Task:1. Database Design:

1. Create the database named "TechShop"

```
mysql> create database TechShop;
Query OK, 1 row affected (0.10 sec)
```

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

```
mysql> create table Customers( CustomerID int NOT NULL PRIMARY KEY,
    -> FirstName varchar(255),
    -> LastName varchar(255),
    -> Email varchar(255),
    -> Phone varchar(255),
    -> Address varchar(255));
Query OK, 0 rows affected (0.20 sec)
```

```
mysql> create table Products(ProductID int NOT NULL PRIMARY KEY,
    -> ProductName text,
    -> Description varchar(255),
    -> Price INT);
Query OK, 0 rows affected (0.12 sec)
```

```
mysql> create table Orders(OrderID int NOT NULL PRIMARY KEY,
    -> CustomerID int,
    -> FOREIGN KEY (CustomerID) REFERENCES Customers (CustomerID),
    -> OrderDate DATE,
    -> TotalAmount int);
Query OK, 0 rows affected (0.29 sec)
```

```
mysql> create table OrderDetails(OrderDetailID INT NOT NULL PRIMARY KEY,
    -> OrderID int,
    -> FOREIGN KEY (OrderID) REFERENCES orders(OrderID),
    -> ProductID INT,
    -> FOREIGN KEY (ProductID) REFERENCES Products (ProductID),
    -> Quantity INT);
Query OK, 0 rows affected (0.08 sec)
```

```
mysql> create table Inventory(InventoryID INT NOT NULL PRIMARY KEY,
    -> ProductID INT,
    -> FOREIGN KEY (ProductID) REFERENCES Products (ProductID),
    -> QuantityInStock INT,
    -> LastStockUpdate VARCHAR(255));
Query OK, 0 rows affected (0.05 sec)
```

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

a. Customers

b. Products

c. Orders

d. OrderDetails

```
mysql> INSERT INTO Customers(CustomerID, FirstName, LastName, Email, Phone, Address)
    -> VALUES(101, "Riya", "Sen", "rs@gmail.com", "56789090", "BBSR"),
    -> (201, "Deep", "Kumar", "rd@gmail.com", "67896899", "RKL"),
    -> (301, "Raj", "Dip", "rds@gmail.com", "67846312", "Rkl"),
    -> (401, "Pratap", "Sen", "Ps@gmail.com", "12346312", "Berhampur"),
    -> (501, "Prapti", "Sen", "prs@gmail.com", "16786312", "Rampur"),
    -> (601, "Arti", "Das", "ad@gmail.com", "16782342", "Sonpur"),
    -> (701, "Iti", "Dash", "itd@gmail.com", "34756982", "Raipur"),
    -> (801, "Dipa", "Das", "dd@gmail.com", "34767845", "BBSR"),
    -> (901, "Puja", "Rai", "pr@gmail.com", "78947845", "BBSR"),
    -> (1001, "Piku", "Pani", "pp@gmail.com", "89767845", "BBSR");
Query OK, 10 rows affected (0.07 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO Products(ProductID, ProductName, Description, Price)
    -> VALUES(1, "Fan", "cooling", 5000),
    -> (2, "AC", "cooling", 9000),
    -> (3, "Fridge", "store", 9500),
    -> (4, "Table", "desc", 3000),
    -> (5, "Lamp", "Light", 2000),
    -> (6, "Lamp-2", "medium size", 5000),
    -> (7, "Washing machine -1", "manual", 9000),
    -> (8, "Washing machine -2", "automatic", 15000),
    -> (9, "TV-1", "50 inches", 60000),
    -> (10, "TV-2", "64 inches", 90000);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO Orders(OrderID, CustomerID, OrderDate, TotalAmount)
    -> VALUES(11, 101, "2024-02-12", 5000),
    -> (21, 201, "2024-02-14",9000),
    -> (31, 301, "2024-02-24",9500),
    -> (41, 401, "2024-03-14",3000),
    -> (51, 501, "2024-03-20",2000),
    -> (61, 601, "2024-03-30",5000),
    -> (71, 701, "2024-03-28",9000),
    -> (81, 801, "2024-02-10",15000),
    -> (91, 901, "2024-02-15",60000),
    -> (101, 1001, "2024-03-15",90000);
Query OK, 10 rows affected (0.05 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO OrderDetails(OrderDetailID, OrderID, ProductID, Quantity)
    -> VALUES(110, 11, 1, 1),
    -> (220, 21, 2, 1),
    -> (330, 31, 3, 1),
    -> (440, 41, 4, 2),
    -> (550, 51, 5, 2),
    -> (660, 61, 6, 1),
    -> (770, 71, 7, 1),
    -> (880, 81, 8, 1),
    -> (990, 91, 9, 1),
    -> (1110, 101, 10, 1);
Query OK, 10 rows affected (0.03 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO Inventory(InventoryID, ProductID, QuantityInStock, LastStockUpdate)
    -> VALUES(550, 1, 10, '2024-01-12'),
    -> (551, 2, 55, '2024-01-13'),
    -> (552, 3, 45, '2024-01-13'),
    -> (553, 4, 40, '2024-01-15'),
    -> (554, 5, 20, '2024-01-16'),
    -> (555, 6, 35, '2024-01-17'),
    -> (556, 7, 45, '2024-01-17'),
    -> (557, 8, 45, '2024-01-18'),
    -> (558, 9, 15, '2024-01-18'),
    -> (559, 10, 45, '2024-01-20');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

Task 2

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

