

TASK 1

1. Create the database named "TechShop"

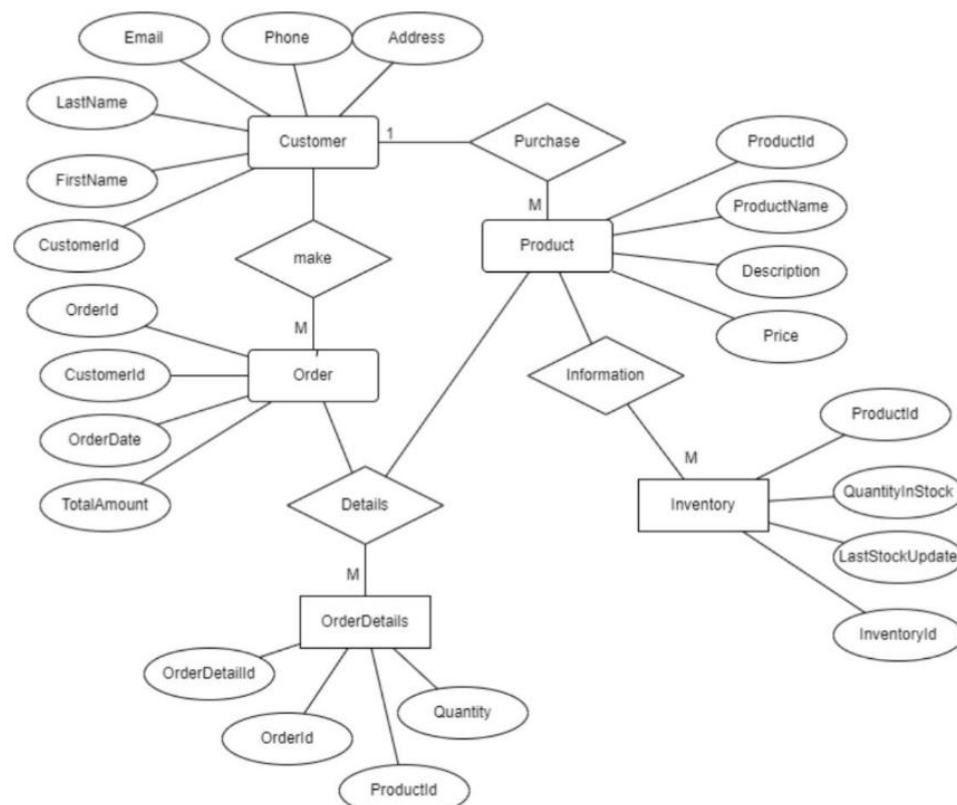
```
1 • CREATE DATABASE TechShop;
2
3 • USE TechShop;
4
5 • CREATE TABLE Customers (
6     CustomerID INT PRIMARY KEY,
7     FirstName VARCHAR(50),
8     LastName VARCHAR(50),
9     Email VARCHAR(100),
10    Phone VARCHAR(20),
11    Address VARCHAR(255)
12 );
13
14 • CREATE TABLE Products (
15     ProductID INT PRIMARY KEY,
16     ProductName VARCHAR(100),
17     Description TEXT,
18     Price DECIMAL(10, 2)
```

Output

Action Output

#	Time	Action	Message
✓ 48	10:57:00	Select * from payments LIMIT 0, 1000	10 row(s) returned
✓ 49	19:48:22	CREATE DATABASE TechShop	1 row(s) affected
✓ 50	19:48:22	USE TechShop	0 row(s) affected
✓ 51	19:48:22	CREATE TABLE Customers (CustomerID INT PRIMARY KEY, FirstName VARCHAR(50), LastName V...	0 row(s) affected
✓ 52	19:48:22	CREATE TABLE Products (ProductID INT PRIMARY KEY, ProductName VARCHAR(100), Descriptio...	0 row(s) affected
✓ 53	19:48:22	CREATE TABLE Orders (OrderID INT PRIMARY KEY, CustomerID INT, OrderDate DATE, TotalAm...	0 row(s) affected
✓ 54	19:48:22	CREATE TABLE OrderDetails (OrderDetailID INT PRIMARY KEY, OrderID INT, ProductID INT, Qu...	0 row(s) affected
✓ 55	19:48:22	CREATE TABLE Inventory (InventoryID INT PRIMARY KEY, ProductID INT, QuantityInStock INT, ...	0 row(s) affected

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



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

- a. Customers
- b. Products
- c. Orders
- d. OrderDetails
- e. Inventory

Limit to 1000 rows

```
1 • INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address)
2 VALUES
3 (1, 'John', 'Doe', 'john.doe@email.com', '123-456-7890', '123 Main St'),
4 (2, 'Jane', 'Smith', 'jane.smith@email.com', '987-654-3210', '456 Oak St'),
5 (3, 'Bob', 'Johnson', 'bob.johnson@email.com', '555-123-4567', '789 Elm St'),
6 (4, 'Alice', 'Williams', 'alice.williams@email.com', '333-555-7890', '987 Pine St'),
7 (5, 'Charlie', 'Brown', 'charlie.brown@email.com', '222-444-6666', '345 Cedar St'),
8 (6, 'Eva', 'Davis', 'eva.davis@email.com', '111-999-8888', '654 Birch St'),
9 (7, 'David', 'Miller', 'david.miller@email.com', '777-888-9999', '234 Maple St'),
10 (8, 'Sophie', 'Anderson', 'sophie.anderson@email.com', '555-666-7777', '876 Oak St'),
11 (9, 'Tom', 'Wilson', 'tom.wilson@email.com', '111-222-3333', '543 Pine St'),
12 (10, 'Mia', 'Jones', 'mia.jones@email.com', '999-777-5555', '432 Elm St');
13
14 • Select * from Customers
```

Result Grid

CustomerID	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St
2	Jane	Smith	jane.smith@email.com	987-654-3210	456 Oak St
3	Bob	Johnson	bob.johnson@email.com	555-123-4567	789 Elm St
4	Alice	Williams	alice.williams@email.com	333-555-7890	987 Pine St
5	Charlie	Brown	charlie.brown@email.com	222-444-6666	345 Cedar St
6	Eva	Davis	eva.davis@email.com	111-999-8888	654 Birch St
7	David	Miller	david.miller@email.com	777-888-9999	234 Maple St
8	Sophie	Anderson	sophie.anderson@email.com	555-666-7777	876 Oak St
9	Tom	Wilson	tom.wilson@email.com	111-222-3333	543 Pine St
10	Mia	Jones	mia.jones@email.com	999-777-5555	432 Elm St

Limit to 1000 rows

```
1 • INSERT INTO Products (ProductID, ProductName, Description, Price)
2 VALUES
3 (1, 'Smartphone X', 'High-end smartphone', 799.99),
4 (2, 'Laptop Pro', 'Powerful laptop with advanced features', 1299.99),
5 (3, 'Wireless Earbuds', 'Noise-canceling wireless earbuds', 149.99),
6 (4, 'Smartwatch Plus', 'Fitness tracking smartwatch', 199.99),
7 (5, 'Gaming Console', 'Latest gaming console with 4K support', 499.99),
8 (6, 'Ultra HD TV', '65-inch Ultra HD smart TV', 899.99),
9 (7, 'Digital Camera', 'Professional-grade digital camera', 699.99),
10 (8, 'Portable Speaker', 'Bluetooth portable speaker', 79.99),
11 (9, 'Tablet Pro', 'High-performance tablet with stylus', 499.99),
12 (10, 'Fitness Tracker', 'Water-resistant fitness tracker', 89.99);
13
14
15 • Select * from Products
```

