DATASTAX ACADEMY

# Exercise 7 – Ring

In this exercise, you will:

- Understand the Apache Cassandra™ token ring
- Create a two-node cluster

One of the secrets to Apache Cassandra's™ performance is the use of a ring that keeps track of tokens. This ring enables Apache Cassandra™ to know exactly which nodes contain which partitions. The ring also eliminates any single points of failure.

## Steps

1) First, we will shut down your current node. Do so by executing the following command:

   ```
   /home/ubuntu/node/resources/cassandra/bin/nodetool stopdaemon
   ```

   Wait for the node to terminate before continuing.

2) Delete the `/home/ubuntu/node` folder by executing the following commands:

   ```
   cd /home/ubuntu
   rm -rf node
   ```

3) To make a two-node cluster, we will unzip the DataStax Enterprise™ tarball twice making two folders: node1 and node2. In your terminal, execute the following commands within the `/home/ubuntu` directory:

   ```
   tar -xf dse-6.0.0-bin.tar.gz
   mv dse-6.0.0 node1
   labwork/config_node 1
   tar -xf dse-6.0.0-bin.tar.gz
   mv dse-6.0.0 node2
   labwork/config_node 2
   ```

4) Open the `/home/ubuntu/node2/resources/cassandra/conf/cassandra.yaml` file in `vi`, nano or any other text editor.

5) Change the `initial_token` value and set it to 9223372036854775807 (note you need a space between the colon and the value). This node will manage the second half of the token range--the positive tokens.

```
# initial_token allows you to specify tokens manually.  While you can use it with
# vnodes (num_tokens > 1, above) -- in which case you should provide a
# comma-separated list -- it's primarily used when adding nodes to legacy clusters
# that do not have vnodes enabled.
initial_token: 9223372036854775807
```

Save the changes to the file and exit the text editor.

6) In your terminal, start the first node via `/home/ubuntu/node1/bin/dse cassandra`

Wait for the node to start up. Once it has done so, press enter to get back to the prompt.

7) Run the `/home/ubuntu/node1/resources/cassandra/bin/nodetool status` command to verify the node is working properly.

```
ubuntu@ds201-node1:~$ /home/ubuntu/node1/resources/cassandra/bin/nodetool status
Datacenter: Cassandra
=====================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address    Load       Owns  Host ID                               Token
Rack
UN  127.0.0.1  114.83 KiB  ?    e2ba30fc-1589-4ae4-8f98-69051151c44f  0
rack1
```

The UN indicates UP NORMAL meaning the node is ready to go. Load indicates current disk space usage. Owns indicates how many tokens this node is responsible for (it is the only node in the ring at the moment). Token should be 0 (the same value set in the cassandra.yaml file). We discuss racks later in the course.

8) Start the second node via `/home/ubuntu/node2/bin/dse cassandra` command. This node will take longer to bootstrap and join the cluster.

Wait for the second node to finish bootstrapping before continuing.

9) Use `/home/ubuntu/node1/resources/cassandra/bin/nodetool status` again to view the current state of the cluster.

Notice both nodes are now up and normal. If not, please consult your instructor.

10) Now let's recreate the two tables we made in previous exercises and import their data. Start `cqlsh` and execute the commands that follow:

```
CREATE KEYSPACE killrvideo
WITH replication = {'class': 'SimpleStrategy',
'replication_factor': 1 };

USE killrvideo;

CREATE TABLE videos (
    id uuid,
    added_date timestamp,
    title text,
    PRIMARY KEY ((id))
);

COPY videos(id, added_date, title)
FROM '/home/ubuntu/labwork/data-files/videos.csv'
WITH HEADER=TRUE;

CREATE TABLE videos_by_tag (
    tag text,
    video_id uuid,
    added_date timestamp,
    title text,
    PRIMARY KEY ((tag), added_date, video_id))
    WITH CLUSTERING ORDER BY(added_date DESC);

COPY videos_by_tag(tag, video_id, added_date, title)
FROM '/home/ubuntu/labwork/data-files/videos-by-tag.csv'
WITH HEADER=TRUE;
```

11) Now let's determine which nodes own which partitions in the `videos_by_tag table`. Execute the following query:

```
SELECT token(tag), tag
FROM videos_by_tag;
```

- How many partitions are there?
- On which node does each partition reside?

12) You can refresh your memory as to which nodes own which token ranges by running the following in the terminal:

```
/home/ubuntu/node1/resources/cassandra/bin/nodetool ring
```

13) You can further prove which partition resides on which node by executing the following nodetool commands in the terminal:

```
/home/ubuntu/node1/resources/cassandra/bin/nodetool getendpoints killrvideo
videos_by_tag 'cassandra'
/home/ubuntu/node1/resources/cassandra/bin/nodetool getendpoints killrvideo
videos_by_tag 'datastax'
```

`getendpoints` returns the IP addresses of the node(s) which store the partitions with the respective partition key value (the last argument in single quotes: `cassandra` and `datastax` respectively). Notice we must also supply the keyspace and table name we are interested in since we set replication on a per-keyspace basis. There is more on replication to come later in this course.