});
dba.stopSandboxInstance(3308,{password:' ngqpwd '});
dba.stopSandboxInstance(3307,{password:' ngqpwd '});
dba.deleteSandboxInstance(3307);
dba.deleteSandboxInstance(3308);
dba.deleteSandboxInstance(3309);
 ./ngqrouter/stop.sh
rm -rf ngqrouter
```

(8). (可选)更新身份验证方式：caching_sha2_password-->mysql_native_password

更新配置并添加以下行:

```
default_authentication_plugin=mysql_native_password
```

到所有实例（配置位于 ~$HOME/mysql-sandboxes/$PORT/my.cnf。确保更新所有 3 个节点

并 mysqlsh 中重启所有 MySQL 实例:

```
mysqlsh> dba.stopSandboxInstance(3307);
mysqlsh> dba.startSandboxInstance(3307);
mysqlsh> \c 'root'@127.0.0.1:3307
```

更新我们的“root”用户以使用旧插件

```
mysql> ALTER USER 'root'@'%' IDENTIFIED WITH mysql_native_password BY 'ngqpwd';
mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'ngqpwd';
 show global variables like 'default_authentication_plugin';
```

路由日志及端口等：

```
#ngqrouter/log/mysqlrouter.log
$ mysqlrouter -c /root/Software/ngqrouter/mysqlrouter.conf
InnoDB Cluster 'ngqcluster' can be reached by connecting to:
## MySQL Classic protocol
- Read/Write Connections: localhost:6446
- Read/Only Connections:    localhost:6447
## MySQL X protocol
- Read/Write Connections: localhost:6448
- Read/Only Connections:    localhost:6449
```

二、采用 MySQL 数据库服务器的 Docker 容器实例来搭建集群(InnoDB Cluster)

--先需要安装配置好 Docker 运行环境，然后下载或更新 MySQL 最新镜像

```
docker search mysql
#docker pull mysql/mysql-server
#docker pull mysql/mysql-router
mysql/mysql-server          8.0.28    434c35b82b08    4 weeks ago    417MB
mysql/mysql-router          8.0.28    acccea81feeb    4 weeks ago    241MB
```

```
docker images
#docker rmi hello-world

more /etc/docker/daemon.json
{
    "dns": ["114.114.114.114", "8.8.8.8"],
    "registry-mirrors": ["https://ldcfwmse.mirror.aliyuncs.com"]
}
#systemctl restart docker
```

-----docker Single-Node MySQL DB 单节点数据库服务器的安装配置----------

```
#docker rm mysqlnie
#docker run --name=mysqlnie -p 4432:3306 -d mysql/mysql-server
#docker ps  -a
#重启 Server 后，需要 docker start mysqlnie 启动容器
docker exec -it mysqlnie mysql -uroot -p
semanage port -a -t mysqld_port_t -p tcp 4432
firewall-cmd --zone=public --permanent --add-port=4432/tcp
firewall-cmd --reload

ip addr |grep docker
docker inspect mysqlnie |grep IP
docker cp mysqlnie:/etc/my.cnf .
```

----------MySQL InnoDB Cluster 集群环境的搭建----------

--基于同一网络创建并启动三个以上的 mysql 容器

```
docker network create ngqclusternet
docker run -d --name=ngqsvr1 --hostname=ngqsvr1 --net=ngqclusternet -e
MYSQL_ROOT_PASSWORD=rootpwd mysql/mysql-server
for N in 2 3
  do docker run -d --name=ngqsvr$N --hostname=ngqsvr$N --net=ngqclusternet \
      -e MYSQL_ROOT_PASSWORD=rootpwd mysql/mysql-server
done
docker ps -a
```

--创建用户并授权并初始化 binlog

```
for N in 1 2 3
do docker exec -it ngqsvr$N mysql -uroot -prootpwd \
  -e "CREATE USER 'ngq'@'%' IDENTIFIED BY 'ngqpwd';" \
  -e "GRANT ALL privileges ON *.* TO 'ngq'@'%' with grant option;" \
  -e "reset master;"
done
for N in 1 2
do docker exec -it ngqsvr$N mysql -ungq -pngqpwd \
  -e "SHOW VARIABLES like 'hostname';" \
```

```
  -e "SELECT user FROM mysql.user where user = 'ngq';"
Done
```

--配置服务器

```
docker exec -it ngqsvr1 mysqlsh -uroot -prootpwd -S/var/run/mysqld/mysqlx.sock
dba.checkInstanceConfiguration("ngq@ngqsvr1:3306")
dba.configureInstance("ngq@ngqsvr1:3306")
docker start ngqsvr1
dba.configureInstance("ngq@ngqsvr2:3306")
dba.configureInstance("ngq@ngqsvr3:3306")
docker restart ngqsvr1 ngqsvr2 ngqsvr3
```

--创建集群(InnoDB Cluster)

