ASSIGNMENT TITLE:

TechShop : MySQL Assignment – 1

SUBMITED BY:

BALAKUMARAN P

DATE OF SUBMISSION:

22/01/2024

TechShop : MySQL Assignment - 1

Task:1. Database Design: 1. Create the database named "TechShop"

```
mysql> create database TechShop;
Query OK, 1 row affected (0.02 sec)

mysql> use TechShop;
Database changed
mysql>
```

2. Define the schema for the Customers, Products, Orders, OrderDetails and Inventory tables based on the provided Schema.

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

```
mysql> CREATE TABLE Products (
    -> ProductId INT PRIMARY KEY,
    -> ProductName TEXT,
    -> Description TEXT,
    -> Price BIGINT
    -> );
Query OK, 0 rows affected (0.04 sec)
```

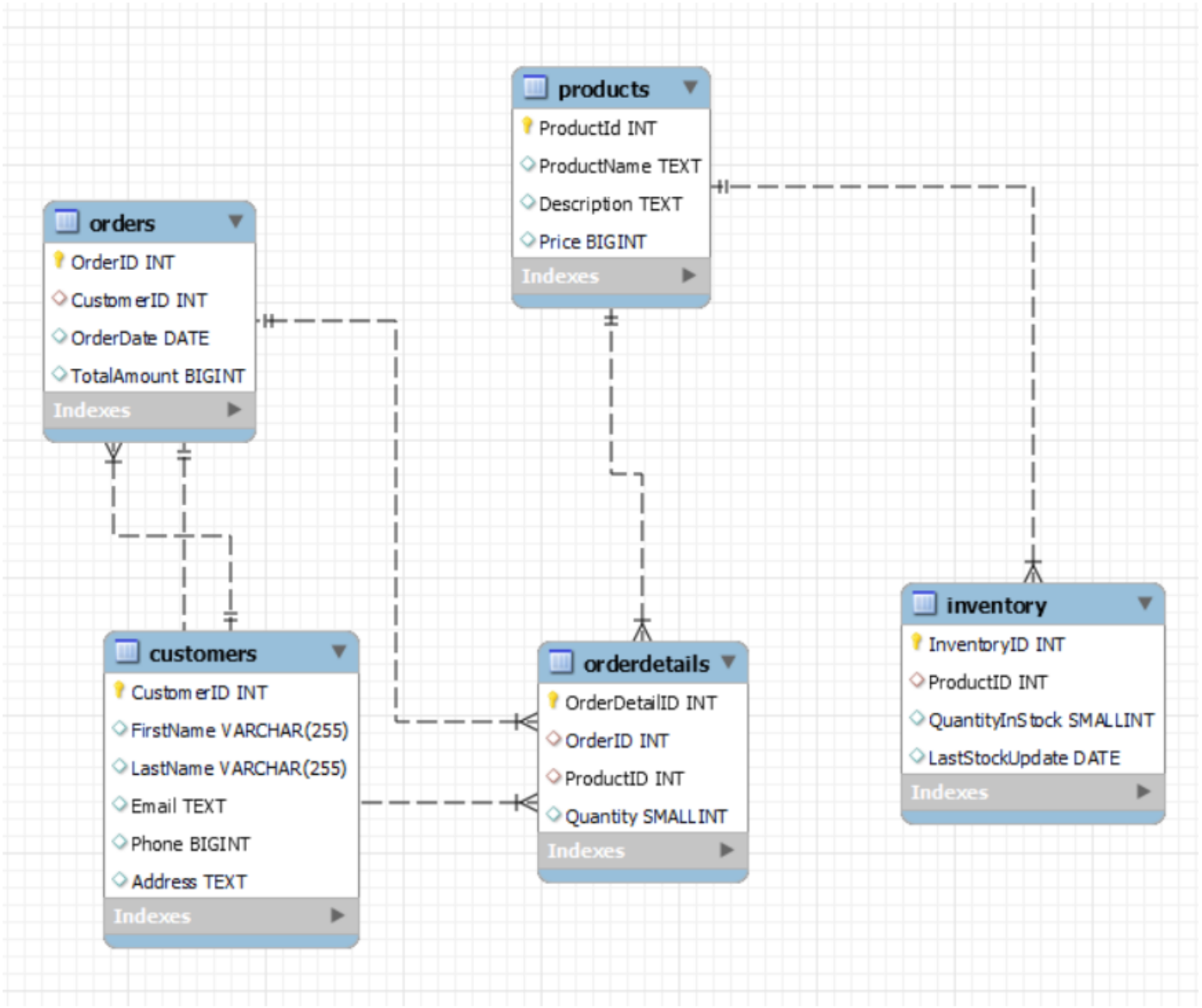
```
mysql> CREATE TABLE Orders (
    -> OrderID INT PRIMARY KEY,
    -> CustomerID INT,
    -> OrderDate DATE,
    -> TotalAmount BIGINT,
    -> FOREIGN KEY (CustomerId) REFERENCES Customers(CustomerID)
    -> );
Query OK, 0 rows affected (0.07 sec)
```

```
mysql> CREATE TABLE OrderDetails (
    -> OrderDetailID INT PRIMARY KEY,
    -> OrderID INT,
    -> ProductID INT,
    -> Quantity SMALLINT,
    -> FOREIGN KEY(OrderID) REFERENCES Orders(OrderID),
    -> FOREIGN KEY(ProductID) REFERENCES Products(ProductID)
    -> );
Query OK, 0 rows affected (0.09 sec)

mysql> CREATE TABLE Inventory(
    -> InventoryID INT PRIMARY KEY,
    -> ProductID INT,
    -> QuantityInStock SMALLINT,
    -> LastStockUpdate SMALLINT,
    -> FOREIGN KEY(ProductID) REFERENCES Products(ProductID)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> desc Customers;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| CustomerID | int         | NO   | PRI | NULL    |       |
| FirstName  | varchar(255) | YES  |     | NULL    |       |
| LastName   | varchar(255) | YES  |     | NULL    |       |
| Email      | text        | YES  |     | NULL    |       |
| Phone      | bigint      | YES  |     | NULL    |       |
| Address    | text        | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
6 rows in set (0.01 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 | bigint | 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      | smallint | YES  |     | NULL    |       |
+---------------+----------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> Desc Products;
+-------------+--------+------+-----+---------+-------+
| Field       | Type   | Null | Key | Default | Extra |
+-------------+--------+------+-----+---------+-------+
| ProductId   | int    | NO   | PRI | NULL    |       |
| ProductName | text   | YES  |     | NULL    |       |
| Description | text   | YES  |     | NULL    |       |
| Price       | bigint | 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 | smallint | YES  |     | NULL    |       |
| LastStockUpdate | date     | YES  |     | NULL    |       |
+-----------------+----------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

3. Create an ERD (Entity Relationship Diagram) for the database.



4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.

Customers Table:

Primary Key (CustomerID): Uniquely identifies customers.

Products Table:

Primary Key (ProductID): Uniquely identifies products.

Orders Table:

Primary Key (OrderID): Uniquely identifies orders.

Foreign Key (CustomerID): Links orders to customers.

OrderDetails Table:

Primary Key (OrderDetailID): Uniquely identifies order details.

Foreign Keys (OrderID, ProductID): Link order details to orders and products.

Inventory Table:

Primary Key (InventoryID): Uniquely identifies inventory records.

Foreign Key (ProductID): Links inventory to products.

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

