集群的部署

MySQL Cluster有3类节点，分别是管理节点，数据节点以及SQL节点。

实验环境：

节点配置：1个管理节点，2个数据节点，2个SQL节点

虚拟机：VirtualBox

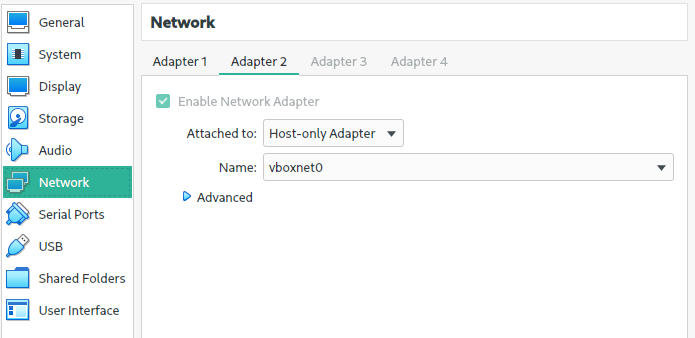
虚拟机运行的系统：Ubuntu 18.04 server

1. 虚拟机的安装和配置

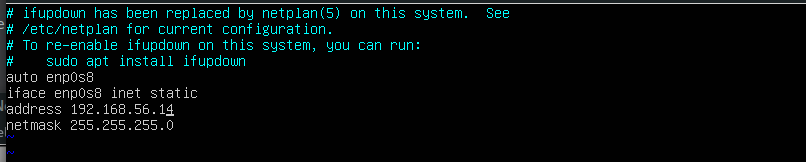
STEP1：在网站<https://ubuntu.com/download>下载ubuntu18.04 server镜像

STEP2：在虚拟机中安装系统

STEP3：为了虚拟机可以互相访问，需要对网络进行配置。在设置里添加Host-only Adapter

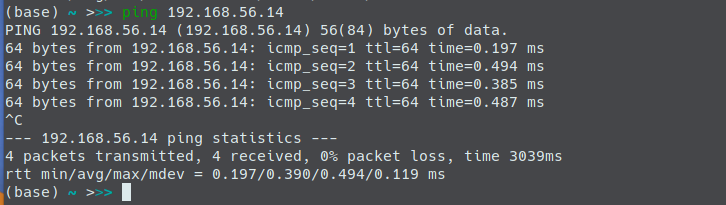


STEP4:启动虚拟机，编辑/etc/network/interfaces



STEP5：运行sudo ifup enp0s8启动配置

STEP6：ping 192.168.56.14，发现可以通，至此虚拟机以及相关网络配置结束



1. 节点的安装

本次实验使用预编译好的程序进行安装

首先访问<https://dev.mysql.com/downloads/cluster/> 下载mysql-cluster-8.0.19-linux-glibc2.12-x86\_64.tar.gz到/var/tmp目录下

* 1. SQL节点的安装

STEP1：创建名为mysql的user以及group，并将其添加到group中

Sudo groupadd mysql

Sudo useradd -g mysql -s /bin/false mysql

STEP2：到文件下载目录下，解压文件到/usr/local/目录下，重命名为mysql

Cd /var/tmp

Sudo Tar -C /usr/local -zxvf mysql-cluster-8.0.19-linux-glibc2.12-x86\_64

.tar.gz

Cd /usr/local

Sudo mv mysql-cluster-8.0.19-linux-glibc2.12-x86\_64 mysql

STEP3:到mysql目录下，初始化数据库

Cd mysql

./bin/mysqld --initialize

STEP4：给MySQL server以及data目录提供必要的权限

Sudo chown -R root .

Sudo chown -R mysql data

Sudo chgrp -R mysql .