```
mysql> SELECT FirstName, LastName, Email FROM Customers;
+-----------+----------+---------------+
| FirstName | LastName | Email         |
+-----------+----------+---------------+
| Riya      | Sen      | rs@gmail.com  |
| Deep      | Kumar    | rd@gmail.com  |
| Raj       | Dip      | rds@gmail.com |
| Pratap    | Sen      | Ps@gmail.com  |
| Prapti    | Sen      | prs@gmail.com |
| Arti      | Das      | ad@gmail.com  |
| Iti       | Dash     | itd@gmail.com |
| Dipa      | Das      | dd@gmail.com  |
| Puja      | Rai      | pr@gmail.com  |
| Piku      | Pani     | pp@gmail.com  |
+-----------+----------+---------------+
10 rows in set (0.03 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, Customers.LastName
    -> FROM Orders, Customers
    -> WHERE Orders.CustomerID = Customers.CustomerID;
+---------+------------+-----------+----------+
| OrderID | OrderDate  | FirstName | LastName |
+---------+------------+-----------+----------+
|      11 | 2024-02-12 | Riya      | Sen      |
|      21 | 2024-02-14 | Deep      | Kumar    |
|      31 | 2024-02-24 | Raj       | Dip      |
|      41 | 2024-03-14 | Pratap    | Sen      |
|      51 | 2024-03-20 | Prapti    | Sen      |
|      61 | 2024-03-30 | Arti      | Das      |
|      71 | 2024-03-28 | Iti       | Dash     |
|      81 | 2024-02-10 | Dipa      | Das      |
|      91 | 2024-02-15 | Puja      | Rai      |
|     101 | 2024-03-15 | Piku      | Pani     |
+---------+------------+-----------+----------+
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, Phone, Address)
    -> VALUES(1101, "Tara", "Dey", "td@gmail.com", "895858427", "Rkl");
Query OK, 1 row affected (0.10 sec)
```

```
mysql> SELECT Price*1.10 FROM Products;
+------------+
| Price*1.10 |
+------------+
|    5500.00 |
|    9900.00 |
|   10450.00 |
|    3300.00 |
|    2200.00 |
|    5500.00 |
|    9900.00 |
|   16500.00 |
|   66000.00 |
|   99000.00 |
+------------+
10 rows in set (0.23 sec)
```