```
mysql> CREATE TABLE Products (
    -> ProductId INT PRIMARY KEY,
    -> ProductName TEXT,
    -> Description TEXT,
    -> Price BIGINT
    -> );
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> CREATE TABLE Orders (
    -> OrderID INT PRIMARY KEY,
    -> CustomerID INT,
    -> OrderDate DATE,
    -> TotalAmount BIGINT,
    -> FOREIGN KEY (CustomerId) REFERENCES Customers(CustomerID)
    -> );
Query OK, 0 rows affected (0.07 sec)
```

```
mysql> CREATE TABLE OrderDetails (
    -> OrderDetailID INT PRIMARY KEY,
    -> OrderID INT,
    -> ProductID INT,
    -> Quantity SMALLINT,
    -> FOREIGN KEY(OrderID) REFERENCES Orders(OrderID),
    -> FOREIGN KEY(ProductID) REFERENCES Products(ProductID)
    -> );
Query OK, 0 rows affected (0.09 sec)

mysql> CREATE TABLE Inventory(
    -> InventoryID INT PRIMARY KEY,
    -> ProductID INT,
    -> QuantityInStock SMALLINT,
    -> LastStockUpdate SMALLINT,
    -> FOREIGN KEY(ProductID) REFERENCES Products(ProductID)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

