L1_Exercise_2_Creating_a_Table_with_Apache_Cassandra

January 17, 2023

1 Lesson 1 Exercise 2: Creating a Table with Apache Cassandra

Walk through the basics of Apache Cassandra. Complete the following tasks:
Create a table in Apache Cassandra,
Insert rows of data,
Run a simple SQL query to validate the information. ##### denotes where the code needs to be completed.

Import Apache Cassandra python package

```
In [3]: import cassandra
```

1.0.1 Create a connection to the database

```
In [4]: from cassandra.cluster import Cluster
        try:
            cluster = Cluster(['127.0.0.1']) #If you have a locally installed Apache Cassandra i
            session = cluster.connect()
        except Exception as e:
            print(e)
```

1.0.2 TO-DO: Create a keyspace to do the work in

```
In [8]: ## TO-DO: Create the keyspace
        try:
            session.execute("""
            CREATE KEYSPACE IF NOT EXISTS udacity
            WITH REPLICATION =
            { 'class' : 'SimpleStrategy', 'replication_factor' : 1 }"""
        )

        except Exception as e:
            print(e)
```

1.0.3 TO-DO: Connect to the Keyspace

```
In [12]: ## To-Do: Add in the keyspace you created
         try:
             session.set_keyspace('udacity')
         except Exception as e:
             print(e)
```

1.0.4 Create a Song Library that contains a list of songs, including the song name, artist name, year, album it was from, and if it was a single.

```
song_title artist_name year album_name single
```

1.0.5 TO-DO: You need to create a table to be able to run the following query:

```
select * from songs WHERE year=1970 AND artist_name="The Beatles"
```

```
In [13]: ## TO-DO: Complete the query below
         query = "CREATE TABLE IF NOT EXISTS song_library"
         query = query + "(song_title text, artist_name text, year int, album_name text, single
         try:
             session.execute(query)
         except Exception as e:
             print(e)
```

1.0.6 TO-DO: Insert the following two rows in your table

```
First Row: "1970", "Let It Be", "The Beatles", "Across The Universe", "False",
   Second Row: "1965", "Think For Yourself", "The Beatles", "Rubber Soul", "False"
```

```
In [15]: ## Add in query and then run the insert statement
         query = "INSERT INTO song_library (year, album_name, artist_name, song_title, single)"
         query = query + " VALUES (%s, %s, %s, %s, %s)"

         try:
             session.execute(query, (1970, "Let It Be", "The Beatles", "Across The Universe", Fa
         except Exception as e:
             print(e)

         try:
             session.execute(query, (1965, "Think For Yourself", "The Beatles", "Rubber Soul", F
         except Exception as e:
             print(e)
```

1.0.7 TO-DO: Validate your data was inserted into the table.

```
In [16]: ## TO-DO: Complete and then run the select statement to validate the data was inserted
         query = 'SELECT * FROM song_library'
         try:
```

```
            rows = session.execute(query)
        except Exception as e:
            print(e)

        for row in rows:
            print (row.year, row.album_name, row.artist_name)
```

```
1965 Think For Yourself The Beatles
1970 Let It Be The Beatles
```

1.0.8 TO-DO: Validate the Data Model with the original query.

```
select * from songs WHERE YEAR=1970 AND artist_name="The Beatles"
```

```
In [19]: ##TO-DO: Complete the select statement to run the query
         query = "SELECT * FROM song_library WHERE year=1970 AND artist_name='The Beatles'"
         try:
             rows = session.execute(query)
         except Exception as e:
             print(e)

         for row in rows:
             print (row.year, row.album_name, row.artist_name)
```

```
1970 Let It Be The Beatles
```

1.0.9 And Finally close the session and cluster connection

```
In [20]: session.shutdown()
         cluster.shutdown()
```