STEP5:配置启动脚本

Sudo cp support-files/mysql.server /etc/init.d/

Sudo chmod+x /etc/init.d/mysql.server

Sudo update-rc.d mysql.server defaults

以上步骤在每个SQL节点都需要进行

* 1. 数据节点的安装

STEP1：到文件的下载目录，解压文件并复制ndbd以及nbdmtd到/usr/local/bin

Cd /var/tmp

Tar -zxvf mysql-cluster-8.0.19-linux-glibc2.12-x86\_64.tar.gz

Cd mysql-cluster-8.0.19-linux-glibc2.12-x86\_64

Sudo cp bin/ndbd /usr/local/bin/ndbd

Sudo cp bin/ndbmtd /usr/local/bin/ndbmtd

STEP2:为nbbd和ndbmtd添加执行权限

Cd /usr/local/bin

Sudo chmod +x ./ndb\*

以上步骤在每个数据节点都要进行

* 1. 管理节点的安装

STEP1：到文件的下载目录，解压文件并复制nbb\_mgm和ndb\_mgmd到/usr/local/bin

Cd /var/tmp

Tar -zxvf mysql-cluster-8.0.19-linux-glibc2.12-x86\_64.tar.gz

Cd mysql-cluster-8.0.19-linux-glibc2.12-x86\_64

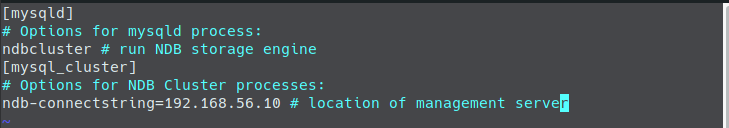
Sudo cp bin/ndb\_mgm\* /usr/local/bin

STEP2:添加执行权限

Sudo chmod +x /usr/local/bin/ndb\_mgm\*

1. 配置节点

数据节点和SQL节点：编辑/etc/my.cnf

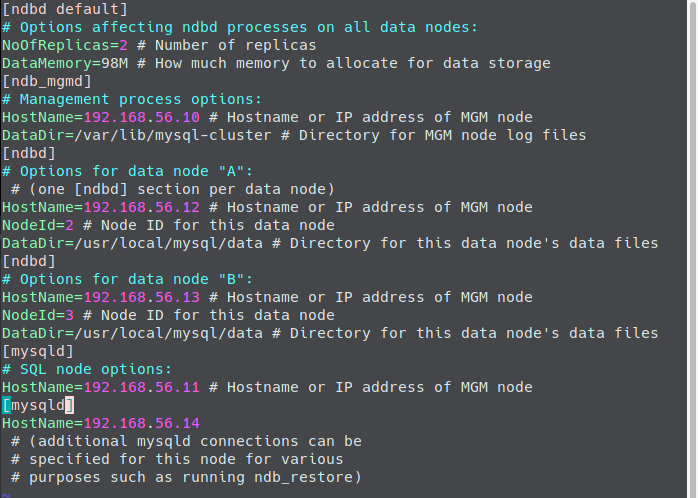


管理节点：创建/var/lib/mysql-cluster目录，编辑/var/lib/mysql-cluster/config.ini

Sudo mkdir /var/lib/mysql-cluster

Sudo cd /var/lib/mysql-cluster

Sudo vim config.ini



1. 启动节点

启动顺序：管理节点，数据节点，SQL节点。

管理节点：sudo ndb\_mgmd -f /var/lib/mysql-cluster/config.ini

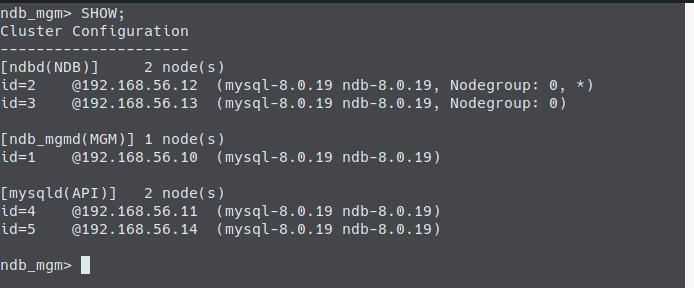
数据节点：sudo ndbd

SQL节点：mysqld\_safe -user=mysql &

在管理节点上运行

sudo ndb\_mgm

SHOW;



可以看到，各个节点都已经成功上线

1. 相关实验

5.1添加数据

在其中一个SQL节点上进行如下操作：

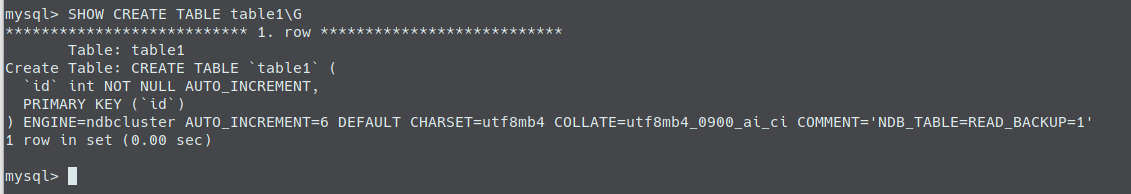
mysql> CREATE DATABASE mydb;

mysql> use mydb;