```
mysql> SELECT * FROM Customers
    -> ;
+------------+-----------+----------+---------------+-----------+-----------+
| CustomerID | FirstName | LastName | Email         | Phone     | Address   |
+------------+-----------+----------+---------------+-----------+-----------+
|        101 | Riya      | Sen      | rs@gmail.com  | 56789090  | BBSR      |
|        201 | Deep      | Kumar    | rd@gmail.com  | 67896899  | RKL       |
|        301 | Raj       | Dip      | rds@gmail.com | 67846312  | Rkl       |
|        401 | Pratap    | Sen      | Ps@gmail.com  | 12346312  | Berhampur |
|        501 | Prapti    | Sen      | prs@gmail.com | 16786312  | Rampur    |
|        601 | Arti      | Das      | ad@gmail.com  | 16782342  | Sonpur    |
|        701 | Iti       | Dash     | itd@gmail.com | 34756982  | Raipur    |
|        801 | Dipa      | Das      | dd@gmail.com  | 34767845  | BBSR      |
|        901 | Puja      | Rai      | pr@gmail.com  | 78947845  | BBSR      |
|       1001 | Piku      | Pani     | pp@gmail.com  | 89767845  | BBSR      |
|       1101 | Tara      | Dey      | td@gmail.com  | 895858427 | Rkl       |
+------------+-----------+----------+---------------+-----------+-----------+
11 rows in set (0.01 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 = 21;
Query OK, 1 row affected (0.03 sec)

mysql> DELETE FROM Orders
    -> WHERE OrderID = 21;
Query OK, 1 row affected (0.01 sec)
```

```
mysql> select * from Orders;
+---------+------------+------------+-------------+
| OrderID | CustomerID | OrderDate  | TotalAmount |
+---------+------------+------------+-------------+
|      11 |        101 | 2024-02-12 |        5000 |
|      31 |        301 | 2024-02-24 |        9500 |
|      41 |        401 | 2024-03-14 |        3000 |
|      51 |        501 | 2024-03-20 |        2000 |
|      61 |        601 | 2024-03-30 |        5000 |
|      71 |        701 | 2024-03-28 |        9000 |
|      81 |        801 | 2024-02-10 |       15000 |
|      91 |        901 | 2024-02-15 |       60000 |
|     101 |       1001 | 2024-03-15 |       90000 |
+---------+------------+------------+-------------+
9 rows in set (0.01 sec)

mysql> select * from OrderDetails;
+--------------+---------+-----------+----------+
| OrderDetailID | OrderID | ProductID | Quantity |
+--------------+---------+-----------+----------+
|          110 |      11 |         1 |        1 |
|          330 |      31 |         3 |        1 |
|          440 |      41 |         4 |        2 |
|          550 |      51 |         5 |        2 |
|          660 |      61 |         6 |        1 |
|          770 |      71 |         7 |        1 |
|          880 |      81 |         8 |        1 |
|          990 |      91 |         9 |        1 |
|         1110 |     101 |        10 |        1 |
+--------------+---------+-----------+----------+
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> INSERT INTO Orders(OrderID, CustomerID, OrderDate, TotalAmount)
    -> VALUES(111,1101, '2024-03-16', 1000000);
Query OK, 1 row affected (0.10 sec)

mysql> select * from Orders;
+---------+------------+------------+-------------+
| OrderID | CustomerID | OrderDate  | TotalAmount |
+---------+------------+------------+-------------+
|      11 |        101 | 2024-02-12 |        5000 |
|      31 |        301 | 2024-02-24 |        9500 |
|      41 |        401 | 2024-03-14 |        3000 |
|      51 |        501 | 2024-03-20 |        2000 |
|      61 |        601 | 2024-03-30 |        5000 |
|      71 |        701 | 2024-03-28 |        9000 |
|      81 |        801 | 2024-02-10 |       15000 |
|      91 |        901 | 2024-02-15 |       60000 |
|     101 |       1001 | 2024-03-15 |       90000 |
|     111 |       1101 | 2024-03-16 |     1000000 |
+---------+------------+------------+-------------+
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 = 'mnn@gmail.com', Address = 'Mumbai'
    -> WHERE CustomerID = 301;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

