

## Assignment 1 (SQL)

### Task 1

Q1:-Create the database named "TechShop"

```
mysql> create database TechShop;  
Query OK, 1 row affected (0.01 sec)
```

Q2 :-Define the schema for the Customers, Products, Orders, OrderDetails and Inventory tables based on the provided schema

```
mysql> desc customers;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| CustomerID | int           | NO   | PRI | NULL    |       |  
| FirstName  | varchar(50)   | YES  |     | NULL    |       |  
| LastName   | varchar(50)   | YES  |     | NULL    |       |  
| Email      | varchar(100)  | YES  |     | NULL    |       |  
| Phone      | varchar(20)   | YES  |     | NULL    |       |  
| Address    | varchar(255)  | YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
6 rows in set (0.00 sec)
```

```
mysql> desc products;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| ProductID  | int           | NO   | PRI | NULL    |       |  
| ProductName | varchar(50)   | YES  |     | NULL    |       |  
| Description | varchar(100)  | YES  |     | NULL    |       |  
| Price      | decimal(10,2) | YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.00 sec)
```

```
mysql> desc orders;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| OrderID    | int           | NO   | PRI | NULL    |       |  
| CustomerID | int           | YES  | MUL | NULL    |       |  
| OrderDate  | date          | YES  |     | NULL    |       |  
| TotalAmount | decimal(10,2) | YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.00 sec)
```

```
mysql> desc orderdetails;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| OrderDetailID | int | NO   | PRI | NULL    |       |  
| OrderID       | int | YES  | MUL | NULL    |       |  
| ProductID     | int | YES  | MUL | NULL    |       |  
| Quantity      | int | YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.00 sec)
```

```
mysql> desc inventory;
```

| Field           | Type | Null | Key | Default | Extra |
|-----------------|------|------|-----|---------|-------|
| InventoryID     | int  | NO   | PRI | NULL    |       |
| ProductID       | int  | YES  | MUL | NULL    |       |
| QuantityInStock | int  | YES  |     | NULL    |       |
| LastStockUpdate | date | YES  |     | NULL    |       |

```
4 rows in set (0.00 sec)
```

**Q4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.**

**Customers table:-**

```
mysql> CREATE TABLE Customers (
  ->   CustomerID INT PRIMARY KEY,
  ->   FirstName VARCHAR(50),
  ->   LastName VARCHAR(50),
  ->   Email VARCHAR(100),
  ->   Phone VARCHAR(20),
  ->   Address VARCHAR(255)
  -> );
Query OK, 0 rows affected (0.04 sec)
```

**Products table:-**

```
mysql> create table Products(
  -> ProductID int PRIMARY KEY,
  -> ProductName varchar(50),
  -> Description varchar(50),
  -> price varchar(20)
  -> );
Query OK, 0 rows affected (0.02 sec)
```

**Order table:-**

```
mysql> CREATE TABLE Orders (
  ->   OrderID INT PRIMARY KEY,
  ->   CustomerID INT,
  ->   OrderDate DATE,
  ->   TotalAmount DECIMAL(10, 2),
  ->   FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
  -> );
Query OK, 0 rows affected (0.03 sec)
```

## Assignment 1 (SQL)

### OrderDetails table:-

```
mysql> create table OrderDetails(  
  -> OrderDetailID INT PRIMARY KEY,  
  -> OrderID INT,  
  -> ProductID INT,  
  -> Quantity INT,  
  -> FOREIGN KEY(OrderID) REFERENCES Orders(OrderID),  
  -> FOREIGN KEY(ProductID) REFERENCES Products(ProductID)  
  -> );  
Query OK, 0 rows affected (0.04 sec)
```

### Inventory tables:-

```
mysql> CREATE TABLE Inventory (  
  -> InventoryID INT PRIMARY KEY,  
  -> ProductID INT,  
  -> QuantityInStock INT,  
  -> LastStockUpdate DATE,  
  -> FOREIGN KEY (ProductID) REFERENCES Products(ProductID)  
  -> );  
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> show tables;  
+-----+  
| Tables_in_techshop |  
+-----+  
| customers          |  
| inventory           |  
| orderdetails       |  
| orders             |  
| products           |  
+-----+  
5 rows in set (0.00 sec)
```

## Assignment 1 (SQL)

**Q 5. Insert at least 10 sample records into each of the following tables.**

### a. Customers

```
mysql> insert into customers values(1,"a","k","a@gmail.com",11111111,"jhajh  
a"),(2,"b","kb","b@gmail.com",11111112,"khajha"),(3,"c","kc","ac@gmail.com",  
11111113,"jhajh3"),(4,"d","kd","ad@gmail.com",1111111d,"jhajhd"),(5,"e","ke"  
,"ae@gmail.com",1111111e,"jhajhae"),(6,"f","kf","af@gmail.com",1111111f,"j  
hajhaf"),(7,"g","kg","ag@gmail.com",1111111g,"jhajhag"),(8,"h","kh","ah@hma  
il.com",1111111h,"jhajhah"),(9,"i","ki","ai@gmail.com",1111111i,"jhajhai")  
,(10,"j","kj","aj@gmail.com",1111111j,"jhajhaj");  
ERROR 1054 (42S22): Unknown column '1111111d' in 'field list'  
mysql> insert into customers values(1,"a","k","a@gmail.com",11111111,"jhajh  
a"),(2,"b","kb","b@gmail.com",11111112,"khajha"),(3,"c","kc","ac@gmail.com",  
11111113,"jhajh3"),(4,"d","kd","ad@gmail.com",11111114,"jhajhd"),(5,"e","ke"  
,"ae@gmail.com",11111115,"jhajhae"),(6,"f","kf","af@gmail.com",11111116,"j  
hajhaf"),(7,"g","kg","ag@gmail.com",11111117,"jhajhag"),(8,"h","kh","ah@hma  
il.com",11111118,"jhajhah"),(9,"i","ki","ai@gmail.com",11111119,"jhajhai")  
,(10,"j","kj","aj@gmail.com",111111110,"jhajhaj");  
Query OK, 10 rows affected (0.00 sec)  
Records: 10 Duplicates: 0 Warnings: 0
```

```
mysql> select * from customers;  
+-----+-----+-----+-----+-----+-----+  
| CustomerID | FirstName | LastName | Email | Phone | Address |  
+-----+-----+-----+-----+-----+-----+  
| 1 | a | k | a@gmail.com | 11111111 | jhajha |  
| 2 | b | kb | b@gmail.com | 11111112 | khajha |  
| 3 | c | kc | ac@gmail.com | 11111113 | jhajh3 |  
| 4 | d | kd | ad@gmail.com | 11111114 | jhajhd |  
| 5 | e | ke | ae@gmail.com | 11111115 | jhajhae |  
| 6 | f | kf | af@gmail.com | 11111116 | jhajhaf |  
| 7 | g | kg | ag@gmail.com | 11111117 | jhajhag |  
| 8 | h | kh | ah@gmail.com | 11111118 | jhajhah |  
| 9 | i | ki | ai@gmail.com | 11111119 | jhajhai |  
| 10 | j | kj | aj@gmail.com | 111111110 | jhajhaj |  
+-----+-----+-----+-----+-----+-----+  
10 rows in set (0.00 sec)
```

