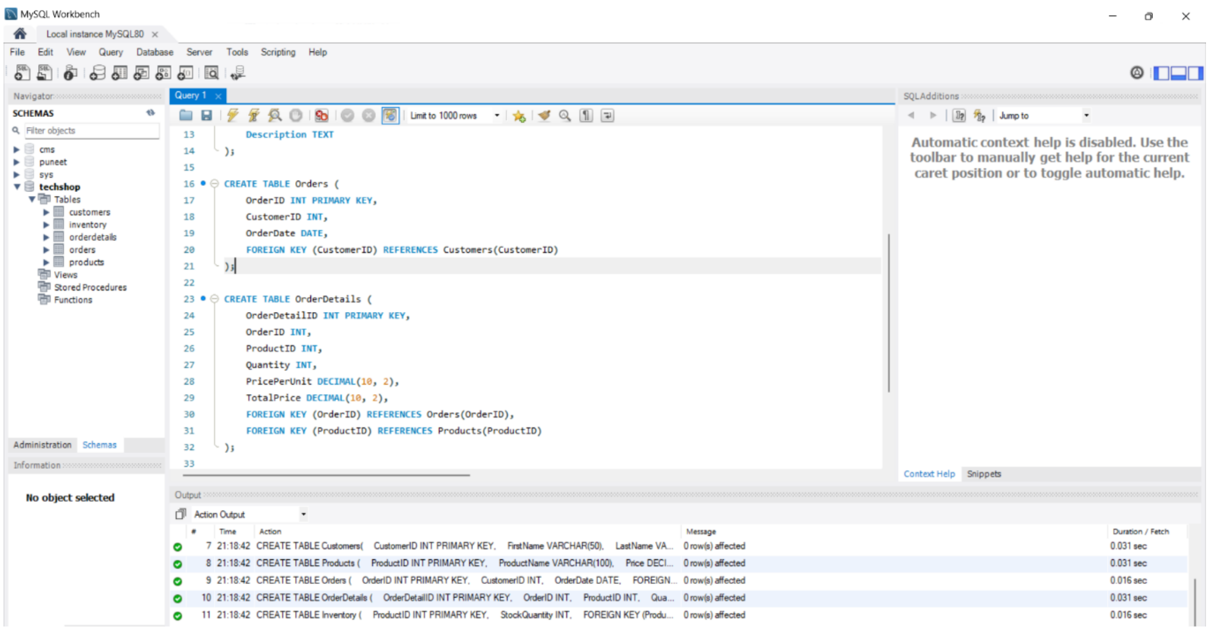
Assignment 1

Task:1. Database Design:

1. **Create the database named "TechShop"**
2. **Define the schema for the Customers, Products, Orders, OrderDetails and Inventory tables based on the provided schema.**
3. **Create appropriate Primary Key and Foreign Key constraints for referential integrity.**

```
CREATE DATABASE TechShop;


CREATE TABLE Customers (

   CustomerID INT PRIMARY KEY,

   FirstName VARCHAR(50),

   LastName VARCHAR(50),

   Email VARCHAR(100)

);


CREATE TABLE Products (

   ProductID INT PRIMARY KEY,

   ProductName VARCHAR(100),

   Price DECIMAL(10, 2),

   Description TEXT

);


CREATE TABLE Orders (

   OrderID INT PRIMARY KEY,

   CustomerID INT,

   OrderDate DATE,

   FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);
```

```
CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY,
    OrderID INT,
    ProductID INT,
    Quantity INT,
    PricePerUnit DECIMAL(10, 2),
    TotalPrice DECIMAL(10, 2),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);

CREATE TABLE Inventory (
    ProductID INT PRIMARY KEY,
    StockQuantity INT,
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
```



Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
        Description TEXT
    );

    CREATE TABLE Orders (
        OrderID INT PRIMARY KEY,
        CustomerID INT,
        OrderDate DATE,
        FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
    );

    CREATE TABLE OrderDetails (
        OrderDetailID INT PRIMARY KEY,
        OrderID INT,
        ProductID INT,
        Quantity INT,
        PricePerUnit DECIMAL(10, 2),
        TotalPrice DECIMAL(10, 2),
        FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
        FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
    );
```