Result Grid

ProductID	ProductName	Description	Price
1	Smartphone X	High-end smartphone	799.99
2	Laptop Pro	Powerful laptop with advanced features	1299.99
3	Wireless Earbuds	Noise-canceling wireless earbuds	149.99
4	Smartwatch Plus	Fitness tracking smartwatch	199.99
5	Gaming Console	Latest gaming console with 4K support	499.99
6	Ultra HD TV	65-inch Ultra HD smart TV	899.99
7	Digital Camera	Professional-grade digital camera	699.99
8	Portable Speaker	Bluetooth portable speaker	79.99
9	Tablet Pro	High-performance tablet with stylus	499.99
10	Fitness Tracker	Water-resistant fitness tracker	89.99

Limit to 1000 rows

```
1 • INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
2 VALUES
3 (1, 1, '2024-01-15', 1599.98),
4 (2, 2, '2024-01-16', 899.99),
5 (3, 3, '2024-01-17', 749.97),
6 (4, 4, '2024-01-18', 299.99),
7 (5, 5, '2024-01-19', 599.98),
8 (6, 6, '2024-01-20', 1299.99),
9 (7, 7, '2024-01-21', 469.98),
10 (8, 8, '2024-01-22', 159.98),
11 (9, 9, '2024-01-23', 149.99),
12 (10, 10, '2024-01-24', 179.98);
13
14 • Select * from Orders
15
```

Result Grid

	OrderID	CustomerID	OrderDate	TotalAmount
▶	1	1	2024-01-15	1599.98
	2	2	2024-01-16	899.99
	3	3	2024-01-17	749.97
	4	4	2024-01-18	299.99
	5	5	2024-01-19	599.98
	6	6	2024-01-20	1299.99
	7	7	2024-01-21	469.98
	8	8	2024-01-22	159.98
	9	9	2024-01-23	149.99
	10	10	2024-01-24	179.98
*	NULL	NULL	NULL	NULL

Limit to 1000 rows

```
1 • INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity)
2 VALUES
3 (1, 1, 1, 2),
4 (2, 1, 2, 1),
5 (3, 2, 3, 1),
6 (4, 3, 4, 1),
7 (5, 3, 5, 1),
8 (6, 4, 6, 1),
9 (7, 5, 7, 1),
10 (8, 6, 8, 2),
11 (9, 7, 9, 1),
12 (10, 8, 10, 3);
13
14
15 • Select * from OrderDetails
16
```

Result Grid

	OrderDetailID	OrderID	ProductID	Quantity
▶	1	1	1	2
	2	1	2	1
	3	2	3	1
	4	3	4	1
	5	3	5	1
	6	4	6	1
	7	5	7	1
	8	6	8	2
	9	7	9	1
	10	8	10	3
*	NULL	NULL	NULL	NULL

Limit to 1000 rows

```

1 • INSERT INTO Inventory (InventoryID, ProductID, QuantityInStock, LastStockUpdate)
2   VALUES
3   (1, 1, 50, '2024-01-15'),
4   (2, 2, 25, '2024-01-15'),
5   (3, 3, 100, '2024-01-15'),
6   (4, 4, 30, '2024-01-15'),
7   (5, 5, 20, '2024-01-15'),
8   (6, 6, 15, '2024-01-15'),
9   (7, 7, 40, '2024-01-15'),
10  (8, 8, 75, '2024-01-15'),
11  (9, 9, 50, '2024-01-15'),
12  (10, 10, 60, '2024-01-15');
13
14
15 • Select * from Inventory;

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

	InventoryID	ProductID	QuantityInStock	LastStockUpdate
▶	1	1	50	2024-01-15
	2	2	25	2024-01-15
	3	3	100	2024-01-15
	4	4	30	2024-01-15
	5	5	20	2024-01-15
	6	6	15	2024-01-15
	7	7	40	2024-01-15
	8	8	75	2024-01-15
	9	9	50	2024-01-15
	10	10	60	2024-01-15
•	NULL	NULL	NULL	NULL

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

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

Limit to 1000 rows

```

1 • SELECT FirstName, LastName, Email
2   FROM Customers;
3
4
5

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |










	FirstName	LastName	Email
▶	John	Doe	john.doe@email.com
	Jane	Smith	jane.smith@email.com
	Bob	Johnson	bob.johnson@email.com
	Alice	Williams	alice.williams@email.com
	Charlie	Brown	charlie.brown@email.com
	Eva	Davis	eva.davis@email.com
	David	Miller	david.miller@email.com
	Sophie	Anderson	sophie.anderson@email.com
	Tom	Wilson	tom.wilson@email.com
	Mia	Jones	mia.jones@email.com

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






1	•	SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName, Customers.LastName
2		FROM Orders
3		JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
4		
5		

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
OrderID	OrderDate	FirstName	LastName
1	2024-01-15	John	Doe
2	2024-01-16	Jane	Smith
3	2024-01-17	Bob	Johnson
4	2024-01-18	Alice	Williams
5	2024-01-19	Charlie	Brown
6	2024-01-20	Eva	Davis
7	2024-01-21	David	Miller
8	2024-01-22	Sophie	Anderson
9	2024-01-23	Tom	Wilson
10	2024-01-24	Mia	Jones

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






Limit to 1000 rows





```
1 • INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address)
2   VALUES ('11', 'New', 'Customer', 'new.customer@email.com', '555-123-4567', '789 Broadway St');
3
4 • select * from customers
5
6
7
-
```

Result Grid

Filter Rows:

Edit:   

Export/Import:  

Wrap Cell Content: ☐

	CustomerID	FirstName	LastName	Email	Phone	Address
▶	1	John	Doe	john.doe@email.com	123-456-7890	123 Main St
	2	Jane	Smith	jane.smith@email.com	987-654-3210	456 Oak St
	3	Bob	Johnson	bob.johnson@email.com	555-123-4567	789 Elm St
	4	Alice	Williams	alice.williams@email.com	333-555-7890	987 Pine St
	5	Charlie	Brown	charlie.brown@email.com	222-444-6666	345 Cedar St
	6	Eva	Davis	eva.davis@email.com	111-999-8888	654 Birch St
	7	David	Miller	david.miller@email.com	777-888-9999	234 Maple St
	8	Sophie	Anderson	sophie.anderson@email.com	555-666-7777	876 Oak St
	9	Tom	Wilson	tom.wilson@email.com	111-222-3333	543 Pine St
	10	Mia	Jones	mia.jones@email.com	999-777-5555	432 Elm St
	11	New	Customer	new.customer@email.com	555-123-4567	789 Broadw...
•	NULL	NULL	NULL	NULL	NULL	NULL

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

Limit to 1000 rows

```

1 • UPDATE Products
2   SET Price = Price * 1.1;
3
4
5 • select * from Products
6

```

ProductID	ProductName	Description	Price
1	Smartphone X	High-end smartphone	879.99
2	Laptop Pro	Powerful laptop with advanced features	1429.99
3	Wireless Earbuds	Noise-canceling wireless earbuds	164.99
4	Smartwatch Plus	Fitness tracking smartwatch	219.99
5	Gaming Console	Latest gaming console with 4K support	549.99
6	Ultra HD TV	65-inch Ultra HD smart TV	989.99
7	Digital Camera	Professional-grade digital camera	769.99
8	Portable Speaker	Bluetooth portable speaker	87.99
9	Tablet Pro	High-performance tablet with stylus	549.99
10	Fitness Tracker	Water-resistant fitness tracker	98.99
NULL	NULL	NULL	NULL

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.
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.

