L1_Exercise_1_Creating_a_Table_with_Postgres

January 6, 2023

1 Lesson 1 Exercise 1: Creating a Table with PostgreSQL

Walk through the basics of PostgreSQL. You will need to complete the following tasks:
Create a table in PostgreSQL,
Insert rows of data
Run a simple SQL query to validate the information. ##### denotes where the code needs to be completed.

Import the library *Note:* An error might popup after this command has executed. If it does, read it carefully before ignoring.

```
In [1]: import psycopg2
```

```
In [2]: !echo "alter user student createdb;" | sudo -u postgres psql
```

```
ALTER ROLE
```

1.0.1 Create a connection to the database

```
In [3]: try:
            conn = psycopg2.connect("host=127.0.0.1 dbname=studentdb user=student password=stude
        except psycopg2.Error as e:
            print("Error: Could not make connection to the Postgres database")
            print(e)
```

1.0.2 Use the connection to get a cursor that can be used to execute queries.

```
In [4]: try:
            cur = conn.cursor()
        except psycopg2.Error as e:
            print("Error: Could not get curser to the Database")
            print(e)
```

1.0.3 TO-DO: Set automatic commit to be true so that each action is committed without having to call conn.commit() after each command.

```
In [5]: # TO-DO: set automatic commit to be true
        conn.set_session(autocommit=True)
```

1.0.4 TO-DO: Create a database to do the work in.

```
In [6]: ## TO-DO: Add the database name within the CREATE DATABASE statement. You can choose you
        try:
            cur.execute("create database stack")
        except psycopg2.Error as e:
            print(e)
```

TO-DO: Add the database name in the connect statement. Let's close our connection to the default database, reconnect to the Udacity database, and get a new cursor.

```
In [7]: ## TO-DO: Add the database name within the connect statement
        try:
            conn.close()
        except psycopg2.Error as e:
            print(e)

        try:
            conn = psycopg2.connect("host=127.0.0.1 dbname=stack user=student password=student")
        except psycopg2.Error as e:
            print("Error: Could not make connection to the Postgres database")
            print(e)

        try:
            cur = conn.cursor()
        except psycopg2.Error as e:
            print("Error: Could not get curser to the Database")
            print(e)

        conn.set_session(autocommit=True)
```

1.0.5 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
```

```
In [8]: ## TO-DO: Finish writing the CREATE TABLE statement with the correct arguments
        try:
            cur.execute("CREATE TABLE IF NOT EXISTS songs (song_title varchar, \
                        artist_name varchar, \
                        year int, \
                        album_name varchar, \
                        single boolean);")
        except psycopg2.Error as e:
            print("Error: Issue creating table")
            print (e)
```

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

```
First Row:  "Across The Universe", "The Beatles", "1970", "Let It Be", "False"
```

Second Row: "Think For Yourself", "The Beatles", "1965", "Rubber Soul", "False"

In [9]:
```
## TO-DO: Finish the INSERT INTO statement with the correct arguments

        try:
            cur.execute("INSERT INTO songs (song_title, artist_name, year, album_name, single) \
                         VALUES (%s, %s, %s, %s, %s)", \
                         ("Across The Universe", "The Beatles", 1970, "Let It Be", False))
        except psycopg2.Error as e:
            print("Error: Inserting Rows")
            print (e)

        try:
            cur.execute("INSERT INTO songs (song_title, artist_name, year, album_name, single) \
                          VALUES (%s, %s, %s, %s, %s)",
                          ("Think For Yourself", "The Beatles", 1965, "Rubber Soul", False))
        except psycopg2.Error as e:
            print("Error: Inserting Rows")
            print (e)
```

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

In [10]:
```
## TO-DO: Finish the SELECT * Statement
        try:
            cur.execute("SELECT * FROM songs")
        except psycopg2.Error as e:
            print("Error: select *")
            print (e)

        row = cur.fetchone()
        while row:
           print(row)
           row = cur.fetchone()
```

```
('Across The Universe', 'The Beatles', 1970, 'Let It Be', False)
('Think For Yourself', 'The Beatles', 1965, 'Rubber Soul', False)
```

1.0.8 And finally close your cursor and connection.

In [11]:
```
cur.close()
conn.close()
```

In [ ]: