

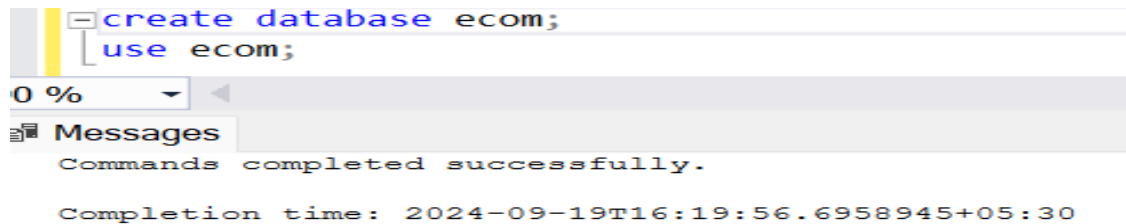
# SQL CODING CHALLENGE - ECOM

HARSHINI V

-- Create a database "ecom"

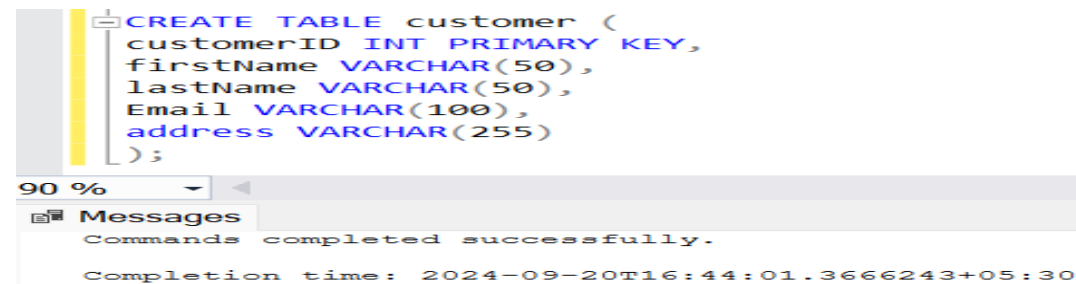
```
create database ecom;
```

```
use ecom;
```

A screenshot of a SQL command window. The command bar shows 'create database ecom;' and 'use ecom;'. Below the command bar, a scroll bar is at 0%. A 'Messages' tab is active, displaying 'Commands completed successfully.' and 'Completion time: 2024-09-19T16:19:56.6958945+05:30'.

-- Create table customer

```
CREATE TABLE customer (  
customerID INT PRIMARY KEY,  
firstName VARCHAR(50),  
lastName VARCHAR(50),  
Email VARCHAR(100),  
address VARCHAR(255)  
);
```

A screenshot of a SQL command window. The command bar shows 'CREATE TABLE customer (' followed by column definitions: 'customerID INT PRIMARY KEY,', 'firstName VARCHAR(50),', 'lastName VARCHAR(50),', 'Email VARCHAR(100),', and 'address VARCHAR(255)'. The command ends with ');'. Below the command bar, a scroll bar is at 90%. A 'Messages' tab is active, displaying 'Commands completed successfully.' and 'Completion time: 2024-09-20T16:44:01.3666243+05:30'.

-- Create table product

```
CREATE TABLE product (  
productID INT PRIMARY KEY,  
name VARCHAR(100),  
Description VARCHAR(255),  
price DECIMAL(10, 2),  
stockQuantity INT  
);
```

```
CREATE TABLE product (  
  productID INT PRIMARY KEY,  
  name VARCHAR(100),  
  Description VARCHAR(255),  
  price DECIMAL(10, 2),  
  stockQuantity INT  
);
```

90 %

#### Messages

Commands completed successfully.

Completion time: 2024-09-20T16:48:20.1787531+05:30

#### -- Create table cart

```
CREATE TABLE cart (  
  
  cartID INT PRIMARY KEY,  
  
  customerID INT,  
  
  productID INT,  
  
  quantity INT,  
  
  FOREIGN KEY (customerID) REFERENCES customer(customerID),  
  
  FOREIGN KEY (productID) REFERENCES product(productID)  
  
);
```

```
CREATE TABLE cart (  
  cartID INT PRIMARY KEY,  
  customerID INT,  
  productID INT,  
  quantity INT,  
  FOREIGN KEY (customerID) REFERENCES customer(customerID),  
  FOREIGN KEY (productID) REFERENCES product(productID)  
);
```

90 %

#### Messages

Commands completed successfully.

Completion time: 2024-09-20T16:49:05.5648434+05:30

#### -- Create table orders

```
CREATE TABLE orders(  
  
  orderID INT PRIMARY KEY,  
  
  customerID INT,  
  
  orderDate DATE,  
  
  totalAmount DECIMAL(10, 2),  
  
  FOREIGN KEY (customerID) REFERENCES customer(customerID)  
  
);
```

```
CREATE TABLE orders(  
  orderID INT PRIMARY KEY,  
  customerID INT,  
  orderDate DATE,  
  totalAmount DECIMAL(10, 2),  
  FOREIGN KEY (customerID) REFERENCES customer(customerID)  
);
```

90 %

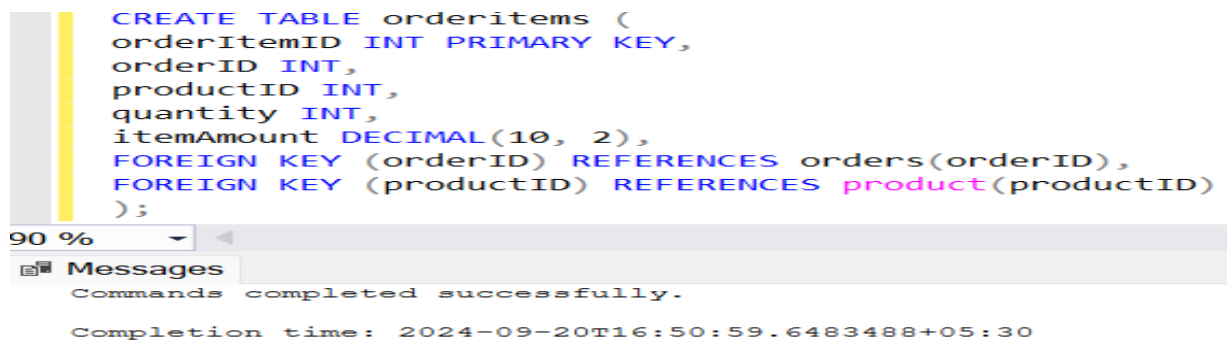
#### Messages

Commands completed successfully.