```
 #   Time      Action                                                                                   Message              Duration / Fetch
 7   21:18:42  CREATE TABLE Customers(   CustomerID INT PRIMARY KEY,   FirstName VARCHAR(50),   LastName VA...   0 row(s) affected   0.031 sec
 8   21:18:42  CREATE TABLE Products (   ProductID INT PRIMARY KEY,   ProductName VARCHAR(100),   Price DECI...  0 row(s) affected   0.031 sec
 9   21:18:42  CREATE TABLE Orders (   OrderID INT PRIMARY KEY,   CustomerID INT,   OrderDate DATE,   FOREIGN...   0 row(s) affected   0.016 sec
10   21:18:42  CREATE TABLE OrderDetails (   OrderDetailID INT PRIMARY KEY,   OrderID INT,   ProductID INT,   Qua...   0 row(s) affected   0.031 sec
11   21:18:42  CREATE TABLE Inventory (   ProductID INT PRIMARY KEY,   StockQuantity INT,   FOREIGN KEY (Produ...   0 row(s) affected   0.016 sec
```

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

```
INSERT INTO Customers (CustomerID, FirstName, LastName, Email)
VALUES
   (1, 'Puneet', 'Vashistha', 'varneeet@gmail.com'),
   (2, 'Jane', 'Smith', 'jane.smith@example.com'),
   (3, 'Bob', 'Johnson', 'bob.johnson@example.com'),
   (4, 'Alice', 'Williams', 'alice.williams@example.com'),
   (5, 'Charlie', 'Brown', 'charlie.brown@example.com'),
   (6, 'Eva', 'Davis', 'eva.davis@example.com'),
   (7, 'Frank', 'Miller', 'frank.miller@example.com'),
   (8, 'Grace', 'Clark', 'grace.clark@example.com'),
   (9, 'Henry', 'Wilson', 'henry.wilson@example.com'),
   (10, 'Ivy', 'Moore', 'ivy.moore@example.com');

INSERT INTO Products (ProductID, ProductName, Price, Description)
VALUES
   (101, 'Laptop', 1200.00, 'High-performance laptop'),
   (102, 'Smartphone', 800.00, 'Latest smartphone model'),
   (103, 'Headphones', 150.00, 'Noise-canceling headphones'),
   (104, 'Tablet', 400.00, 'Portable tablet device'),
   (105, 'Camera', 700.00, 'Digital camera with advanced features'),
   (106, 'Printer', 250.00, 'Wireless color printer'),
   (107, 'External Hard Drive', 120.00, '1TB external hard drive'),
   (108, 'Monitor', 300.00, '24-inch LED monitor'),
   (109, 'Keyboard', 50.00, 'Mechanical gaming keyboard'),
   (110, 'Mouse', 30.00, 'Wireless optical mouse');
```

```sql
INSERT INTO Orders (OrderID, CustomerID, OrderDate)
VALUES
    (1001, 1, '2023-01-15'),
    (1002, 2, '2023-02-20'),
    (1003, 3, '2023-03-25'),
    (1004, 4, '2023-04-10'),
    (1005, 5, '2023-05-05'),
    (1006, 6, '2023-06-18'),
    (1007, 7, '2023-07-22'),
    (1008, 8, '2023-08-30'),
    (1009, 9, '2023-09-14'),
    (1010, 10, '2023-10-01');

INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity, PricePerUnit,
TotalPrice)
VALUES
    (5001, 1001, 101, 2, 1200.00, 2400.00),
    (5002, 1001, 102, 1, 800.00, 800.00),
    (5003, 1002, 103, 3, 150.00, 450.00),
    (5004, 1003, 104, 1, 400.00, 400.00),
    (5005, 1003, 105, 2, 700.00, 1400.00),
    (5006, 1004, 106, 1, 250.00, 250.00),
    (5007, 1005, 107, 2, 120.00, 240.00),
    (5008, 1006, 108, 1, 300.00, 300.00),
    (5009, 1007, 109, 3, 50.00, 150.00),
    (5010, 1008, 110, 1, 30.00, 30.00);

