DATASTAX ACADEMY

Exercise 12 – Replication

In this exercise, you will:

- Understand replication
- Understand how replication aides in reliability

Apache Cassandra™ provides scalability and fault tolerance. Replication is the secret-sauce that makes this work. In this exercise we will add a third node to our cluster and experiment with replication settings.

Steps

1) In your terminal, extract the tarball again to make a third node by executing the following commands (be sure you are in the `/home/ubuntu/` directory).

   ```
   tar -xf dse-6.0.0-bin.tar.gz
   mv dse-6.0.0 node3
   labwork/config_node 3
   ```

2) Ensure that both of your previous two nodes are up. We need to configure this new third node to play nicely with these other nodes already running on your machine.

3) Edit `/home/ubuntu/node3/resources/cassandra/conf/cassandra.yaml`. Change `num_tokens` to 128 and comment out `initial_token`.

4) Change the `endpoint_snitch` to `GossipingPropertyFileSnitch`. Save your changes and exit the editor.

5) Now, edit `/home/ubuntu/node3/resources/cassandra/conf/cassandra-rackdc.properties`. Change the `dc` to `west-side` and the `rack` to `hakuna-matata`. Save and close the file.

6) Start your third node by executing `/home/ubuntu/node3/bin/dse cassandra`

7) Once the node is up, ensure all three nodes are in your cluster by executing `/home/ubuntu/node1/bin/dsetool status`. (You can use any node's dsetool to do this if you like.)

8) Now we will re-import our KillVideo data. Open `cqlsh`. Execute the following CQL CREATE KEYSPACE statement to use `NetworkTopologyStrategy` with replication set to store one replica per data center:

```
CREATE KEYSPACE killrvideo
WITH replication = {
    'class': 'NetworkTopologyStrategy',
    'east-side': 1,
    'west-side': 1
};
```

9) Switch to the `killrvideo` keyspace.

```
USE killrvideo;
```

10) Now let's recreate our `videos_by_tag` table and re-import the data. Execute the following commands:

```
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) Let's determine which nodes our replicas ended up on. Execute the following 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'
```

NOTE: `nodetool` displays the IP addresses of the nodes containing our data.

Notice Apache Cassandra™ stores each replica twice, and each replica is in a different data center. Your results may vary due to randomness in choosing tokens for vnodes.

12) Apache Cassandra™ doesn't have to have an actual partition with a key value to determine which nodes will store that partition. You can try any partition key value you like. For example, try the following:

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