Limit to 1000 rows

```

1 • INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
2   VALUES (11 , 1, '2024-01-25', 129.99);
3
4 • select * from orders
5
6

```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2024-01-15	1599.98
2	2	2024-01-16	899.99
3	3	2024-01-17	749.97
4	4	2024-01-18	299.99
5	5	2024-01-19	599.98
6	6	2024-01-20	1299.99
7	7	2024-01-21	469.98
8	8	2024-01-22	159.98
9	9	2024-01-23	149.99
10	10	2024-01-24	179.98
11	1	2024-01-25	129.99
NULL	NULL	NULL	NULL

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.
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.

SQL Query Editor showing an UPDATE statement:

```

1 • UPDATE Orders
2   SET TotalAmount = (
3     SELECT SUM(OrderDetails.Quantity * Products.Price)
4     FROM OrderDetails
5     JOIN Products ON OrderDetails.ProductID = Products.ProductID
6     WHERE OrderDetails.OrderID = Orders.OrderID
7   )
8   WHERE EXISTS (
9     SELECT 1
10    FROM OrderDetails
11    WHERE OrderDetails.OrderID = Orders.OrderID
12  );

```

Result Grid:

	OrderID	CustomerID	OrderDate	TotalAmount
▶	1	1	2024-01-15	3189.97
	2	2	2024-01-16	164.99
	3	3	2024-01-17	769.98
	4	4	2024-01-18	989.99
	5	5	2024-01-19	769.99
	6	6	2024-01-20	175.98
	7	7	2024-01-21	549.99
	8	8	2024-01-22	296.97
	9	9	2024-01-23	149.99
	10	10	2024-01-24	179.98
	11	1	2024-01-25	129.99
•	NULL	NULL	NULL	NULL

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.

SQL Query Editor showing a DELETE statement:

```

1 • DELETE FROM OrderDetails
2   WHERE OrderID IN (
3     SELECT OrderID
4     FROM Orders
5     WHERE CustomerID = 11);
6
7 • DELETE FROM Orders
8   WHERE CustomerID = 11;
9

```

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.

SQL Query Editor showing an INSERT and SELECT statement:

```

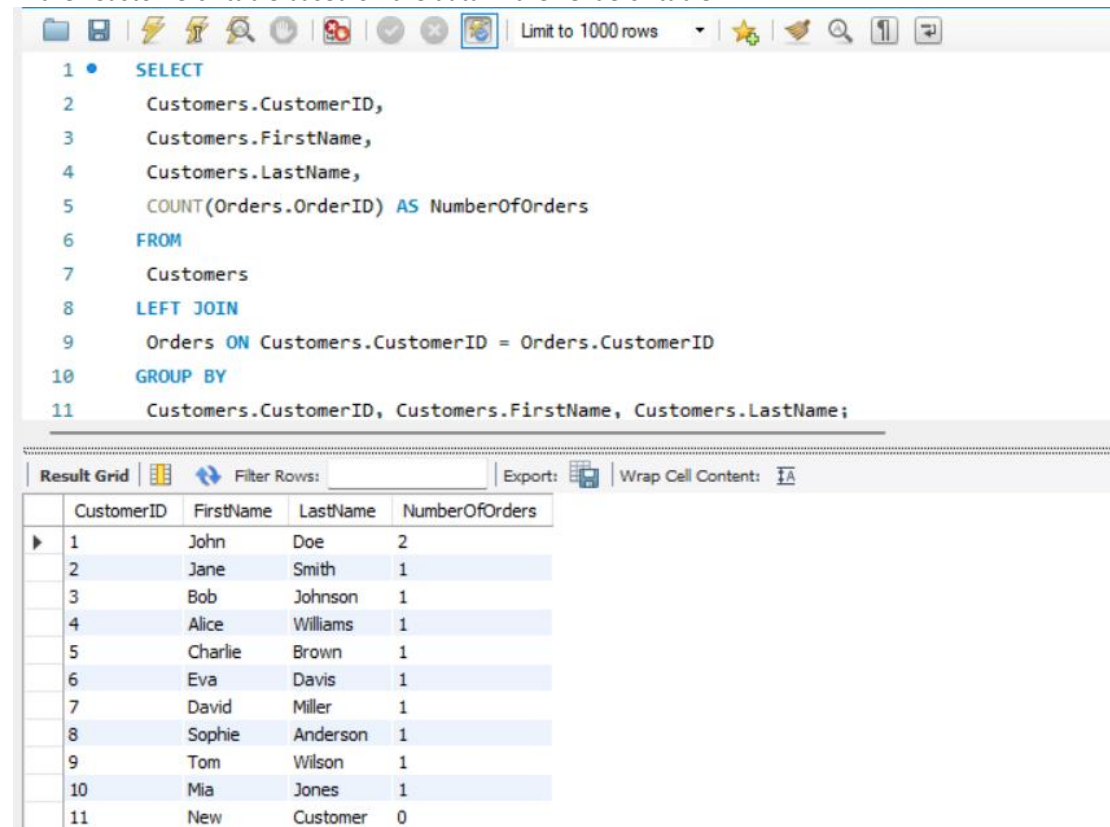
1 • INSERT INTO Products (ProductID, ProductName, Description, Price)
2   VALUES (11, 'New Gadget', 'High-tech electronic gadget', 499.99);
3
4 • select * from products
5

```

Result Grid:

	ProductID	ProductName	Description	Price
▶	1	Smartphone X	High-end smartphone	879.99
	2	Laptop Pro	Powerful laptop with advanced features	1429.99
	3	Wireless Earbuds	Noise-canceling wireless earbuds	164.99
	4	Smartwatch Plus	Fitness tracking smartwatch	219.99
	5	Gaming Console	Latest gaming console with 4K support	549.99
	6	Ultra HD TV	65-inch Ultra HD smart TV	989.99
	7	Digital Camera	Professional-grade digital camera	769.99
	8	Portable Speaker	Bluetooth portable speaker	87.99
	9	Tablet Pro	High-performance tablet with stylus	549.99
	10	Fitness Tracker	Water-resistant fitness tracker	98.99
	11	New Gadget	High-tech electronic gadget	499.99
•	NULL	NULL	NULL	NULL