Completion time: 2024-09-20T16:49:52.1171114+05:30

### -- Create table orderitems

```
CREATE TABLE orderitems (  
orderItemID INT PRIMARY KEY,  
orderID INT,  
productID INT,  
quantity INT,  
itemAmount DECIMAL(10, 2),  
FOREIGN KEY (orderID) REFERENCES orders(orderID),  
FOREIGN KEY (productID) REFERENCES product(productID)  
);
```



```
CREATE TABLE orderitems (  
orderItemID INT PRIMARY KEY,  
orderID INT,  
productID INT,  
quantity INT,  
itemAmount DECIMAL(10, 2),  
FOREIGN KEY (orderID) REFERENCES orders(orderID),  
FOREIGN KEY (productID) REFERENCES product(productID)  
);
```

90 %


Messages

Commands completed successfully.

Completion time: 2024-09-20T16:50:59.6483488+05:30

### -- insert values in table customer

```
INSERT INTO customer (customerID, firstName, lastName, Email, address) VALUES  
(1, 'John', 'Doe', 'johndoe@example.com', '123 Main St, City'),  
(2, 'Jane', 'Smith', 'janesmith@example.com', '456 Elm St, Town'),  
(3, 'Robert', 'Johnson', 'robert@example.com', '789 Oak St, Village'),  
(4, 'Sarah', 'Brown', 'sarah@example.com', '101 Pine St, Suburb'),  
(5, 'David', 'Lee', 'david@example.com', '234 Cedar St, District'),  
(6, 'Laura', 'Hall', 'laura@example.com', '567 Birch St, County'),  
(7, 'Michael', 'Davis', 'michael@example.com', '890 Maple St, State'),  
(8, 'Emma', 'Wilson', 'emma@example.com', '321 Redwood St, Country'),  
(9, 'William', 'Taylor', 'william@example.com', '432 Spruce St, Province'),  
(10, 'Olivia', 'Adams', 'olivia@example.com', '765 Fir St, Territory');
```



```
INSERT INTO customer (customerID, firstName, lastName, Email, address) VALUES  
(1, 'John', 'Doe', 'johndoe@example.com', '123 Main St, City'),  
(2, 'Jane', 'Smith', 'janesmith@example.com', '456 Elm St, Town'),  
(3, 'Robert', 'Johnson', 'robert@example.com', '789 Oak St, Village'),  
(4, 'Sarah', 'Brown', 'sarah@example.com', '101 Pine St, Suburb'),  
(5, 'David', 'Lee', 'david@example.com', '234 Cedar St, District'),  
(6, 'Laura', 'Hall', 'laura@example.com', '567 Birch St, County'),  
(7, 'Michael', 'Davis', 'michael@example.com', '890 Maple St, State'),  
(8, 'Emma', 'Wilson', 'emma@example.com', '321 Redwood St, Country'),  
(9, 'William', 'Taylor', 'william@example.com', '432 Spruce St, Province'),  
(10, 'Olivia', 'Adams', 'olivia@example.com', '765 Fir St, Territory');
```

30 %

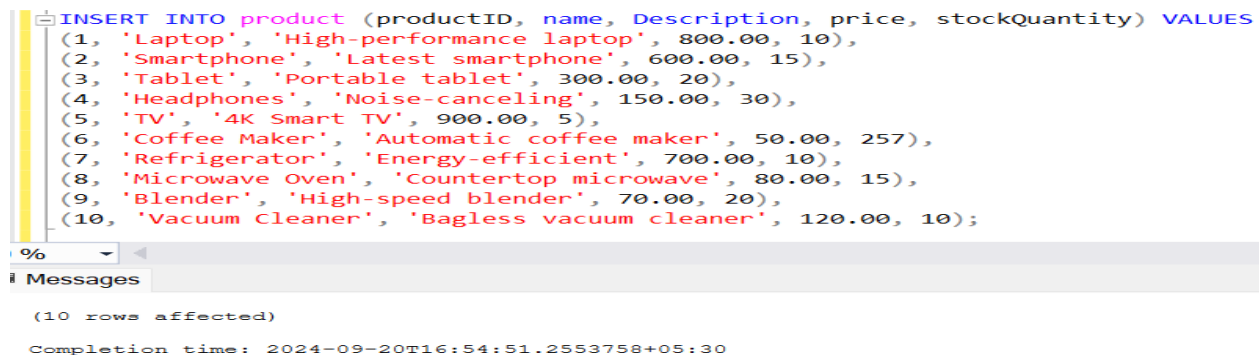
Messages

(10 rows affected)

Completion time: 2024-09-20T16:53:29.4467679+05:30

### -- insert values in table product

```
INSERT INTO product (productID, name, Description, price, stockQuantity) VALUES
(1, 'Laptop', 'High-performance laptop', 800.00, 10),
(2, 'Smartphone', 'Latest smartphone', 600.00, 15),
(3, 'Tablet', 'Portable tablet', 300.00, 20),
(4, 'Headphones', 'Noise-canceling', 150.00, 30),
(5, 'TV', '4K Smart TV', 900.00, 5),
(6, 'Coffee Maker', 'Automatic coffee maker', 50.00, 257),
(7, 'Refrigerator', 'Energy-efficient', 700.00, 10),
(8, 'Microwave Oven', 'Countertop microwave', 80.00, 15),
(9, 'Blender', 'High-speed blender', 70.00, 20),
(10, 'Vacuum Cleaner', 'Bagless vacuum cleaner', 120.00, 10);
```