Mysql> CREATE TABLE table1(id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY) ENGINE=ndbcluster;

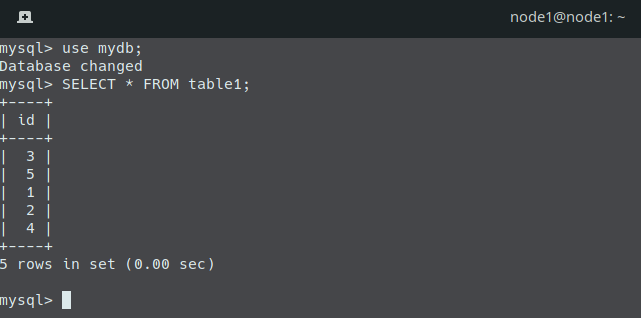
Mysql> INSERT INTO table1 VALUES (1),(2),(3),(4),(5);

Mysql> SHOW CREATE TABLE table1\G

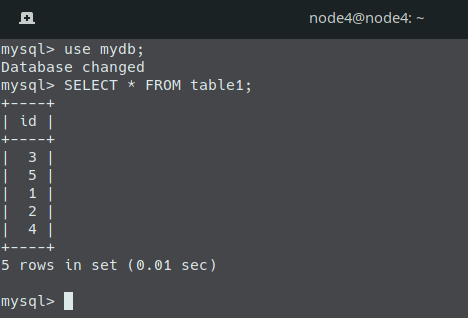


在每个SQL节点查看数据：

SQL节点1:



SQL节点2：



可以发现，在每个SQL节点中都可以获得数据。

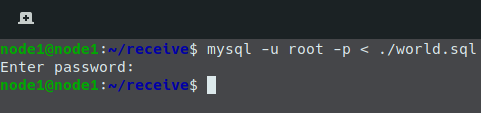
5.2 导入官方示例world数据

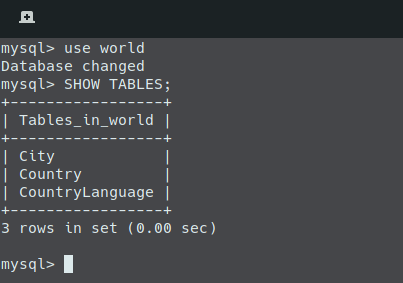
SQL节点1：

创建数据库world

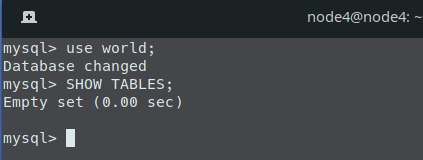
Mysql> CREATE DATABASE WORLD;

