DATASTAX ACADEMY

Exercise 3.3 – Built Statements and Query Builder

This exercise is designed to familiarize you with the basic code required to execute queries using Built Statements and the Query Builder to read data from and write data to DSE.

In this exercise, you will use add additional code to the `addVideo()` function of the `CassandraVideoDAO` class. This new code will use the Query Builder functionality to write information about the "Pirates of the Caribbean" video to the *latest_videos* table. You will then add code to read that data.

This exercise uses two (2) files in the *Session 2* exercise project:

- *CassandraVideoDAO.java:* The file that defines the class responsible for reading and writing data to the *video* table. Additionally, by the end of this exercise it will feature a query builder statement that will write data to the *latest_videos* table as well as a query builder statement in method `getLatestVideos()` that will read the data in the table.

- *QueryBuilderVideoDAOTest.java:* This file defines a class that first simulates data being received from a webpage and then initiates the process of writing said data to the *video* table via the `addVideo()` function of a `CassandraVideoDAO` instance. Additionally, it executes code that uses the `getLatestVideos()` method in `CassandraVideoDAO` to fetch the latest videos for display on the website.

All steps will have most of the required code already written out. Your task is to place the additional code where it is needed and understand how to create and test a Query Builder query. Here is a brief summary of the steps you will need to perform:

STEPS:

Step 1: Examine the files *CassandraVideoDAO.java* and *QueryBuilderVideoDAOTest.java*
Step 2: Truncate the *video* and *latest_video* tables and add Query Builder code to the `addVideo()` method
Step 3: Add code to the `testAddLatestVideo()` method to read data from the *latest_video* table
Step 4: Add a new video and confirm it is present

Step 1: Examine CassandraVideoDAO.java and QueryBuilderVideoDAOTest.java

The purpose of this step is for you to become familiar with the code you are running so you understand what is happening and why it's happening.

1. Launch the IDE and open the *killrvideo session2* project, if necessary.

2. In the Theia editor, navigate to and open both file:

   ~/session2/src/main/java/com/datastax/training/killrvideo/model/dao/cassandra/CassandraVideoDAO.java.

   ~/session2/src/test/java/com/datastax/training/killrvideo/dao/cassandra/QueryBuilderVideoDAOTest.java

3. Within a terminal, back up both files per the instructions in the previous exercises.

4. Open the file *QueryBuilderVideoDAOTest.java*. Note that the `testAddLatestVideo()` method features the same three (3) lines of code found in the `testAddVideo()` method in *PreparedStatementVideoDAOTest.java*.

```
Video originalVideo = createVideo();

CassandraVideoDAO videoDAO = new CassandraVideoDAO();
videoDAO.addVideo(originalVideo);
```

5. After these three (3) lines, the file contains additional lines of code that will use the `getLatestVideos()` function to fetch the latest videos for display on the website:

```
// getVideos
LatestVideo latestOriginal = new LatestVideo(originalVideo);
Iterable<LatestVideo> videos = videoDAO.getLatestVideos();
boolean found = false;
for (LatestVideo retrievedVideo: videos) {
    if(retrievedVideo.getVideoId().equals(latestOriginal.getVideoId()))
{
        assertEquals(latestOriginal, retrievedVideo);
        found = true;
        break;
    }
}
assertTrue(found);
```

6. Examine the constructor for the file *CassandraVideoDAO.java*. These lines are located at the end of the method and are used to set the class's private variable *current_date*, which will be used by the Query Builder Statement:

```
Date date = new Date();
    Calendar cal = Calendar.getInstance();
    cal.setTime(date);
    currentDate = cal.get(Calendar.YEAR)*10000 +
(cal.get(Calendar.MONTH)+1)*100 + cal.get(Calendar.DAY_OF_MONTH);
```

7. Note the empty `getLatestVideo()` method. We will add a Query Builder statement to make this return a list of latest videos.

8. Also, note the table schema for *latest_videos* below. This is the table we will be populating with the Query Builder code.

```
    CREATE TABLE killrvideo_test.latest_videos (
    video_bucket int,
    video_id timeuuid,
    preview_thumbnail blob,
    tags set<text>,
    title text,
    type text,
    PRIMARY KEY (video_bucket, video_id)
)
```

Step 2: Truncate video and latest_video tables and add Query Builder code to the addVideo() method

The purpose of this step is to add the Query Builder code in order for the *latest_videos* table to be updated when a new video is added.