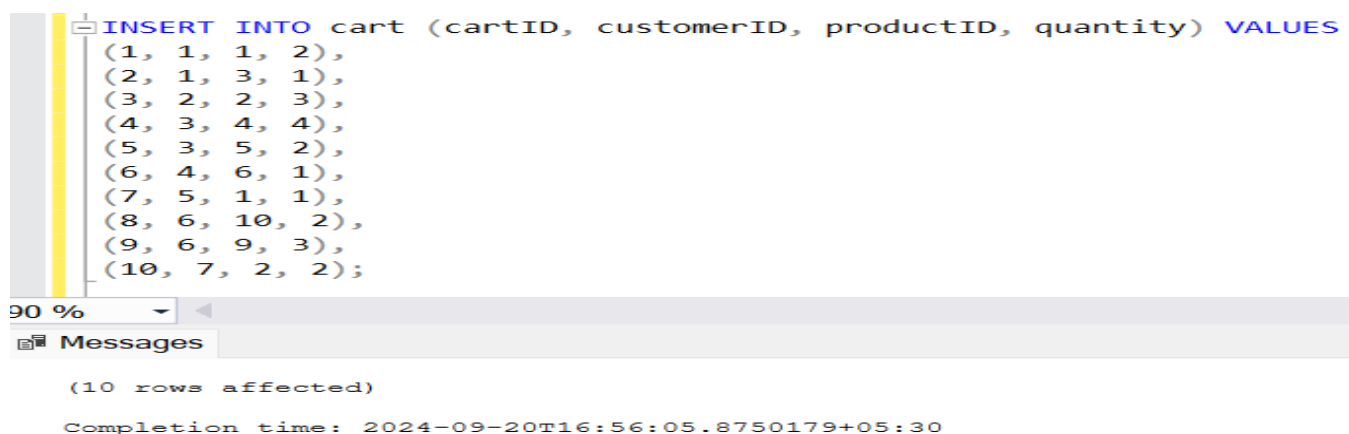
```
INSERT INTO product (productID, name, Description, price, stockQuantity) VALUES
(1, 'Laptop', 'High-performance laptop', 800.00, 10),
(2, 'Smartphone', 'Latest smartphone', 600.00, 15),
(3, 'Tablet', 'Portable tablet', 300.00, 20),
(4, 'Headphones', 'Noise-canceling', 150.00, 30),
(5, 'TV', '4K Smart TV', 900.00, 5),
(6, 'Coffee Maker', 'Automatic coffee maker', 50.00, 257),
(7, 'Refrigerator', 'Energy-efficient', 700.00, 10),
(8, 'Microwave Oven', 'Countertop microwave', 80.00, 15),
(9, 'Blender', 'High-speed blender', 70.00, 20),
(10, 'Vacuum Cleaner', 'Bagless vacuum cleaner', 120.00, 10);
```

(10 rows affected)

Completion time: 2024-09-20T16:54:51.2553758+05:30

### -- insert values in table cart

```
INSERT INTO cart (cartID, customerID, productID, quantity) VALUES
(1, 1, 1, 2),
(2, 1, 3, 1),
(3, 2, 2, 3),
(4, 3, 4, 4),
(5, 3, 5, 2),
(6, 4, 6, 1),
(7, 5, 1, 1),
(8, 6, 10, 2),
(9, 6, 9, 3),
(10, 7, 2, 2);
```



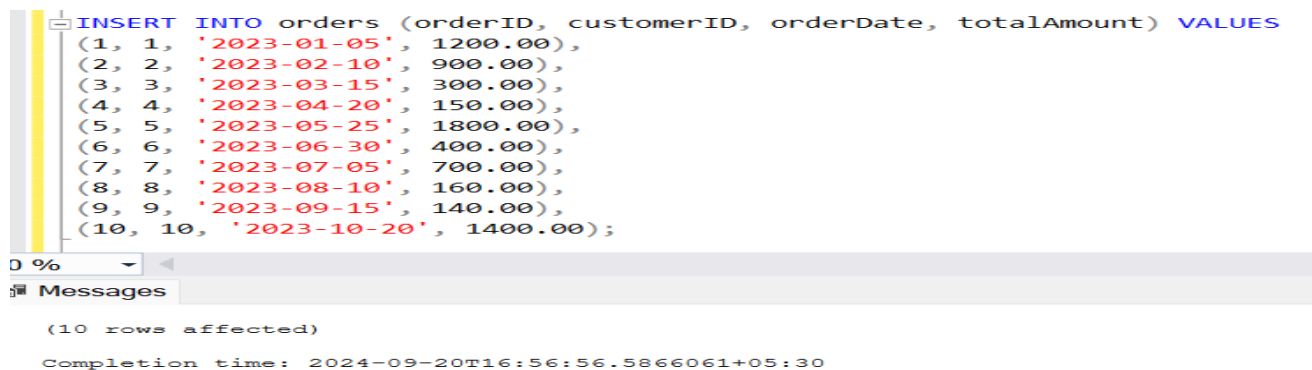
```
INSERT INTO cart (cartID, customerID, productID, quantity) VALUES
(1, 1, 1, 2),
(2, 1, 3, 1),
(3, 2, 2, 3),
(4, 3, 4, 4),
(5, 3, 5, 2),
(6, 4, 6, 1),
(7, 5, 1, 1),
(8, 6, 10, 2),
(9, 6, 9, 3),
(10, 7, 2, 2);
```

(10 rows affected)