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> DELETE FROM OrderDetails
    -> WHERE OrderID IN(
    -> SELECT OrderID
    -> FROM Orders
    -> WHERE CustomerID = 1001);
Query OK, 1 row affected (0.10 sec)

mysql> DELETE FROM Orders
    -> WHERE CustomerID = 1001;
Query OK, 1 row affected (0.01 sec)

mysql> select * from Orders;
+---------+------------+------------+-------------+
| OrderID | CustomerID | OrderDate  | TotalAmount |
+---------+------------+------------+-------------+
|      11 |        101 | 2024-02-12 |        5000 |
|      31 |        301 | 2024-02-24 |        9500 |
|      41 |        401 | 2024-03-14 |        3000 |
|      51 |        501 | 2024-03-20 |        2000 |
|      61 |        601 | 2024-03-30 |        5000 |
|      71 |        701 | 2024-03-28 |        9000 |
|      81 |        801 | 2024-02-10 |       15000 |
|      91 |        901 | 2024-02-15 |       60000 |
|     111 |       1101 | 2024-03-16 |     1000000 |
+---------+------------+------------+-------------+
9 rows in set (0.01 sec)

mysql> select * from OrderDetails;
+--------------+---------+-----------+----------+
| OrderDetailID | OrderID | ProductID | Quantity |
+--------------+---------+-----------+----------+
|          110 |      11 |         1 |        1 |
|          330 |      31 |         3 |        1 |
|          440 |      41 |         4 |        2 |
|          550 |      51 |         5 |        2 |
|          660 |      61 |         6 |        1 |
|          770 |      71 |         7 |        1 |
|          880 |      81 |         8 |        1 |
|          990 |      91 |         9 |        1 |
+--------------+---------+-----------+----------+
8 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> INSERT INTO Products( ProductID, ProductName, Description, Price)
    -> VALUES(11, 'Vaccum Cleaner', 'cleaning', '6000');
Query OK, 1 row affected (0.01 sec)