### b. Products

```
mysql> insert into products values  
-> (1,'laptop','high-pef',199),  
-> (2,'desktop','med-pef',299),  
-> (3,'smartphone','low-pef',399),  
-> (4,'heater','high-pef',499),  
-> (5,'tab','med-pef',599),  
-> (6,'monitor','low-pef',699),  
-> (7,'cpu','high-pef',799),  
-> (8,'mouse','low-pef',899),  
-> (9,'keyboard','med-pef',999),  
-> (10,'telivision','high-pef',1099);  
Query OK, 10 rows affected (0.01 sec)  
Records: 10 Duplicates: 0 Warnings: 0
```

## Assignment 1 (SQL)

```
mysql> select * from products;
```

| ProductID | ProductName | Description | Price   |
|-----------|-------------|-------------|---------|
| 1         | laptop      | high-pef    | 199.00  |
| 2         | desktop     | med-pef     | 299.00  |
| 3         | smartphone  | low-pef     | 399.00  |
| 4         | heater      | high-pef    | 499.00  |
| 5         | tab         | med-pef     | 599.00  |
| 6         | monitor     | low-pef     | 699.00  |
| 7         | cpu         | high-pef    | 799.00  |
| 8         | mouse       | low-pef     | 899.00  |
| 9         | keyboard    | med-pef     | 999.00  |
| 10        | television  | high-pef    | 1099.00 |

10 rows in set (0.00 sec)

### c. Orders

```
mysql> insert into Orders values
-> (10,1,'2024-01-10',299.00),
-> (20,2,'2024-01-11',399.00),
-> (30,3,'2024-01-12',499.00),
-> (40,4,'2024-01-13',599.00),
-> (50,5,'2024-01-14',699.00),
-> (60,6,'2024-01-15',799.00),
-> (70,7,'2024-01-16',899.00),
-> (80,8,'2024-01-17',999.00),
-> (90,9,'2024-01-18',1099.00),
-> (100,10,'2024-01-19',1199.00);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> select *from orders;
```

| OrderID | CustomerID | OrderDate  | TotalAmount |
|---------|------------|------------|-------------|
| 10      | 1          | 2024-01-10 | 299.00      |
| 20      | 2          | 2024-01-11 | 399.00      |
| 30      | 3          | 2024-01-12 | 499.00      |
| 40      | 4          | 2024-01-13 | 599.00      |
| 50      | 5          | 2024-01-14 | 699.00      |
| 60      | 6          | 2024-01-15 | 799.00      |
| 70      | 7          | 2024-01-16 | 899.00      |
| 80      | 8          | 2024-01-17 | 999.00      |
| 90      | 9          | 2024-01-18 | 1099.00     |
| 100     | 10         | 2024-01-19 | 1199.00     |

10 rows in set (0.00 sec)

**d. OrderDetails**

```
mysql> insert into Orderdetails values(
  -> 11,10,1,5),
  -> (12,20,2,10),
  -> (13,30,3,15),
  -> (14,40,4,20),
  -> (15,50,5,25),
  -> (16,60,6,30),
  -> (17,70,7,35),
  -> (18,80,8,0),
  -> (19,90,9,40),
  -> (20,100,10,45);
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> select * from Orderdetails;
+-----+-----+-----+-----+
| OrderDetailID | OrderID | ProductID | Quantity |
+-----+-----+-----+-----+
|          11 |      10 |          1 |         5 |
|          12 |      20 |          2 |        10 |
|          13 |      30 |          3 |        15 |
|          14 |      40 |          4 |        20 |
|          15 |      50 |          5 |        25 |
|          16 |      60 |          6 |        30 |
|          17 |      70 |          7 |        35 |
|          18 |      80 |          8 |         0 |
|          19 |      90 |          9 |        40 |
|          20 |     100 |         10 |        45 |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

**e. Inventory**

```
mysql> insert into inventory values
  -> (1,1,25,'2024-01-10'),
  -> (2,2,30,'2024-01-11'),
  -> (3,3,35,'2024-01-12'),
  -> (4,4,45,'2024-01-13'),
  -> (5,5,55,'2024-01-14'),
  -> (6,6,26,'2024-01-16'),
  -> (7,7,60,'2024-01-17'),
  -> (8,8,65,'2024-01-18'),
  -> (9,9,70,'2024-01-19'),
  -> (10,10,75,'2024-01-20');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> select * from inventory;
+-----+-----+-----+-----+
| InventoryID | ProductID | QuantityInStock | LastStockUpdate |
+-----+-----+-----+-----+
| 1 | 1 | 25 | 2024-01-10 |
| 2 | 2 | 30 | 2024-01-11 |
| 3 | 3 | 35 | 2024-01-12 |
| 4 | 4 | 45 | 2024-01-13 |
| 5 | 5 | 55 | 2024-01-14 |
| 6 | 6 | 26 | 2024-01-16 |
| 7 | 7 | 60 | 2024-01-17 |
| 8 | 8 | 65 | 2024-01-18 |
| 9 | 9 | 70 | 2024-01-19 |
| 10 | 10 | 75 | 2024-01-20 |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

**Tasks 2: Select, Where, Between, AND, LIKE:**

1. Write an SQL query to retrieve the names and emails of all customers.

```
mysql> select FirstName,Email from customers;
+-----+-----+
| FirstName | Email |
+-----+-----+
| a | a@gmail.com |
| b | b@gmail.com |
| c | ac@gmail.com |
| d | ad@gmail.com |
| e | ae@gmail.com |
| f | af@gmail.com |
| g | ag@gmail.com |
| h | ah@gmail.com |
| i | ai@gmail.com |
| j | aj@gmail.com |
+-----+-----+
10 rows in set (0.00 sec)
```