Completion time: 2024-09-20T16:56:05.8750179+05:30

### -- insert values in table orders

```
INSERT INTO orders (orderId, customerId, orderDate, totalAmount) VALUES
(1, 1, '2023-01-05', 1200.00),
(2, 2, '2023-02-10', 900.00),
(3, 3, '2023-03-15', 300.00),
(4, 4, '2023-04-20', 150.00),
(5, 5, '2023-05-25', 1800.00),
(6, 6, '2023-06-30', 400.00),
(7, 7, '2023-07-05', 700.00),
(8, 8, '2023-08-10', 160.00),
(9, 9, '2023-09-15', 140.00),
(10, 10, '2023-10-20', 1400.00);
```



```
INSERT INTO orders (orderId, customerId, orderDate, totalAmount) VALUES
(1, 1, '2023-01-05', 1200.00),
(2, 2, '2023-02-10', 900.00),
(3, 3, '2023-03-15', 300.00),
(4, 4, '2023-04-20', 150.00),
(5, 5, '2023-05-25', 1800.00),
(6, 6, '2023-06-30', 400.00),
(7, 7, '2023-07-05', 700.00),
(8, 8, '2023-08-10', 160.00),
(9, 9, '2023-09-15', 140.00),
(10, 10, '2023-10-20', 1400.00);
```

0 %

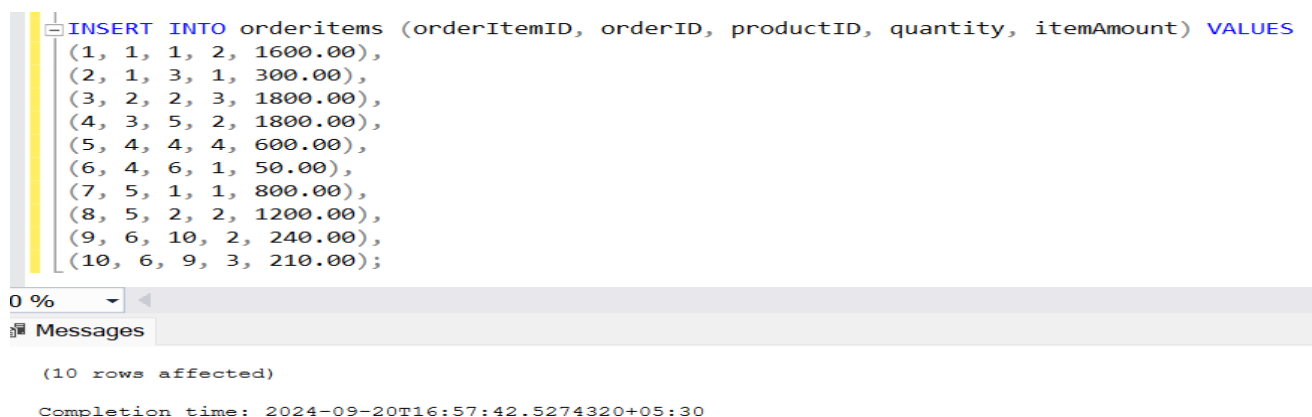
Messages

(10 rows affected)

Completion time: 2024-09-20T16:56:56.5866061+05:30

### -- insert values in table orderitems

```
INSERT INTO orderitems (orderItemID, orderId, productID, quantity, itemAmount) VALUES
(1, 1, 1, 2, 1600.00),
(2, 1, 3, 1, 300.00),
(3, 2, 2, 3, 1800.00),
(4, 3, 5, 2, 1800.00),
(5, 4, 4, 4, 600.00),
(6, 4, 6, 1, 50.00),
(7, 5, 1, 1, 800.00),
(8, 5, 2, 2, 1200.00),
(9, 6, 10, 2, 240.00),
(10, 6, 9, 3, 210.00);
```



```
INSERT INTO orderitems (orderItemID, orderId, productID, quantity, itemAmount) VALUES
(1, 1, 1, 2, 1600.00),
(2, 1, 3, 1, 300.00),
(3, 2, 2, 3, 1800.00),
(4, 3, 5, 2, 1800.00),
(5, 4, 4, 4, 600.00),
(6, 4, 6, 1, 50.00),
(7, 5, 1, 1, 800.00),
(8, 5, 2, 2, 1200.00),
(9, 6, 10, 2, 240.00),
(10, 6, 9, 3, 210.00);
```

0 %

Messages

(10 rows affected)

Completion time: 2024-09-20T16:57:42.5274320+05:30

-- 1) Update refrigerator product price to 800.

UPDATE product

SET price = 800.00

WHERE productID = 7;

```
UPDATE product
SET price = 800.00
WHERE productID = 7;
```

1 %

Messages

(1 row affected)

Completion time: 2024-09-20T16:59:44.4223480+05:30

Results		Messages			
	productID	name	Description	price	stockQuantity
1	1	Laptop	High-performance laptop	800.00	10
2	2	Smartphone	Latest smartphone	600.00	15
3	3	Tablet	Portable tablet	300.00	20
4	4	Headphones	Noise-canceling	150.00	30
5	5	TV	4K Smart TV	900.00	5
6	6	Coffee Maker	Automatic coffee maker	50.00	257
7	7	Refrigerator	Energy-efficient	800.00	10
8	8	Microwave Oven	Countertop microwave	80.00	15
9	9	Blender	High-speed blender	70.00	20
10	10	Vacuum Clean...	Bagless vacuum cleaner	120.00	10

-- 2) Remove all cart items for a specific customer.