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



The screenshot shows a SQL query editor with a query to calculate the number of orders for each customer. The query is as follows:

```

1  SELECT
2      Customers.CustomerID,
3      Customers.FirstName,
4      Customers.LastName,
5      COUNT(Orders.OrderID) AS NumberOfOrders
6  FROM
7      Customers
8  LEFT JOIN
9      Orders ON Customers.CustomerID = Orders.CustomerID
10 GROUP BY
11     Customers.CustomerID, Customers.FirstName, Customers.LastName;

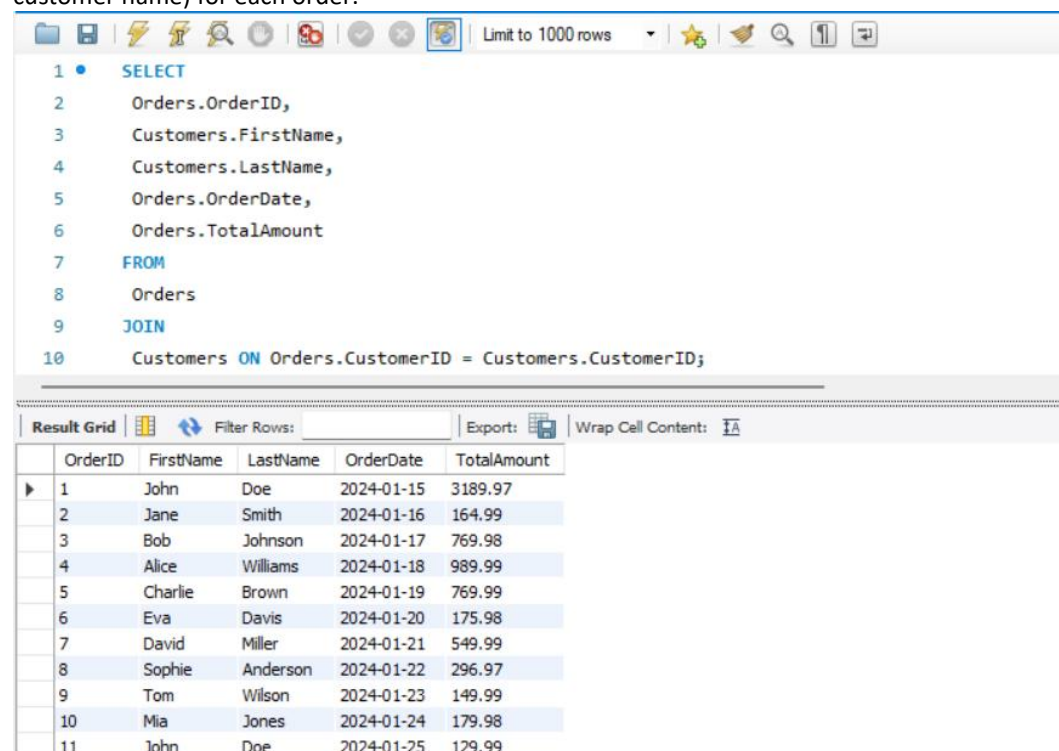
```

Below the query editor, the "Result Grid" is displayed, showing the results of the query. The grid has 5 columns: CustomerID, FirstName, LastName, and NumberOfOrders. The results are as follows:

	CustomerID	FirstName	LastName	NumberOfOrders
1	1	John	Doe	2
2	2	Jane	Smith	1
3	3	Bob	Johnson	1
4	4	Alice	Williams	1
5	5	Charlie	Brown	1
6	6	Eva	Davis	1
7	7	David	Miller	1
8	8	Sophie	Anderson	1
9	9	Tom	Wilson	1
10	10	Mia	Jones	1
11	11	New	Customer	0

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.



The screenshot shows a SQL query editor with a query to retrieve all orders along with customer information. The query is as follows:

```

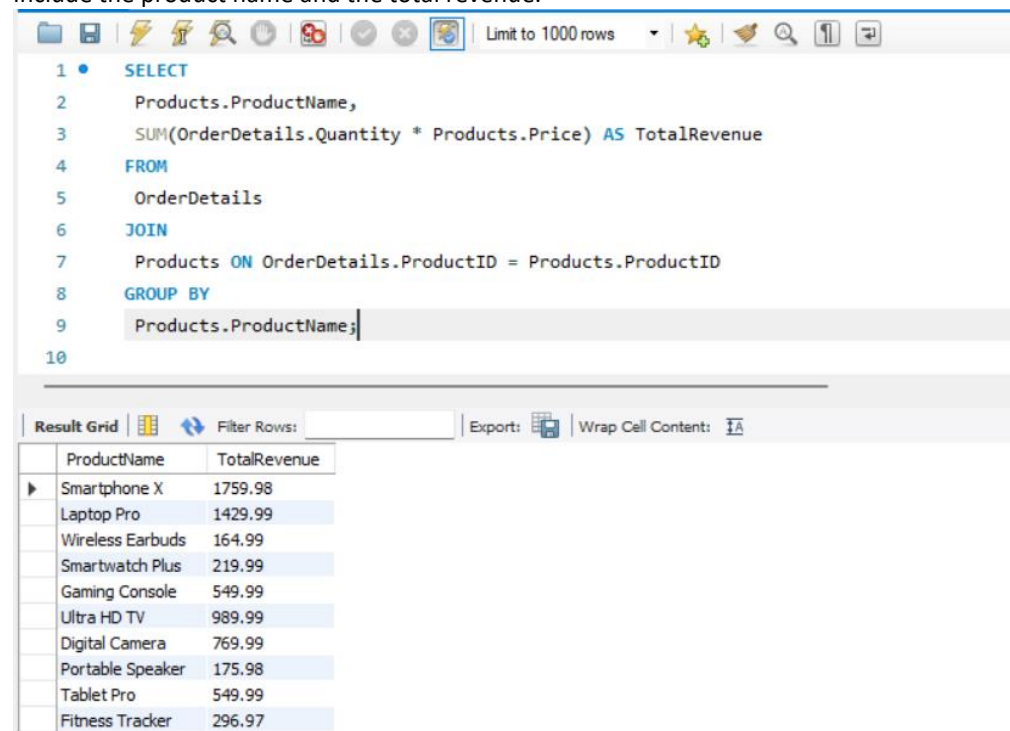
1  SELECT
2      Orders.OrderID,
3      Customers.FirstName,
4      Customers.LastName,
5      Orders.OrderDate,
6      Orders.TotalAmount
7  FROM
8      Orders
9  JOIN
10     Customers ON Orders.CustomerID = Customers.CustomerID;

```

Below the query editor, the "Result Grid" is displayed, showing the results of the query. The grid has 6 columns: OrderID, FirstName, LastName, OrderDate, and TotalAmount. The results are as follows:

	OrderID	FirstName	LastName	OrderDate	TotalAmount
1	1	John	Doe	2024-01-15	3189.97
2	2	Jane	Smith	2024-01-16	164.99
3	3	Bob	Johnson	2024-01-17	769.98
4	4	Alice	Williams	2024-01-18	989.99
5	5	Charlie	Brown	2024-01-19	769.99
6	6	Eva	Davis	2024-01-20	175.98
7	7	David	Miller	2024-01-21	549.99
8	8	Sophie	Anderson	2024-01-22	296.97
9	9	Tom	Wilson	2024-01-23	149.99
10	10	Mia	Jones	2024-01-24	179.98
11	11	John	Doe	2024-01-25	129.99

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



The screenshot shows a SQL query editor with the following query:

```

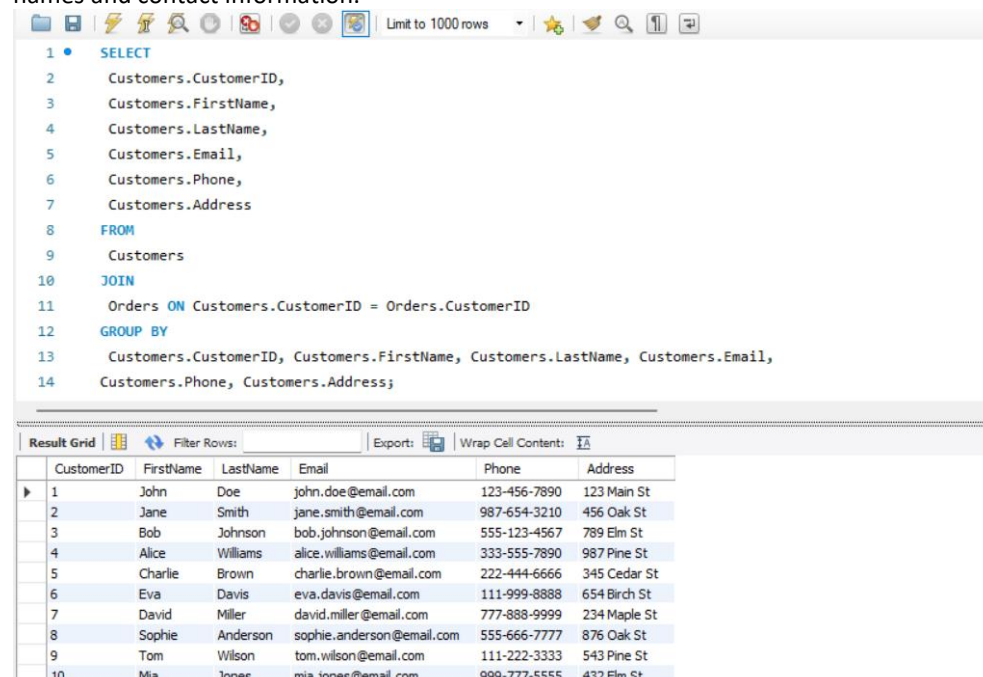
1  SELECT
2  Products.ProductName,
3  SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue
4  FROM
5  OrderDetails
6  JOIN
7  Products ON OrderDetails.ProductID = Products.ProductID
8  GROUP BY
9  Products.ProductName;
10