1. Go to the CQL shell and truncate both the *videos* and *latest_videos* tables. Run select all statements to confirm both are truncated.

```
cqlsh:killrvideo> TRUNCATE VIDEOS;
cqlsh:killrvideo> TRUNCATE latest_videos ;
cqlsh:killrvideo> SELECT * from videos;

 video_id | avg_rating | description | genres | mpaa_rating | preview_thumbnail | release_date | release_year | tags | title | type | url | user_id
----------+------------+-------------+--------+-------------+-------------------+--------------+--------------+------+-------+------+-----+---------

(0 rows)
cqlsh:killrvideo> SELECT * from latest_videos;

 video_bucket | video_id | preview_thumbnail | tags | title | type
--------------+----------+-------------------+------+-------+------

(0 rows)
cqlsh:killrvideo>
```

2. In the IDE, navigate to the following directory: *~/session2/src/main/java/com/datastax/training/killrvideo/model/dao/cassandra/CassandraVideoDAO.java*

3. Find the `addVideo()` method implemented earlier.

4. Add this code **between** the code previously added in the last exercise and the "`//TODO: Your added code ends here`" line:

```
// TODO: Insert your code here

...[previously inserted code]...

// Adding newVideo object and currentDate variable
Insert insertToLatestVideos = QueryBuilder.insertInto("latest_videos")
        .value("video_bucket", currentDate)
        .value("video_id", newVideo.getVideoId())
        .value("title", newVideo.getTitle())
        .value("type", newVideo.getType())
        .value("tags", newVideo.getTags())
        .value("preview_thumbnail", newVideo.getPreviewThumbnail());

        session.execute(insertToLatestVideos);

//TODO: Your added code ends here
```

This code creates an *insert* statement that packages up the required parameter values from the `newVideo` object used in the last exercise. It also uses the `currentDate` private variable to set the `video_bucket` value. Then it is passed to `session.execute()` so the insertion can be carried out.

Step 3: Add code to the getLatestVideos() method to read data from latest_video.

The purpose of this step is to add the code to `testADDLatestVideo()` so the data can be displayed on the website. It will also be implemented as a Query Builder statement.

1. Find the mostly empty `getLatestVideos()` method and insert the code below.

```
@Override
public Iterable<LatestVideo> getLatestVideos() {
    ArrayList<LatestVideo> latestVideos = new ArrayList<LatestVideo>();

    // TODO: Insert your code here

    Statement selectLatestVideos = QueryBuilder
        .select()
        .from("latest_videos")
        .where(QueryBuilder.eq("video_bucket", currentDate))
        .limit(12);

    ResultSet rs = getCassandraSession().execute(selectLatestVideos);

    for (Row row : rs) {
        LatestVideo newVideo = new LatestVideo();
        newVideo.setVideoId(row.getUUID("video_id"));
        newVideo.setTitle(row.getString("title"));
        newVideo.setType(row.getString("type"));
        newVideo.setTags(row.getSet("tags", String.class));

newVideo.setPreviewThumbnail(row.getBytes("preview_thumbnail"));

        latestVideos.add(newVideo);

    }

    //TODO: Your added code ends here
```

Note that the first part is a query builder statement for selecting 12 of the latest videos. It also uses the `currentDate` private variable to obtain the latest videos for the current date only.

Note the second half of the code is a “for-each loop” that populates an array called `latestVideos`, which was already dimensioned before adding this code. The array will be returned to the `calling` function when this method ends.

Your code should appear as follows in your IDE editor:

```java
@Override
public Iterable<LatestVideo> getLatestVideos() {
    ArrayList<LatestVideo> latestVideos = new ArrayList<LatestVideo>();

    // TODO: Insert your code here

Statement selectLatestVideos = QueryBuilder
    .select()
    .from("latest_videos")
    .where(QueryBuilder.eq("video_bucket", currentDate))
    .limit(12);

ResultSet rs = getCassandraSession().execute(selectLatestVideos);

for (Row row : rs) {
    LatestVideo newVideo = new LatestVideo();
    newVideo.setVideoId(row.getUUID("video_id"));
    newVideo.setTitle(row.getString("title"));
    newVideo.setType(row.getString("type"));
    newVideo.setTags(row.getSet("tags", String.class));
    newVideo.setPreviewThumbnail(row.getBytes("preview_thumbnail"));

    latestVideos.add(newVideo);

}


    //TODO: Your added code ends here
```

Step 4: Add a video and verify

1. In your IDE editor, navigate to the following location: *~/session2/src/test/java/com/datastax/training/killrvideo/dao/cassandra/QueryBuilderVideoDAOTest.java*

2. Note the following code:

```
// getVideos
LatestVideo latestOriginal = new LatestVideo(originalVideo);
Iterable<LatestVideo> videos = videoDAO.getLatestVideos();
boolean found = false;
for (LatestVideo retrievedVideo: videos) {
    if(retrievedVideo.getVideoId().equals(latestOriginal.getVideoId()))
{
        assertEquals(latestOriginal, retrievedVideo);
        found = true;
        break;
    }
}
```

   a. This code uses the `getLatestVideos()` function just completed to fetch all of the latest videos. Then it loops through and checks to see if the video you just inserted ("Pirates of the Caribbean") is in the list. If it is the code breaks.

3. Put a breakpoint on this line at the end of the method: `assertTrue(found);`

   a. Add a breakpoint by pressing Ctrl+F8 or by clicking in the left gutter of that line and adding a red dot, i.e. breakpoint.

4. Go to *Run>>Debug 'QueryBuilderVideoDAOTest'*.

5. When the process stops at your breakpoint, proceed to the CQL shell, and confirm both tables have one record each in them.

6. Also, examine the `latestOriginal` object in the variables pane to verify that it is indeed the "Pirates of the Caribbean" video.