MySQL Cluster Configuration

Setup:

Management Server - 1

Data Node- 2

SQL – 2

Reference Links:

1. <http://stepbysteparticles.com/index.php?option=com_content&view=article&id=48&Itemid=54>
2. <https://github.com/zorani/MySqlStarCluster>
3. <https://github.com/zorani/MySqlStarCluster/blob/master/ManagementNode.sh>

Introduction:

The goal of this article is to basically give you a step by step on how to set up a MySQL Cluster. I will not go into a lot of technical details but will try to keep it straight forward and simple. So, if you are looking for a very detailed guide on setting up a MySQL Cluster with explanation on every command, this article is not for you. But, if you are looking for a brief overview on setting up a MySQL Cluster or a practical guide on MySQL clustering, this certainly will help you. So, let's start:

MySQL Cluster is an in-memory highly available database solution. It does not have a central/shared storage architecture which minimizes the risk of single point of failure (critical to web applications). All the transactions replicate to all the nodes from memory thus keeping all the nodes in synch. This type of architecture provides to great features, first as I said it minimizes the risk of single point of failure and second it makes scaling out an easy task (whenever needed, adding another storage node is an easy task). Also, it makes developing load balanced backend solutions possible.

This capability of MySQL gives it a higher ranking (in my view) with respect to SQL Server and Oracle (off course a database engine cannot be judged alone on the basis of its high availability architecture, Oracle and SQL Server both are respectable products and fit the needs of many critical systems worldwide).

NDB

MySQL Cluster is based off of NDB storage engine. It is a shared nothing cluster engine stored in memory. If you have a 2-node cluster, both the nodes would have the data available independently. Data would be committed on all the cluster nodes simultaneously (2-phase commit like SQL Server mirroring). NDB is a powerful engine comparable in performance with SQL Server (again in my view). There are some things you have to remember however, it does not support Full-Text indexing for example, so if your application needs it, either you go with a third-party product to implement that or chose a different solution (Master/Slave replication ?).

Setting up the Cluster

To set up the cluster, you need three servers. Two cluster nodes and one Management node. I should point out that the Management node is not required after the cluster install, but I strongly recommend keeping it as it gives you the automatic failover capability. I will use three servers as examples:

Server1 192.168.2.145 (Cluster Management Server)

Server2 192.168.2.103 (Data Node 1 with SQL)

Server3 192.168.2.161 (Data Node 2 with SQL)

**MySQL Cluster Management Server Configuration:**

First step is to install MySQL Cluster Management Server on Server1. Lets download from MySQL Cluster 6.2 from MySQL website (http://dev.mysql.com/downloads/cluster/). This guide is intended for Debian based systems, so we will download nonrpm package (mysql-cluster-gpl-6.2.15-linux-i686-glibc23.tar.gz). Here are the steps to follow, to set up the MySQL Cluster Management Server (ndb\_mgmd) and the cluster Management client (ndb\_mgm)

*ManagementNode.sh*

*xxxxxxxxxxxxxxxxxxxxxxxxx*

*#!/bin/sh*

*#####################################*

*#Management Node*

*#####################################*

*export DEBIAN\_FRONTEND=noninteractive*

*rm -rf /var/lib/mysql*

*rm -rf /etc/mysql/*

*rm -f /etc/mysql/my.cnf*

*apt-get -y remove mysql\**

*apt-get -y --purge remove*

*apt-get -y autoremove*

*dpkg --get-selections | grep mysql*

*aptitude -y purge $(dpkg --get-selections | grep deinstall | sed s/deinstall//)*

*cd /var/tmp/*

*wget http://dev.mysql.com/get/Downloads/MySQL-Cluster-7.3/mysql-cluster-gpl-7.3.7-linux-glibc2.5-x86\_64.tar.gz*

*tar -xvzf mysql-cluster-gpl-7.3.7-linux-glibc2.5-x86\_64.tar.gz*

*cp /var/tmp/mysql-cluster-gpl-7.3.7-linux-glibc2.5-x86\_64/bin/ndb\_mgm\* /usr/local/bin/*

