Assignment 2

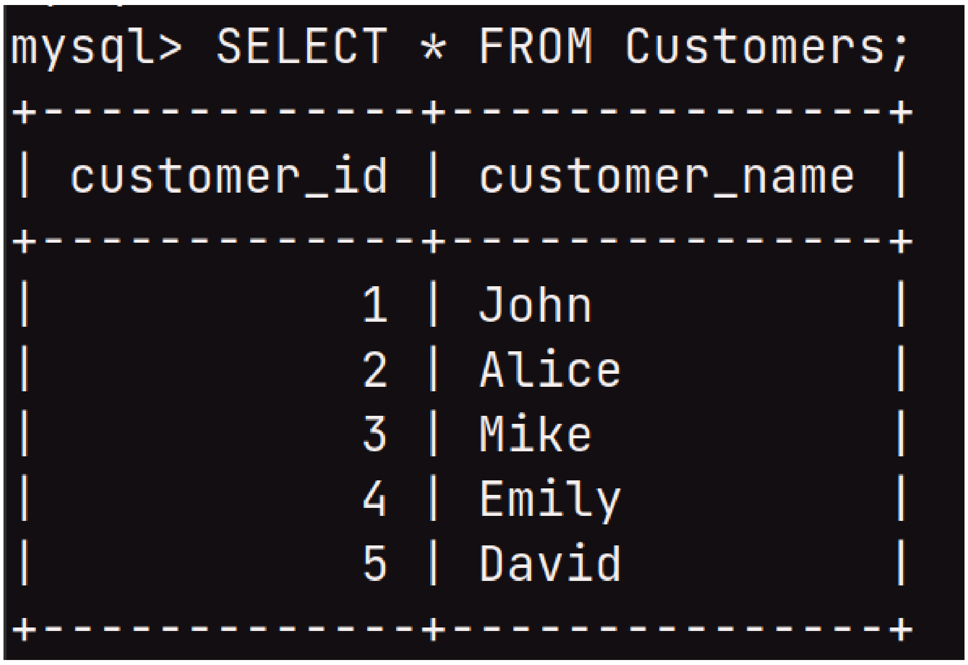
Trainee: Pankaj Kumar Mentor: Sagnik Mandal

Q1)

Find out the total number of orders for each customer, including customers who have not placed any orders.

CREATE TABLE Customers (customer_id INT, customer_name VARCHAR(50));

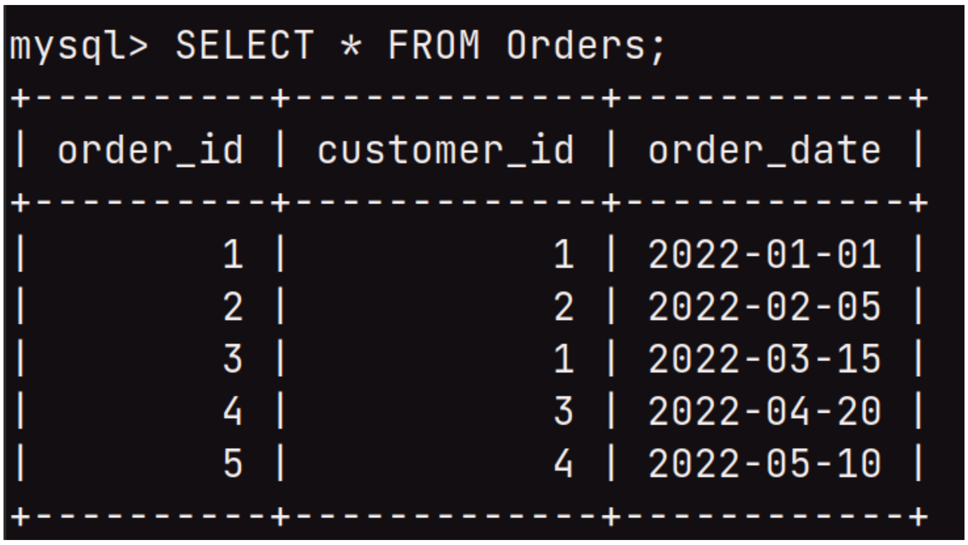
INSERT INTO Customers (customer_id, customer_name) VALUES (1, 'John'), (2, 'Alice'), (3, 'Mike'), (4, 'Emily'), (5, 'David');



```
mysql> SELECT * FROM Customers;
+-------------+---------------+
| customer_id | customer_name |
+-------------+---------------+
|           1 | John          |
|           2 | Alice         |
|           3 | Mike          |
|           4 | Emily         |
|           5 | David         |
+-------------+---------------+
```

