Lesson 3 Exercise 3 Clustering Column

January 28, 2023

1 Lesson 3 Exercise 3: Focus on Clustering Columns

1.0.1 Walk through the basics of creating a table with a good Primary Key and Clustering Columns in Apache Cassandra, inserting rows of data, and doing a simple CQL query to validate the information.

1.0.2 Remember, replace ##### with your own code.

We will use a python wrapper/ python driver called cassandra to run the Apache Cassandra queries. This library should be preinstalled but in the future to install this library you can run this command in a notebook to install locally: ! pip install cassandra-driver #### More documentation can be found here: https://datastax.github.io/python-driver/

Import Apache Cassandra python package

```
In [1]: import cassandra
```

1.0.3 Create a connection to the database

```
In [2]: 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)
```

1.0.4 Create a keyspace to work in

```
In [3]: try:
            session.execute("""
            CREATE KEYSPACE IF NOT EXISTS udacity
            WITH REPLICATION =
            { 'class' : 'SimpleStrategy', 'replication_factor' : 1 }"""
        )

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

Connect to the Keyspace. Compare this to how we had to create a new session in PostgreSQL.

```
In [4]: try:
            session.set_keyspace('udacity')
        except Exception as e:
            print(e)
```

1.0.5 Imagine we would like to start creating a new Music Library of albums.

1.0.6 We want to ask 1 question of our data:

1.0.7 1. Give me all the information from the music library about a given album

`select * from album_library WHERE album_name="Close To You"`

1.0.8 Here is the data:

1.0.9 How should we model this data? What should be our Primary Key and Partition Key?

```
In [5]: query = "CREATE TABLE IF NOT EXISTS album_library "
        query = query + "(year int, artist_name text, album_name text, city text, PRIMARY KEY (a
        try:
            session.execute(query)
        except Exception as e:
            print(e)
```

1.0.10 Insert data into the table

```
In [8]: ## You can opt to change the sequence of columns to match your composite key. \
        ## If you do, make sure to match the values in the INSERT statement

        query = "INSERT INTO album_library (year, artist_name, album_name, city)"
        query = query + " VALUES (%s, %s, %s, %s)"

        try:
            session.execute(query, (1970, "The Beatles", "Let it Be", "Liverpool"))
        except Exception as e:
            print(e)

        try:
            session.execute(query, (1965, "The Beatles", "Rubber Soul", "Oxford"))
        except Exception as e:
            print(e)

        try:
            session.execute(query, (1964, "The Beatles", "Beatles For Sale", "London"))
        except Exception as e:
            print(e)

        try:
            session.execute(query, (1966, "The Monkees", "The Monkees", "Los Angeles"))
        except Exception as e:
            print(e)

        try:
            session.execute(query, (1970, "The Carpenters", "Close To You", "San Diego"))
        except Exception as e:
            print(e)
```

1.0.11 Validate the Data Model -- Did it work?

`select * from album_library WHERE album_name="Close To You"`

```
In [12]: query = "select * from album_library WHERE album_name='Close To You'"
         try:
             rows = session.execute(query)
         except Exception as e:
             print(e)

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

```
The Carpenters Close To You San Diego 1970
```

1.0.12 Your output should be:

('The Carpenters', 'Close to You', 'San Diego', 1970)

1.0.13 OR

('The Carpenters', 'Close to You', 1970, 'San Diego')

1.0.14 Drop the table

```
In [14]: query = "drop table album_library"
         try:
             rows = session.execute(query)
         except Exception as e:
             print(e)
```

1.0.15 Close the session and cluster connection

```
In [15]: session.shutdown()
         cluster.shutdown()
```

```
In [ ]:
```