## Assignment 1 (SQL)

2. Write an SQL query to list all orders with their order dates and corresponding customer names.

```
mysql> select orders.orderID,orders.OrderDate,customers.FirstName from orders
-> JOIN customers ON orders.CustomerID=customers.CustomerID;
```

| orderID | OrderDate  | FirstName |
|---------|------------|-----------|
| 10      | 2024-01-10 | a         |
| 20      | 2024-01-11 | b         |
| 30      | 2024-01-12 | c         |
| 40      | 2024-01-13 | d         |
| 50      | 2024-01-14 | e         |
| 60      | 2024-01-15 | f         |
| 70      | 2024-01-16 | g         |
| 80      | 2024-01-17 | h         |
| 90      | 2024-01-18 | i         |
| 100     | 2024-01-19 | j         |

10 rows in set (0.00 sec)

3. Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address

Ans:-

Before Adding the new customer data

```
mysql> select * from customers;
```

| CustomerID | FirstName | LastName | Email        | Phone     | Address |
|------------|-----------|----------|--------------|-----------|---------|
| 1          | a         | k        | a@gmail.com  | 11111111  | jhajha  |
| 2          | b         | kb       | b@gmail.com  | 11111112  | khajha  |
| 3          | c         | kc       | ac@gmail.com | 11111113  | jhajh3  |
| 4          | d         | kd       | ad@gmail.com | 11111114  | jhajhd  |
| 5          | e         | ke       | ae@gmail.com | 11111115  | jhajhae |
| 6          | f         | kf       | af@gmail.com | 11111116  | jhajhaf |
| 7          | g         | kg       | ag@gmail.com | 11111117  | jhajhag |
| 8          | h         | kh       | ah@gmail.com | 11111118  | jhajhah |
| 9          | i         | ki       | ai@gmail.com | 11111119  | jhajhai |
| 10         | j         | kj       | aj@gmail.com | 111111110 | jhajhaj |

10 rows in set (0.00 sec)

After Adding the new customer data

```
mysql> insert into customers values(11,'Sourav','Kumar','Engineer@gmail.com',9110166666,'Bihar');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from customers;
```

| CustomerID | FirstName | LastName | Email              | Phone      | Address |
|------------|-----------|----------|--------------------|------------|---------|
| 1          | a         | k        | a@gmail.com        | 11111111   | jhajha  |
| 2          | b         | kb       | b@gmail.com        | 11111112   | khajha  |
| 3          | c         | kc       | ac@gmail.com       | 11111113   | jhajh3  |
| 4          | d         | kd       | ad@gmail.com       | 11111114   | jhajhd  |
| 5          | e         | ke       | ae@gmail.com       | 11111115   | jhajhae |
| 6          | f         | kf       | af@gmail.com       | 11111116   | jhajhaf |
| 7          | g         | kg       | ag@gmail.com       | 11111117   | jhajhag |
| 8          | h         | kh       | ah@gmail.com       | 11111118   | jhajhah |
| 9          | i         | ki       | ai@gmail.com       | 11111119   | jhajhai |
| 10         | j         | kj       | aj@gmail.com       | 111111110  | jhajhaj |
| 11         | Sourav    | Kumar    | Engineer@gmail.com | 9110166666 | Bihar   |

11 rows in set (0.00 sec)



4. Write an SQL query to update the prices of all electronic gadgets in the "Products" table by increasing them by 10%.

```
mysql> select * from products;
```

| ProductID | ProductName | Description | Price   |
|-----------|-------------|-------------|---------|
| 1         | laptop      | high-pef    | 199.00  |
| 2         | desktop     | med-pef     | 299.00  |
| 3         | smartphone  | low-pef     | 399.00  |
| 4         | heater      | high-pef    | 499.00  |
| 5         | tab         | med-pef     | 599.00  |
| 6         | monitor     | low-pef     | 699.00  |
| 7         | cpu         | high-pef    | 799.00  |
| 8         | mouse       | low-pef     | 899.00  |
| 9         | keyboard    | med-pef     | 999.00  |
| 10        | television  | high-pef    | 1099.00 |

```
10 rows in set (0.00 sec)
```

```
mysql> UPDATE products
-> SET Price=Price+(Price*10/100)
-> ;
```

```
Query OK, 10 rows affected (0.01 sec)
```

```
Rows matched: 10 Changed: 10 Warnings: 0
```

```
mysql> select * from Products;
```

| ProductID | ProductName | Description | Price   |
|-----------|-------------|-------------|---------|
| 1         | laptop      | high-pef    | 218.90  |
| 2         | desktop     | med-pef     | 328.90  |
| 3         | smartphone  | low-pef     | 438.90  |
| 4         | heater      | high-pef    | 548.90  |
| 5         | tab         | med-pef     | 658.90  |
| 6         | monitor     | low-pef     | 768.90  |
| 7         | cpu         | high-pef    | 878.90  |
| 8         | mouse       | low-pef     | 988.90  |
| 9         | keyboard    | med-pef     | 1098.90 |
| 10        | television  | high-pef    | 1208.90 |

```
10 rows in set (0.00 sec)
```

5. Write an SQL query to delete a specific order and its associated order details from the "Orders" and "OrderDetails" tables. Allow users to input the order ID as a parameter

```
mysql> DELETE FROM OrderDetails
    -> WHERE OrderID=20;
Query OK, 1 row affected (0.01 sec)
```

```
mysql> delete from Orders
    -> where OrderID=20;
Query OK, 1 row affected (0.00 sec)
```

```
mysql> Select *from Orders;
```

| OrderID | CustomerID | OrderDate  | TotalAmount |
|---------|------------|------------|-------------|
| 10      | 1          | 2024-01-10 | 299.00      |
| 30      | 3          | 2024-01-12 | 499.00      |
| 40      | 4          | 2024-01-13 | 599.00      |
| 50      | 5          | 2024-01-14 | 699.00      |
| 60      | 6          | 2024-01-15 | 799.00      |
| 70      | 7          | 2024-01-16 | 899.00      |
| 80      | 8          | 2024-01-17 | 999.00      |
| 90      | 9          | 2024-01-18 | 1099.00     |
| 100     | 10         | 2024-01-19 | 1199.00     |

```
9 rows in set (0.00 sec)
```

```
mysql> select * from OrderDetails;
```

| OrderDetailID | OrderID | ProductID | Quantity |
|---------------|---------|-----------|----------|
| 11            | 10      | 1         | 5        |
| 13            | 30      | 3         | 15       |
| 14            | 40      | 4         | 20       |
| 15            | 50      | 5         | 25       |
| 16            | 60      | 6         | 30       |
| 17            | 70      | 7         | 35       |
| 18            | 80      | 8         | 0        |
| 19            | 90      | 9         | 40       |
| 20            | 100     | 10        | 45       |

```
9 rows in set (0.00 sec)
```