导入数据并查看



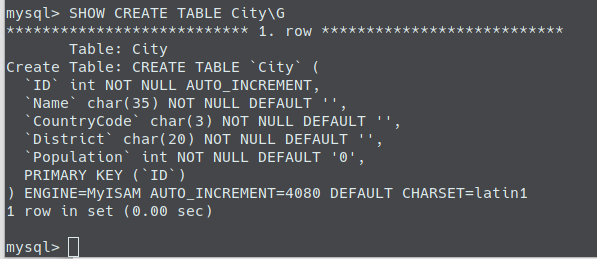


SQL节点2：



结论：从SQL节点1中创建的数据库可以从SQL节点2中获取，但是所导入的数据无法获取。

原因是表的引擎是MyISM而不是ndbcluster,所以数据没有存储到数据节点，因此只能在本地访问。

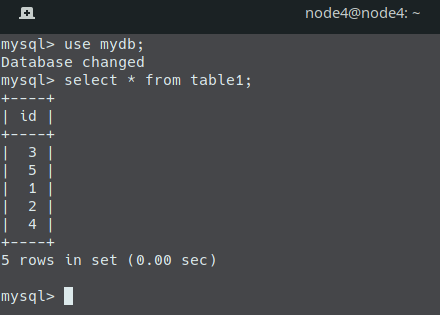


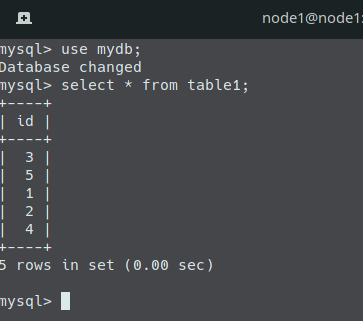
5.3测试可用性

从实验5.1可知，目前mydb中的数据可以从两个SQL节点都能访问。

现在，从管理节点中停掉其中一个数据节点：

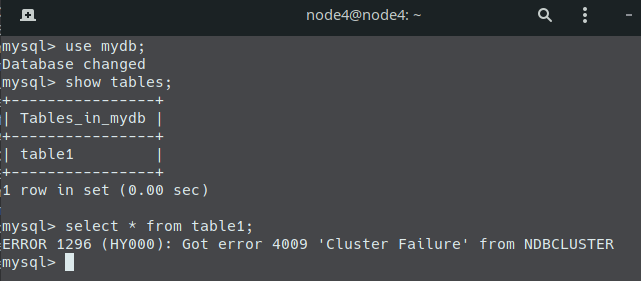
分别测试两个SQL节点能否访问数据



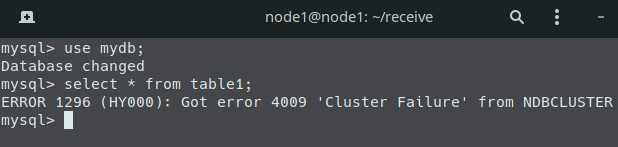


可以发现，两个SQL节点都能访问数据

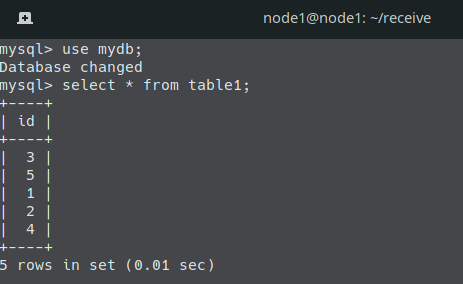
继续停掉另一个节点，再次访问数据，发现两个SQL节点都开始报错



