|  |  |
| --- | --- |
| Project Name: | NHA-Facts Portal |
| Document Name | Production DB Server for Clustering |
| Created By | Mr. Sudhir Kumar Singh |

**InnoDB Clustering in MySQL**

**Check the instance configuration created earlier or not in MySQLSH**

dba.checkInstanceConfiguration ('admin@127.0.0.1:3306')

**If instance is not configured then proceeds bellows processes otherwise monitoring the InnoDB Clustering in MySQL shell**

Configuring Production Instances for InnoDB Cluster

On Master InnoDB Cluster System - 1

dba.configureInstance ('admin@127.0.0.1:3306')

On Replica InnoDB Cluster System – 2

dba.configureInstance ('admin@127.0.0.1:3306')

On Replica InnoDB Cluster System – 3

dba.configureInstance ('admin@127.0.0.1:3306')

### Creating an InnoDB Cluster on Master InnoDB Cluster-1

### var cluster = dba.createCluster ('nhaCluster')

### Adding Instances to an InnoDB Cluster

### cluster.addInstance ('admin@nhadelgccmysql2:3306')

### cluster.addInstance ('admin@nhadelgccmysql3:3306')

### Configuring the Election Process of Primary (Master) System

### Configure how a single-primary cluster elects a new primary to prefer one instance as the new primary to fail over the primary cluster.

### The memberWeight option accepts an integer value between 0 and 100, which is a percentage weight for automatic primary election on failover. When an instance has a higher percentage number set by memberWeight.

dba.createCluster ('nhaCluster', {memberWeight:55})

var mycluster = dba.getCluster ()

mycluster.addInstance ('admin@delnhagcc-prdsascpdbmysql2:3306', {memberWeight:45})

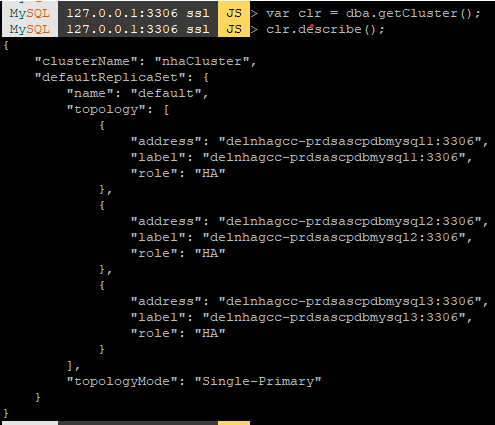
### mycluster.addInstance ('admin@delnhagccc-prdsascpmysql3:3306', {memberWeight:35})

## **Monitoring InnoDB Cluster**

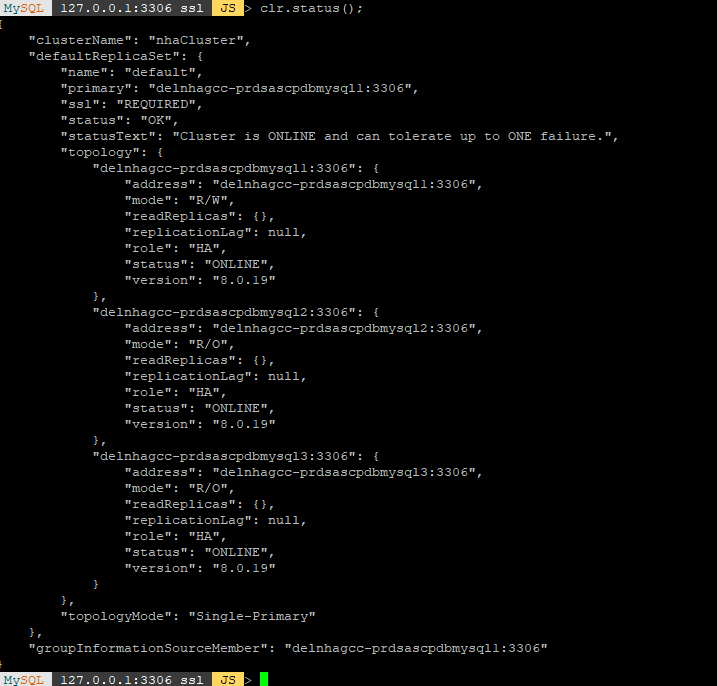
### Using *Cluster*.describe()

### LinuxShell $> mysqlsh <- Enter

### MySQL> \c [admin@127.0.0.1:3306](mailto:admin@127.0.0.1:3306)



### Checking a cluster's Status with *Cluster*.status()



## **If any issues of InnoDB Cluster…**

Firstly we check the systems are in innodb cluster or not through the queries –

* **Re-insert a MISSING instance**

MySQL 127.0.0.1:33060+ ssl JS > cluster.rejoinInstance ('admin@delnhagcc-prdsascpdbmysql1:3306');

If your instance was out of cluster for a while, this action can take long time or fail. In case of fail, please check your logs to determine the problem and fix-it.