6. Write an SQL query to insert a new order into the "Orders" table. Include the customer ID, order date, and any other necessary information.

```
mysql> select * from orders;
```

| OrderID | CustomerID | OrderDate  | TotalAmount |
|---------|------------|------------|-------------|
| 10      | 1          | 2024-01-10 | 299.00      |
| 30      | 3          | 2024-01-12 | 499.00      |
| 40      | 4          | 2024-01-13 | 599.00      |
| 50      | 5          | 2024-01-14 | 699.00      |
| 60      | 6          | 2024-01-15 | 799.00      |
| 70      | 7          | 2024-01-16 | 899.00      |
| 80      | 8          | 2024-01-17 | 999.00      |
| 90      | 9          | 2024-01-18 | 1099.00     |
| 100     | 10         | 2024-01-19 | 1199.00     |

```
9 rows in set (0.00 sec)
```

```
mysql> insert into orders values(110,11,'2024-01-20',1299.00);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from orders;
```

| OrderID | CustomerID | OrderDate  | TotalAmount |
|---------|------------|------------|-------------|
| 10      | 1          | 2024-01-10 | 299.00      |
| 30      | 3          | 2024-01-12 | 499.00      |
| 40      | 4          | 2024-01-13 | 599.00      |
| 50      | 5          | 2024-01-14 | 699.00      |
| 60      | 6          | 2024-01-15 | 799.00      |
| 70      | 7          | 2024-01-16 | 899.00      |
| 80      | 8          | 2024-01-17 | 999.00      |
| 90      | 9          | 2024-01-18 | 1099.00     |
| 100     | 10         | 2024-01-19 | 1199.00     |
| 110     | 11         | 2024-01-20 | 1299.00     |

```
10 rows in set (0.00 sec)
```

7. Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information

```
mysql> UPDATE Customers
-> SET Email='srv@gmail.com',Address='Patna' WHERE CustomerID=1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> select * from customers;
```

| CustomerID | FirstName | LastName | Email              | Phone      | Address |
|------------|-----------|----------|--------------------|------------|---------|
| 1          | a         | k        | srv@gmail.com      | 11111111   | Patna   |
| 2          | b         | kb       | b@gmail.com        | 11111112   | khajha  |
| 3          | c         | kc       | ac@gmail.com       | 11111113   | jhajh3  |
| 4          | d         | kd       | ad@gmail.com       | 11111114   | jhajhd  |
| 5          | e         | ke       | ae@gmail.com       | 11111115   | jhajhae |
| 6          | f         | kf       | af@gmail.com       | 11111116   | jhajhaf |
| 7          | g         | kg       | ag@gmail.com       | 11111117   | jhajhag |
| 8          | h         | kh       | ah@gmail.com       | 11111118   | jhajhah |
| 9          | i         | ki       | ai@gmail.com       | 11111119   | jhajhai |
| 10         | j         | kj       | aj@gmail.com       | 111111110  | jhajhaj |
| 11         | Sourav    | Kumar    | Engineer@gmail.com | 9110166666 | Bihar   |

```
11 rows in set (0.00 sec)
```

8. Write an SQL query to recalculate and update the total cost of each order in the "Orders" table based on the prices and quantities in the "OrderDetails" table.

```
mysql> select * from orders;
```

| OrderID | CustomerID | OrderDate  | TotalAmount |
|---------|------------|------------|-------------|
| 10      | 1          | 2024-01-10 | 299.00      |
| 30      | 3          | 2024-01-12 | 499.00      |
| 40      | 4          | 2024-01-13 | 599.00      |
| 50      | 5          | 2024-01-14 | 699.00      |
| 60      | 6          | 2024-01-15 | 799.00      |
| 70      | 7          | 2024-01-16 | 899.00      |
| 80      | 8          | 2024-01-17 | 999.00      |
| 90      | 9          | 2024-01-18 | 1099.00     |
| 100     | 10         | 2024-01-19 | 1199.00     |
| 110     | 11         | 2024-01-20 | 1299.00     |

```
10 rows in set (0.00 sec)
```

```
mysql> select * from Orderdetails;
```

| OrderDetailID | OrderID | ProductID | Quantity |
|---------------|---------|-----------|----------|
| 11            | 10      | 1         | 5        |
| 13            | 30      | 3         | 15       |
| 14            | 40      | 4         | 20       |
| 15            | 50      | 5         | 25       |
| 16            | 60      | 6         | 30       |
| 17            | 70      | 7         | 35       |
| 18            | 80      | 8         | 0        |
| 19            | 90      | 9         | 40       |
| 20            | 100     | 10        | 45       |

```
9 rows in set (0.00 sec)
```

```
mysql> select * from products;
```

| ProductID | ProductName | Description | Price   |
|-----------|-------------|-------------|---------|
| 1         | laptop      | high-pef    | 218.90  |
| 2         | desktop     | med-pef     | 328.90  |
| 3         | smartphone  | low-pef     | 438.90  |
| 4         | heater      | high-pef    | 548.90  |
| 5         | tab         | med-pef     | 658.90  |
| 6         | monitor     | low-pef     | 768.90  |
| 7         | cpu         | high-pef    | 878.90  |
| 8         | mouse       | low-pef     | 988.90  |
| 9         | keyboard    | med-pef     | 1098.90 |
| 10        | television  | high-pef    | 1208.90 |

```
10 rows in set (0.00 sec)
```

## Assignment 1 (SQL)

```
--> FROM Orderdetails at line 2
mysql> UPDATE Orders
--> SET TotalAmount=(
--> SELECT SUM(od.Quantity*p.Price)
--> FROM Orderdetails od
--> JOIN Products p ON od.ProductID=p.ProductID
--> WHERE od.OrderID=Orders.OrderID
--> )
--> WHERE OrderID IN(SELECT DISTINCT OrderID FROM Orderdetails);
Query OK, 9 rows affected (0.01 sec)
Rows matched: 9  Changed: 9  Warnings: 0

mysql> select * from orders;
+-----+-----+-----+-----+
| OrderID | CustomerID | OrderDate | TotalAmount |
+-----+-----+-----+-----+
| 10 | 1 | 2024-01-10 | 1094.50 |
| 30 | 3 | 2024-01-12 | 6583.50 |
| 40 | 4 | 2024-01-13 | 10978.00 |
| 50 | 5 | 2024-01-14 | 16472.50 |
| 60 | 6 | 2024-01-15 | 23067.00 |
| 70 | 7 | 2024-01-16 | 30761.50 |
| 80 | 8 | 2024-01-17 | 0.00 |
| 90 | 9 | 2024-01-18 | 43956.00 |
| 100 | 10 | 2024-01-19 | 54400.50 |
| 110 | 11 | 2024-01-20 | 1299.00 |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

9. Write an SQL query to delete all orders and their associated order details for a specific customer from the "Orders" and "OrderDetails" tables. Allow users to input the customer ID as a parameter