*rm -rf /var/tmp/\**

*mkdir -p /var/lib/mysql-cluster/*

*cat > /var/lib/mysql-cluster/config.ini << EOF*

*#####################################*

*# Config File For Management Node #*

*#####################################*

*#*

*# Which node?*

*# Answer: Master.*

*#*

*# Which location?*

*# Answer: /var/lib/mysql-cluster/config.ini*

*[ndbd default]*

*NoOfReplicas=1*

*DataMemory=80M*

*IndexMemory=18M*

*[ndb\_mgmd]*

*hostname=master*

*datadir=/var/lib/mysql-cluster*

*[ndbd]*

*hostname=node001*

*datadir=/usr/local/mysql/data*

*[mysqld]*

*hostname=node002*

*EOF*

*ndb\_mgmd --configdir=/var/lib/mysql-cluster/ -f /var/lib/mysql-cluster/config.ini*

Script Link: <https://github.com/zorani/MySqlStarCluster/blob/master/ManagementNode.sh>

Main page Link: <https://github.com/zorani/MySqlStarCluster>

Now let's start the Management Server:

ndb\_mgmd -f /var/lib/mysql-cluster/config.ini

Now, we would want to start the Management Server automatically in case of a system reboot, so we add an init script to do that:

echo 'ndb\_mgmd -f /var/lib/mysql-cluster/config.ini' > /etc/init.d/ndb\_mgmd

chmod 755 /etc/init.d/ndb\_mgmd

update-rc.d ndb\_mgmd defaults

**Data Nodes Configuration (Server2 and Server3):**

Now let's set up the data nodes. Here are the steps to do that (do on both data nodes):

groupadd mysql

useradd -g mysql mysql

cd /usr/local/

wget pick up any mirror from MySQL's website

tar xvfz mysql-cluster-gpl-6.2.15-linux-i686-glibc23.tar.gz

ln -s mysql-cluster-gpl-6.2.15-linux-i686-glibc23 mysql

cd mysql

*Note: Error: libaio.so.1: cannot open shared object file: No such file or directory*

*Solution: https://help.directadmin.com/item.php?id=368*

*CMD: apt-get install libaio1 libaio-dev*

scripts/mysql\_install\_db --user=mysql

chown -R root:mysql .

chown -R mysql data

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

chmod 755 /etc/init.d/mysql.server

update-rc.d mysql.server defaults

cd /usr/local/mysql/bin

mv \* /usr/bin

cd ../

rm -fr /usr/local/mysql/bin

ln -s /usr/bin /usr/local/mysql/bin

Next we need to create the MySQL config file /etc/my.cnf on both nodes:

vi /etc/my.cnf

Here is the sample file:

[mysqld]

ndbcluster

# IP address of the cluster management server (Server1)

ndb-connectstring=192.168.100.1

[mysql\_cluster]

# IP address of the cluster management Server (Server1)

ndb-connectstring=192.168.100.1

Our MySQL installation is almost complete, now let's create the data directories and start the MySQL Server on both nodes:

mkdir /var/lib/mysql-cluster

cd /var/lib/mysql-cluster

ndbd --initial

/etc/init.d/mysql.server start

(Important: we need to run ndbd --initial only when the start MySQL for the first time, and if /var/lib/mysql-cluster/config.ini on Management Server changes.)

MySQL installation is complete, now let's put in a root password for our MySQL Servers:

mysqladmin -u root password newrootpassword

Again, it makes sense to start up the cluster nodes automatically in case of a system restart/failure. Here are the ndbd init script and system startup links for that:

echo 'ndbd' > /etc/init.d/ndbd

chmod 755 /etc/init.d/ndbd

update-rc.d ndbd defaults

this completes are Cluster installation process, next, now let's test it.