DELETE FROM cart

WHERE customerID = 3;

```
DELETE FROM cart
WHERE customerID = 3;
```

90 %

Messages

(2 rows affected)

Completion time: 2024-09-20T17:01:25.6400251+05:30

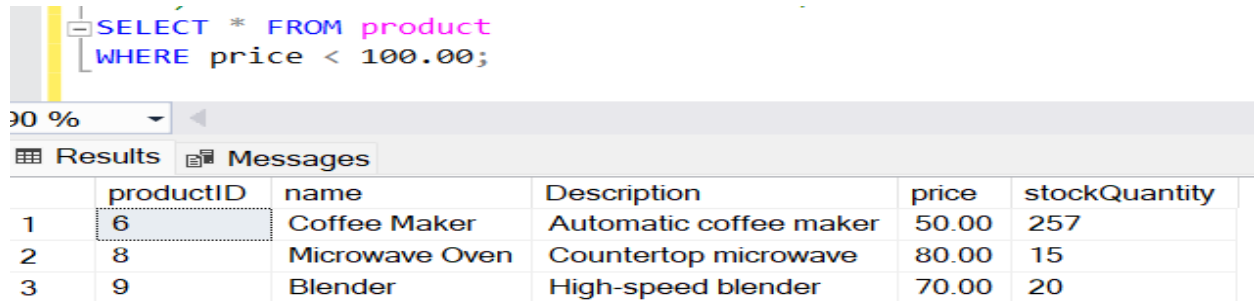
Results		Messages		
	cartID	customerID	productID	quantity
1	1	1	1	2
2	2	1	3	1
3	3	2	2	3
4	6	4	6	1
5	7	5	1	1
6	8	6	10	2
7	9	6	9	3
8	10	7	2	2

-- 3) Retrieve Products Priced Below \$100.

SELECT \*

FROM product

WHERE price < 100.00;



90 %

Results Messages

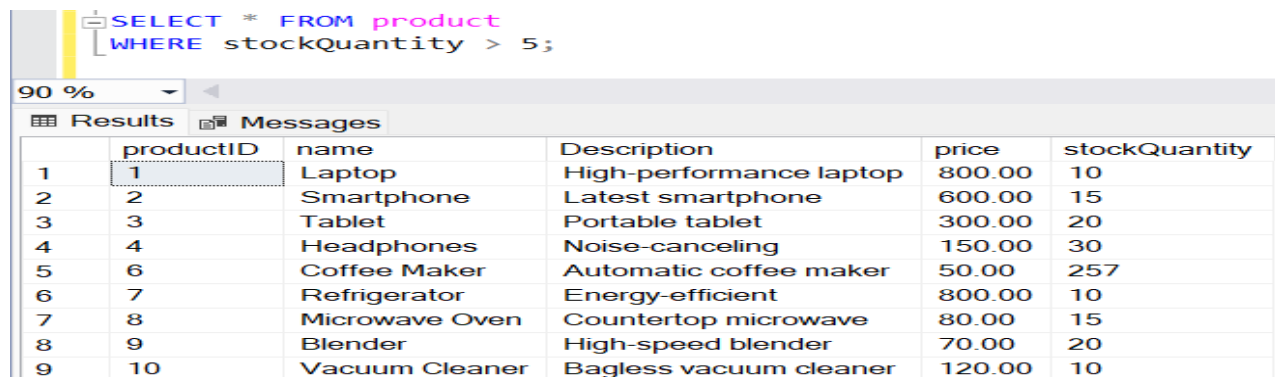
	productID	name	Description	price	stockQuantity
1	6	Coffee Maker	Automatic coffee maker	50.00	257
2	8	Microwave Oven	Countertop microwave	80.00	15
3	9	Blender	High-speed blender	70.00	20

-- 4) Find Products with Stock Quantity Greater Than 5.

SELECT \*

FROM product

WHERE stockQuantity > 5;



90 %

Results Messages

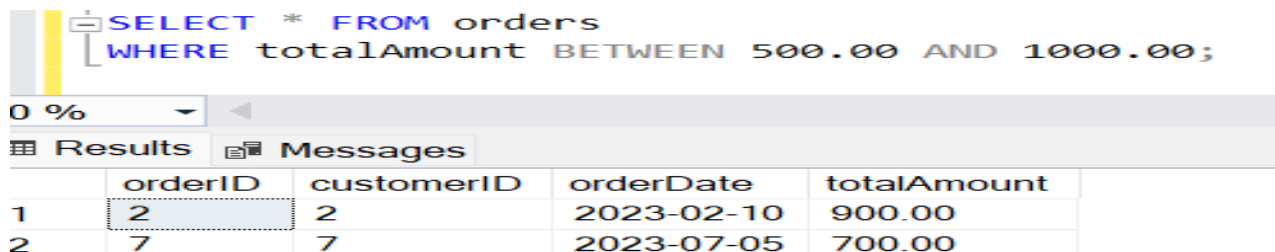
	productID	name	Description	price	stockQuantity
1	1	Laptop	High-performance laptop	800.00	10
2	2	Smartphone	Latest smartphone	600.00	15
3	3	Tablet	Portable tablet	300.00	20
4	4	Headphones	Noise-canceling	150.00	30
5	6	Coffee Maker	Automatic coffee maker	50.00	257
6	7	Refrigerator	Energy-efficient	800.00	10
7	8	Microwave Oven	Countertop microwave	80.00	15
8	9	Blender	High-speed blender	70.00	20
9	10	Vacuum Cleaner	Bagless vacuum cleaner	120.00	10

-- 5) Retrieve Orders with Total Amount Between \$500 and \$1000.

SELECT \*

FROM orders

WHERE totalAmount BETWEEN 500.00 AND 1000.00;



0 %

Results Messages

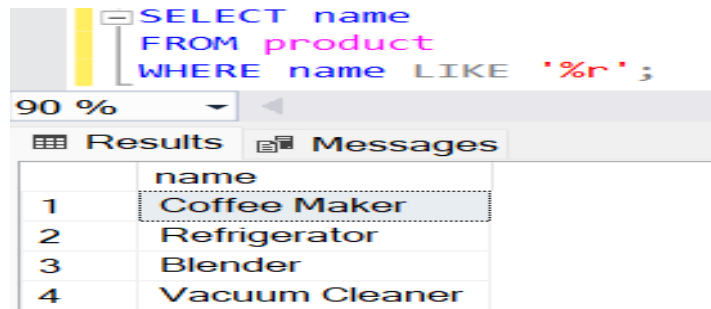
	orderID	customerID	orderDate	totalAmount
1	2	2	2023-02-10	900.00
2	7	7	2023-07-05	700.00

--6) Find Products which name end with letter 'r'.