```

Below the query editor is the Result Grid, which displays the following data:

ProductName	TotalRevenue
Smartphone X	1759.98
Laptop Pro	1429.99
Wireless Earbuds	164.99
Smartwatch Plus	219.99
Gaming Console	549.99
Ultra HD TV	989.99
Digital Camera	769.99
Portable Speaker	175.98
Tablet Pro	549.99
Fitness Tracker	296.97

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



The screenshot shows a SQL query editor with the following query:

```

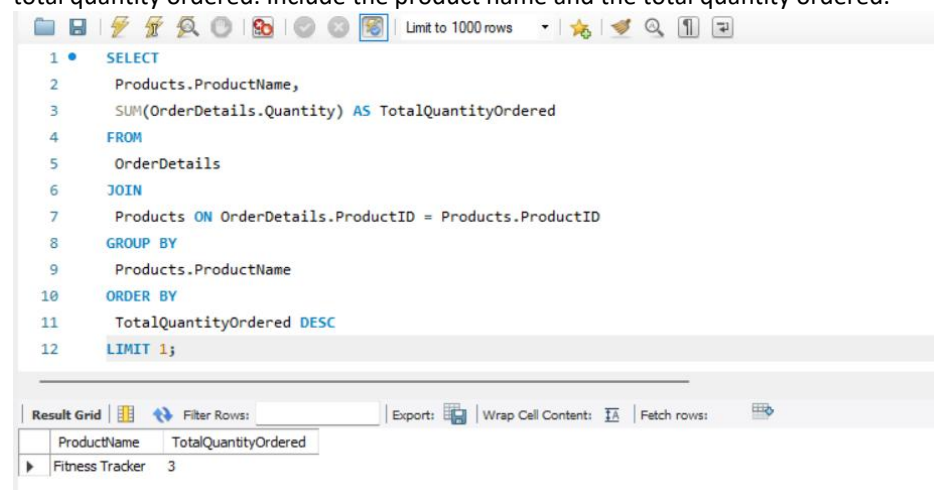
1  SELECT
2  Customers.CustomerID,
3  Customers.FirstName,
4  Customers.LastName,
5  Customers.Email,
6  Customers.Phone,
7  Customers.Address
8  FROM
9  Customers
10 JOIN
11 Orders ON Customers.CustomerID = Orders.CustomerID
12 GROUP BY
13 Customers.CustomerID, Customers.FirstName, Customers.LastName, Customers.Email,
14 Customers.Phone, Customers.Address;

```

Below the query editor is the Result Grid, which displays the following data:

CustomerID	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St
2	Jane	Smith	jane.smith@email.com	987-654-3210	456 Oak St
3	Bob	Johnson	bob.johnson@email.com	555-123-4567	789 Elm St
4	Alice	Williams	alice.williams@email.com	333-555-7890	987 Pine St
5	Charlie	Brown	charlie.brown@email.com	222-444-6666	345 Cedar St
6	Eva	Davis	eva.davis@email.com	111-999-8888	654 Birch St
7	David	Miller	david.miller@email.com	777-888-9999	234 Maple St
8	Sophie	Anderson	sophie.anderson@email.com	555-666-7777	876 Oak St
9	Tom	Wilson	tom.wilson@email.com	111-222-3333	543 Pine St
10	Mia	Jones	mia.jones@email.com	999-777-5555	432 Elm St

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.



```

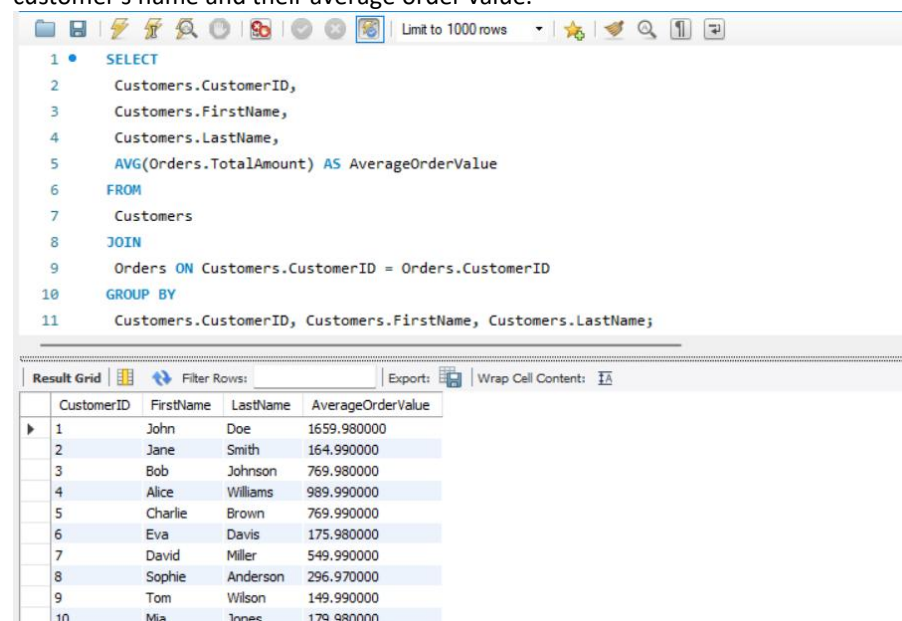
1 • SELECT
2   Products.ProductName,
3   SUM(OrderDetails.Quantity) AS TotalQuantityOrdered
4 FROM
5   OrderDetails
6 JOIN
7   Products ON OrderDetails.ProductID = Products.ProductID
8 GROUP BY
9   Products.ProductName
10 ORDER BY
11   TotalQuantityOrdered DESC
12 LIMIT 1;