5. Insert at least 10 sample records into each of the following tables. a. Customers b. Products c. Orders d. OrderDetails e. Inventory

```
mysql> INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address) VALUES
    ->     (4, 'John', 'Doe', 'john.doe@example.com', 1234567890, 'TamilNadu'),
    ->     (5, 'Charlie', 'Brown', 'charlie.brown@example.com', 1112223333, 'Andhra Pradesh'),
    ->     (6, 'Eva', 'Miller', 'eva.miller@example.com', 9998887777, 'Pondicherry'),
    ->     (7, 'Frank', 'Davis', 'frank.davis@example.com', 3334445555, 'Chennai'),
    ->     (8, 'Grace', 'Thomas', 'grace.thomas@gmail.com', 7776668888, 'Hydrebad'),
    ->     (9, 'Harry', 'Anderson', 'harry.anderson@gmail.com', 2225554444, 'Delhi'),
    ->     (10, 'Ivy', 'Harris', 'ivy.harris@example.com', 6669991111, 'Pondicherry');
Query OK, 7 rows affected (0.02 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql> select * from Customers;
+------------+-------------+----------+--------------------------+------------+----------------+
| CustomerID | FirstName   | LastName | Email                    | Phone      | Address        |
+------------+-------------+----------+--------------------------+------------+----------------+
|          1 | Balakumaran | P        | balabkkumaran@gmail.com  | 6382474871 | Pondicherry    |
|          2 | John        | Snow     | John@gmail.com           | 7382574871 | Kerala         |
|          3 | Arya        | Stark    | aryaw@gmail.com          |  938474871 | Kerala         |
|          4 | John        | Doe      | john.doe@example.com     | 1234567890 | TamilNadu      |
|          5 | Charlie     | Brown    | charlie.brown@example.com | 1112223333 | Andhra Pradesh |
|          6 | Eva         | Miller   | eva.miller@example.com   | 9998887777 | Pondicherry    |
|          7 | Frank       | Davis    | frank.davis@example.com  | 3334445555 | Chennai        |
|          8 | Grace       | Thomas   | grace.thomas@gmail.com   | 7776668888 | Hydrebad       |
|          9 | Harry       | Anderson | harry.anderson@gmail.com | 2225554444 | Delhi          |
|         10 | Ivy         | Harris   | ivy.harris@example.com   | 6669991111 | Pondicherry    |
+------------+-------------+----------+--------------------------+------------+----------------+
10 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Products (ProductId, ProductName, Description, Price) VALUES
    ->     (101, 'Laptop', 'High-performance laptop', 1200),
    ->     (102, 'Smartphone', 'Latest smartphone model', 1800),
    ->     (103, 'Headphones', 'Noise-canceling headphones', 550),
    ->     (104, 'Tablet', '10-inch touchscreen tablet', 3100),
    ->     (105, 'Digital Camera', '20MP digital camera', 2150),
    ->     (106, 'Wireless Mouse', 'Ergonomic wireless mouse', 430),
    ->     (107, 'External Hard Drive', '1TB external hard drive', 480),
    ->     (108, 'Bluetooth Speaker', 'Portable Bluetooth speaker', 950),
    ->     (109, 'Fitness Tracker', 'Water-resistant fitness tracker', 970),
    ->     (110, 'Gaming Console', 'Next-gen gaming console', 9500);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Products;
+-----------+---------------------+---------------------------------+-------+
| ProductId | ProductName         | Description                     | Price |
+-----------+---------------------+---------------------------------+-------+
|       101 | Laptop              | High-performance laptop         |  1200 |
|       102 | Smartphone          | Latest smartphone model         |  1800 |
|       103 | Headphones          | Noise-canceling headphones      |   550 |
|       104 | Tablet              | 10-inch touchscreen tablet      |  3100 |
|       105 | Digital Camera      | 20MP digital camera             |  2150 |
|       106 | Wireless Mouse      | Ergonomic wireless mouse        |   430 |
|       107 | External Hard Drive | 1TB external hard drive         |   480 |
|       108 | Bluetooth Speaker   | Portable Bluetooth speaker      |   950 |
|       109 | Fitness Tracker     | Water-resistant fitness tracker |   970 |
|       110 | Gaming Console      | Next-gen gaming console         |  9500 |
+-----------+---------------------+---------------------------------+-------+
10 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount) VALUES
    ->     (1, 1, '2024-01-12', 150),
    ->     (2, 2, '2024-01-13', 200),
    ->     (3, 3, '2024-01-14', 100),
    ->     (4, 4, '2024-01-15', 300),
    ->     (5, 5, '2024-01-16', 250),