Test:

On Cluster Management Server, run the Cluster Management Client:

ndb\_mgm

It will take you to the ndb\_mgm prompt:

-- NDB Cluster -- Management Client --

ndb\_mgm>

Now type show on the prompt:

ndb\_mgm> show;

You should see an output similar to this:

ndb\_mgm> show;

Connected to Management Server at: localhost:1186

Cluster Configuration

---------------------

[ndbd(NDB)] 2 node(s)

id=2 @192.168.2.161 (Version: version number, Nodegroup: 0, Master)

id=3 @192.168.2.103(Version: version number, Nodegroup: 0)

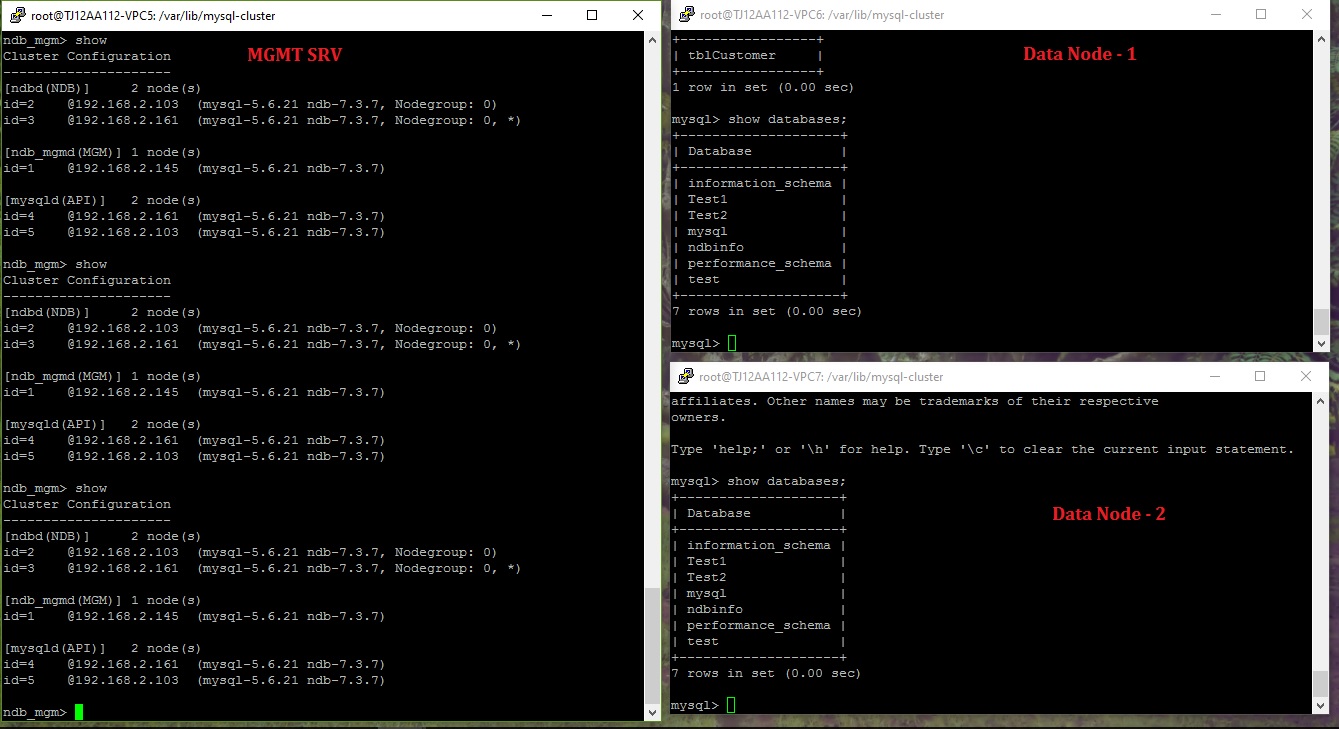
[ndb\_mgmd(MGM)] 1 node(s)

id=1 @192.168.2.145(Version: version number)

[mysqld(API)] 2 node(s)

id=4 @192.168.2.161 (Version: version number)

id=5 @192.168.2.103(Version: version number)



ndb\_mgm>

We should see our data nodes connected in the previous screen. Now type quit to close the Management client:

ndb\_mgm>quit;

Test the Cluster:

Now, let's create a Test database on Server2 (192.168.2.161) and run some tests:

On Server2:

mysql -u root -p

CREATE DATABASE testdb;

USE testdb;

CREATE TABLE tblCustomer (ID INT) ENGINE=NDBCLUSTER;

INSERT INTO tblCustomer VALUES (1);

SELECT \* FROM tblCustomer;

quit;

pay attention to the create table statement, we must specify ENGINE=NDBCLUSTER for all tables that we want to clustered. As stated earlier, MySQL cluster only saupports NDB engine, so if you use any other engine, table simply wont get clustered.