Check the active instances on primary server –

MySQL 127.0.0.1:3306 ssl JS > var cluster = dba.getCluster ()

MySQL 127.0.0.1:3306 ssl JS > cluster.status ()

{

"clusterName": "nhaCluster",

"defaultReplicaSet": {

"name": "default",

"primary": "delnhagcc-prdsascpdbmysql1:3306",

"ssl": "REQUIRED",

"status": "OK ",

"statusText": "Cluster is ONLINE and tolerate up to ONE failure",

"topology": {

“delnhagcc-prdsascpdbmysql1:3306": {

"address": delnhagcc-prdsascpdbmysql1:3306",

"mode": "R/W",

"readReplicas": {},

"role": "HA",

"status": "(MISSING)"

},

"delnhagcc-prdsascpdbmysql2:3306": {

"address": " delnhagcc-prdsascpdbmysql2:3306",

"mode": "R/O",

"readReplicas": {},

"role": "HA",

"status": "(MISSING)"

},

"delnhagcc-prdsascpdbmysql3:3306": {

"address": "delnhagcc-prdsascpdbmysql3:3306",

"mode": "R/W",

"readReplicas": {},

"role": "HA",

"status": "ONLINE"

}

}

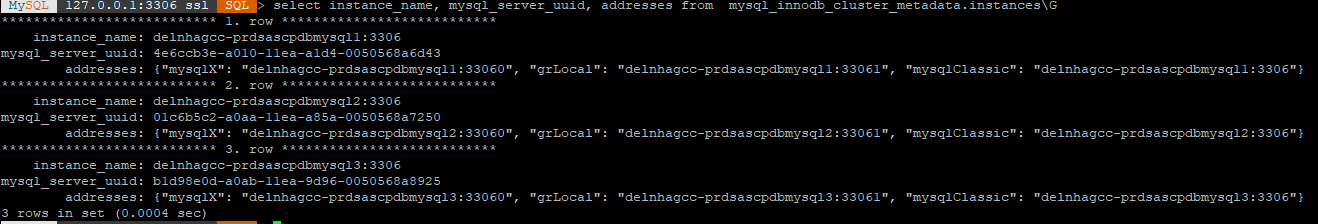
“topologyMode”: ”single-primary”

},

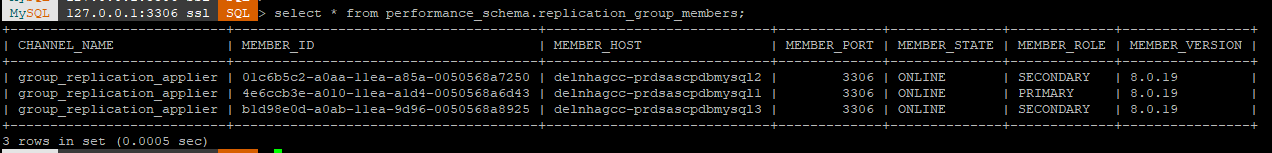
"groupInformationSourceMember": " delnhagcc-prdsascpdbmysql1:3306"

}

SELECT instance\_name, mysql\_server\_uuid, addresses FROM mysql\_innodb\_cluster\_metadata- .instances;



SELECT \* FROM performance\_schema.replication\_group\_members;



When MEMBER\_STATE is ONLINE and MEMBER\_ROLE is shown as PRIMARY & SECONDARY as MEMBER\_HOST then we check the MySQL Connections on 3306 or 6446 port numbers.

When MEMBER\_STATE is OFFLINE and MEMBER\_ROLE is shown as STANDLONE as MEMBER\_HOST then we need to run the dba.rebootClusterFromCompleteOutage (); command from the mysqlsh-js console.

After reboot Outage cluster, again we check the metadata instances and state of replication group members.

**Check the status of mysqlrouter for connection pooling…**

**If any issues of MySQL Connection on 6446 port**, then

Firstly we check status of MySQL Router …

$**sudo systemctl status mysqlrouter or #systemctl status mysqlrouter**

If Status is Active: active (running), again check the MySQL Connection through Workbench tool.

If MySQL Connection is not connected then we run the command in user terminal

mysqlrouter --bootstrap root@localhost:3306 --directory /tmp/router –user=root --force

#--force – option use for overriding the directory and files

cd /tmp/router

./start.sh <-(Enter)

MySQL Connection is connected on port numbers 6446 and 3306, Cluster systems are working fine.

**For DB Server Patching follows some steps:-**

When we start database server patching or server maintenance activities of MySQL Clustering systems like 100.65.132.254, \*.\*.\*.229, \*.\*.\*.230, so, we carefully monitoring database server.

|  |  |  |
| --- | --- | --- |
| S. No | DB Server IP | Type of System |
| 1. | 100.65.132.254 | Master System |
| 2. | 100.65.132.229 | Slave1 System |
| 3. | 100.65.132.230 | Slave2 System |

Firstly we stop or start the slave2 system, then stop or start the slave1 system, finally we stop or start the master system and check above procedure and for the MySQL Router service status. If router service is stop or inactive (dead) system, we restart the service of MySQL Router.

Check the port 6446 of MySQL Router is active or not, and check connection of Master DB System, through the MySQL Workbench, DB Connection is connected or not.

**How to take the down time of MySQL Clustering system?**

Firstly, we will be down Primary Cluster (master-100.65.132.254) System and check SQL Query which system became as primary cluster system in Other Cluster system MySQL Shell. Then we will be up this cluster system.

Secondly, we will be down any Secondary Cluster (slave1/2-100.65.132.229/30) System and check SQL Query which system became as primary cluster system in Other Cluster system MySQL Shell. Then we will be up this cluster system.

Finally, we check the system-100.65.132.254, became as primary cluster system and check DB Connection is working at port no- 6446, if it is not running, then check MySQLRouter Service is running or not and follows the above MySQLRouter process.