```
docker exec -it ngqsvr1 mysqlsh -uroot -prootpwd -S/var/run/mysqld/mysqlx.sock
\c ngq@ngqsvr1:3306
cluster = dba.createCluster("ngqcluster")
cluster.addInstance("ngq@ngqsvr2:3306")
cluster.addInstance("ngq@ngqsvr3:3306",{password:'ngqpwd'})
--restart timeout
docker start ngqsvr2
cluster.rescan()
cluster.describe()
```

--路由启动器

```
docker run -d --name ngqrouter --net=ngqclusternet \
    -e MYSQL_HOST=ngqsvr1 \
    -e MYSQL_PORT=3306 \
    -e MYSQL_USER=ngq \
    -e MYSQL_PASSWORD=ngqpwd \
    -e MYSQL_INNODB_CLUSTER_MEMBERS=3 \
    mysql/mysql-router
docker logs ngqrouter
## MySQL Classic protocol
- Read/Write Connections: localhost:6446
- Read/Only Connections:    localhost:6447

docker ps
mysql/mysql-router          "/run.sh    mysqlrouter"                6446-6449/tcp,    8443/tcp
mysql-router
mysql/mysql-server              "/entrypoint.sh     mysq…"              3306/tcp,     33060/tcp
ngqsvr3
mysql/mysql-server              "/entrypoint.sh     mysq…"              3306/tcp,     33060/tcp
ngqsvr2
mysql/mysql-server              "/entrypoint.sh     mysq…"              3306/tcp,     33060/tcp
```

ngqsvr1

--再配置一 mysql 作为路由连接用服务器

```
docker run -d --name=ngqclient --hostname=ngqclient --net=ngqclusternet \
    -e MYSQL_ROOT_PASSWORD=rootpwd mysql/mysql-server
docker exec -it ngqclient mysql -h ngqrouter -P 6446 -ungq -pngqpwd \
  -e "create database ngqdb; use ngqdb; CREATE table ngq_tab (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, name varchar(100) not null) ENGINE=InnoDB; show tables;"
docker exec -it ngqclient mysql -h ngqrouter -P 6446 -ungq -pngqpwd  \
 -e "insert into ngqdb.ngq_tab (name) values (concat('hostname:',@@hostname));" -e "select * from ngqdb.ngq_tab;"
for N in 1 2 3
do docker exec -it ngqsvr$N mysql -ungq -pngqpwd \
  -e "SHOW VARIABLES like 'hostname';" \
  -e "SELECT * FROM ngqdb.ngq_tab;"
done
```

--故障切换

```
docker stop ngqsvr1
docker exec -it ngqclient mysqlsh -h ngqrouter -P 6447 -ungq -pngqpwd
cluster = dba.getCluster()
cluster.status()
docker start ngqsvr1
"status": "(MISSING)"-->"RECOVERING"-->"RECOVERING"
#cluster.rescan()
#cluster.addInstance("ngq@ngqsvr1:3306",{password:'ngqpwd'})
```

--多活与单活的转换测试