```

Result Grid

ProductName	TotalQuantityOrdered
Fitness Tracker	3

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



```

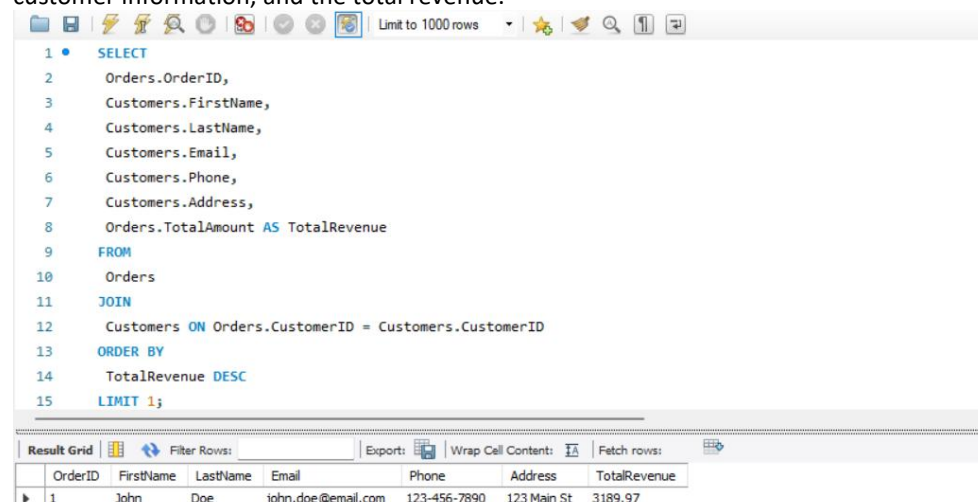
1 • SELECT
2   Customers.CustomerID,
3   Customers.FirstName,
4   Customers.LastName,
5   AVG(Orders.TotalAmount) AS AverageOrderValue
6 FROM
7   Customers
8 JOIN
9   Orders ON Customers.CustomerID = Orders.CustomerID
10 GROUP BY
11   Customers.CustomerID, Customers.FirstName, Customers.LastName;

```

Result Grid

CustomerID	FirstName	LastName	AverageOrderValue
1	John	Doe	1659.980000
2	Jane	Smith	164.990000
3	Bob	Johnson	769.980000
4	Alice	Williams	989.990000
5	Charlie	Brown	769.990000
6	Eva	Davis	175.980000
7	David	Miller	549.990000
8	Sophie	Anderson	296.970000
9	Tom	Wilson	149.990000
10	Mia	Jones	179.980000

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



```

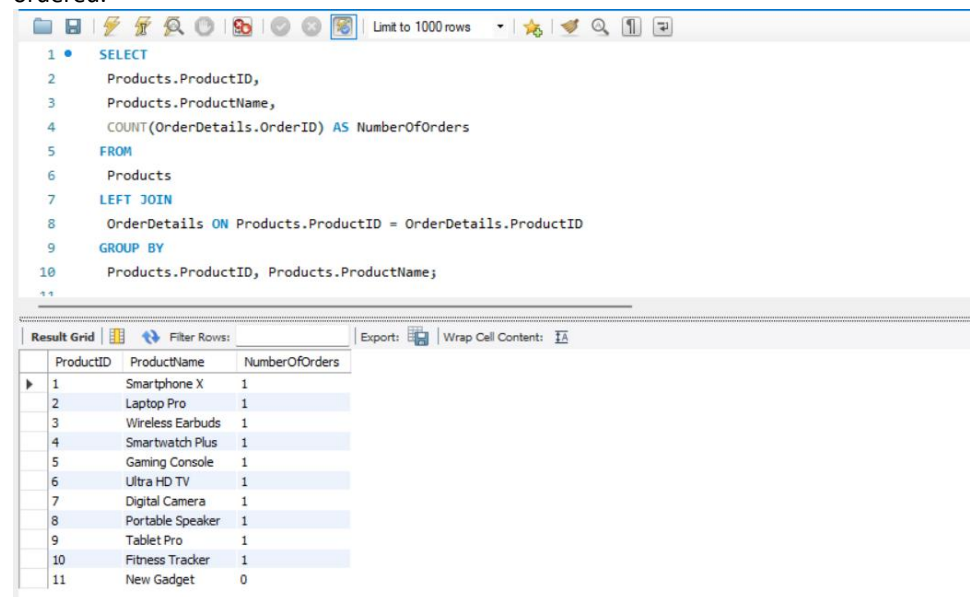
1 • SELECT
2   Orders.OrderID,
3   Customers.FirstName,
4   Customers.LastName,
5   Customers.Email,
6   Customers.Phone,
7   Customers.Address,
8   Orders.TotalAmount AS TotalRevenue
9 FROM
10  Orders
11 JOIN
12  Customers ON Orders.CustomerID = Customers.CustomerID
13 ORDER BY
14   TotalRevenue DESC
15 LIMIT 1;

```

Result Grid

OrderID	FirstName	LastName	Email	Phone	Address	TotalRevenue
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St	3189.97

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



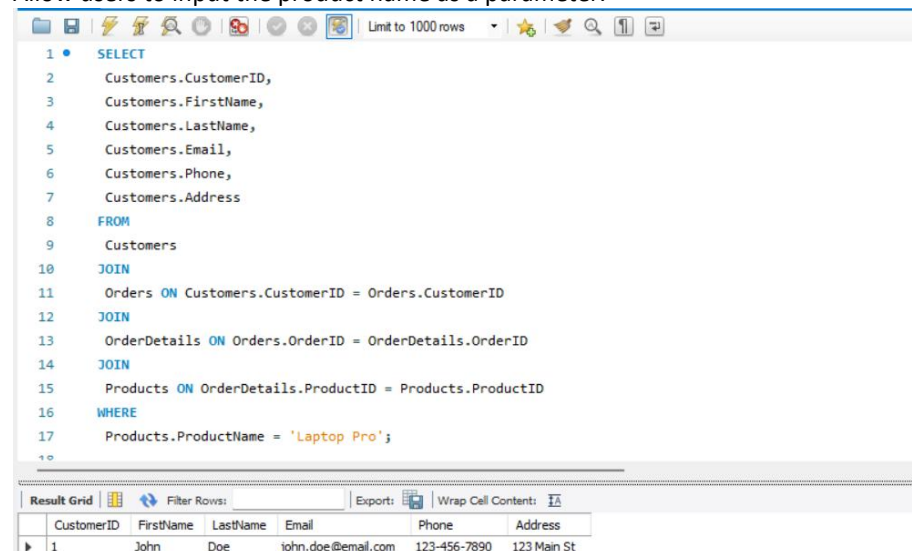
```

1 • SELECT
2     Products.ProductID,
3     Products.ProductName,
4     COUNT(OrderDetails.OrderID) AS NumberOfOrders
5 FROM
6     Products
7 LEFT JOIN
8     OrderDetails ON Products.ProductID = OrderDetails.ProductID
9 GROUP BY
10    Products.ProductID, Products.ProductName;

```

ProductID	ProductName	NumberOfOrders
1	Smartphone X	1
2	Laptop Pro	1
3	Wireless Earbuds	1
4	Smartwatch Plus	1
5	Gaming Console	1
6	Ultra HD TV	1
7	Digital Camera	1
8	Portable Speaker	1
9	Tablet Pro	1
10	Fitness Tracker	1
11	New Gadget	0

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.