SELECT name

FROM product

WHERE name LIKE '%r';



The screenshot shows a SQL query editor with the following query: `SELECT name FROM product WHERE name LIKE '%r';`. Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with 4 rows and 1 column named 'name'. The rows contain the following values: 'Coffee Maker', 'Refrigerator', 'Blender', and 'Vacuum Cleaner'.

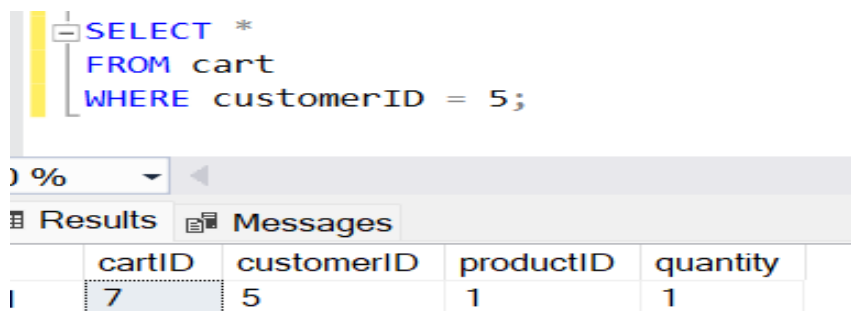
	name
1	Coffee Maker
2	Refrigerator
3	Blender
4	Vacuum Cleaner

--7) Retrieve Cart Items for Customer 5.

SELECT \*

FROM cart

WHERE customerID = 5;



The screenshot shows a SQL query editor with the following query: `SELECT * FROM cart WHERE customerID = 5;`. Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with 4 columns: 'cartID', 'customerID', 'productID', and 'quantity'. The first row contains the values: 7, 5, 1, and 1.

	cartID	customerID	productID	quantity
1	7	5	1	1

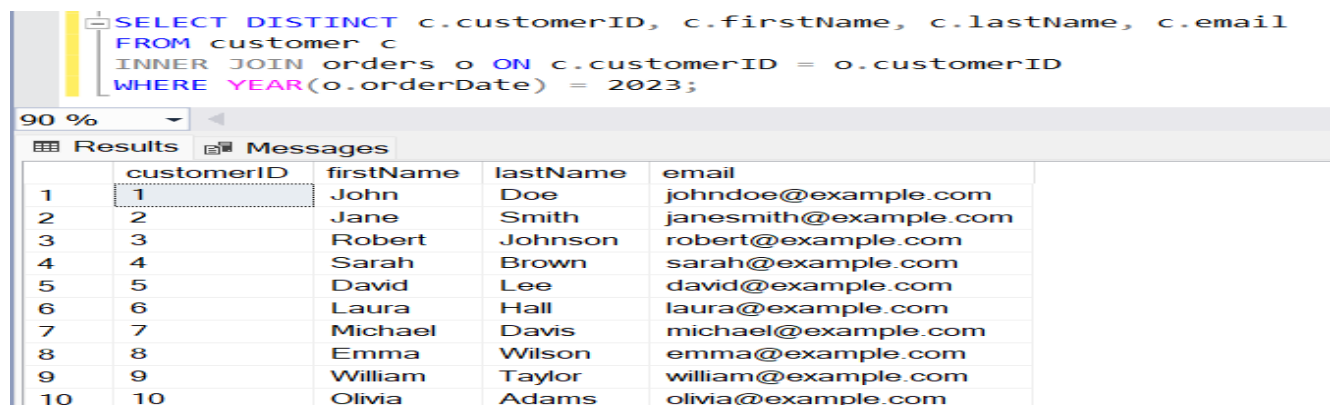
-- 8) Find Customers Who Placed Orders in 2023.

SELECT DISTINCT c.customerID, c.firstName, c.lastName, c.email

FROM customer c

INNER JOIN orders o ON c.customerID = o.customerID

WHERE YEAR(o.orderDate) = 2023;



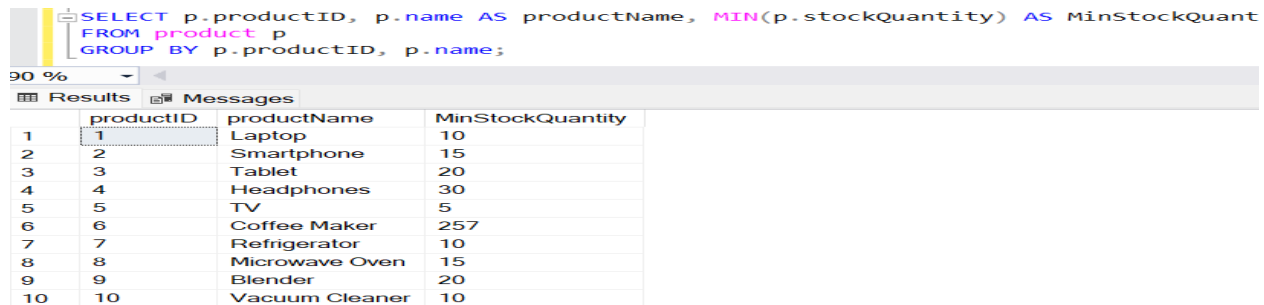
The screenshot shows a SQL query editor with the following query: `SELECT DISTINCT c.customerID, c.firstName, c.lastName, c.email FROM customer c INNER JOIN orders o ON c.customerID = o.customerID WHERE YEAR(o.orderDate) = 2023;`. Below the query, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with 5 columns: 'customerID', 'firstName', 'lastName', and 'email'. The table contains 10 rows of data.

	customerID	firstName	lastName	email
1	1	John	Doe	johndoe@example.com
2	2	Jane	Smith	janesmith@example.com
3	3	Robert	Johnson	robert@example.com
4	4	Sarah	Brown	sarah@example.com
5	5	David	Lee	david@example.com
6	6	Laura	Hall	laura@example.com
7	7	Michael	Davis	michael@example.com
8	8	Emma	Wilson	emma@example.com
9	9	William	Taylor	william@example.com
10	10	Olivia	Adams	olivia@example.com