```
mysql> delimiter @@
mysql> create procedure del_all_orders(in custid int)
--> begin
--> delete from orderdetails where OrderID in(select OrderID from Orders where CustomerID=custid)
--> delete from Orders where CustomerID=custid;
--> end @@
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> call del_all_orders(2);
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> select * from orders;
+-----+-----+-----+-----+
| OrderID | CustomerID | OrderDate | TotalAmount |
+-----+-----+-----+-----+
| 10 | 1 | 2024-01-10 | 1094.50 |
| 30 | 3 | 2024-01-12 | 6583.50 |
| 40 | 4 | 2024-01-13 | 10978.00 |
| 50 | 5 | 2024-01-14 | 16472.50 |
| 60 | 6 | 2024-01-15 | 23067.00 |
| 70 | 7 | 2024-01-16 | 30761.50 |
| 80 | 8 | 2024-01-17 | 0.00 |
| 90 | 9 | 2024-01-18 | 43956.00 |
| 100 | 10 | 2024-01-19 | 54400.50 |
| 110 | 11 | 2024-01-20 | 1299.00 |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

```
mysql> select * from orderdetails;
```

| OrderDetailID | OrderID | ProductID | Quantity |
|---------------|---------|-----------|----------|
| 11            | 10      | 1         | 5        |
| 13            | 30      | 3         | 15       |
| 14            | 40      | 4         | 20       |
| 15            | 50      | 5         | 25       |
| 16            | 60      | 6         | 30       |
| 17            | 70      | 7         | 35       |
| 18            | 80      | 8         | 0        |
| 19            | 90      | 9         | 40       |
| 20            | 100     | 10        | 45       |

```
9 rows in set (0.00 sec)
```

10. Write an SQL query to insert a new electronic gadget product into the "Products" table, including product name, category, price, and any other relevant details.

```
mysql> alter table orderdetails
-> drop foreign key orderdetails_ibfk_2;
Query OK, 0 rows affected (0.03 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

```
mysql> alter table inventory
-> drop foreign key inventory_ibfk_1;
Query OK, 0 rows affected (0.02 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> alter table products
-> modify column productID INT auto_increment;
Query OK, 10 rows affected (0.06 sec)
Records: 10 Duplicates: 0 Warnings: 0
```

```
mysql> alter table orderdetails
-> add foreign key(productID) references products(productID);
Query OK, 9 rows affected (0.07 sec)
Records: 9 Duplicates: 0 Warnings: 0

mysql> alter table inventory
-> add foreign key(productID) references products(productID);
Query OK, 10 rows affected (0.06 sec)
Records: 10 Duplicates: 0 Warnings: 0
```

## Assignment 1 (SQL)

```
mysql> insert into products(ProductName,Description,Price)values('Phone','medium_to_high',1199.50);
Query OK, 1 row affected (0.00 sec)

mysql> select * from products;
```

| productID | ProductName | Description    | Price   |
|-----------|-------------|----------------|---------|
| 1         | laptop      | high-pef       | 218.90  |
| 2         | desktop     | med-pef        | 328.90  |
| 3         | smartphone  | low-pef        | 438.90  |
| 4         | heater      | high-pef       | 548.90  |
| 5         | tab         | med-pef        | 658.90  |
| 6         | monitor     | low-pef        | 768.90  |
| 7         | cpu         | high-pef       | 878.90  |
| 8         | mouse       | low-pef        | 988.90  |
| 9         | keyboard    | med-pef        | 1098.90 |
| 10        | television  | high-pef       | 1208.90 |
| 11        | Phone       | medium_to_high | 1199.50 |

```
11 rows in set (0.00 sec)
```

11. Write an SQL query to update the status of a specific order in the "Orders" table (e.g., from "Pending" to "Shipped"). Allow users to input the order ID and the new status

```
mysql> alter table orders
-> add column status varchar(20) DEFAULT 'pending';
Query OK, 0 rows affected (0.04 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> select * from orders;
```

| OrderID | CustomerID | OrderDate  | TotalAmount | status  |
|---------|------------|------------|-------------|---------|
| 10      | 1          | 2024-01-10 | 1094.50     | pending |
| 30      | 3          | 2024-01-12 | 6583.50     | pending |
| 40      | 4          | 2024-01-13 | 10978.00    | pending |
| 50      | 5          | 2024-01-14 | 16472.50    | pending |
| 60      | 6          | 2024-01-15 | 23067.00    | pending |
| 70      | 7          | 2024-01-16 | 30761.50    | pending |
| 80      | 8          | 2024-01-17 | 0.00        | pending |
| 90      | 9          | 2024-01-18 | 43956.00    | pending |
| 100     | 10         | 2024-01-19 | 54400.50    | pending |
| 110     | 11         | 2024-01-20 | 1299.00     | pending |

```
10 rows in set (0.00 sec)
```

```
mysql> delimiter ##
mysql> create procedure updateorderstatus(in ord_id int,in new_status varchar(20))
-> begin
-> update orders set status=new_status where OrderID=ord_id;
-> end ##
Query OK, 0 rows affected (0.01 sec)
```

## Assignment 1 (SQL)

```
mysql> delimiter ;
mysql> call updateorderstatus(10,'shipped');
Query OK, 1 row affected (0.00 sec)

mysql> select * from orders;
+-----+-----+-----+-----+-----+
| OrderID | CustomerID | OrderDate | TotalAmount | status |
+-----+-----+-----+-----+-----+
| 10 | 1 | 2024-01-10 | 1094.50 | shipped |
| 30 | 3 | 2024-01-12 | 6583.50 | pending |
| 40 | 4 | 2024-01-13 | 10978.00 | pending |
| 50 | 5 | 2024-01-14 | 16472.50 | pending |
| 60 | 6 | 2024-01-15 | 23067.00 | pending |
| 70 | 7 | 2024-01-16 | 30761.50 | pending |
| 80 | 8 | 2024-01-17 | 0.00 | pending |
| 90 | 9 | 2024-01-18 | 43956.00 | pending |
| 100 | 10 | 2024-01-19 | 54400.50 | pending |
| 110 | 11 | 2024-01-20 | 1299.00 | pending |
+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

