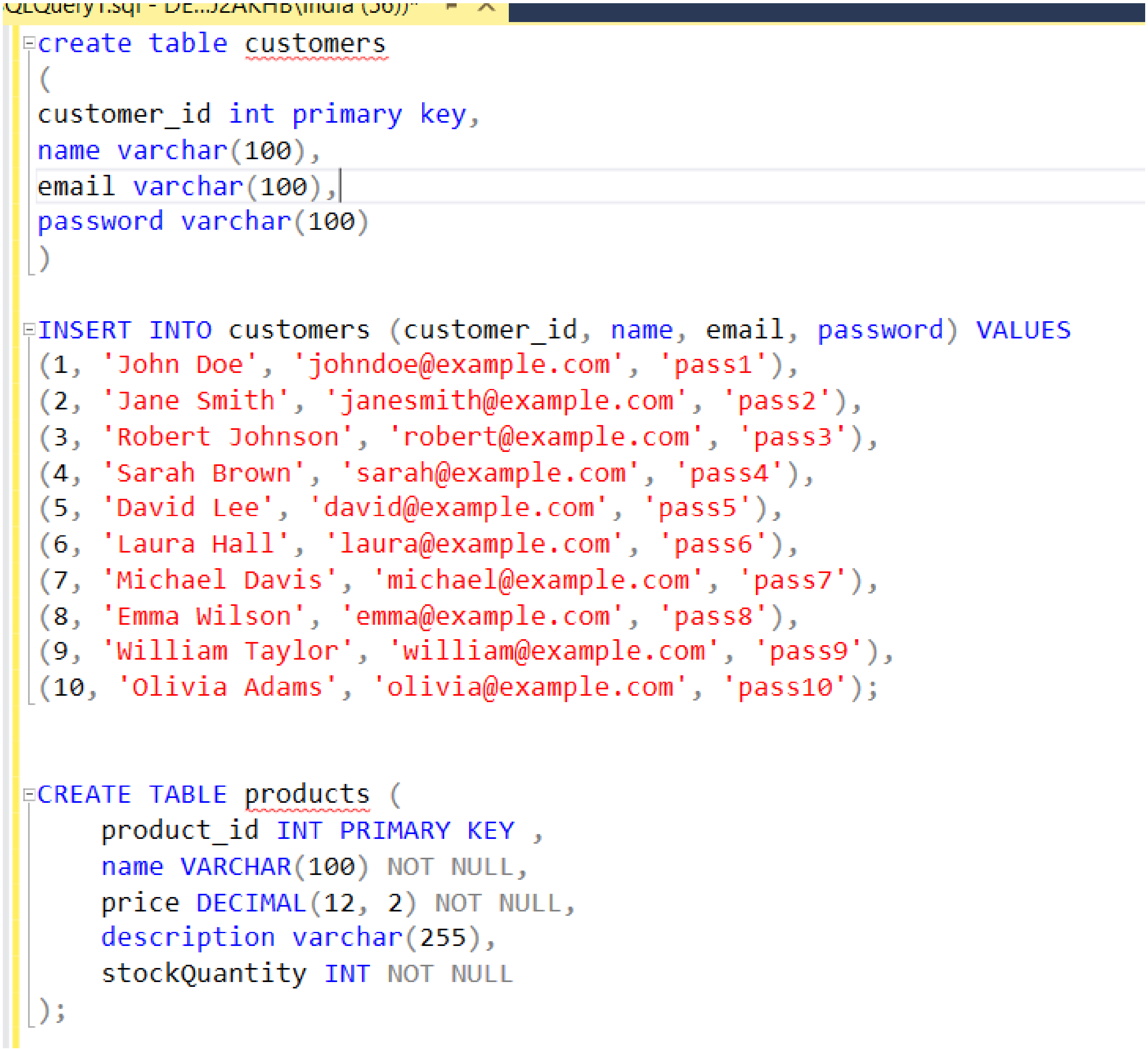
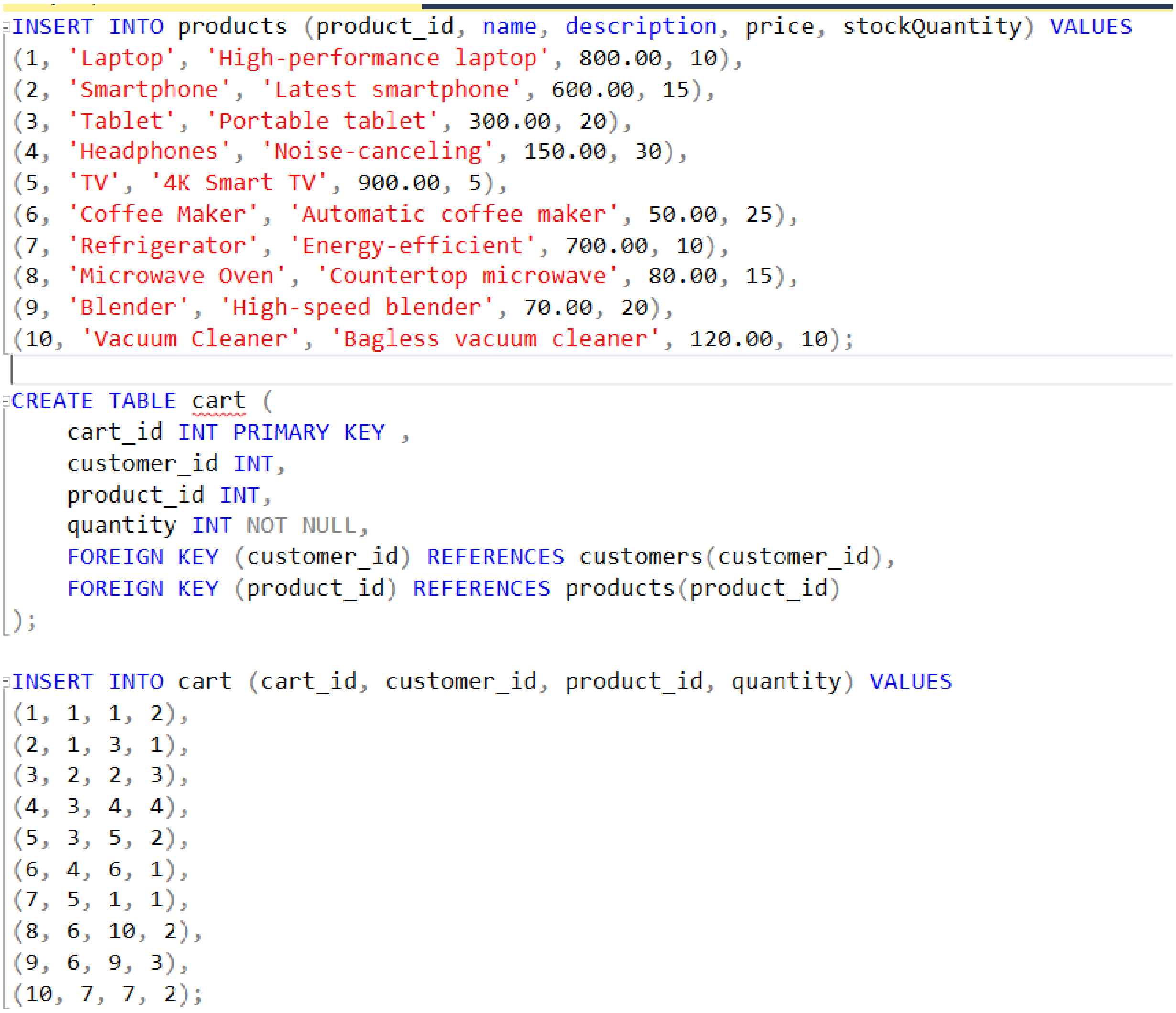
