Running Cassandra Multinode Cluster

[4 Nodes & 2 Datacenters]

Install Cassandra on all Nodes as follows:

```
echo "deb http://www.apache.org/dist/cassandra/debian 311x main" | sudo tee -a /etc
/apt/sources.list.d/cassandra.sources.list
curl https://www.apache.org/dist/cassandra/KEYS | sudo apt-key add -
sudo apt-get update

sudo apt-get install cassandra
```

- You can start Cassandra with `sudo service cassandra start`
- Stop it with `sudo service cassandra stop`

Normally the service will start automatically. For this reason be sure to stop it if you need to make any configuration changes.

- Verify that Cassandra is running by invoking `nodetool status` from the command line.

```
nodetool status
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address    Load       Tokens       Owns    Host ID
  Rack
UN  127.0.0.1  172.18 KiB  256          ?       60bc1910-eb04-4c7f-bc44-4de3643ce4f
4  rack1

Note: Non-system keyspaces don't have the same replication settings, effective owne
rship information is meaningless
```

- The default location of configuration files is `/etc/cassandra`.
- The default location of log and data directories is `/var/log/cassandra/` and `/var/lib/cassandra`.
- Start-up options (heap size, etc) can be configured in `/etc/default/cassandra`.

1. Stop Cassandra-daemon-

```
sudo service cassandra stop
```

2. Delete the default dataset:

```
sudo rm -rf /var/lib/cassandra/data/system/*
```

3. Edit the cassandra.yaml file as follows-

```
sudo vim /etc/cassandra/cassandra.yaml
```

4. The contents should look like the below:

```
  cluster_name: 'CassandraDOCluster'

  seed_provider:
    - class_name: org.apache.cassandra.locator.SimpleSeedProvider
      parameters:
           - seeds: "your_server_ip,your_server_ip_2,...your_server_ip_n"

  listen_address: your_server_ip

  rpc_address: your_server_ip

  endpoint_snitch: GossipingPropertyFileSnitch
```

- at the end of the cassandra.yaml file add the following :
  `auto_bootstrap: false`
- edit the file below
  `sudo vim /etc/cassandra/cassandra-env.sh`
- search for **hostname** and place your IP address in it

5. In the cassandra-rackdc.properties file, assign the data center and rack names you determined in the Prerequisites.

For example:

Nodes 0 to 2

```
indicate the rack and dc for this node
dc=DC1
rack=RAC1
```

Nodes 3 to 5

```
indicate the rack and dc for this node
dc=DC2
rack=RAC1
```

5. Restart the cassandra-daemon

```
sudo service cassandra start

sudo service cassandra restart
```

6. Check status of the cluster

```
sudo nodetool status
sudo nodetool status <keyspace-name> (if the keyspaces don't have the same replication factor)
```

O/P

```
  Datacenter: dc1
===============
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address         Load       Tokens       Owns (effective)  Host ID
                Rack
UN  172.31.86.204   456.99 MiB  256          23.6%             c03141fc-ae28-4d4c-b6
58-cb949e5ccc57  rack1
UN  172.31.90.24    107.44 KiB  256          27.5%             8c143d7a-69d2-48c1-8a
23-dcda6ce9dfa5  rack1
Datacenter: dc2
===============
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address         Load       Tokens       Owns (effective)  Host ID
                Rack
UN  172.31.88.141   297.85 MiB  256          23.9%             f16bf414-f528-49fe-90
6c-53092f6fe957  rack1
UN  172.31.88.19    355.46 MiB  256          25.1%             fee3502e-8cd7-4433-af
b6-8216e6d8dd66  rack1
```

10. Check cluster status-

```
sudo nodetool status
```

11. Configuring Vnodes in Cassandra-

REFERENCE LINK–1

REFERENCE LINK–2

- Virtual nodes have been enabled by default since 2.0
- you can enable them as follows

  ```
  sudo vim /etc/cassandra/cassandra.yaml
  ```

- set number of tokens as required

  `num_tokens: 256`
- Uncomment the initial_token property and set it to 1 or to the value of a generated token for a multi-node cluster

12. Partitioner-

- A partitioner determines how data is distributed across the nodes in the cluster
- **Default partition**

  Murmur3Partitioner was added in **1.2**
- **Before that**

  RandomPartitioner was the **default**

13. Replication Strategies-

- A node serves as a replica for different ranges of data
- If one node goes down, other replicas can respond to queries for that range of data
- **replication factor** is the number of nodes in your cluster that will receive copies (replicas) of the same data
- **2** implementations of **AbstractReplicationStrategy** are

  SimpleStrategy

  NetworkTopologyStrategy

14. Consistency levels-

- Available consistency levels

```
ONE (requires 1 replica to respond to request)
TWO (requires 2 replicas to respond to request)
THREE (requires 3 replicas to respond to request)
ALL (requires a response from all of the replicas)
```

eg

```
Connected to 02-04-18-Admatic-Cluster at 172.31.92.220:9042.
[cqlsh 5.0.1 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh> consistency;
Current consistency level is ONE.
cqlsh> CONSISTENCY LOCAL_TWO;
Improper CONSISTENCY command.
cqlsh> CONSISTENCY LOCAL_ONE;
Consistency level set to LOCAL_ONE.
```

```
cqlsh> CONSISTENCY TWO;
Consistency level set to TWO.
cqlsh> consistency;
Current consistency level is TWO.
cqlsh> CONSISTENCY Three;
Consistency level set to THREE.
cqlsh> consistency;
Current consistency level is THREE.
```

15. Durable writes-

- It is a keyspace option
- By default, durable writes is set to true
- When a write request is received, the node first writes a copy of the data to an on-disk append-only structure called commitlog
- Then, it writes the data to an in-memory structure called memtable
- When memtable is full, it writes it to SStable
- Setting `durable_writes : true` will ensure data is written to commitlog
- Incase of abrupt restart of nodes, memtables will be lost as they exist in the memory
- So, the message consistency can be maintained by replaying data from commitlogs to the memtable

Reference links:

Reference link 1