INSERT INTO Inventory (ProductID, StockQuantity)
VALUES
    (101, 50),
```

(102, 30),

(103, 100),

(104, 20),

(105, 40),

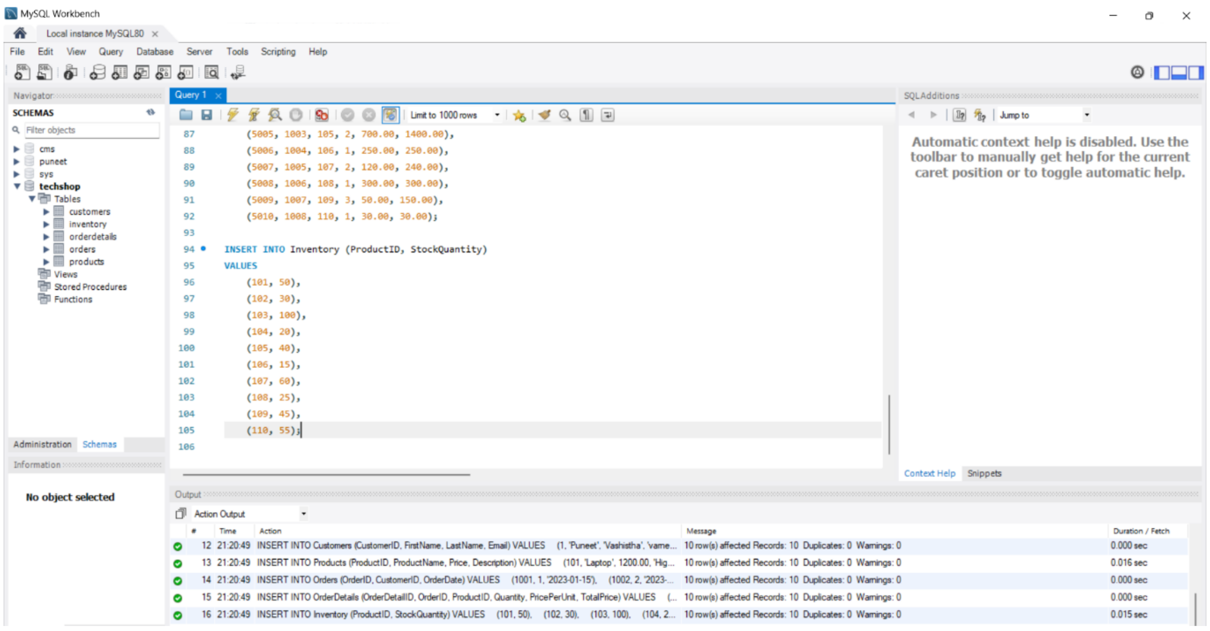
(106, 15),

(107, 60),

(108, 25),

(109, 45),

(110, 55);



Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

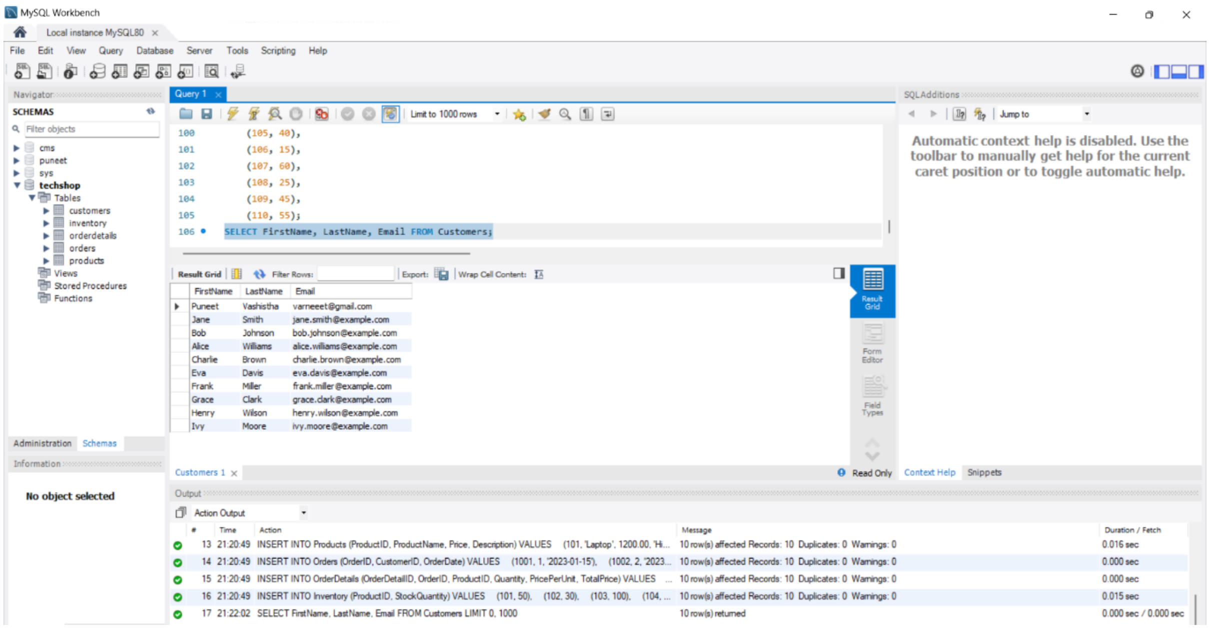
```
    (5005, 1003, 105, 2, 700.00, 1400.00),
    (5006, 1004, 106, 1, 250.00, 250.00),
    (5007, 1005, 107, 2, 120.00, 240.00),
    (5008, 1006, 108, 1, 300.00, 300.00),
    (5009, 1007, 109, 3, 50.00, 150.00),
    (5010, 1008, 110, 1, 30.00, 30.00);

