CASSANDRA - ALTER KEYSPACE

Alter Keyspace using Cqlsh

ALTER KEYSPACE can be used to alter properties such as the number of replicas and the durable_writes of a KeySpace. Given below is the syntax of this command.

Syntax

```
ALTER KEYSPACE <identifier> WITH <properties>
```

i.e.

```
ALTER KEYSPACE “KeySpace Name”
WITH replication = {'class': ‘Strategy name’, 'replication_factor' : ‘No.Of  replicas’};
```

The properties of **ALTER KEYSPACE** are same as CREATE KEYSPACE. It has two properties: **replication** and **durable_writes**.

Durable_writes

Using this option, you can instruct Cassandra whether to use commitlog for updates on the current KeySpace. This option is not mandatory and by default, it is set to true.

Example

Given below is an example of altering a KeySpace.

- Here we are altering a KeySpace named **TutorialsPoint**.
- We are changing the replication factor from 1 to 3.

```
cqlsh.> ALTER KEYSPACE tutorialspoint
WITH replication = {'class':'NetworkTopologyStrategy', 'replication_factor' : 3};
```

Altering Durable_writes

You can also alter the durable_writes property of a KeySpace. Given below is the durable_writes property of the **test** KeySpace.

```
SELECT * FROM system.schema_keyspaces;

  keyspace_name | durable_writes |                                       strategy_class
| strategy_options
----------------+----------------+-----------------------------------------------------
--+----------------------------
           test |          False | org.apache.cassandra.locator.NetworkTopologyStrategy |
{"datacenter1":"3"}

 tutorialspoint |           True |          org.apache.cassandra.locator.SimpleStrategy |
{"replication_factor":"4"}

         system |           True |           org.apache.cassandra.locator.LocalStrategy |
{ }

  system_traces |           True |          org.apache.cassandra.locator.SimpleStrategy |
{"replication_factor":"2"}
(4 rows)
```

```
ALTER KEYSPACE test
```

```
WITH REPLICATION = {'class' : 'NetworkTopologyStrategy', 'datacenter1' : 3}
AND DURABLE_WRITES = true;
```

Once again, if you verify the properties of KeySpaces, it will produce the following output.

```
SELECT * FROM system.schema_keyspaces;
  keyspace_name | durable_writes |                                       strategy_class
| strategy_options
----------------+----------------+----------------------------------------------------
--+----------------------------
           test |           True | org.apache.cassandra.locator.NetworkTopologyStrategy |
{"datacenter1":"3"}

 tutorialspoint |           True |          org.apache.cassandra.locator.SimpleStrategy |
{"replication_factor":"4"}

         system |           True |           org.apache.cassandra.locator.LocalStrategy |
{ }

  system_traces |           True |          org.apache.cassandra.locator.SimpleStrategy |
{"replication_factor":"2"}

(4 rows)
```

Altering a Keyspace using Java API

You can alter a keyspace using the **execute** method of **Session** class. Follow the steps given below to alter a keyspace using Java API

Step1: Create a Cluster Object

First of all, create an instance of **Cluster.builder** class of **com.datastax.driver.core** package as shown below.

```
//Creating Cluster.Builder object
Cluster.Builder builder1 = Cluster.builder();
```

Add a contact point *IPaddressofthenode* using the **addContactPoint** method of **Cluster.Builder** object. This method returns **Cluster.Builder**.

```
//Adding contact point to the Cluster.Builder object
Cluster.Builder builder2 = build.addContactPoint( "127.0.0.1" );
```

Using the new builder object, create a cluster object. To do so, you have a method called **build** in the **Cluster.Builder** class. The following code shows how to create a cluster object.

```
//Building a cluster

Cluster cluster = builder.build();
```

You can build the cluster object using a single line of code as shown below.

```
Cluster cluster = Cluster.builder().addContactPoint("127.0.0.1").build();
```

Step 2: Create a Session Object

Create an instance of **Session** object using the **connect** method of **Cluster**class as shown below.

```
Session session = cluster.connect( );
```

This method creates a new session and initializes it. If you already have a keyspace, you can set it to the existing one by passing the keyspace name in string format to this method as shown below.

```
Session session = cluster.connect(“ Your keyspace name ” );
```

Step 3: Execute Query

You can execute CQL queries using the execute method of Session class. Pass the query either in string format or as a **Statement** class object to the execute method. Whatever you pass to this method in string format will be executed on the **cqlsh**.

In this example,

- We are altering a keyspace named **tp**. We are altering the replication option from Simple Strategy to Network Topology Strategy.
- We are altering the **durable_writes** to false

You have to store the query in a string variable and pass it to the execute method as shown below.

```
//Query
String query = "ALTER KEYSPACE tp WITH replication " + "=
{'class':'NetworkTopologyStrategy', 'datacenter1':3}" +" AND DURABLE_WRITES = false;";
session.execute(query);
```

Given below is the complete program to create and use a keyspace in Cassandra using Java API.

```
import com.datastax.driver.core.Cluster;
import com.datastax.driver.core.Session;

public class Alter_KeySpace {
   public static void main(String args[]){

      //Query
      String query = "ALTER KEYSPACE tp WITH replication " + "=
{'class':'NetworkTopologyStrategy', 'datacenter1':3}"
         + "AND DURABLE_WRITES = false;";

      //Creating Cluster object
      Cluster cluster = Cluster.builder().addContactPoint("127.0.0.1").build();

      //Creating Session object
      Session session = cluster.connect();

      //Executing the query
      session.execute(query);

      System.out.println("Keyspace altered");
   }
}
```

Save the above program with the class name followed by .java, browse to the location where it is saved. Compile and execute the program as shown below.

```
$javac Alter_KeySpace.java
$java Alter_KeySpace
```

Under normal conditions, it produces the following output:

```
Keyspace Altered
```

Loading [MathJax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js