    ->     (6, 6, '2024-01-17', 180),
    ->     (7, 7, '2024-01-18', 220),
    ->     (8, 8, '2024-01-19', 120),
    ->     (9, 9, '2024-01-20', 90),
    ->     (10, 10, '2024-01-21', 350);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Orders;
+---------+------------+------------+-------------+
| OrderID | CustomerID | OrderDate  | TotalAmount |
+---------+------------+------------+-------------+
|       1 |          1 | 2024-01-12 |         150 |
|       2 |          2 | 2024-01-13 |         200 |
|       3 |          3 | 2024-01-14 |         100 |
|       4 |          4 | 2024-01-15 |         300 |
|       5 |          5 | 2024-01-16 |         250 |
|       6 |          6 | 2024-01-17 |         180 |
|       7 |          7 | 2024-01-18 |         220 |
|       8 |          8 | 2024-01-19 |         120 |
|       9 |          9 | 2024-01-20 |          90 |
|      10 |         10 | 2024-01-21 |         350 |
+---------+------------+------------+-------------+
10 rows in set (0.00 sec)
```

```
mysql> INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity) VALUES
    ->     (1, 1, 101, 2),
    ->     (2, 1, 103, 1),
    ->     (3, 2, 105, 3),
    ->     (4, 3, 107, 1),
    ->     (5, 3, 102, 2),
    ->     (6, 4, 108, 1),
    ->     (7, 5, 104, 4),
    ->     (8, 6, 106, 1),
    ->     (9, 7, 109, 2),
    ->     (10, 8, 110, 3);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from OrderDetails;
+---------------+---------+-----------+----------+
| OrderDetailID | OrderID | ProductID | Quantity |
+---------------+---------+-----------+----------+
|             1 |       1 |       101 |        2 |
|             2 |       1 |       103 |        1 |
|             3 |       2 |       105 |        3 |
|             4 |       3 |       107 |        1 |
|             5 |       3 |       102 |        2 |
|             6 |       4 |       108 |        1 |
|             7 |       5 |       104 |        4 |
|             8 |       6 |       106 |        1 |
|             9 |       7 |       109 |        2 |
|            10 |       8 |       110 |        3 |
+---------------+---------+-----------+----------+
10 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Inventory (InventoryID, ProductID, QuantityInStock, LastStockUpdate) VALUES
    ->     (1, 101, 50, '2024-01-12'),
    ->     (2, 102, 30, '2024-01-13'),
    ->     (3, 103, 20, '2024-01-14'),
    ->     (4, 104, 40, '2024-01-15'),
    ->     (5, 105, 25, '2024-01-16'),
    ->     (6, 106, 15, '2024-01-17'),
    ->     (7, 107, 35, '2024-01-18'),
    ->     (8, 108, 10, '2024-01-19'),
    ->     (9, 109, 28, '2024-01-20'),
    ->     (10, 110, 18, '2024-01-21');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Inventory;
+-------------+-----------+-----------------+-----------------+
| InventoryID | ProductID | QuantityInStock | LastStockUpdate |
+-------------+-----------+-----------------+-----------------+
|           1 |       101 |              50 | 2024-01-12      |
|           2 |       102 |              30 | 2024-01-13      |
|           3 |       103 |              20 | 2024-01-14      |
|           4 |       104 |              40 | 2024-01-15      |
|           5 |       105 |              25 | 2024-01-16      |
|           6 |       106 |              15 | 2024-01-17      |
|           7 |       107 |              35 | 2024-01-18      |
|           8 |       108 |              10 | 2024-01-19      |
|           9 |       109 |              28 | 2024-01-20      |
|          10 |       110 |              18 | 2024-01-21      |
+-------------+-----------+-----------------+-----------------+
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,LastName,Email from Customers
    -> ;
+-------------+----------+--------------------------+
| FirstName   | LastName | Email                    |
+-------------+----------+--------------------------+
| Balakumaran | P        | balabkkumaran@gmail.com  |
| John        | Snow     | John@gmail.com           |
| Arya        | Stark    | aryaw@gmail.com          |
| John        | Doe      | john.doe@example.com     |
| Charlie     | Brown    | charlie.brown@example.com |
| Eva         | Miller   | eva.miller@example.com   |
| Frank       | Davis    | frank.davis@example.com  |
| Grace       | Thomas   | grace.thomas@gmail.com   |
| Harry       | Anderson | harry.anderson@gmail.com |
| Ivy         | Harris   | ivy.harris@example.com   |
+-------------+----------+--------------------------+
10 rows in set (0.00 sec)
```