INSERT INTO Inventory (ProductID, StockQuantity)
VALUES
    (101, 50),
    (102, 30),
    (103, 100),
    (104, 20),
    (105, 40),
    (106, 15),
    (107, 60),
    (108, 25),
    (109, 45),
    (110, 55);
```

```
#   Time      Action                                                                                              Message                                                        Duration / Fetch
12  21:20:49  INSERT INTO Customers (CustomerID, FirstName, LastName, Email) VALUES    (1, 'Puneet', 'Vashistha', 'vame...   10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0   0.000 sec
13  21:20:49  INSERT INTO Products (ProductID, ProductName, Price, Description) VALUES    (101, 'Laptop', 1200.00, 'Hig...  10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0   0.016 sec
14  21:20:49  INSERT INTO Orders (OrderID, CustomerID, OrderDate) VALUES    (1001, 1, '2023-01-15'),    (1002, 2, '2023-...  10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0   0.000 sec
15  21:20:49  INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity, PricePerUnit, TotalPrice) VALUES    (...  10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0   0.000 sec
16  21:20:49  INSERT INTO Inventory (ProductID, StockQuantity) VALUES    (101, 50),    (102, 30),    (103, 100),    (104, 2...  10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0   0.015 sec
```

Task 2:

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

SELECT FirstName, LastName, Email FROM Customers;



```
       (105, 40),
       (106, 15),
       (107, 60),
       (108, 25),
       (109, 45),
       (110, 55);
 SELECT FirstName, LastName, Email FROM Customers;
```

	FirstName	LastName	Email
▶	Puneet	Vashistha	varneeet@gmail.com
	Jane	Smith	jane.smith@example.com
	Bob	Johnson	bob.johnson@example.com
	Alice	Williams	alice.williams@example.com
	Charlie	Brown	charlie.brown@example.com
	Eva	Davis	eva.davis@example.com
	Frank	Miller	frank.miller@example.com
	Grace	Clark	grace.clark@example.com
	Henry	Wilson	henry.wilson@example.com
	Ivy	Moore	ivy.moore@example.com

No object selected

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

	#	Time	Action	Message	Duration / Fetch
✓	13	21:20:49	INSERT INTO Products (ProductID, ProductName, Price, Description) VALUES (101, 'Laptop', 1200.00, 'Hi...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
✓	14	21:20:49	INSERT INTO Orders (OrderID, CustomerID, OrderDate) VALUES (1001, 1, '2023-01-15'), (1002, 2, '2023...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec
✓	15	21:20:49	INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity, PricePerUnit, TotalPrice) VALUES ...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec
✓	16	21:20:49	INSERT INTO Inventory (ProductID, StockQuantity) VALUES (101, 50), (102, 30), (103, 100), (104, ...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.015 sec
✓	17	21:22:02	SELECT FirstName, LastName, Email FROM Customers LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

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

SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName, Customers.LastName

FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

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

INSERT INTO Customers (FirstName, LastName, Email, Address)

VALUES ('Henry', 'Patel', 'new.customer@example.com', '123 Main Street, Cityville');

4. **Write an SQL query to update the prices of all electronics gadgets in the “Products” table by increasing them by 10%.**

```
UPDATE Products
SET Price = Price * 1.10
WHERE Category = 'Electronics';
```

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

```
DECLARE @OrderIDToDelete INT;

SET @OrderIDToDelete = 1001;
DELETE FROM OrderDetails
WHERE OrderID = @OrderIDToDelete;

DELETE FROM Orders
WHERE OrderID = @OrderIDToDelete;
```

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

```
INSERT INTO Orders (OrderID, CustomerID, OrderDate)
VALUES (1013, 23, '2023-09-15'),
(1014, 24, '2023-12-20');
```

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

```
SET @customerid=4;

update customers
set email='varneeet@gmail.com' , address='123 baraut'
where customerid=@customerid;
```

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.**