-- 9) Determine the Minimum Stock Quantity for Each Product Category.

```
SELECT p.productID, p.name AS productName, MIN(p.stockQuantity) AS MinStockQuantity
FROM product p
GROUP BY p.productID, p.name;
```

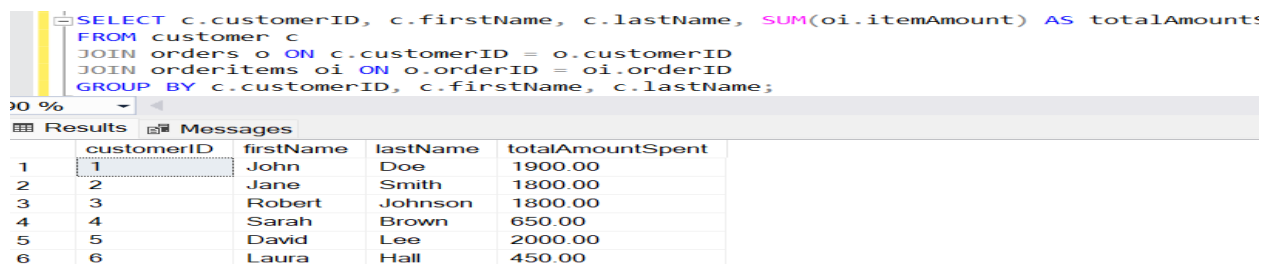


The screenshot shows a SQL query window with the following query: `SELECT p.productID, p.name AS productName, MIN(p.stockQuantity) AS MinStockQuantity FROM product p GROUP BY p.productID, p.name;` The results pane displays a table with 10 rows. The first row is highlighted.

	productID	productName	MinStockQuantity
1	1	Laptop	10
2	2	Smartphone	15
3	3	Tablet	20
4	4	Headphones	30
5	5	TV	5
6	6	Coffee Maker	257
7	7	Refrigerator	10
8	8	Microwave Oven	15
9	9	Blender	20
10	10	Vacuum Cleaner	10

-- 10) Calculate the Total Amount Spent by Each Customer.

```
SELECT c.customerID, c.firstName, c.lastName, SUM(oi.itemAmount) AS totalAmountSpent
FROM customer c
JOIN orders o ON c.customerID = o.customerID
JOIN orderitems oi ON o.orderID = oi.orderID
GROUP BY c.customerID, c.firstName, c.lastName;
```

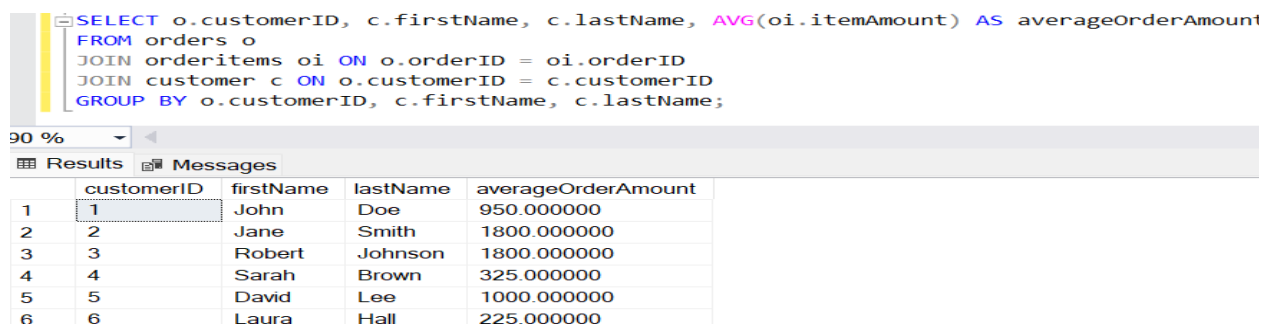


The screenshot shows a SQL query window with the following query: `SELECT c.customerID, c.firstName, c.lastName, SUM(oi.itemAmount) AS totalAmountSpent FROM customer c JOIN orders o ON c.customerID = o.customerID JOIN orderitems oi ON o.orderID = oi.orderID GROUP BY c.customerID, c.firstName, c.lastName;` The results pane displays a table with 6 rows. The first row is highlighted.

	customerID	firstName	lastName	totalAmountSpent
1	1	John	Doe	1900.00
2	2	Jane	Smith	1800.00
3	3	Robert	Johnson	1800.00
4	4	Sarah	Brown	650.00
5	5	David	Lee	2000.00
6	6	Laura	Hall	450.00

-- 11) Find the Average Order Amount for Each Customer.

```
SELECT o.customerID, c.firstName, c.lastName, AVG(oi.itemAmount) AS averageOrderAmount
FROM orders o
JOIN orderitems oi ON o.orderID = oi.orderID
JOIN customer c ON o.customerID = c.customerID
GROUP BY o.customerID, c.firstName, c.lastName;
```



The screenshot shows a SQL query window with the following query: `SELECT o.customerID, c.firstName, c.lastName, AVG(oi.itemAmount) AS averageOrderAmount FROM orders o JOIN orderitems oi ON o.orderID = oi.orderID JOIN customer c ON o.customerID = c.customerID GROUP BY o.customerID, c.firstName, c.lastName;` The results pane displays a table with 6 rows. The first row is highlighted.

	customerID	firstName	lastName	averageOrderAmount
1	1	John	Doe	950.000000
2	2	Jane	Smith	1800.000000
3	3	Robert	Johnson	1800.000000
4	4	Sarah	Brown	325.000000
5	5	David	Lee	1000.000000
6	6	Laura	Hall	225.000000

**--12) Count the Number of Orders Placed by Each Customer.**

```
SELECT o.customerID, c.firstName, c.lastName, COUNT(o.orderID) AS numberOfOrders
FROM orders o
JOIN customer c ON o.customerID = c.customerID
GROUP BY o.customerID, c.firstName, c.lastName;
```

```
SELECT o.customerID, c.firstName, c.lastName, COUNT(o.orderID) AS numberOfOrders
FROM orders o
JOIN customer c ON o.customerID = c.customerID
GROUP BY o.customerID, c.firstName, c.lastName;
```

