Executing Batch Statements Using Cqlsh

Using **BATCH,** you can execute multiple modification statements *insert*, *update*, *delete* simultaneiously. Its syntax is as follows:

```
BEGIN BATCH
<insert-stmt>/ <update-stmt>/ <delete-stmt>
APPLY BATCH
```

Example

Assume there is a table in Cassandra called emp having the following data:

emp_id	emp_name	emp_city	emp_phone	emp_sal
1	ram	Hyderabad	9848022338	50000
2	robin	Delhi	9848022339	50000
3	rahman	Chennai	9848022330	45000

In this example, we will perform the following operations:

- Insert a new row with the following details 4, *rajeev*, *pune*, 9848022331, 30000.
- Update the salary of employee with row id 3 to 50000.
- Delete city of the employee with row id 2.

To perform the above operations in one go, use the following BATCH command:

```
cqlsh:tutorialspoint> BEGIN BATCH
... INSERT INTO emp (emp_id, emp_city, emp_name, emp_phone, emp_sal) values(
4,'Pune','rajeev',9848022331, 30000);
... UPDATE emp SET emp_sal = 50000 WHERE emp_id =3;
... DELETE emp_city FROM emp WHERE emp_id = 2;
... APPLY BATCH;
```

Verification

After making changes, verify the table using the SELECT statement. It should produce the following output:

```
cqlsh:tutorialspoint> select * from emp;

 emp_id |  emp_city | emp_name |  emp_phone | emp_sal
--------+-----------+----------+------------+---------
      1 | Hyderabad | ram      | 9848022338 | 50000
      2 | null      | robin    | 9848022339 | 50000
      3 | Chennai   | rahman   | 9848022330 | 50000
      4 | Pune      | rajeev   | 9848022331 | 30000

(4 rows)
```

Here you can observe the table with modified data.

Batch Statements using Java API

Batch statements can be written programmatically in a table using the execute method of Session class. Follow the steps given below to execute multiple statements using batch statement with the help of Java API.

Step1: Create a Cluster Object

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. Use the following code to create the 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, then 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 ”);
```

Here we are using the KeySpace named **tp**. Therefore, create the session object as shown below.

```
Session session = cluster.connect(“tp”);
```

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 will perform the following operations:

- Insert a new row with the following details 4, *rajeev*, *pune*, 9848022331, 30000.
- Update the salary of employee with row id 3 to 50000.
- Delete the city of the employee with row id 2.

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

```
String query1 = ” BEGIN BATCH INSERT INTO emp (emp_id, emp_city, emp_name,
emp_phone, emp_sal) values( 4,'Pune','rajeev',9848022331, 30000);
```

```
UPDATE emp SET emp_sal = 50000 WHERE emp_id =3;
DELETE emp_city FROM emp WHERE emp_id = 2;
APPLY BATCH;”;
```

Given below is the complete program to execute multiple statements simultaneously on a table in Cassandra using Java API.

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

public class Batch {

   public static void main(String args[]){

      //query
      String query =" BEGIN BATCH INSERT INTO emp (emp_id, emp_city,
         emp_name, emp_phone, emp_sal) values( 4,'Pune','rajeev',9848022331, 30000);"

         + "UPDATE emp SET emp_sal = 50000 WHERE emp_id =3;"
         + "DELETE emp_city FROM emp WHERE emp_id = 2;"
         + "APPLY BATCH;";

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

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

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

      System.out.println("Changes done");
   }
}
```

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 Batch.java
$java Batch
```

Under normal conditions, it should produce the following output:

```
Changes done
```