12. Write an SQL query to calculate and update the number of orders placed by each customer in the "Customers" table based on the data in the "Orders" table

```
mysql> ALTER TABLE Customers
-> ADD COLUMN OrdersPlaced INT DEFAULT 0;
Query OK, 0 rows affected (0.02 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> select * from customers;
+-----+-----+-----+-----+-----+-----+
| CustomerID | FirstName | LastName | Email | Phone | Address | OrdersPlaced |
+-----+-----+-----+-----+-----+-----+
| 1 | a | k | srv@gmail.com | 11111111 | Patna | 0 |
| 2 | b | kb | b@gmail.com | 11111112 | khajha | 0 |
| 3 | c | kc | ac@gmail.com | 11111113 | jhajh3 | 0 |
| 4 | d | kd | ad@gmail.com | 11111114 | jhajhd | 0 |
| 5 | e | ke | ae@gmail.com | 11111115 | jhajhae | 0 |
| 6 | f | kf | af@gmail.com | 11111116 | jhajhaf | 0 |
| 7 | g | kg | ag@gmail.com | 11111117 | jhajhag | 0 |
| 8 | h | kh | ah@gmail.com | 11111118 | jhajhah | 0 |
| 9 | i | ki | ai@gmail.com | 11111119 | jhajhai | 0 |
| 10 | j | kj | aj@gmail.com | 111111110 | jhajhaj | 0 |
| 11 | Sourav | Kumar | Engineer@gmail.com | 9110166666 | Bihar | 0 |
+-----+-----+-----+-----+-----+-----+
11 rows in set (0.00 sec)
```

```
mysql> -- Update the number of orders placed by each customer in the Customers table
mysql> UPDATE Customers
-> SET OrdersPlaced = (
-> SELECT COUNT(*)
-> FROM Orders
-> WHERE Orders.CustomerID = Customers.CustomerID
-> );
Query OK, 10 rows affected (0.00 sec)
Rows matched: 11 Changed: 10 Warnings: 0
```



## Assignment 1 (SQL)

```
mysql> select * from customers;
```

| CustomerID | FirstName | LastName | Email              | Phone      | Address | OrdersPlaced |
|------------|-----------|----------|--------------------|------------|---------|--------------|
| 1          | a         | k        | srv@gmail.com      | 11111111   | Patna   | 1            |
| 2          | b         | kb       | b@gmail.com        | 11111112   | khajha  | 0            |
| 3          | c         | kc       | ac@gmail.com       | 11111113   | jhajh3  | 1            |
| 4          | d         | kd       | ad@gmail.com       | 11111114   | jhajhd  | 1            |
| 5          | e         | ke       | ae@gmail.com       | 11111115   | jhajhae | 1            |
| 6          | f         | kf       | af@gmail.com       | 11111116   | jhajhaf | 1            |
| 7          | g         | kg       | ag@gmail.com       | 11111117   | jhajhag | 1            |
| 8          | h         | kh       | ah@gmail.com       | 11111118   | jhajhah | 1            |
| 9          | i         | ki       | ai@gmail.com       | 11111119   | jhajhai | 1            |
| 10         | j         | kj       | aj@gmail.com       | 111111110  | jhajhaj | 1            |
| 11         | Sourav    | Kumar    | Engineer@gmail.com | 9110166666 | Bihar   | 1            |

```
11 rows in set (0.00 sec)
```

### Task 3. Aggregate functions, Having, Order By, GroupBy and Joins:

1. Write an SQL query to retrieve a list of all orders along with customer information (e.g., customer name) for each order.

```
mysql> select orders.orderID,customers.CustomerID,orders.OrderDate,orders.TotalAmount,customers.FirstName,customers.LastName from orders
-> join customers on orders.CustomerID=customers.CustomerID;
```

| orderID | CustomerID | OrderDate  | TotalAmount | FirstName | LastName |
|---------|------------|------------|-------------|-----------|----------|
| 10      | 1          | 2024-01-10 | 1094.50     | a         | k        |
| 30      | 3          | 2024-01-12 | 6583.50     | c         | kc       |
| 40      | 4          | 2024-01-13 | 10978.00    | d         | kd       |
| 50      | 5          | 2024-01-14 | 16472.50    | e         | ke       |
| 60      | 6          | 2024-01-15 | 23067.00    | f         | kf       |
| 70      | 7          | 2024-01-16 | 30761.50    | g         | kg       |
| 80      | 8          | 2024-01-17 | 0.00        | h         | kh       |
| 90      | 9          | 2024-01-18 | 43956.00    | i         | ki       |
| 100     | 10         | 2024-01-19 | 54400.50    | j         | kj       |
| 110     | 11         | 2024-01-20 | 1299.00     | Sourav    | Kumar    |

```
10 rows in set (0.00 sec)
```

2. Write an SQL query to find the total revenue generated by each electronic gadget product. Include the product name and the total revenue

```
mysql> select products.productID,products.ProductName,SUM(orderdetails.Quantity*products.price) AS totalrevenue from products
-> join orderdetails ON products.productID=orderdetails.productID
-> join orders ON orderdetails.orderID=orders.orderID
-> group by products.productID,products.productName;
```

| productID | ProductName | totalrevenue |
|-----------|-------------|--------------|
| 1         | laptop      | 1094.50      |
| 3         | smartphone  | 6583.50      |
| 4         | heater      | 10978.00     |
| 5         | tab         | 16472.50     |
| 6         | monitor     | 23067.00     |
| 7         | cpu         | 30761.50     |
| 8         | mouse       | 0.00         |
| 9         | keyboard    | 43956.00     |
| 10        | television  | 54400.50     |

```
9 rows in set (0.00 sec)
```

3. Write an SQL query to list all customers who have made at least one purchase. Include their names and contact information.

```
mysql> select customers.CustomerID,customers.FirstName,customers.LastName,customers.Email,customers.Phone,customers.Address from customers
-> join orders on customers.CustomerID=orders.OrderID
-> group by customers.CustomerID,customers.FirstName,customers.LastName,customers.Email,customers.Phone;

+-----+-----+-----+-----+-----+-----+
| CustomerID | FirstName | LastName | Email | Phone | Address |
+-----+-----+-----+-----+-----+-----+
| 10 | j | kj | aj@gmail.com | 1111111110 | jhajhaj |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

4. Write an SQL query to find the most popular electronic gadget, which is the one with the highest total quantity ordered. Include the product name and the total quantity ordered

```
mysql> SELECT
-> Products.ProductName,
-> SUM(OrderDetails.Quantity) AS TotalQuantityOrdered
-> FROM
-> Products
-> JOIN
-> OrderDetails ON Products.ProductID = OrderDetails.ProductID
-> GROUP BY
-> Products.ProductName
-> ORDER BY
-> TotalQuantityOrdered DESC
-> LIMIT 1;

