CASSANDRA - READ DATA

Reading Data using Select Clause

SELECT clause is used to read data from a table in Cassandra. Using this clause, you can read a whole table, a single column, or a particular cell. Given below is the syntax of SELECT clause.

```
SELECT FROM <tablename>
```

Example

Assume there is a table in the keyspace named **emp** with the following details:

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

The following example shows how to read a whole table using SELECT clause. Here we are reading a table called **emp**.

```
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)
```

Reading Required Columns

The following example shows how to read a particular column in a table.

```
cqlsh:tutorialspoint> SELECT emp_name, emp_sal from emp;

 emp_name | emp_sal
----------+---------
      ram | 50000
    robin | 50000
   rajeev | 30000
   rahman | 50000

(4 rows)
```

Where Clause

Using WHERE clause, you can put a constraint on the required columns. Its syntax is as follows:

```
SELECT FROM <table name> WHERE <condition>;
```

Note: A WHERE clause can be used only on the columns that are a part of primary key or have a secondary index on them.

In the following example, we are reading the details of an employee whose salary is 50000. First of all, set secondary index to the column emp_sal.

```
cqlsh:tutorialspoint> CREATE INDEX ON emp(emp_sal);
cqlsh:tutorialspoint> SELECT * FROM emp WHERE emp_sal=50000;

 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
```

Reading Data using Java API

You can read data from 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 called **tp**. Therefore, create the session object as shown below.

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

Step 3: Execute Query

You can execute CQL queries using 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 retrieving the data from **emp** table. Store the query in a string and pass it to the execute method of session class as shown below.

```
String query = ”SELECT 8 FROM emp”;
session.execute(query);
```

Execute the query using the execute method of Session class.

Step 4: Get the ResultSet Object

The select queries will return the result in the form of a **ResultSet** object, therefore store the result in the object of **RESULTSET** class as shown below.

```
ResultSet result = session.execute( );
```

Given below is the complete program to read data from a table.

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

public class Read_Data {

   public static void main(String args[])throws Exception{

      //queries
      String query = "SELECT * FROM emp";

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

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

      //Getting the ResultSet
      ResultSet result = session.execute(query);

      System.out.println(result.all());
   }
}
```

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 Read_Data.java
$java Read_Data
```

Under normal conditions, it should produce the following output:

```
[Row[1, Hyderabad, ram, 9848022338, 50000], Row[2, Delhi, robin,
9848022339, 50000], Row[4, Pune, rajeev, 9848022331, 30000], Row[3,
Chennai, rahman, 9848022330, 50000]]
```