Zabala, Rhaldynyl Brian F.
C203

Finals Lab Task 6. Connection Mysql using Python using CLI

1. Make sure you have installed the following pre-requisites before proceeding:
    a. Mysql-connector
    b. Mysql-connector-python
    c. Xampp is running along with Apache and Mysql in the background
    d. Make sure you have a database in Mysql named testDB with an employees table with at-least 10 records (You may import the attached SQL file to start with

2. Guided by the Demo code attached in this task. Test_DemoDB.py
3. Kindly continue working on the code that will allow the user to navigate through the Database and perform CRUD operations. Follow the following **CLI Menu Options:**

```
----- Employee Manager -----
1. Add Employee
2. View Employees
3. Update Employee
4. Delete Employee
5. Exit
Select an option (1-5):
```

4. The user should be able to add New Records when 1 is selected and show the records, 2- Display all records, 3 Update a Record and show the updates, 4 – Delete a record 5- Exit
5. For additional challenge, Task – You are tasked to add a **SEARCH option** in the MENU that will allow the user to search by Name or emp_id, then display the information about the record being search.

Code:

```
import mysql.connector
import sys
# Connect to the SQLite database (it will create the DB file if not exists)

conn= mysql.connector.connect(
   host="localhost",  # Replace with your MySQL host (e.g., IP address or
hostname)
   user="root",  # Replace with your MySQL username
   password="",  # Replace with your MySQL password
   database="testdb")  # Replace with the name of your database
```

```
cursor = conn.cursor()

#Insert a New Record
def add_employee():
   name = input("Enter name: ")
   emp_id = input("Enter employee ID: ")
   salary = float(input("Enter salary: "))
   cursor.execute("INSERT INTO employees (name, emp_id, salary)
VALUES(%s,%s,%s)", (name, emp_id, salary))
   conn.commit()
   print("Employee added successfully!\n")


# View all employees
def view_employees():
   cursor.execute("SELECT * FROM employees")
   rows = cursor.fetchall()
   if rows:
       print("\nEmployees List:")
       for row in rows:
           print(row)
   else:
       print("\nNo employees found.")

# Update employee
def update_employee():
   emp_id = input("Enter employee ID to update: ")
   name = input("Enter new name: ")
   salary = float(input("Enter new salary: "))
   cursor.execute("UPDATE employees SET name=%s, salary=%s WHERE emp_id=%s",
(name, salary, emp_id))
   conn.commit()
   print("Employee updated successfully!\n")

# Delete employee
def delete_employee():
   emp_id = input("Enter employee ID to delete: ")
   cursor.execute("DELETE FROM employees WHERE emp_id=%s", (emp_id,))
   conn.commit()
   print("Employee deleted successfully!\n")


def search_employee():
   print("Search by:")
   print("1. Name")
   print("2. Employee ID")
   choice = input("Enter your choice: ")
```

```
    if choice == "1":
        name = input("Enter name: ")
        query = "SELECT * FROM employees WHERE name LIKE %s"
        cursor.execute(query, ('%' + name + '%',))
    elif choice == "2":
        emp_id = input("Enter employee ID: ")
        query = "SELECT * FROM employees WHERE emp_id = %s"
        cursor.execute(query, (emp_id,))
    else:
        print("Invalid choice.")
        return
    rows = cursor.fetchall()
    if rows:
        print("\nEmployee(s) found:")
        for row in rows:
            print(row)
    else:
        print("\nNo employees found.")
    show_menu()

def show_menu():
    choice = True
    while choice != '5':
        print("-----Employee Manager-----")
        print("1. Add Employee")
        print("2. View Employee")
        print("3. Update Employee")
        print("4. Delete Employee")
        print("5. Search Employee")
        print("6. Exit")
        choice = input("Select an option (1-5):")

        if choice == '1':
            add_employee()
        elif choice == '2':
            view_employees()
        elif choice == '3':
            update_employee()
        elif choice == '4':
            delete_employee()
        elif choice == '5':
            search_employee()
        elif choice == '6':
            sys.exit()
        else:
            print("Invalid Choice")
```

```
#Test the methods
if __name__ == '__main__':
    show_menu()
```

Output:

```
-----Employee Manager-----
1. Add Employee
2. View Employee
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Select an option (1-5):1
Enter name: Paul
Enter employee ID: 10
Enter salary: 8000
Employee added successfully!

-----Employee Manager-----
1. Add Employee
2. View Employee
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Select an option (1-5):2

Employees List:
(1, 'Russel', 10000.0)
(2, 'Jean', 5000.0)
(3, 'Mark', 69000.0)
(4, 'Lance', 10000.0)
(5, 'Prince', 1000.0)
(6, 'Toby', 9000.0)
(7, 'Bruce', 1000000.0)
(8, 'Steve', 69000.0)
(9, 'lee', 10000.0)
(10, 'Paul', 8000.0)
(222, 'daniel', 1000.0)
```

```
-----Employee Manager-----
1. Add Employee
2. View Employee
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Select an option (1-5):3
Enter employee ID to update: 10
Enter new name: paul
Enter new salary: 2000
Employee updated successfully!

-----Employee Manager-----
1. Add Employee
2. View Employee
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Select an option (1-5):4
Enter employee ID to delete: 222
Employee deleted successfully!

-----Employee Manager-----
1. Add Employee
2. View Employee
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
```

```
-----Employee Manager-----
1. Add Employee
2. View Employee
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Select an option (1-5):5
Search by:
1. Name
2. Employee ID
Enter your choice: 1
Enter name: Steve

Employee(s) found:
(8, 'Steve', 69000.0)
-----Employee Manager-----
1. Add Employee
2. View Employee
3. Update Employee
4. Delete Employee
5. Search Employee
6. Exit
Select an option (1-5):6

Process finished with exit code 0
```