+-----+-----+
| ProductName | TotalQuantityOrdered |
+-----+-----+
| television | 45 |
+-----+-----+
1 row in set (0.00 sec)
```

5. Write an SQL query to retrieve a list of electronic gadgets along with their corresponding categories

Ans:- I have consider and use description instead of category---

```
mysql> SELECT
-> Products.ProductID,Products.ProductName,products.description
-> FROM
-> Products
-> WHERE
-> products.description = 'low-pef';

+-----+-----+-----+
| ProductID | ProductName | description |
+-----+-----+-----+
| 3 | smartphone | low-pef |
| 6 | monitor | low-pef |
| 8 | mouse | low-pef |
+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> |
```

6. Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value

```
mysql> SELECT
->     Customers.CustomerID,
->     Customers.FirstName,
->     AVG(Orders.TotalAmount) AS AverageOrderValue
-> FROM
->     Customers
-> JOIN
->     Orders ON Customers.CustomerID = Orders.CustomerID
-> GROUP BY
->     Customers.CustomerID, Customers.FirstName;
```

| CustomerID | FirstName | AverageOrderValue |
|------------|-----------|-------------------|
| 1          | a         | 1094.500000       |
| 3          | c         | 6583.500000       |
| 4          | d         | 10978.000000      |
| 5          | e         | 16472.500000      |
| 6          | f         | 23067.000000      |
| 7          | g         | 30761.500000      |
| 8          | h         | 0.000000          |
| 9          | i         | 43956.000000      |
| 10         | j         | 54400.500000      |
| 11         | Sourav    | 1299.000000       |

10 rows in set (0.00 sec)

7. Write an SQL query to find the order with the highest total revenue. Include the order ID, customer information, and the total revenue.

```
mysql> SELECT
->     Orders.OrderID,
->     Customers.CustomerID,
->     Customers.FirstName,
->     Customers.Phone,
->     SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue
-> FROM
->     Orders
-> JOIN
->     Customers ON Orders.CustomerID = Customers.CustomerID
-> JOIN
->     OrderDetails ON Orders.OrderID = OrderDetails.OrderID
-> JOIN
->     Products ON OrderDetails.ProductID = Products.ProductID
-> GROUP BY
->     Orders.OrderID, Customers.CustomerID, Customers.FirstName, Custom
ers.Phone
-> ORDER BY
->     TotalRevenue DESC
-> LIMIT 1;
```

| OrderID | CustomerID | FirstName | Phone      | TotalRevenue |
|---------|------------|-----------|------------|--------------|
| 100     | 10         | j         | 1111111110 | 54400.50     |

1 row in set (0.00 sec)

8. Write an SQL query to list electronic gadgets and the number of times each product has been ordered

```
mysql> SELECT
->     Products.ProductID,
->     Products.ProductName,
->     COUNT(OrderDetails.OrderID) AS NumberOfOrders
-> FROM
->     Products
-> JOIN
->     OrderDetails ON Products.ProductID = OrderDetails.ProductID
-> GROUP BY
->     Products.ProductID, Products.ProductName;
```

| ProductID | ProductName | NumberOfOrders |
|-----------|-------------|----------------|
| 1         | laptop      | 1              |
| 3         | smartphone  | 1              |
| 4         | heater      | 1              |
| 5         | tab         | 1              |
| 6         | monitor     | 1              |
| 7         | cpu         | 1              |
| 8         | mouse       | 1              |
| 9         | keyboard    | 1              |
| 10        | television  | 1              |

9 rows in set (0.00 sec)

9. Write an SQL query to find customers who have purchased a specific electronic gadget product. Allow users to input the product name as a parameter.

```
mysql>
mysql> CREATE PROCEDURE finds1product(IN p_ProductName VARCHAR(255))
-> BEGIN
->     SELECT
->         Customers.CustomerID,
->         Customers.FirstName,
->         Customers.phone,
->         products.ProductName
->     FROM
->         Customers
->     JOIN
->         Orders ON Customers.CustomerID = Orders.CustomerID
->     JOIN
->         OrderDetails ON Orders.OrderID = OrderDetails.OrderID
->     JOIN
->         Products ON OrderDetails.ProductID = Products.ProductID
->     WHERE
->         Products.ProductName = p_ProductName;
-> END @@
```

Query OK, 0 rows affected (0.01 sec)

```
mysql>
mysql> DELIMITER ;
mysql> call finds1product('keyboard');
```

| CustomerID | FirstName | phone     | ProductName |
|------------|-----------|-----------|-------------|
| 9          | i         | 111111119 | keyboard    |

1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

10. Write an SQL query to calculate the total revenue generated by all orders placed within a specific time period. Allow users to input the start and end dates as parameters.

```
mysql> DELIMITER **
mysql> CREATE PROCEDURE CalculateTotalRevenue(
    ->     IN p_StartDate DATE,
    ->     IN p_EndDate DATE
    -> )
    -> BEGIN
    ->     SELECT
    ->         SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue
    ->     FROM
    ->         Orders
    ->     JOIN
    ->         OrderDetails ON Orders.OrderID = OrderDetails.OrderID
    ->     JOIN
    ->         Products ON OrderDetails.ProductID = Products.ProductID
    ->     WHERE
    ->         Orders.OrderDate BETWEEN p_StartDate AND p_EndDate;
    -> END **
ERROR 1304 (42000): PROCEDURE CalculateTotalRevenue already exists
mysql>
mysql> DELIMITER ;
mysql> CALL CalculateTotalRevenue('2022-01-01', '2022-12-31');
+-----+
| TotalRevenue |
+-----+
|          NULL |
+-----+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

#### Task 4. Subquery and its type:

1. Write an SQL query to find out which customers have not placed any orders.

```
mysql> SELECT
    ->     CustomerID,
    ->     FirstName,
    ->     phone
    -> FROM
    ->     Customers
    -> WHERE
    ->     CustomerID NOT IN (SELECT DISTINCT CustomerID FROM Orders);
+-----+-----+-----+
| CustomerID | FirstName | phone    |
+-----+-----+-----+
|          2 | b        | 11111112 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

2. Write an SQL query to find the total number of products available for sale.

```
mysql> select count(*) As Totalproduct from products;
+-----+
| Totalproduct |
+-----+
|          11 |
+-----+
1 row in set (0.00 sec)
```

3. Write an SQL query to calculate the total revenue generated by TechShop

```
mysql> select sum(orderdetails.Quantity*products.price) As totalrevenue
-> from orderdetails
-> JOIN products ON orderdetails.productID=products.productID;
+-----+
| totalrevenue |
+-----+
| 187313.50 |
+-----+
1 row in set (0.00 sec)
```

4. Write an SQL query to calculate the average quantity ordered for products in a specific category. Allow users to input the category name as a parameter.