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 Customers
    -> JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
+---------+------------+-------------+
| OrderID | OrderDate  | FirstName   |
+---------+------------+-------------+
|       1 | 2024-01-12 | Balakumaran |
|       2 | 2024-01-13 | John        |
|       3 | 2024-01-14 | Arya        |
|       4 | 2024-01-15 | John        |
|       5 | 2024-01-16 | Charlie     |
|       6 | 2024-01-17 | Eva         |
|       7 | 2024-01-18 | Frank       |
|       8 | 2024-01-19 | Grace       |
|       9 | 2024-01-20 | Harry       |
|      10 | 2024-01-21 | Ivy         |
+---------+------------+-------------+
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.

```
mysql> INSERT INTO Customers(CustomerID,FirstName,LastName,Email,Address) VA
LUES(11,'Karan','K','karan@gmail.com','Pondicherry');
Query OK, 1 row affected (0.01 sec)
```

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

```
mysql> UPDATE Products SET Price=Price*1.1;
Query OK, 10 rows affected (0.01 sec)
Rows matched: 10  Changed: 10  Warnings: 0

mysql> select Price from Products;
+--------+
| Price |
+--------+
|  1320 |
|  1980 |
|   605 |
|  3410 |
|  2365 |
|   473 |
|   528 |
|  1045 |
|  1067 |
| 10450 |
+--------+
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> DELIMITER //
mysql> CREATE PROCEDURE orders_delete(IN Order_ID INT)
    -> BEGIN
    -> DELETE orders, orderdetails
    -> FROM orders
    -> JOIN orderdetails ON orders.orderid = orderdetails.orderid
    -> WHERE orders.orderid = Order_ID;
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> CALL orders_delete(7);
Query OK, 2 rows affected (0.02 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> INSERT INTO Orders (OrderID,CustomerID,OrderDate,TotalAmount) VALUES
    -> (12,10,'2024-02-02',500);
Query OK, 1 row affected (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> DELIMITER //
mysql> CREATE PROCEDURE update_info(IN Cust_ID INT,IN email TEXT,IN adr TEXT)
    -> UPDATE Customers SET email=email,Address=adr
    -> WHERE CustomerID=Cust_ID;
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> DELIMITER ;
mysql> CALL update_info(11,'karan@gmail.com','Pondicherry');
Query OK, 1 row affected (0.02 sec)

mysql> select * from Customers;
+------------+-------------+----------+--------------------------+------------+----------------+
| CustomerID | FirstName   | LastName | Email                    | Phone      | Address        |
+------------+-------------+----------+--------------------------+------------+----------------+
|          1 | Balakumaran | P        | balabkkumaran@gmail.com  | 6382474871 | Pondicherry    |
|          2 | John        | Snow     | John@gmail.com           | 7382574871 | Kerala         |
|          3 | Arya        | Stark    | aryaw@gmail.com          |  938474871 | Kerala         |
|          4 | John        | Doe      | john.doe@example.com     | 1234567890 | TamilNadu      |
|          5 | Charlie     | Brown    | charlie.brown@example.com| 1112223333 | Andhra Pradesh |
|          6 | Eva         | Miller   | eva.miller@example.com   | 9998887777 | Pondicherry    |
|          7 | Frank       | Davis    | dhonifan@gmail.com       | 3334445555 | ERODE          |
|          8 | Grace       | Thomas   | grace.thomas@gmail.com   | 7776668888 | Hydrebad       |
|          9 | Harry       | Anderson | harry.anderson@gmail.com | 2225554444 | Delhi          |
|         10 | Ivy         | Harris   | ivy.harris@example.com   | 6669991111 | Pondicherry    |
|         11 | Karan       | K        | karan@gmail.com          |       NULL | Pondicherry    |
+------------+-------------+----------+--------------------------+------------+----------------+
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> UPDATE Orders o
    -> SET TotalAmount = (
    -> SELECT SUM(oi.Quantity * p.Price)
    -> FROM OrderDetails oi
    -> JOIN Products p ON p.ProductId = oi.ProductId
    -> WHERE oi.OrderID = o.OrderID
    -> );
Query OK, 2 rows affected (0.02 sec)
Rows matched: 10  Changed: 2  Warnings: 0

mysql> select * from orders;
+---------+------------+------------+-------------+
| OrderID | CustomerID | OrderDate  | TotalAmount |
+---------+------------+------------+-------------+
|       1 |          1 | 2024-01-12 |        3570 |
|       2 |          2 | 2024-01-13 |        7806 |
|       3 |          3 | 2024-01-14 |        4937 |
|       4 |          4 | 2024-01-15 |        1150 |
|       6 |          6 | 2024-01-17 |         520 |
|       7 |          7 | 2024-01-18 |        2348 |
|       8 |          8 | 2024-01-19 |       34485 |
|       9 |          9 | 2024-01-20 |        NULL |
|      10 |         10 | 2024-01-21 |        NULL |
|      12 |         10 | 2024-02-02 |        NULL |
+---------+------------+------------+-------------+
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 orders_delete(IN Order_ID INT)
    -> BEGIN
    -> DELETE orders, orderdetails
    -> FROM orders
    -> JOIN orderdetails ON orders.orderid = orderdetails.orderid
    -> WHERE orders.orderid = Order_ID;
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> CALL orders_delete(7);
Query OK, 2 rows affected (0.02 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> insert into Products values(12,'HP Latop','Windows latest version','40000');
Query OK, 1 row affected (0.02 sec)

mysql> select * from products;
+-----------+---------------------+----------------------------------+-------+
| ProductId | ProductName         | Description                      | Price |
+-----------+---------------------+----------------------------------+-------+
|        12 | HP Latop            | Windows latest version           | 40000 |
|       101 | Laptop              | High-performance laptop          |  1452 |
|       102 | Smartphone          | Latest smartphone model          |  2178 |
|       103 | Headphones          | Noise-canceling headphones       |   666 |