现在启动一个节点，再次访问数据，发现还是报错

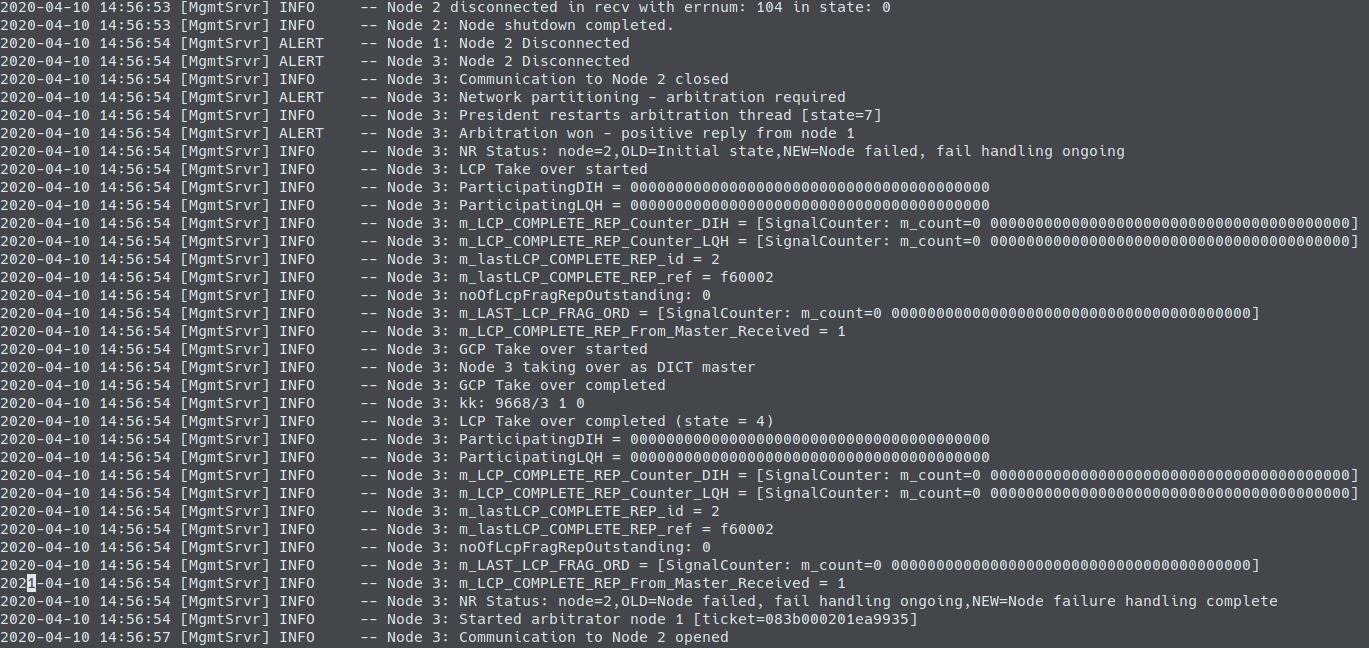


启动另一个数据节点，访问数据



此时两个SQL节点都能访问数据

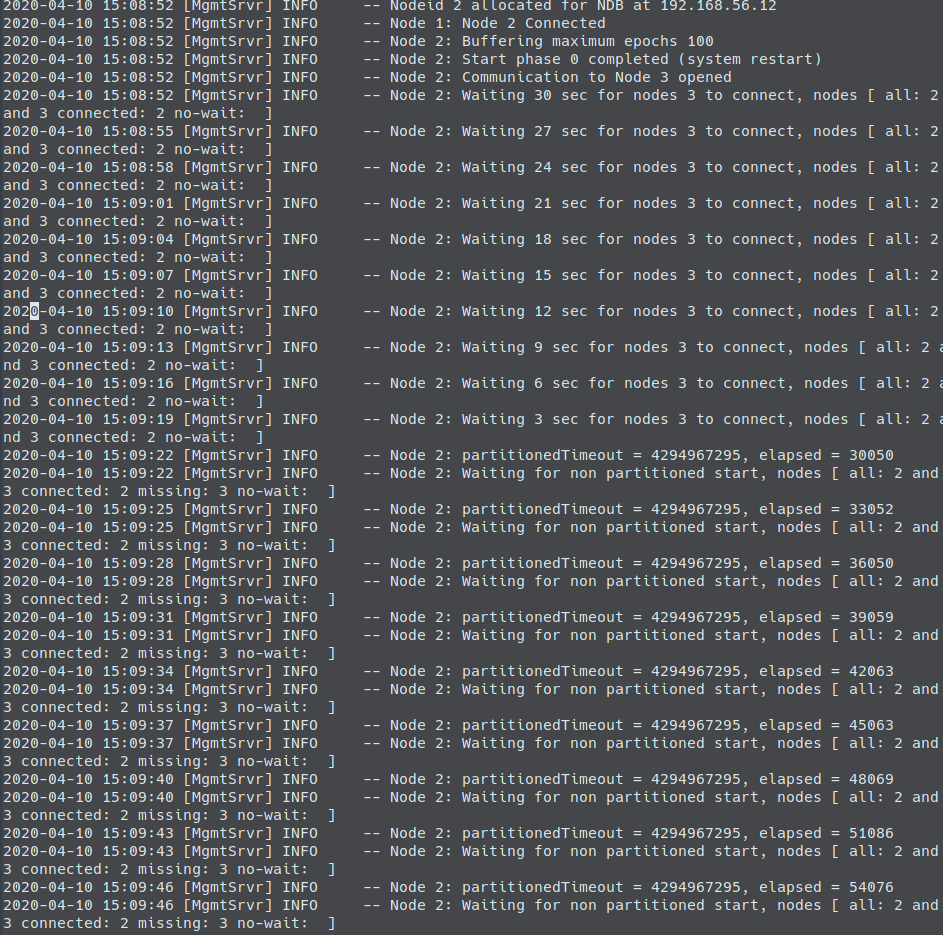
在关闭其中1个数据节点后，log如下



发现：停掉数据节点1(Node2)后，管理节点(Node1)和数据节点2(Node 3)都收到数据节点1断开连接的信息。此时数据节点2数据节点1的通信关闭。数据节点2开始请求网络分区的仲裁。总统重启仲裁进程。仲裁得到管理节点1的肯定回复，进行分区操作，完成后，数据节点2与数据节点1的通信打开。

而在停掉数据节点2后，管理节点只收到数据节点2断开连接的消息，并未采取任何操作。同时，在ndb\_mdm上运行show命令，发现两个SQL节点都已断开连接。

此时开启数据节点1，log如下



其中数据节点1一直等待数据节点2上线，等待分区开始。

启动数据节点2成功启动后，开始恢复数据库，最终SQL节点也都上线

心的体会：

我学会了如何部署MySQL NDB Cluster，并且了解了数据同步的过程。