mysql> select * from Products;
+-----------+--------------------+-------------+-------+
| ProductID | ProductName        | Description | Price |
+-----------+--------------------+-------------+-------+
|         1 | Fan                | cooling     |  5000 |
|         2 | AC                 | cooling     |  9000 |
|         3 | Fridge             | store       |  9500 |
|         4 | Table              | desc        |  3000 |
|         5 | Lamp               | Light       |  2000 |
|         6 | Lamp-2             | medium size |  5000 |
|         7 | Washing machine -1 | manual      |  9000 |
|         8 | Washing machine -2 | automatic   | 15000 |
|         9 | TV-1               | 50 inches   | 60000 |
|        10 | TV-2               | 64 inches   | 90000 |
|        11 | Vaccum Cleaner     | cleaning    |  6000 |
+-----------+--------------------+-------------+-------+
11 rows in set (0.00 sec)
```

Task 3

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.OrderDate, CONCAT(c.FirstName, ' ', c.LastName) AS CustomerName
    -> FROM Orders o
    -> JOIN Customers c ON o.CustomerID = c.CustomerID;
+---------+------------+--------------+
| OrderID | OrderDate  | CustomerName |
+---------+------------+--------------+
|      11 | 2024-02-12 | Riya Sen     |
|      31 | 2024-02-24 | Raj Dip      |
|      41 | 2024-03-14 | Pratap Sen   |
|      51 | 2024-03-20 | Prapti Sen   |
|      61 | 2024-03-30 | Arti Das     |
|      71 | 2024-03-28 | Iti Dash     |
|      81 | 2024-02-10 | Dipa Das     |
|      91 | 2024-02-15 | Puja Rai     |
|     111 | 2024-03-16 | Tara Dey     |
+---------+------------+--------------+
9 rows in set (0.07 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
    ->     Products p
    -> JOIN
    ->     OrderDetails od ON p.ProductID = od.ProductID
    -> WHERE
    ->     p.Description LIKE '%electronic%'
    -> GROUP BY
    ->     p.ProductID, p.ProductName;
Empty 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.LastName,
    ->     c.Email,
    ->     c.Phone
    -> FROM
    ->     Customers c
    -> JOIN
    ->     Orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    ->     c.CustomerID, c.FirstName, c.LastName, c.Email, c.Phone;
+-----------+----------+--------------+-----------+
| FirstName | LastName | Email        | Phone     |
+-----------+----------+--------------+-----------+
| Riya      | Sen      | rs@gmail.com | 56789090  |
| Raj       | Dip      | mnn@gmail.com | 67846312  |
| Pratap    | Sen      | Ps@gmail.com | 12346312  |
| Prapti    | Sen      | prs@gmail.com | 16786312  |
| Arti      | Das      | ad@gmail.com | 16782342  |
| Iti       | Dash     | itd@gmail.com | 34756982  |
| Dipa      | Das      | dd@gmail.com | 34767845  |
| Puja      | Rai      | pr@gmail.com | 78947845  |
| Tara      | Dey      | td@gmail.com | 895858427 |
+-----------+----------+--------------+-----------+
9 rows in set (0.01 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,
    ->     SUM(od.Quantity) AS TotalQuantityOrdered
    -> FROM
    ->     OrderDetails od
    -> JOIN
    ->     Products p ON od.ProductID = p.ProductID
    -> WHERE
    ->     p.ProductName = 'Lamp'
    -> GROUP BY
    ->     p.ProductName
    -> ORDER BY
    ->     TotalQuantityOrdered DESC
    -> LIMIT 1;
+-------------+----------------------+
| ProductName | TotalQuantityOrdered |
+-------------+----------------------+
| Lamp        |                    2 |
+-------------+----------------------+
1 row in set (0.15 sec)
```