```
docker exec -it ngqclient mysql -h ngqrouter -P 6446 -ungq -pngqpwd    -e"select * from performance_schema.replication_group_members;"
docker exec -it ngqclient mysqlsh -h ngqrouter -P 6447 -ungq -pngqpwd
cluster = dba.getCluster()
cluster.switchToMultiPrimaryMode()
cluster.status()
docker stop ngqrouter
docker start ngqrouter
docker logs ngqrouter
docker exec -it ngqclient mysql -h ngqrouter -P 6446 -ungq -pngqpwd -e "insert into ngqdb.ngq_tab (name) values (concat('hostname:',@@hostname));" -e "select * from ngqdb.ngq_tab;"
docker exec -it ngqsvr3 mysql -ungq -pngqpwd    -e "insert into ngqdb.ngq_tab (name) values (concat('hostname:',@@hostname));" -e "select * from ngqdb.ngq_tab;"
cluster.switchToSinglePrimaryMode('ngq@ngqsvr1:3306')
```

docker exec -it ngqclient mysql -h ngqrouter -P 6447 -ungq -pngqpwd -e "select * from ngqdb.ngq_tab; select @@hostname;"

--删除集群和测试用服务器容器

docker stop ngqsvr1 ngqsvr2 ngqsvr3 ngqrouter ngqclient
docker rm ngqsvr1 ngqsvr2 ngqsvr3 ngqrouter ngqclient
或者
docker rm -f ngqsvr1 ngqsvr2 ngqsvr3 ngqrouter ngqclient

小提示：

当我们使用 docker 的默认网络模式时，第一个安装并启动的服务 ip 为`172.17.0.2`， 第二个安装并启动的服务 ip 为`172.17.0.3`
docker exec -it ngqsvr1 bash
cat /etc/hosts
docker inspect ngqsvr1
yum --disablerepo=mysql80-server-minimal yum install net-tools
yum-config-manager --disable mysql80-server-minimal
yum install net-tools
yum-config-manager --enable mysql80-server-minimal

三、InnoDB Cluster 维护常用操作和命令

mysqlsh>dba.help();
dba.createCluster()
dba.createCluster('testCluster', {replicationAllowedHost:'192.0.2.0/24'})
replicationAllowedHost 选项意味着自动创建的所有帐户只能从允许的主机连接（8.0.28）
cluster.checkInstanceState('icadmin@ic-4:3306')可以将状态为 OK 的实例添加到集群中
cluster.addInstance('icadmin@ic-2:3306')
group_replication_local_address= "[2001:db8:85a3:8d3:1319:8a2e:370:7348]:33061"
Cluster.switchToMultiPrimaryMode()，这会将集群切换到多主模式。所有实例都成为主要实例。
Cluster.switchToSinglePrimaryMode([instance])，这会将集群切换到单主模式
cluster.removeInstance('root@localhost:3310') 从集群中删除实例
Cluster.dissolve() Cluster.dissolve({force: true}) Cluster.dissolve({interactive: true}) 解散 InnoDB 集群，需连接到读写实例

InnoDB Cluster 常用操作

常见操作	命令示例	备注
连接实例	\connect root@192.168.1.12:3306	
检测实例状态	dba.checkInstanceConfiguration('root@192.168.1.12:3306')	
自动修正实例设置	dba.configureInstance('root@192.168.1.12:3306')	

删除集群元数据	dba.dropMetadataSchema()	
创建集群	var cluster = dba.createCluster('ngqcluster')	已存在 getCluster()
向集群中添加实例	cluster.addInstance('root@192.168.1.13:3306')	密码加{password:'pwd'}
从集群中删除实例	cluster.removeInstance('root@192.168.1.13:3306')	强制加{force:true}
查看集群状态	cluster.status()|status({extended:1|2|3})	描述：cluster.describe()
重启集群	var cluster = dba.rebootClusterFromCompleteOutage() 停电重启	
解散集群	cluster.dissolve({force:true}) 删除元数据和配置关闭组复制但不删除数据	
指定新主节点	cluster.setPrimaryInstance('root@192.168.1.13:3306')	
切换到多主模式	cluster.switchToMultiPrimaryMode()	
切换到单主模式	cluster.switchToSinglePrimaryMode('root@192.168.1.12:3306')	
查看配置选项	cluster.forceQuorumUsingPartitionOf("root@192.168.1.12:3306") 仅存的节点	
查看配置选项	cluster. rejoinInstance('root@192.168.1.12:3306') 重新加入集群	
查看配置选项	cluster. options()	
更改集群设置	cluster.setOption('clusterName','nieCluster')	设置实例：.setInstanceOption
停止沙漏实例	dba.stopSandboxInstance(3309,{password:'pw'});	启动：startSandboxInstance

常见错误处理方法：
防火墙配置：3306/33061

metadata exists, instance belongs to that metadata, but GR is not active
连接到主节点，重启集群（rebootClusterFromCompleteQutage()）
或者 mysql 登录到此节点，启动此节点的 group Replication(start group_replication;必须先设置 group_replication_bootstrap_group 为 on)：
Sql>set global group_replication_bootstrap_group=on;
Sql>start group_replication;
Sql>set global group_replication_bootstrap_group=off;

metadata exists, instance does not belong to that metadata, and GR is not active
连接到错误节点删除节点元数据（dropMetadataSchema()），再重新加入集群：

参考资料

Mysql InnoDB cluster Document：

https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-innodb-cluster.html

https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-innodb-replicaset.html
https://dev.mysql.com/doc/mysql-shell/8.0/en/deploying-production-innodb-cluster.html
https://dev.mysql.com/doc/mysql-shell/8.0/en/configuring-innodb-cluster.html
https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-innodb-cluster-working-with-cluster.html
https://dev.mysql.com/doc/mysql-shell/8.0/en/admin-api-integrating-router.html
https://dev.mysql.com/doc/mysql-router/8.0/en/mysql-router-innodb-cluster.html
https://github.com/wagnerjfr/mysql-innodb-cluster
https://github.com/wwwted/mysql-innodb-cluster-local-sandbox
https://dev.mysql.com/doc/mysql-port-reference/en/mysql-ports-reference-tables.html#mysql-shell-ports
https://thesubtlepath.com/mysql/building-out-the-mysql-innodb-clusterset-for-high-availability-disaster-recovery-in-one-fully-supported-platform/
https://thesubtlepath.com/mysql/clusterset-router-integration-operational-details-uncovered-part2-of-a-series/