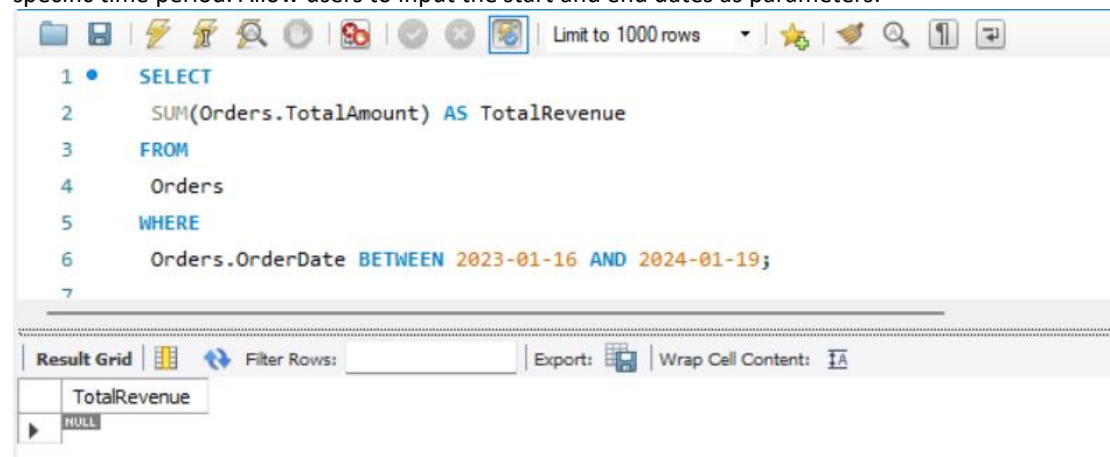
```

1 • SELECT
2     Customers.CustomerID,
3     Customers.FirstName,
4     Customers.LastName,
5     Customers.Email,
6     Customers.Phone,
7     Customers.Address
8 FROM
9     Customers
10 JOIN
11     Orders ON Customers.CustomerID = Orders.CustomerID
12 JOIN
13     OrderDetails ON Orders.OrderID = OrderDetails.OrderID
14 JOIN
15     Products ON OrderDetails.ProductID = Products.ProductID
16 WHERE
17     Products.ProductName = 'Laptop Pro';

```

CustomerID	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St

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.



```

1 • SELECT
2     SUM(Orders.TotalAmount) AS TotalRevenue
3 FROM
4     Orders
5 WHERE
6     Orders.OrderDate BETWEEN 2023-01-16 AND 2024-01-19;
7

```

TotalRevenue
NULL

Task 4. Subquery and its type:

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

```
3  FirstName,  
4  LastName,  
5  Email,  
6  Phone,  
7  Address  
8  FROM  
9  Customers  
10 WHERE  
11 NOT EXISTS (  
12 SELECT 1  
13 FROM Orders  
14 WHERE Customers.CustomerID = Orders.CustomerID  
15 );
```

Result Grid

CustomerID	FirstName	LastName	Email	Phone	Address
11	New	Customer	new.customer@email.com	555-123-4567	789 Broadway St

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

```
1 • SELECT COUNT(*) AS TotalProducts  
2 FROM Products;  
3  
4  
5
```

Result Grid

TotalProducts
11

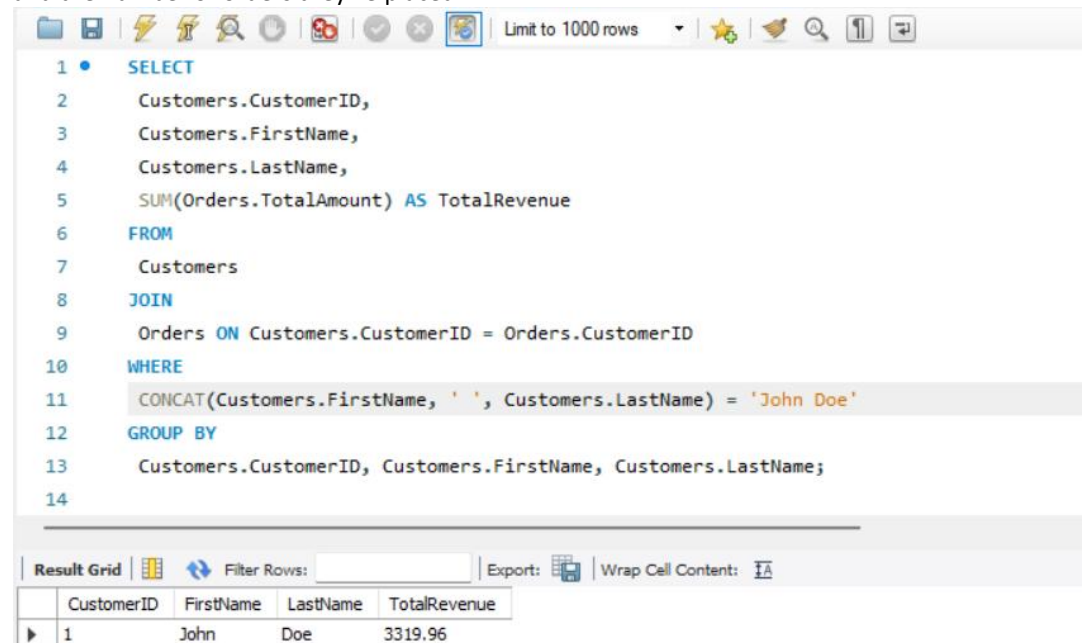
3. Write an SQL query to calculate the total revenue generated by TechShop.

```
1 • SELECT SUM(TotalAmount) AS TotalRevenue  
2 FROM Orders;  
3  
4  
5
```

Result Grid

TotalRevenue
7367.82

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.



The screenshot shows a SQL query in a text editor. The query selects customer information and calculates the total revenue for each customer. The result grid below the query shows one row for John Doe with a total revenue of 3319.96.

```

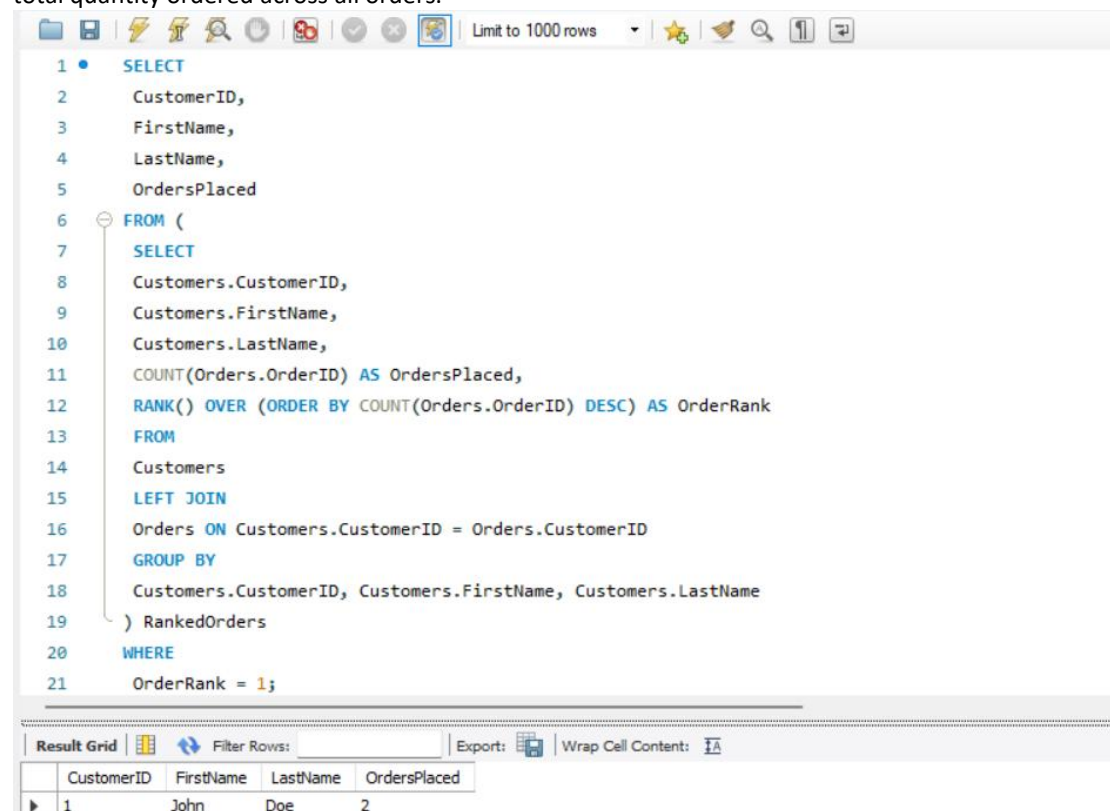
1 • SELECT
2   Customers.CustomerID,
3   Customers.FirstName,
4   Customers.LastName,
5   SUM(Orders.TotalAmount) AS TotalRevenue
6 FROM
7   Customers
8 JOIN
9   Orders ON Customers.CustomerID = Orders.CustomerID
10 WHERE
11   CONCAT(Customers.FirstName, ' ', Customers.LastName) = 'John Doe'
12 GROUP BY
13   Customers.CustomerID, Customers.FirstName, Customers.LastName;
14

```

Result Grid

	CustomerID	FirstName	LastName	TotalRevenue
▶	1	John	Doe	3319.96

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.