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

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,
    ->     c.LastName,
    ->     AVG(od.Quantity * p.Price) AS AverageOrderValue
    -> 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
    -> GROUP BY
    ->     c.FirstName, c.LastName, c.CustomerID;
+-----------+----------+-------------------+
| FirstName | LastName | AverageOrderValue |
+-----------+----------+-------------------+
| Riya      | Sen      |         5000.0000 |
| Raj       | Dip      |         9500.0000 |
| Pratap    | Sen      |         6000.0000 |
| Prapti    | Sen      |         4000.0000 |
| Arti      | Das      |         5000.0000 |
| Iti       | Dash     |         9000.0000 |
| Dipa      | Das      |        15000.0000 |
| Puja      | Rai      |        60000.0000 |
+-----------+----------+-------------------+
8 rows in set (0.02 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
    ->     o.OrderID,
    ->     c.FirstName,
    ->     c.LastName,
    ->     SUM(od.Quantity * p.Price) AS TotalRevenue
    -> FROM
    ->     Orders o
    -> JOIN
    ->     Customers c ON o.CustomerID = c.CustomerID
    -> JOIN
    ->     OrderDetails od ON o.OrderID = od.OrderID
    -> JOIN
    ->     Products p ON od.ProductID = p.ProductID
    -> GROUP BY
    ->     o.OrderID, c.FirstName, c.LastName
    -> ORDER BY
    ->     TotalRevenue DESC
    -> LIMIT 1;
+---------+-----------+----------+--------------+
| OrderID | FirstName | LastName | TotalRevenue |
+---------+-----------+----------+--------------+
|      91 | Puja      | Rai      |        60000 |
+---------+-----------+----------+--------------+
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
    ->     p.ProductID,
    ->     p.ProductName,
    ->     p.Description,
    ->     COALESCE(NumberOfOrders, 0) AS NumberOfOrders
    -> FROM
    ->     Products p
    -> LEFT JOIN (
    ->     SELECT
    ->         ProductID,
    ->         COUNT(OrderDetailID) AS NumberOfOrders
    ->     FROM
    ->         OrderDetails
    ->     GROUP BY
    ->         ProductID
    -> ) AS OrderCounts ON p.ProductID = OrderCounts.ProductID;
+-----------+--------------------+-------------+----------------+
| ProductID | ProductName        | Description | NumberOfOrders |
+-----------+--------------------+-------------+----------------+
|         1 | Fan                | cooling     |              1 |
|         2 | AC                 | cooling     |              0 |
|         3 | Fridge             | store       |              1 |
|         4 | Table              | desc        |              1 |
|         5 | Lamp               | Light       |              1 |
|         6 | Lamp-2             | medium size |              1 |
|         7 | Washing machine -1 | manual      |              1 |
|         8 | Washing machine -2 | automatic   |              1 |
|         9 | TV-1               | 50 inches   |              1 |
|        10 | TV-2               | 64 inches   |              0 |
|        11 | Vaccum Cleaner     | cleaning    |              0 |
+-----------+--------------------+-------------+----------------+
11 rows in set (0.01 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> SELECT
    ->     c.CustomerID,
    ->     c.FirstName,
    ->     c.LastName,
    ->     c.Email,
    ->     c.Phone,
    ->     c.Address
    -> 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.ProductName = 'AC';
Empty set (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> SELECT
    ->     SUM(od.Quantity * p.Price) AS TotalRevenue
    -> FROM
    ->     Orders o
    -> JOIN
    ->     OrderDetails od ON o.OrderID = od.OrderID
    -> JOIN
    ->     Products p ON od.ProductID = p.ProductID
    -> WHERE
    ->     o.OrderDate BETWEEN '2024-02-12' AND '2024-03-30';
+--------------+
| TotalRevenue |
+--------------+
|        98500 |
+--------------+
1 row in set (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
    ->     c.CustomerID,
    ->     c.FirstName,
    ->     c.LastName,
    ->     c.Email,
    ->     c.Phone,
    ->     c.Address
    -> FROM
    ->     Customers c
    -> LEFT JOIN
    ->     Orders o ON c.CustomerID = o.CustomerID
    -> WHERE
    ->     o.OrderID IS NULL;
+------------+-----------+----------+--------------+----------+---------+
| CustomerID | FirstName | LastName | Email        | Phone    | Address |
+------------+-----------+----------+--------------+----------+---------+
|        201 | Deep      | Kumar    | rd@gmail.com | 67896899 | RKL     |
|       1001 | Piku      | Pani     | pp@gmail.com | 89767845 | BBSR    |
+------------+-----------+----------+--------------+----------+---------+
2 rows in set (0.01 sec)
```

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

```
mysql> SELECT COUNT(*) AS TotalProducts
    -> FROM Products;
+---------------+
| TotalProducts |
+---------------+
|            11 |
+---------------+
1 row in set (0.04 sec)
```

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