```
UPDATE Orders
SET TotalCost = (
   SELECT SUM(Quantity * PricePerUnit)
   FROM OrderDetails
   WHERE OrderDetails.OrderID = Orders.OrderID
)
WHERE EXISTS (
   SELECT 1
   FROM OrderDetails
   WHERE OrderDetails.OrderID = Orders.OrderID
);
```

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.**

```
DECLARE @CustomerIDToDelete INT;

SET @CustomerIDToDelete = 1001;

DELETE FROM OrderDetails
WHERE OrderID IN (SELECT OrderID FROM Orders WHERE CustomerID =
@CustomerIDToDelete);

DELETE FROM Orders
WHERE CustomerID = @CustomerIDToDelete;
```

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.

```
INSERT INTO Products (ProductName, Category, Price, Description)

VALUES ('New Electronic Gadget', 'Electronics', 499.99, ‘new gadget.');
```

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.

```
DECLARE @OrderIDToUpdate INT;

DECLARE @NewStatus VARCHAR(50);


SET @OrderIDToUpdate = 1008;

SET @NewStatus = 'Shipped';


UPDATE Orders

SET Status = @NewStatus

WHERE OrderID = @OrderIDToUpdate;
```

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.

```
UPDATE Customers

SET NumberOfOrders = (

   SELECT COUNT(OrderID)

   FROM Orders

   WHERE Orders.CustomerID = Customers.CustomerID

);
```

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.**

```
SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName, Customers.LastName,
Customers.Email

FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

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

```
SELECT

  Products.ProductName,

  SUM(OrderDetails.Quantity * OrderDetails.PricePerUnit) AS TotalRevenue

FROM

  OrderDetails

JOIN

  Products ON OrderDetails.ProductID = Products.ProductID
```

```
WHERE
   Products.Category = 'Electronics'
GROUP BY
   Products.ProductName;
```

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

```
SELECT DISTINCT
   Customers.CustomerID,
   Customers.FirstName,
   Customers.LastName,
   Customers.Email
FROM
   Customers
JOIN
   Orders ON Customers.CustomerID = Orders.CustomerID;
```

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.**

```
SELECT
   Products.ProductName,
   SUM(OrderDetails.Quantity) AS TotalQuantityOrdered
FROM
   OrderDetails
JOIN
   Products ON OrderDetails.ProductID = Products.ProductID
WHERE
   Products.Category = 'Electronics'
GROUP BY
   Products.ProductName
```

```
ORDER BY
    TotalQuantityOrdered DESC
LIMIT 1;
```

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

```
SELECT
    ProductID,
    ProductName,
    Category
FROM
    Products
WHERE
    Category = 'Electronics';
```

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

```
SELECT
    Customers.CustomerID,
    Customers.FirstName,
    Customers.LastName,
    AVG(OrderDetails.TotalPrice) AS AverageOrderValue
FROM
    Customers
JOIN
    Orders ON Customers.CustomerID = Orders.CustomerID
JOIN
    OrderDetails ON Orders.OrderID = OrderDetails.OrderID
GROUP BY
    Customers.CustomerID, Customers.FirstName, Customers.LastName;
```

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

```
SELECT
   Orders.OrderID,
   Customers.CustomerID,
   Customers.FirstName,
   Customers.LastName,
   SUM(OrderDetails.TotalPrice) AS TotalRevenue
FROM
   Orders
JOIN
   Customers ON Orders.CustomerID = Customers.CustomerID
JOIN
   OrderDetails ON Orders.OrderID = OrderDetails.OrderID
GROUP BY
   Orders.OrderID, Customers.CustomerID, Customers.FirstName, Customers.LastName
ORDER BY
   TotalRevenue DESC
LIMIT 1;
```

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

```
SELECT
   Products.ProductID,
   Products.ProductName,
   COUNT(OrderDetails.OrderID) AS NumberOfOrders
FROM
   Products
JOIN
   OrderDetails ON Products.ProductID = OrderDetails.ProductID
WHERE
```

```
    Products.Category = 'Electronics'
GROUP BY
    Products.ProductID, Products.ProductName;
```

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.**

```
DECLARE @ProductNameParam VARCHAR(100);

SET @ProductNameParam = 'Laptop';

SELECT
    Customers.CustomerID,
    Customers.FirstName,
    Customers.LastName,
    Customers.Email
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 = @ProductNameParam;
```

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.**

```
Select sum(totalamount) as totalrevenue
```

```
From orders
Where orderdate between ‘2023-01-13’ AND ‘2023-12-31’;
```