|       104 | Tablet              | 10-inch touchscreen tablet       |  3751 |
|       105 | Digital Camera      | 20MP digital camera              |  2602 |
|       106 | Wireless Mouse      | Ergonomic wireless mouse         |   520 |
|       107 | External Hard Drive | 1TB external hard drive          |   581 |
|       108 | Bluetooth Speaker   | Portable Bluetooth speaker       |  1150 |
|       109 | Fitness Tracker     | Water-resistant fitness tracker  |  1174 |
|       110 | Gaming Console      | Next-gen gaming console          | 11495 |
+-----------+---------------------+----------------------------------+-------+
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> use techshop;
Database changed
mysql> DELIMITER //
mysql> CREATE PROCEDURE status_update(IN o_id INT)
    -> BEGIN
    -> SELECT orderID,
    -> IF(orderDate<= CURDATE(), "shipped","pending") AS Status
    -> FROM Orders WHERE orderID=o_id;
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER ;
mysql> CALL status_update(4);
+---------+---------+
| orderID | Status  |
+---------+---------+
|       4 | shipped |
+---------+---------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (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> SELECT
    -> c.CustomerID,c.FirstName AS Name,
    -> COUNT(*) AS NO_ORDERS_PLACED
    -> FROM
    -> Customers c
    -> JOIN
    -> Orders o
    -> ON
    -> c.CustomerID=o.CustomerID
    -> GROUP BY
    -> CustomerID;
+------------+-------------+------------------+
| CustomerID | Name        | NO_ORDERS_PLACED |
+------------+-------------+------------------+
|          1 | Balakumaran |                1 |
|          2 | John        |                1 |
|          3 | Arya        |                1 |
|          4 | John        |                1 |
|          6 | Eva         |                1 |
|          7 | Frank       |                1 |
|          8 | Grace       |                1 |
|          9 | Harry       |                1 |
|         10 | Ivy         |                2 |
+------------+-------------+------------------+
9 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 o.OrderID,o.CustomerID,o.OrderDate,c.FirstName,c.Email
    -> FROM
    -> Customers c
    -> JOIN
    -> Orders o
    -> WHERE
    -> c.CustomerID=o.CustomerID;
+---------+------------+------------+-------------+--------------------------+
| OrderID | CustomerID | OrderDate  | FirstName   | Email                    |
+---------+------------+------------+-------------+--------------------------+
|       1 |          1 | 2024-01-12 | Balakumaran | balabkkumaran@gmail.com  |
|       2 |          2 | 2024-01-13 | John        | John@gmail.com           |
|       3 |          3 | 2024-01-14 | Arya        | aryaw@gmail.com          |
|       4 |          4 | 2024-01-15 | John        | john.doe@example.com     |
|       6 |          6 | 2024-01-17 | Eva         | eva.miller@example.com   |
|       7 |          7 | 2024-01-18 | Frank       | frank.davis@example.com  |
|       8 |          8 | 2024-01-19 | Grace       | grace.thomas@gmail.com   |
|       9 |          9 | 2024-01-20 | Harry       | harry.anderson@gmail.com |
|      10 |         10 | 2024-01-21 | Ivy         | ivy.harris@example.com   |
|      12 |         10 | 2024-02-02 | Ivy         | ivy.harris@example.com   |
+---------+------------+------------+-------------+--------------------------+
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
    -> p.ProductName,
    -> SUM(od.Quantity * p.Price) AS TotalRevenue
    -> FROM
    -> orderdetails od
    -> JOIN
    -> products p ON od.ProductID = p.ProductId
    -> GROUP BY p.ProductName;
+---------------------+--------------+
| ProductName         | TotalRevenue |
+---------------------+--------------+
| Laptop              |         2904 |
| Headphones          |          666 |
| Digital Camera      |         7806 |
| External Hard Drive |          581 |
| Smartphone          |         4356 |
| Bluetooth Speaker   |         1150 |
| Wireless Mouse      |          520 |
| Fitness Tracker     |         2348 |
| Gaming Console      |        34485 |
+---------------------+--------------+
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
    -> c.FirstName,
    -> c.Phone,
    -> c.Email,
    -> COUNT(o.OrderID) AS Purchase_Count
    -> FROM
    -> Customers c
    -> JOIN
    -> Orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    -> c.CustomerID, c.FirstName, c.Phone, c.Email
    -> HAVING Purchase_Count>=1;
+-------------+------------+-------------------------+----------------+
| FirstName   | Phone      | Email                   | Purchase_Count |
+-------------+------------+-------------------------+----------------+
| Balakumaran | 6382474871 | balabkkumaran@gmail.com |              1 |
| John        | 7382574871 | John@gmail.com          |              1 |
| Arya        |  938474871 | aryaw@gmail.com         |              1 |
| John        | 1234567890 | john.doe@example.com    |              1 |
| Eva         | 9998887777 | eva.miller@example.com  |              1 |
| Frank       | 3334445555 | frank.davis@example.com |              1 |
| Grace       | 7776668888 | grace.thomas@gmail.com  |              1 |
| Harry       | 2225554444 | harry.anderson@gmail.com|              1 |
| Ivy         | 6669991111 | ivy.harris@example.com  |              2 |
+-------------+------------+-------------------------+----------------+
9 rows 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
    -> p.productname AS Popular_gadget,
    -> oi.quantity AS total_quantity_ordered
    -> FROM
    -> products p
    -> JOIN
    -> orderdetails oi ON p.productid = oi.productid
    -> ORDER BY
    -> total_quantity_ordered DESC
    -> LIMIT 1;
+---------------+------------------------+
| Popular_gadget | total_quantity_ordered |
+---------------+------------------------+
| Digital Camera |                      3 |
+---------------+------------------------+
1 row in set (0.00 sec)
```