```
mysql> SELECT
    ->     SUM(od.Quantity * p.Price) AS TotalRevenue
    -> FROM
    ->     OrderDetails od
    -> JOIN
    ->     Products p ON od.ProductID = p.ProductID
    -> JOIN
    ->     Orders o ON od.OrderID = o.OrderID;
+--------------+
| TotalRevenue |
+--------------+
|       113500 |
+--------------+
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> SELECT
    ->     AVG(od.Quantity) AS AverageQuantityOrdered
    -> FROM
    ->     OrderDetails od
    -> JOIN
    ->     Products p ON od.ProductID = p.ProductID
    -> WHERE
    ->     p.ProductName = 'Washing machine -2';
+------------------------+
| AverageQuantityOrdered |
+------------------------+
|                 1.0000 |
+------------------------+
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> SELECT
    ->     SUM(od.Quantity * p.Price) AS TotalRevenue
    -> FROM
    ->     OrderDetails od
    -> JOIN
    ->     Products p ON od.ProductID = p.ProductID
    -> JOIN
    ->     Orders o ON od.OrderID = o.OrderID
    -> WHERE
    ->     o.CustomerID = 301;
+--------------+
| TotalRevenue |
+--------------+
|         9500 |
+--------------+
1 row in set (0.01 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,
    ->     c.LastName,
    ->     COUNT(o.OrderID) AS NumberOfOrders
    -> FROM
    ->     Customers c
    -> JOIN
    ->     Orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    ->     c.CustomerID, c.FirstName, c.LastName
    -> ORDER BY
    ->     NumberOfOrders DESC
    -> LIMIT 1;
+------------+-----------+----------+----------------+
| CustomerID | FirstName | LastName | NumberOfOrders |
+------------+-----------+----------+----------------+
|        101 | Riya      | Sen      |              1 |
+------------+-----------+----------+----------------+
1 row in set (0.01 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 AS MostPopularCategory,
    ->     SUM(od.Quantity) AS TotalQuantityOrdered
    -> FROM
    ->     OrderDetails od
    -> JOIN
    ->     Products p ON od.ProductID = p.ProductID
    -> GROUP BY
    ->     p.ProductName
    -> ORDER BY
    ->     TotalQuantityOrdered DESC
    -> LIMIT 1;
+--------------------+----------------------+
| MostPopularCategory | TotalQuantityOrdered |
+--------------------+----------------------+
| Table              |                    2 |
+--------------------+----------------------+
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,
    ->     c.LastName,
    ->     SUM(od.Quantity * p.Price) 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
    -> GROUP BY
    ->     c.CustomerID, c.FirstName, c.LastName
    -> ORDER BY
    ->     TotalSpending ASC
    -> LIMIT 1;
+------------+-----------+----------+---------------+
| CustomerID | FirstName | LastName | TotalSpending |
+------------+-----------+----------+---------------+
|        501 | Prapti    | Sen      |          4000 |
+------------+-----------+----------+---------------+
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
    ->     AVG(TotalRevenue / NumOrders) AS AverageOrderValue
    -> FROM (
    ->     SELECT
    ->         c.CustomerID,
    ->         COUNT(DISTINCT o.OrderID) AS NumOrders,
    ->         SUM(od.Quantity * p.Price) AS TotalRevenue
    ->     FROM
    ->         Customers c
    ->     LEFT JOIN
    ->         Orders o ON c.CustomerID = o.CustomerID
    ->     LEFT JOIN
    ->         OrderDetails od ON o.OrderID = od.OrderID
    ->     LEFT JOIN
    ->         Products p ON od.ProductID = p.ProductID
    ->     GROUP BY
    ->         c.CustomerID
    -> ) AS CustomerOrderTotals;
+-------------------+
| AverageOrderValue |
+-------------------+
|    14187.50000000 |
+-------------------+
1 row in set (0.01 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,
    ->     c.LastName,
    ->     COUNT(o.OrderID) AS OrderCount
    -> FROM  •
    ->     Customers c
    -> LEFT JOIN
    ->     Orders o ON c.CustomerID = o.CustomerID
    -> GROUP BY
    ->     c.CustomerID, c.FirstName, c.LastName
    -> ORDER BY
    ->     OrderCount DESC;
+------------+-----------+----------+------------+
| CustomerID | FirstName | LastName | OrderCount |
+------------+-----------+----------+------------+
|        101 | Riya      | Sen      |          1 |
|        301 | Raj       | Dip      |          1 |
|        401 | Pratap    | Sen      |          1 |
|        501 | Prapti    | Sen      |          1 |
|        601 | Arti      | Das      |          1 |
|        701 | Iti       | Dash     |          1 |
|        801 | Dipa      | Das      |          1 |
|        901 | Puja      | Rai      |          1 |
|       1101 | Tara      | Dey      |          1 |
|        201 | Deep      | Kumar    |          0 |
|       1001 | Piku      | Pani     |          0 |
+------------+-----------+----------+------------+
11 rows in set (0.00 sec)
```