Prerequisites:

To complete this lab, you must have access to the little\_lemon database. As an authorized user, you need to establish a connection between Python and the database via the MySQL Connector/Python API using the following code:

!pip install mysql-connector-python

import mysql.connector as connector

connection=connector.connect(user="your\_username",password="your\_password")

Learning objectives:

* Create a cursor object
* Use a cursor object to make use of a MySQL database using Python

Scenario:

Little Lemon needs to perform some basic tasks on its databases such as setting up the database and checking the names of the tables in the database. For this purpose, they have established a connection with the MySQL database using Python. In order to perform a task they need to communicate with the database.

You are tasked to help Little Lemon set up their database in use and confirm the existence of tables to perform tasks. This needs to happen according to their requirements in their Python-based application.

Task 1:

Little Lemon need to know what tables currently exist in the database. You need to help them retrieve the names of all the existing tables in their database.

To access the names of the existing tables in the Little Lemon database, set the database little\_lemon in use. Then, create a cursor object and execute SHOW TABLES to retrieve the names of the tables in the database. Fetch all the names in a variable and use the for loop to print the output.

TIP: Create the standard cursor object with the default set of arguments. This is done by invoking the cursor module on the connection that works with the MySQL database using Python.

Task 2:

Creating the cursor is an important step for communicating with the entire MySQL database using Python.

You have learned about the different approaches for creating cursors. Which approach you use depends on your application and resource optimization.

Run a test between the standard and the buffered cursor to check what type of cursor will work for the situation given below:

Create a cursor

* Execute USE little\_lemon
* Execute SELECT \* FROM Bookings
* Execute SELECT \* FROM Orders

Task 3:

Little Lemon will soon have multiple databases. They need to plan for a scalable solution.  This information can be tracked in a Python dictionary. A dictionary cursor is helpful as it returns a dictionary object.

Create a cursor with the argument [dictionary = True] and retrieve the names of all the tables in the form of a dictionary object where the name of the tables is a value, and the database name is a key.

TIP: Explore the arguments that you can pass to the cursor module.

**Solution**

# Establish connection between Python and MySQL database via connector API

connection=connector.connect(

user="root", # use your own

password="", # use your own

)

# Create cursor object to communicate with entire MySQL database

cursor = connection.cursor()

# If exist, drop the database first, and create again

try:

cursor.execute("CREATE DATABASE little\_lemon")

except:

cursor.execute("drop database little\_lemon")

cursor.execute("CREATE DATABASE little\_lemon")

print("The database little\_lemon is created.\n")

# Set little\_lemon database for use

cursor.execute("USE little\_lemon")

print("The database little\_lemon is set for use.\n")

# The SQL query for MenuItems table is:

create\_menuitem\_table="""

CREATE TABLE MenuItems (

ItemID INT AUTO\_INCREMENT,

Name VARCHAR(200),

Type VARCHAR(100),

Price INT,

PRIMARY KEY (ItemID)

);"""

# Create MenuItems table

cursor.execute(create\_menuitem\_table)

print("MenuItmes table is created.\n")

# The SQL query for Menu table is:

create\_menu\_table="""

CREATE TABLE Menus (

MenuID INT,

ItemID INT,

Cuisine VARCHAR(100),

PRIMARY KEY (MenuID,ItemID)

);"""

# Create Menu table

cursor.execute(create\_menu\_table)

print("Menu table is created.\n")

# The SQL query for Bookings table is:

create\_booking\_table="""

CREATE TABLE Bookings (

BookingID INT AUTO\_INCREMENT,

TableNo INT,

GuestFirstName VARCHAR(100) NOT NULL,

GuestLastName VARCHAR(100) NOT NULL,

BookingSlot TIME NOT NULL,

EmployeeID INT,

PRIMARY KEY (BookingID)

);"""

# Create Bookings table

cursor.execute(create\_booking\_table)

print("Bookings table is created.\n")

# The SQL query for Bookings table is:

create\_orders\_table="""

CREATE TABLE Orders (

OrderID INT,

TableNo INT,

MenuID INT,

BookingID INT,

BillAmount INT,

Quantity INT,

PRIMARY KEY (OrderID,TableNo)

);"""

# Create Orders table

cursor.execute(create\_orders\_table)

print("Orders table is created.\n")

# Confirm if the tables are created

print("Following tables are created in the little\_lemon database.\n")

cursor.execute("SHOW TABLES")

for table in cursor:

print(table)