

# Lesson 3 Exercise 2 Primary Key

January 28, 2023

## 1 Lesson 3 Exercise 2: Focus on Primary Key

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

1.0.2 Replace ##### with your own answers.

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 [71]: import cassandra
```

1.0.3 Create a connection to the database

```
In [72]: from cassandra.cluster import Cluster
        try:
            cluster = Cluster(['127.0.0.1']) #If you have a locally installed Apache Cassandra
            session = cluster.connect()
        except Exception as e:
            print(e)
```

1.0.4 Create a keyspace to work in

```
In [73]: 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 [74]: try:
        session.set_keyspace('udacity')
    except Exception as e:
        print(e)
```

**1.0.5 Imagine you need to create a new Music Library of albums**

**1.0.6 Here is the information asked of the data:**

**1. Give every album in the music library that was created by a given artist** `select * from music_library WHERE artist_name="The Beatles"`

**1.0.7 Here is the collection of data**

**Practice by making the PRIMARY KEY only 1 Column (not 2 or more)**

```
In [76]: query = "DROP TABLE music_library"
        try:
            session.execute(query)
        except Exception as e:
            print(e)

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

**1.0.8 Let's insert the data into the table**

```
In [77]: query = "INSERT INTO music_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, (1965, "The Who", "My Generation", "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)

```

### 1.0.9 Validate the Data Model -- Does it give you two rows?

```

In [78]: query = "select * from music_library WHERE artist_name='The Beatles'"
try:
    rows = session.execute(query)
except Exception as e:
    print(e)

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

```

1965 The Beatles Rubber Soul Oxford

### 1.0.10 If you used just one column as your PRIMARY KEY, your output should be:

1965 The Beatles Rubber Soul Oxford

### 1.0.11 That didn't work out as planned! Why is that? Did you create a unique primary key?

### 1.0.12 Try again - Create a new table with a composite key this time

```

In [79]: query = "DROP TABLE music_library2"
try:
    session.execute(query)
except Exception as e:
    print(e)

query = "CREATE TABLE IF NOT EXISTS music_library2 "
query = query + "(year int, artist_name text, album_name text, city text, PRIMARY KEY (
try:
    session.execute(query)
except Exception as e:
    print(e)

```

Error from server: code=2200 [Invalid query] message="unconfigured table music\_library2"

```

In [80]: ## You can opt to change the sequence of columns to match your composite key. |
        ## Make sure to match the values in the INSERT statement

query = "INSERT INTO music_library2 (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, (1965, "The Who", "My Generation", "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)

```

### 1.0.13 Validate the Data Model -- Did it work?

```

In [81]: query = "select * from music_library2 WHERE artist_name='The Beatles'"
try:
    rows = session.execute(query)
except Exception as e:
    print(e)

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

```

1970 The Beatles Let it Be Liverpool

1965 The Beatles Rubber Soul Oxford

### 1.0.14 Your output should be:

1970 The Beatles Let it Be Liverpool 1965 The Beatles Rubber Soul Oxford

### 1.0.15 Drop the tables

```
In [82]: query = "DROP TABLE music_library"
        try:
            rows = session.execute(query)
        except Exception as e:
            print(e)

        query = "DROP TABLE music_library2"
        try:
            rows = session.execute(query)
        except Exception as e:
            print(e)
```

### 1.0.16 Close the session and cluster connection

```
In [83]: session.shutdown()
        cluster.shutdown()
```

```
In [ ]:
```