```
mysql> DELIMITER &&
mysql> CREATE PROCEDURE CalculateAvgQuantity(IN p_DescriptionName VARCHAR(255))
-> BEGIN
->     SELECT
->         AVG(OrderDetails.Quantity) AS AverageQuantity
->     FROM
->         OrderDetails
->     JOIN
->         Products ON OrderDetails.ProductID = Products.ProductID
->     WHERE
->         Products.Description = p_DescriptionName;
-> END &&
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql> CALL CalculateAvgQuantity('high-pef');
+-----+
| AverageQuantity |
+-----+
| 26.2500 |
+-----+
1 row in set (0.00 sec)
```

5. Write an SQL query to calculate the total revenue generated by a specific customer. Allow users to input the customer ID as a parameter.

```
mysql> CREATE PROCEDURE RevenueByCustomer(IN p_CustomerID INT)
-> BEGIN
->     SELECT
->         SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue
->     FROM
->         Orders
->     JOIN
->         OrderDetails ON Orders.OrderID = OrderDetails.OrderID
->     JOIN
->         Products ON OrderDetails.ProductID = Products.ProductID
->     WHERE
->         Orders.CustomerID = p_CustomerID;
-> END $$
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql> CALL RevenueByCustomer(10);
+-----+
| TotalRevenue |
+-----+
|      54400.50 |
+-----+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

6. Write an SQL query to find the customers who have placed the most orders. List their names and the number of orders they've placed

```
mysql> SELECT
->     Customers.CustomerID,
->     Customers.FirstName,
->     COUNT(Orders.OrderID) AS NumberOfOrders
-> FROM
->     Customers
-> JOIN
->     Orders ON Customers.CustomerID = Orders.CustomerID
-> GROUP BY
->     Customers.CustomerID, Customers.FirstName
-> ORDER BY
->     NumberOfOrders DESC
-> LIMIT 1;
+-----+-----+-----+
| CustomerID | FirstName | NumberOfOrders |
+-----+-----+-----+
|          1 | a         |              1 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

7. Write an SQL query to find the most popular product category, which is the one with the highest total quantity ordered across all orders

```
mysql> SELECT
->     products.productName,
->     SUM(OrderDetails.Quantity) AS TotalQuantityOrdered
-> FROM
->     products
-> JOIN
->     OrderDetails ON Products.ProductID = OrderDetails.ProductID
-> GROUP BY
->     products.productName
-> ORDER BY
->     TotalQuantityOrdered DESC
-> LIMIT 1;
```

| productName | TotalQuantityOrdered |
|-------------|----------------------|
| television  | 45                   |

1 row in set (0.00 sec)

8. Write an SQL query to find the customer who has spent the most money (highest total revenue) on electronic gadgets. List their name and total spending

```
mysql> SELECT
->     Customers.CustomerID,
->     Customers.FirstName,
->     SUM(OrderDetails.Quantity * Products.Price) AS TotalSpending
-> FROM
->     Customers
-> JOIN
->     Orders ON Customers.CustomerID = Orders.CustomerID
-> JOIN
->     OrderDetails ON Orders.OrderID = OrderDetails.OrderID
-> JOIN
->     Products ON OrderDetails.ProductID = Products.ProductID
-> GROUP BY
->     Customers.CustomerID, Customers.FirstName
-> ORDER BY
->     TotalSpending DESC
-> LIMIT 1;
```

| CustomerID | FirstName | TotalSpending |
|------------|-----------|---------------|
| 10         | j         | 54400.50      |

1 row in set (0.00 sec)



**9. Write an SQL query to calculate the average order value (total revenue divided by the number of orders) for all customers**

```
mysql> SELECT
->   Customers.CustomerID,
->   Customers.FirstName,
->   COUNT(DISTINCT Orders.OrderID) AS NumberOfOrders,
->   SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue,
->   SUM(OrderDetails.Quantity * Products.Price) / COUNT(DISTINCT Orders.OrderID) AS AverageOrderValue
-> FROM
->   Customers
-> LEFT JOIN
->   Orders ON Customers.CustomerID = Orders.CustomerID
-> LEFT JOIN
->   OrderDetails ON Orders.OrderID = OrderDetails.OrderID
-> LEFT JOIN
->   Products ON OrderDetails.ProductID = Products.ProductID
-> GROUP BY
->   Customers.CustomerID, Customers.FirstName
-> ORDER BY
->   AverageOrderValue DESC;
```

| CustomerID | FirstName | NumberOfOrders | TotalRevenue | AverageOrderValue |
|------------|-----------|----------------|--------------|-------------------|
| 10         | j         | 1              | 54400.50     | 54400.500000      |
| 9          | i         | 1              | 43956.00     | 43956.000000      |
| 7          | g         | 1              | 30761.50     | 30761.500000      |
| 6          | f         | 1              | 23067.00     | 23067.000000      |
| 5          | e         | 1              | 16472.50     | 16472.500000      |
| 4          | d         | 1              | 10978.00     | 10978.000000      |
| 3          | c         | 1              | 6583.50      | 6583.500000       |
| 1          | a         | 1              | 1094.50      | 1094.500000       |
| 8          | h         | 1              | 0.00         | 0.000000          |
| 2          | b         | 0              | NULL         | NULL              |
| 11         | Sourav    | 1              | NULL         | NULL              |

11 rows in set (0.00 sec)

10. Write an SQL query to find the total number of orders placed by each customer and list their names along with the order count.

```
mysql> SELECT
->     Customers.CustomerID,
->     Customers.FirstName,
->     COUNT(Orders.OrderID) AS OrderCount
-> FROM
->     Customers
-> LEFT JOIN
->     Orders ON Customers.CustomerID = Orders.CustomerID
-> GROUP BY
->     Customers.CustomerID, Customers.FirstName
-> ORDER BY
->     OrderCount DESC;
```

| CustomerID | FirstName | OrderCount |
|------------|-----------|------------|
| 1          | a         | 1          |
| 3          | c         | 1          |
| 4          | d         | 1          |
| 5          | e         | 1          |
| 6          | f         | 1          |
| 7          | g         | 1          |
| 8          | h         | 1          |
| 9          | i         | 1          |
| 10         | j         | 1          |
| 11         | Sourav    | 1          |
| 2          | b         | 0          |

11 rows in set (0.00 sec)

## Assignment 1 (SQL)