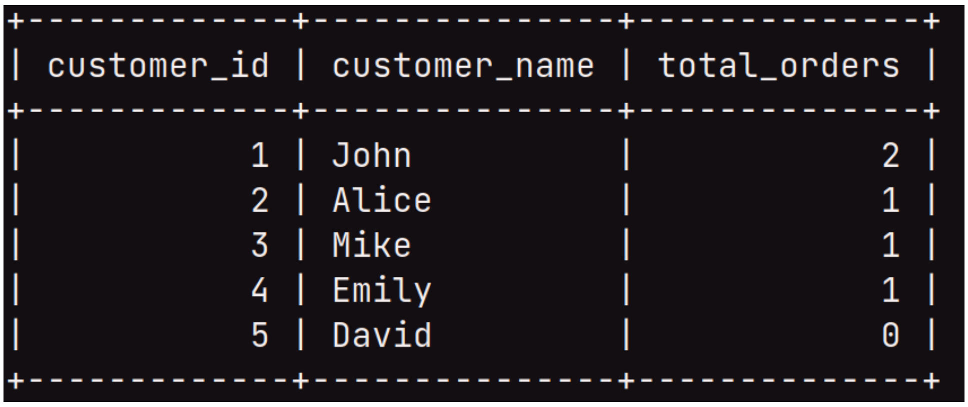
CREATE TABLE Orders (order_id INT, customer_id INT, order_date DATE);

INSERT INTO Orders (order_id, customer_id, order_date) VALUES (1, 1, '2022-01-01'), (2, 2, '2022-02-05'), (3, 1, '2022-03-15'), (4, 3, '2022-04-20'), (5, 4, '2022-05-10');



```
mysql> SELECT * FROM Orders;
+----------+-------------+------------+
| order_id | customer_id | order_date |
+----------+-------------+------------+
|        1 |           1 | 2022-01-01 |
|        2 |           2 | 2022-02-05 |
|        3 |           1 | 2022-03-15 |
|        4 |           3 | 2022-04-20 |
|        5 |           4 | 2022-05-10 |
+----------+-------------+------------+
```

Output:



```
+-------------+---------------+--------------+
| customer_id | customer_name | total_orders |
+-------------+---------------+--------------+
|           1 | John          |            2 |
|           2 | Alice         |            1 |
|           3 | Mike          |            1 |
|           4 | Emily         |            1 |
|           5 | David         |            0 |
+-------------+---------------+--------------+
```

Q2)

Write a Python program that organizes files in a directory based on their file extensions. The program should create separate folders for each file extension and move the corresponding files into their respective folders.

Requirements:

- Create separate folders for each unique file extension found in the directory.
- Move the files into their respective folders based on their file extensions.
- Display the total number of files moved

Q3)

Implement an inheritance hierarchy for shapes in Python. The hierarchy should include a base class called **Shape** and three derived classes: **Rectangle**, **Circle**, and **Triangle**.

Requirements:

Shape class:

- Attributes: None
- Methods:
    - **get_area()**: Returns the area of the shape (abstract method)

Rectangle class:

- Inherits from **Shape**
- Attributes: **length** (float), **width** (float)
- Implements the **get_area()** method to calculate and return the area of the rectangle (length * width)

Circle class:

- Inherits from **Shape**
- Attribute: **radius** (float)
- Implements the **get_area()** method to calculate and return the area of the circle (π * radius^2)
- Use the value of π as 3.14159

Triangle class:

- Inherits from **Shape**
- Attributes: **base** (float), **height** (float)
- Implements the **get_area()** method to calculate and return the area of the triangle (0.5 * base * height)

Your task is to:

1. Implement the **Shape**, **Rectangle**, **Circle**, and **Triangle** classes as described above.
2. Create instances of each shape class with sample data.
3. Call the **get_area**() method on each instance and print the returned area.

Q4)

Using the Python "requests" library, implement the following endpoints:

GET	https://jsonplaceholder.typicode.com/posts	Get all posts
GET	https://jsonplaceholder.typicode.com/posts/2	Get a particular post by ID
POST	https://jsonplaceholder.typicode.com/posts	Create post
PUT	https://jsonplaceholder.typicode.com/posts/2	Update post by ID
DELETE	https://jsonplaceholder.typicode.com/posts/2	Delete post by ID

Q5)

Given an integer list of numbers, return true if any value appears at least twice in the array, and return false if every element is distinct.

Input: numbers = [7,8,3,4]
Output: false
Input: numbers = [5,2,5,3]
Output: true

Q6)

Write a Python function called calculate_sum that accepts variable-length arguments (*args) and keyword arguments (**kwargs). The function should calculate the sum of all the arguments passed, where each argument can be either a number or a key-value pair with the key being a string and the value being a number. The function should return the total sum as the output.

```
# Example 1: Combination of numbers and key-value pairs

result = calculate_sum(10, d=11, e=12, 13)

print(result) # Output: 46
```

Q7)

You are working on a program that needs to calculate the squares of multiple numbers concurrently to improve performance. Your task is to implement a solution using Python's multiprocessing module.

Write a Python script that does the following:

1. Define a function calculate_square(num) that takes an integer num as input and calculates the square of the number.
2. Implement a multiprocessing solution that uses 3 separate processes to calculate the squares of the numbers in the list [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].
3. Each process should calculate the square of one element from the list and print the result along with the process name.
4. Ensure that the processes are executed concurrently to maximize efficiency and take advantage of multiple CPU cores.