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

```
mysql> SELECT * FROM Products WHERE Description LIKE 'Electronic Gadgets';
+-----------+--------------------+--------------------+-------+
| ProductId | ProductName        | Description        | Price |
+-----------+--------------------+--------------------+-------+
|       103 | Headphones         | Electronic Gadgets |   666 |
|       104 | Tablet             | Electronic Gadgets |  3751 |
|       106 | Wireless Mouse     | Electronic Gadgets |   520 |
|       107 | External Hard Drive | Electronic Gadgets |   581 |
|       109 | Fitness Tracker    | Electronic Gadgets |  1174 |
+-----------+--------------------+--------------------+-------+
5 rows in set (0.01 sec)
```

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
    -> c.FirstName AS Custome_Name,
    -> AVG(o.TotalAmount) AS avg_order_value
    -> FROM
    -> customers c
    -> JOIN
    -> orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    -> c.CustomerID, c.FirstName, c.LastName;
+--------------+-----------------+
| Custome_Name | avg_order_value |
+--------------+-----------------+
| Balakumaran  |       3570.0000 |
| John         |       7806.0000 |
| Arya         |       4937.0000 |
| John         |       1150.0000 |
| Eva          |        520.0000 |
| Frank        |       2348.0000 |
| Grace        |      34485.0000 |
| Harry        |            NULL |
| Ivy          |            NULL |
+--------------+-----------------+
9 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
    -> c.FirstName,c.Email,o.OrderID,p.price*oi.Quantity AS Revenue_Order
    -> FROM
    -> Customers c
    -> JOIN
    -> Orders o
    -> ON
    -> c.CustomerID=o.CustomerID
    -> JOIN
    -> OrderDetails oi
    -> ON
    -> o.OrderID=oi.OrderID
    -> JOIN
    -> Products p
    -> ON
    -> p.ProductID=oi.ProductID ORDER BY Revenue_Order DESC;
+-------------+-------------------------+---------+---------------+
| FirstName   | Email                   | OrderID | Revenue_Order |
+-------------+-------------------------+---------+---------------+
| Grace       | grace.thomas@gmail.com  |       8 |         34485 |
| John        | John@gmail.com          |       2 |          7806 |
| Arya        | aryaw@gmail.com         |       3 |          4356 |
| Balakumaran | balabkkumaran@gmail.com |       1 |          2904 |
| Frank       | frank.davis@example.com |       7 |          2348 |
| John        | john.doe@example.com    |       4 |          1150 |
| Balakumaran | balabkkumaran@gmail.com |       1 |           666 |
| Arya        | aryaw@gmail.com         |       3 |           581 |
| Eva         | eva.miller@example.com  |       6 |           520 |
+-------------+-------------------------+---------+---------------+
9 rows 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
    -> p.ProductID,p.ProductName,p.Description,p.Price,COUNT(oi.ProductID)
    -> AS No_Times_Order_Placed
    -> FROM
    -> Products p
    -> JOIN
    -> OrderDetails oi
    -> ON
    -> p.ProductID=oi.ProductID
    -> GROUP BY oi.ProductID;
+-----------+--------------------+--------------------------------+-------+----------------------+
| ProductID | ProductName        | Description                    | Price | No_Times_Order_Placed |
+-----------+--------------------+--------------------------------+-------+----------------------+
|       101 | Laptop             | High-performance laptop        |  1452 |                    1 |
|       102 | Smartphone         | Latest smartphone model        |  2178 |                    1 |
|       103 | Headphones         | Noise-canceling headphones     |   666 |                    1 |
|       105 | Digital Camera     | 20MP digital camera            |  2602 |                    1 |
|       106 | Wireless Mouse     | Ergonomic wireless mouse       |   520 |                    1 |
|       107 | External Hard Drive | 1TB external hard drive       |   581 |                    1 |
|       108 | Bluetooth Speaker  | Portable Bluetooth speaker     |  1150 |                    1 |
|       109 | Fitness Tracker    | Water-resistant fitness tracker |  1174 |                    1 |
|       110 | Gaming Console     | Next-gen gaming console        | 11495 |                    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> use techshop;
Database changed
mysql> DELIMITER ??
mysql> CREATE PROCEDURE finding_customer(IN p_name TEXT)
    -> BEGIN
    -> SELECT c.FirstName, p.ProductName AS Product_Purchased
    -> FROM Customers c
    -> JOIN orders o ON c.CustomerID = o.CustomerID
    -> JOIN orderdetails oi ON oi.OrderID = o.OrderID
    -> JOIN Products p ON p.ProductID = oi.ProductID
    -> WHERE p.ProductName LIKE CONCAT('%', p_name, '%');
    -> END ??
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL finding_customer("Laptop");
+-------------+------------------+
| FirstName   | Product_Purchased |
+-------------+------------------+
| Balakumaran | Laptop           |
+-------------+------------------+
1 row in set (0.01 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 calc_revenue(IN start_date DATE,IN end_date DATE)
    -> BEGIN
    -> SELECT SUM(TotalAmount) AS Total_Amount
    -> FROM orders
    -> WHERE OrderDate BETWEEN start_date AND end_date;
    -> END **
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL calc_revenue('2024-01-01',CURDATE());
+--------------+
| Total_Amount |
+--------------+
|        17983 |
+--------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

Task 4. Subquery and its type:

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

```
mysql> select * from customers where customerid not in (select customerid from orders);
+------------+-----------+----------+--------------------------+------------+----------------+
| CustomerID | FirstName | LastName | Email                    | Phone      | Address        |
+------------+-----------+----------+--------------------------+------------+----------------+
|          5 | Charlie   | Brown    | charlie.brown@example.com | 1112223333 | Andhra Pradesh |
|          7 | Frank     | Davis    | dhonifan@gmail.com       | 3334445555 | ERODE          |
|          8 | Grace     | Thomas   | grace.thomas@gmail.com   | 7776668888 | Hydrebad       |
|         11 | Karan     | K        | karan@gmail.com          |       NULL | Pondicherry    |
+------------+-----------+----------+--------------------------+------------+----------------+
4 rows in set (0.01 sec)
```

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

```
mysql> SELECT *
    -> FROM inventory
    -> WHERE productid IN (SELECT productid FROM products);
+-------------+-----------+-----------------+-----------------+
| InventoryID | ProductID | QuantityInStock | LastStockUpdate |
+-------------+-----------+-----------------+-----------------+
|           1 |       101 |              50 | 2024-01-12      |
|           2 |       102 |              30 | 2024-01-13      |
|           3 |       103 |              20 | 2024-01-14      |
|           4 |       104 |              40 | 2024-01-15      |
|           5 |       105 |              25 | 2024-01-16      |
|           6 |       106 |              15 | 2024-01-17      |
|           7 |       107 |              35 | 2024-01-18      |
|           8 |       108 |              10 | 2024-01-19      |
|           9 |       109 |              28 | 2024-01-20      |
|          10 |       110 |              18 | 2024-01-21      |
+-------------+-----------+-----------------+-----------------+
10 rows in set (0.00 sec)
```