The screenshot shows a SQL query in a text editor. The query uses a subquery to rank customers by the number of orders placed. The result grid below the query shows one row for John Doe with 2 orders placed.

```

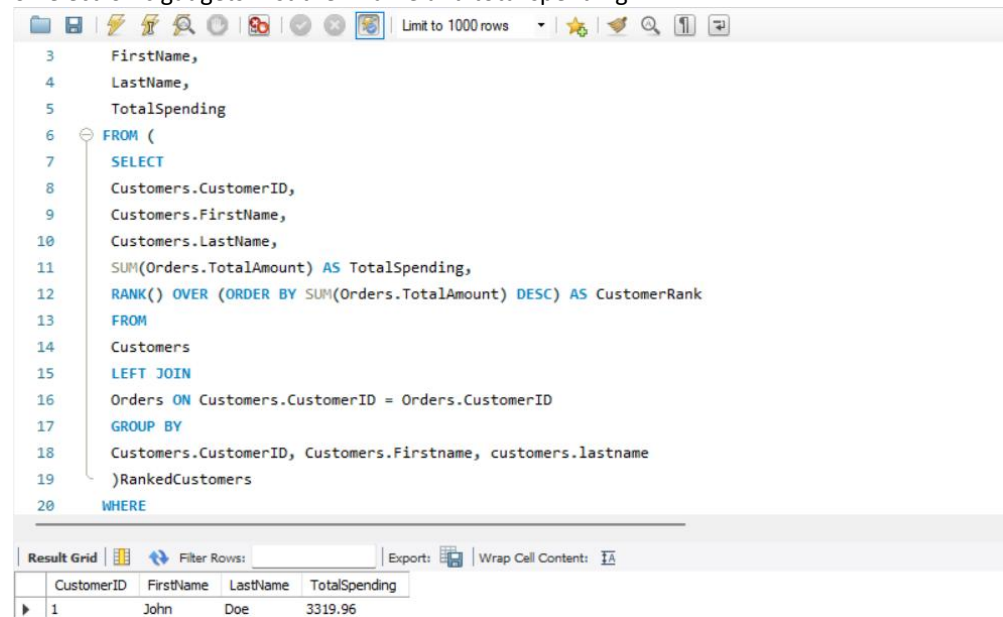
1 • SELECT
2   CustomerID,
3   FirstName,
4   LastName,
5   OrdersPlaced
6 FROM (
7   SELECT
8     Customers.CustomerID,
9     Customers.FirstName,
10    Customers.LastName,
11    COUNT(Orders.OrderID) AS OrdersPlaced,
12    RANK() OVER (ORDER BY COUNT(Orders.OrderID) DESC) AS OrderRank
13   FROM
14     Customers
15   LEFT JOIN
16     Orders ON Customers.CustomerID = Orders.CustomerID
17   GROUP BY
18     Customers.CustomerID, Customers.FirstName, Customers.LastName
19 ) RankedOrders
20 WHERE
21   OrderRank = 1;

```

Result Grid

	CustomerID	FirstName	LastName	OrdersPlaced
▶	1	John	Doe	2

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.



```

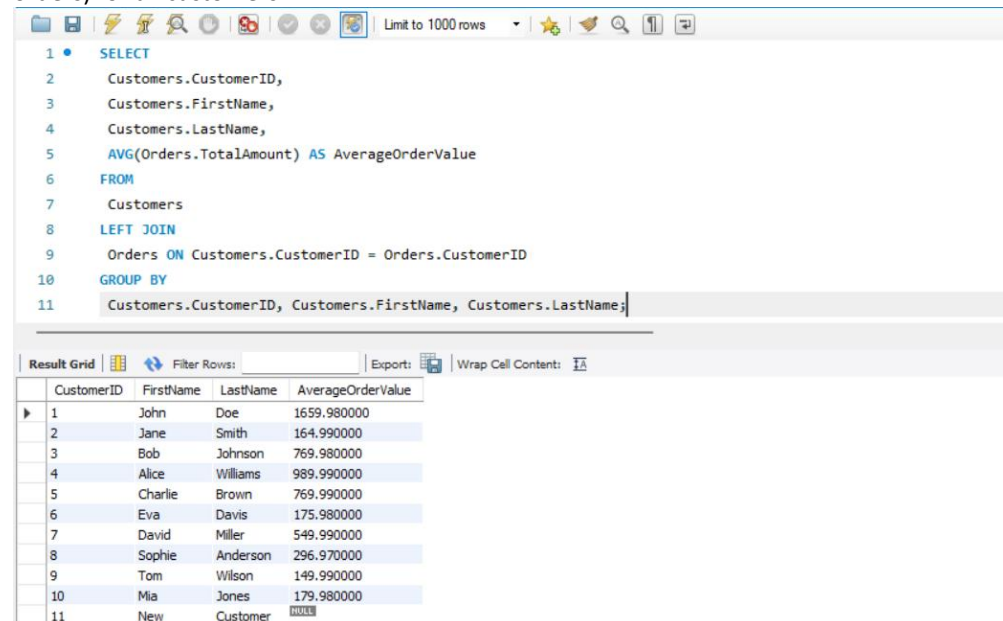
3  FirstName,
4  LastName,
5  TotalSpending
6  FROM (
7  SELECT
8  Customers.CustomerID,
9  Customers.FirstName,
10 Customers.LastName,
11 SUM(Orders.TotalAmount) AS TotalSpending,
12 RANK() OVER (ORDER BY SUM(Orders.TotalAmount) DESC) AS CustomerRank
13 FROM
14 Customers
15 LEFT JOIN
16 Orders ON Customers.CustomerID = Orders.CustomerID
17 GROUP BY
18 Customers.CustomerID, Customers.FirstName, customers.lastname
19 )RankedCustomers
20 WHERE

```

Result Grid

	CustomerID	FirstName	LastName	TotalSpending
1	John	Doe	3319.96	

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



```

1  SELECT
2  Customers.CustomerID,
3  Customers.FirstName,
4  Customers.LastName,
5  AVG(Orders.TotalAmount) AS AverageOrderValue
6  FROM
7  Customers
8  LEFT JOIN
9  Orders ON Customers.CustomerID = Orders.CustomerID
10 GROUP BY
11 Customers.CustomerID, Customers.FirstName, Customers.LastName;

```

Result Grid

	CustomerID	FirstName	LastName	AverageOrderValue
1	John	Doe	1659.980000	
2	Jane	Smith	164.990000	
3	Bob	Johnson	769.980000	
4	Alice	Williams	989.990000	
5	Charlie	Brown	769.990000	
6	Eva	Davis	175.980000	
7	David	Miller	549.990000	
8	Sophie	Anderson	296.970000	
9	Tom	Wilson	149.990000	
10	Mia	Jones	179.980000	
11	New	Customer	NULL	