90 %				
Results Messages				
	customerID	firstName	lastName	numberOfOrders
1	1	John	Doe	1
2	2	Jane	Smith	1
3	3	Robert	Johnson	1
4	4	Sarah	Brown	1
5	5	David	Lee	1
6	6	Laura	Hall	1
7	7	Michael	Davis	1
8	8	Emma	Wilson	1
9	9	William	Taylor	1
10	10	Olivia	Adams	1

**-- 13) Find the Maximum Order Amount for Each Customer.**

```
SELECT o.customerID, c.firstName, c.lastName, MAX(oi.itemAmount) AS maxOrderAmount
FROM orders o
JOIN orderitems oi ON o.orderID = oi.orderID
JOIN customer c ON o.customerID = c.customerID
GROUP BY o.customerID, c.firstName, c.lastName;
```

```
SELECT o.customerID, c.firstName, c.lastName, MAX(oi.itemAmount) AS maxOrderAmount
FROM orders o
JOIN orderitems oi ON o.orderID = oi.orderID
JOIN customer c ON o.customerID = c.customerID
GROUP BY o.customerID, c.firstName, c.lastName;
```

90 %				
Results Messages				
	customerID	firstName	lastName	maxOrderAmount
1	1	John	Doe	1600.00
2	2	Jane	Smith	1800.00
3	3	Robert	Johnson	1800.00
4	4	Sarah	Brown	600.00
5	5	David	Lee	1200.00
6	6	Laura	Hall	240.00

**-- 14) Get Customers Who Placed Orders Totaling Over \$1000.**

```
SELECT c.customerID, c.firstName, c.lastName, SUM(o.totalAmount) AS totalAmount
FROM customer c
JOIN orders o ON c.customerID = o.customerID
GROUP BY c.customerID, c.firstName, c.lastName
HAVING SUM(o.totalAmount) > 1000;
```

```

SELECT c.customerID, c.firstName, c.lastName, SUM(o.totalAmount) AS totalAmount
FROM customer c
JOIN orders o ON c.customerID = o.customerID
GROUP BY c.customerID, c.firstName, c.lastName
HAVING SUM(o.totalAmount) > 1000;

```

90 %

Results Messages

	customerID	firstName	lastName	totalAmount
1	1	John	Doe	1200.00
2	5	David	Lee	1800.00
3	10	Olivia	Adams	1400.00

-- 15) Subquery to Find Products Not in the Cart.

SELECT \*

FROM product

WHERE productID NOT IN (

SELECT DISTINCT productID

FROM cart );

```

SELECT * FROM product
WHERE productID NOT IN (
SELECT DISTINCT productID
FROM cart );

```

90 %

Results Messages

	productID	name	Description	price	stockQuantity
1	4	Headphones	Noise-canceling	150.00	30
2	5	TV	4K Smart TV	900.00	5
3	7	Refrigerator	Energy-efficient	800.00	10
4	8	Microwave Oven	Countertop microwave	80.00	15

-- 16) Subquery to Find Customers Who Haven't Placed Orders.

SELECT \*

FROM customer

WHERE customerID NOT IN (

SELECT DISTINCT customerID

FROM orders );

```

SELECT * FROM customer
WHERE customerID NOT IN (
SELECT DISTINCT customerID
FROM orders );

```

90 %

Results Messages

customerID	firstName	lastName	Email	address
------------	-----------	----------	-------	---------

-- 17) Subquery to Calculate the Percentage of Total Revenue for a Product.

```
SELECT p.productID, p.name, (SUM(oi.itemAmount) / (SELECT SUM(itemAmount) FROM orderitems)) *  
100 AS revenuePercentage
```

```
FROM orderitems oi
```

```
JOIN product p ON oi.productID = p.productID
```

```
GROUP BY p.productID, p.name;
```

```
SELECT p.productID, p.name, (SUM(oi.itemAmount) / (SELECT SUM(itemAmount) FROM orderitems)) * 100 AS revenuePercentage  
FROM orderitems oi  
JOIN product p ON oi.productID = p.productID  
GROUP BY p.productID, p.name;
```

90 %

	productID	name	revenuePercentage
1	1	Laptop	27.906900
2	2	Smartphone	34.883700
3	3	Tablet	3.488300
4	4	Headphones	6.976700
5	5	TV	20.930200
6	6	Coffee Maker	0.581300
7	9	Blender	2.441800
8	10	Vacuum Cleaner	2.790600

-- 18) Subquery to Find Products with Low Stock.

```
SELECT * FROM product
```

```
WHERE stockQuantity < ( SELECT AVG(stockQuantity) * 0.2 FROM product );
```

```
SELECT * FROM product  
WHERE stockQuantity < ( SELECT AVG(stockQuantity) * 0.2 FROM product );
```

10 %

	productID	name	Description	price	stockQuantity
1	5	TV	4K Smart TV	900.00	5

-- 19) Subquery to Find Customers Who Placed High-Value Orders.

```
SELECT customerID, firstName, lastName FROM customer
```

```
WHERE customerID IN (SELECT customerID FROM orders
```

```
WHERE totalAmount > 1000);
```

```
SELECT * FROM customer  
WHERE customerID IN ( SELECT o.customerID FROM orders o  
JOIN ( SELECT orderID, SUM(itemAmount) AS totalAmount FROM orderitems  
GROUP BY orderID ) AS order_total ON o.orderID = order_total.orderID  
WHERE order_total.totalAmount > 1000 );
```

90 %

	customerID	firstName	lastName	Email	address
1	1	John	Doe	johndoe@example.com	123 Main St, City
2	2	Jane	Smith	janesmith@example.com	456 Elm St, Town
3	3	Robert	Johnson	robert@example.com	789 Oak St, Village
4	5	David	Lee	david@example.com	234 Cedar St, District