## #install connector api using the command below.

!pip install mysql-connector-python

# Import the MySQL Connector/Python

import mysql.connector as connector

# Establish connection between Python and MySQL database via connector API

connection=connector.connect(

user="root", # use your own

password="", # use your own

)

print("Connection between MySQL and Python is established.\n")

# Create cursor object to communicate with entire MySQL database

cursor = connection.cursor()

print("Cursor is created to communicate with the MySQL using Python.\n")

# 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")

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

# Insert query to populate "MenuItems" table is:

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

insert\_menuitmes="""

INSERT INTO MenuItems (ItemID, Name, Type, Price)

VALUES

(1,'Olives','Starters',5),

(2,'Flatbread','Starters', 5),

(3, 'Minestrone', 'Starters', 8),

(4, 'Tomato bread','Starters', 8),

(5, 'Falafel', 'Starters', 7),

(6, 'Hummus', 'Starters', 5),

(7, 'Greek salad', 'Main Courses', 15),

(8, 'Bean soup', 'Main Courses', 12),

(9, 'Pizza', 'Main Courses', 15),

(10,'Greek yoghurt','Desserts', 7),

(11, 'Ice cream', 'Desserts', 6),

(12, 'Cheesecake', 'Desserts', 4),

(13, 'Athens White wine', 'Drinks', 25),

(14, 'Corfu Red Wine', 'Drinks', 30),

(15, 'Turkish Coffee', 'Drinks', 10),

(16, 'Turkish Coffee', 'Drinks', 10),

(17, 'Kabasa', 'Main Courses', 17);"""

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

# Insert query to populate "Menu" table is:

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

insert\_menu="""

INSERT INTO Menus (MenuID,ItemID,Cuisine)

VALUES

(1, 1, 'Greek'),

(1, 7, 'Greek'),

(1, 10, 'Greek'),

(1, 13, 'Greek'),

(2, 3, 'Italian'),

(2, 9, 'Italian'),

(2, 12, 'Italian'),

(2, 15, 'Italian'),

(3, 5, 'Turkish'),

(3, 17, 'Turkish'),

(3, 11, 'Turkish'),

(3, 16, 'Turkish');"""

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

# Insert query to populate "Bookings" table is:

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

insert\_bookings="""

INSERT INTO Bookings (BookingID, TableNo, GuestFirstName,

GuestLastName, BookingSlot, EmployeeID)

VALUES

(1,12,'Anna','Iversen','19:00:00',1),

(2, 12, 'Joakim', 'Iversen', '19:00:00', 1),

(3, 19, 'Vanessa', 'McCarthy', '15:00:00', 3),

(4, 15, 'Marcos', 'Romero', '17:30:00', 4),

(5, 5, 'Hiroki', 'Yamane', '18:30:00', 2),

(6, 8, 'Diana', 'Pinto', '20:00:00', 5);"""

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

# Insert query to populate "Orders" table is:

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

insert\_orders="""

INSERT INTO Orders (OrderID, TableNo, MenuID, BookingID, Quantity, BillAmount)

VALUES

(1, 12, 1, 1, 2, 86),

(2, 19, 2, 2, 1, 37),

(3, 15, 2, 3, 1, 37),

(4, 5, 3, 4, 1, 40),

(5, 8, 1, 5, 1, 43);"""

print("Inserting data in MenuItems table.")

# Populate MenuItems table

cursor.execute(insert\_menuitmes)

print("Total number of rows in MenuItem table: {}\n".format(cursor.rowcount))

# Once the query is executed, you commit the change into the database

connection.commit()

print("Inserting data in Menus table.")

# Populate MenuItems table

cursor.execute(insert\_menu)

print("Total number of rows in Menu table: {}\n".format(cursor.rowcount))

connection.commit()

print("Inserting data in Bookings table.")

# Populate Bookings table

cursor.execute(insert\_bookings)

print("Total number of rows in Bookings table: {}\n".format(cursor.rowcount))

connection.commit()

print("Inserting data in Orders table.")

# Populate Orders table

cursor.execute(insert\_orders)

print("Total number of rows in Orders table: {}\n".format(cursor.rowcount))

connection.commit()

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

## **Task 1**

Little Lemon need you to help them query their database to find out many bookings they have today for table number 12. They also need to know the names of the guests booked for table 12 and who their servers are. These records are located within the following tables in the database:

* TableNo,
* GuestFirstName,
* GuestLastName,
* and EmployeeID.

**# The SQL query is:**

filtering\_and\_sorting = """SELECT TableNo,

GuestFirstName, GuestLastName, EmployeeID

FROM Bookings

WHERE TableNo= 12;"""

# Execute query

cursor.execute(filtering\_and\_sorting)

# Fetch records

results = cursor.fetchall()

# Retrieve column names

columns = cursor.column\_names

# Print column names

print(columns)

# Print query results

for result in results:

print(result)

## **Task 2**

Little Lemon want to send a coupon to all guests who spent more than $40 at the restaurant today. Help Little Lemon to filter and sort the records of all guests who qualify for coupons.

Step 1: Create a variable called filtering\_and\_sorting that extracts the booking ID and the bill amount from the Orders table:

# The SQL query is:

filtering\_and\_sorting = """SELECT BookingID, BillAmount

FROM

Orders ORDER BY BillAmount DESC;"""

# Execute query

cursor.execute(filtering\_and\_sorting)

# Fetch records

results = cursor.fetchmany(size=2)#fetchall()

# Retrieve column names

columns = cursor.column\_names

# Print column names

print(columns)

# Print query results

for result in results:

print(result)

# Resetting cursor

cursor.fetchall()

Step 2: Append a WHERE clause to your Python string that limits the bill amount to more than 40 dollars:

# Task 2 Option 2

# The SQL query is:

filtering\_and\_sorting = """SELECT BookingID, BillAmount

FROM

Orders ORDER WHERE BillAmount > 40;"""

# Execute query

cursor.execute(filtering\_and\_sorting)

# Fetch records

results = cursor.fetchall()

# Retrieve column names

columns = cursor.column\_names

# Print column names

print(columns)

# Print query results

for result in results:

print(result)

Step 3: Next, ensure that the results are ordered from high to low:

## # Task 2 Option 3

# The SQL query is:

filtering\_and\_sorting = """SELECT BookingID, BillAmount

FROM

Orders WHERE BILLaMOUNT > 40 ORDER BY BillAmount DESC;"""

# Execute query

cursor.execute(filtering\_and\_sorting)

# Fetch records

results = cursor.fetchall()

# Retrieve column names

columns = cursor.column\_names

# Print column names

print(columns)

# Print query results

for result in results:

print(result)

## **Task 3**

A guest wants to know the price of today’s starters and desserts. So Little Lemon need to extract the price of today’s STARTERS and DESSERT options from their database.

Create a query that extracts the prices and details of starters and dessert options only from the MenuItems table. Order these prices from lower to higher.

# The SQL query is:

filtering\_and\_sorting = """SELECT \*

FROM MenuItems

WHERE (Type = 'Starters' OR Type = 'Desserts')

ORDER BY Price ASC;"""

# Execute query

cursor.execute(filtering\_and\_sorting)

# Fetch records

results = cursor.fetchall()

# Retrieve column names

columns = cursor.column\_names

# Print column names

print(columns)

# Print query results

for result in results:

print(result)

# Let's close the cursor and the connection

if connection.is\_connected():

cursor.close()

print("The cursor is closed.")

connection.close()

print("MySQL connection is closed.")

else:

print("Connection is already closed")