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

```
mysql> SELECT SUM(TotalAmount) AS Total_Revenue_Generated
    -> FROM Orders;
+-------------------------+
| Total_Revenue_Generated |
+-------------------------+
|                   54816 |
+-------------------------+
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 $$
mysql> CREATE PROCEDURE calc_avg_quantity(IN category_name TEXT)
    -> BEGIN
    -> SELECT AVG(od.Quantity) AS Average_Quantity
    -> FROM orderdetails od
    -> JOIN products p ON od.ProductID = p.ProductId
    -> WHERE p.Description LIKE CONCAT('%', category_name, '%');
    -> END $$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL calc_avg_quantity("Electronic Gadgets");
+------------------+
| Average_Quantity |
+------------------+
|           1.0000 |
+------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> CALL calc_avg_quantity("Laptop");
+------------------+
| Average_Quantity |
+------------------+
|           2.0000 |
+------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (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> DELIMITER //
mysql> CREATE PROCEDURE calc_rev(IN c_id INT)
    -> BEGIN
    -> SELECT SUM(od.Quantity * p.Price) AS Total_Revenue
    -> FROM orders o
    -> JOIN orderdetails od ON o.OrderID = od.OrderID
    -> JOIN products p ON od.ProductID = p.ProductId
    -> WHERE o.CustomerID = c_id;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL calc_rev(3);
+---------------+
| Total_Revenue |
+---------------+
|          4937 |
+---------------+
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
    -> c.CustomerID,
    -> c.FirstName,
    -> COUNT(o.OrderID) AS OrdersPlaced
    -> FROM
    -> customers c
    -> JOIN
    -> orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    -> c.CustomerID
    -> ORDER BY
    -> OrdersPlaced DESC
    -> LIMIT 1;
+------------+-----------+--------------+
| CustomerID | FirstName | OrdersPlaced |
+------------+-----------+--------------+
|         10 | Ivy       |            2 |
+------------+-----------+--------------+
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
    -> p.ProductName,p.Description,
    -> COUNT(oi.OrderID) AS ordered
    -> FROM
    -> Products p
    -> JOIN
    -> OrderDetails oi ON p.ProductID = oi.ProductID
    -> GROUP BY
    -> p.ProductID
    -> ORDER BY
    -> ordered DESC LIMIT 1;
+-------------+-------------------------+---------+
| ProductName | Description             | ordered |
+-------------+-------------------------+---------+
| Laptop      | High-performance laptop |       1 |
+-------------+-------------------------+---------+
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
    -> c.CustomerID,c.FirstName,
    -> o.TotalAmount AS TotalSpending
    -> FROM
    -> Customers c
    -> JOIN
    -> Orders o ON c.CustomerID = o.CustomerID
    -> JOIN
    -> OrderDetails od ON o.OrderID = od.OrderID
    -> JOIN
    -> Products p ON od.ProductID = p.ProductID
    -> WHERE p.Description LIKE '%speaker%'
    -> ORDER BY TotalSpending DESC
    -> LIMIT 1;
+------------+-----------+---------------+
| CustomerID | FirstName | TotalSpending |
+------------+-----------+---------------+
|          4 | John      |          1150 |
+------------+-----------+---------------+
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
    -> c.CustomerID,c.FirstName,
    -> SUM(o.TotalAmount) / COUNT(o.OrderID) AS AverageOrderValue
    -> FROM
    -> Customers c
    -> JOIN
    -> Orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    -> c.CustomerID
    -> ORDER BY
    -> AverageOrderValue DESC;
+------------+-------------+-------------------+
| CustomerID | FirstName   | AverageOrderValue |
+------------+-------------+-------------------+
|          8 | Grace       |        34485.0000 |
|          2 | John        |         7806.0000 |
|          3 | Arya        |         4937.0000 |
|          1 | Balakumaran |         3570.0000 |
|          7 | Frank       |         2348.0000 |
|          4 | John        |         1150.0000 |
|          6 | Eva         |          520.0000 |
|          9 | Harry       |              NULL |
|         10 | Ivy         |              NULL |
+------------+-------------+-------------------+
9 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
    -> c.CustomerID,c.FirstName,
    -> COUNT(o.OrderID) AS No_Orders_Placed
    -> FROM
    -> Customers c
    -> JOIN
    -> Orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    -> c.CustomerID
    -> ORDER BY
    -> No_Orders_Placed DESC;
+------------+-------------+------------------+
| CustomerID | FirstName   | No_Orders_Placed |
+------------+-------------+------------------+
|         10 | Ivy         |                2 |
|          1 | Balakumaran |                1 |
|          2 | John        |                1 |
|          3 | Arya        |                1 |
|          4 | John        |                1 |
|          6 | Eva         |                1 |
|          7 | Frank       |                1 |
|          8 | Grace       |                1 |
|          9 | Harry       |                1 |
+------------+-------------+------------------+
9 rows in set (0.00 sec)
```