The result of the SELECT statement would be:

mysql> SELECT \* FROM tblCustomer;

+------+

| ID |

+------+

| 1 |

+------+

Since clustering in MySQL is at the "table level" not at the database level, so we would have to create the database sperately on Server3 (192.168.100.3) as well, but afterwards tblCustomer would be replicated with all its data (since the engine is NDBCLUSTER):

On Server3:

mysql -u root -p

CREATE DATABASE testdb;

USE testdb;

SELECT \* FROM tblCustomer;

Now, if we insert a row of data on Server3, it should be replicated back to Server2:

INSERT INTO tblCustomer VALUES (2);

If we run a SELECT query on Server2, here is what we should see:

mysql> SELECT \* FROM testtable;

+------+

| ID |

+------+

| 1 |

| 2 |

+------+

Test Node shutdown:

Now run the following on Server2 to test what happens if a node goes offline:

killall ndbd

and run this command to make sure that all ndbd processes have terminated:

ps aux | grep ndbd | grep -iv grep

If you still see any prpcesses, run this again:

killall ndbd

Now, lets go the management server (Server1) and run the following to check the cluster status:

ndb\_mgm

On the ndb\_mgm console. run:

show;

it should be bring you an output simlar to the following:

ndb\_mgm> show;

Connected to Management Server at: localhost:1186

Cluster Configuration

---------------------

[ndbd(NDB)] 2 node(s)

id=2 (not connected, accepting connect from 192.168.2.161)

id=3 @192.168.2.103(Version: -----, Nodegroup: 0, Master)

[ndb\_mgmd(MGM)] 1 node(s)

id=1 @192.168.2.145(Version: -----)

[mysqld(API)] 2 node(s)

id=4 @192.168.2.161 (Version: --------)

id=5 @192.168.2.103(Version: --------)

ndb\_mgm>

You see, Server2 is not connected anymore.

Type quit; to leave the ndb\_mgm management console. Now, let's check on Server3, if our database is still up and we can make connections to it:

mysql -u root -p

USE testdb;

SELECT \* FROM tblCustomer;

quit;

It should bring up the following result set:

mysql> SELECT \* FROM tblCustomer;

+------+

| ID |

+------+

| 1 |

| 2 |

+------+

Now, let's start MySQL on Server2 again by issuing the following command:

ndbd

How to Restart MySQL Cluster:

In managing a produciton MySQL environment or any other transactional database environment, times come when we have to restart/shutdone our systems. So, let's see how would we shutdown our MySQL Cluster:

On Server1, open the management console:

ndb\_mgm

then type:

shutdown;

it would bring up an output like this:

ndb\_mgm> shutdown;

Node 3: Cluster shutdown initiated

Node 2: Node shutdown completed.

2 NDB Cluster node(s) have shutdown.

NDB Cluster management server shutdown.

ndb\_mgm>

This means that the cluster nodes Server2 and Server3 and also the Management node (Server1) have shut down.

To leave the Management console, run:

quit;

To start the cluster management server again, run the following (on Server1, Management Server):

ndb\_mgmd -f /var/lib/mysql-cluster/config.ini

and on Server2 and Server3, run the following:

ndbd

in case /var/lib/mysql-cluster/config.ini on Management Server changed, you should run the following:

ndbd --initial

You can go back to the Management node and verify if the cluster started ok, without any errors:

ndb\_mgm

on the Management console run the following:

show;

This should bring up the cluster configuration.