SQL Coding Challenge (EComm)

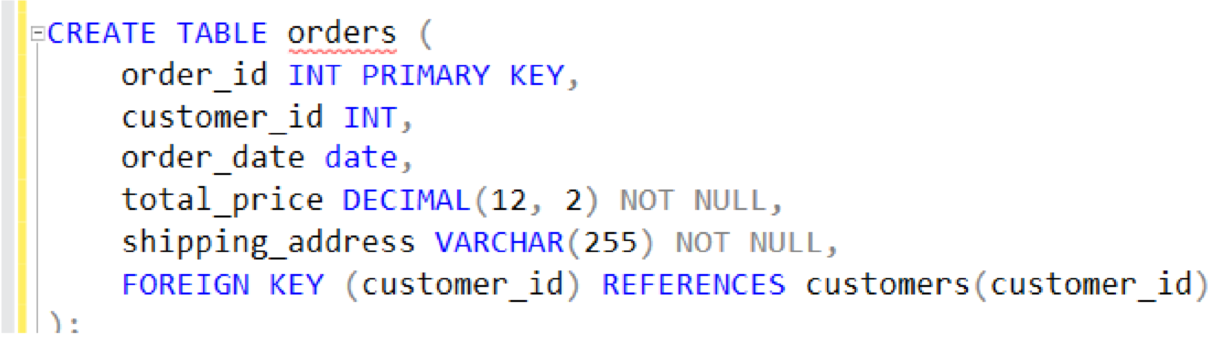
Chirag Bhatia



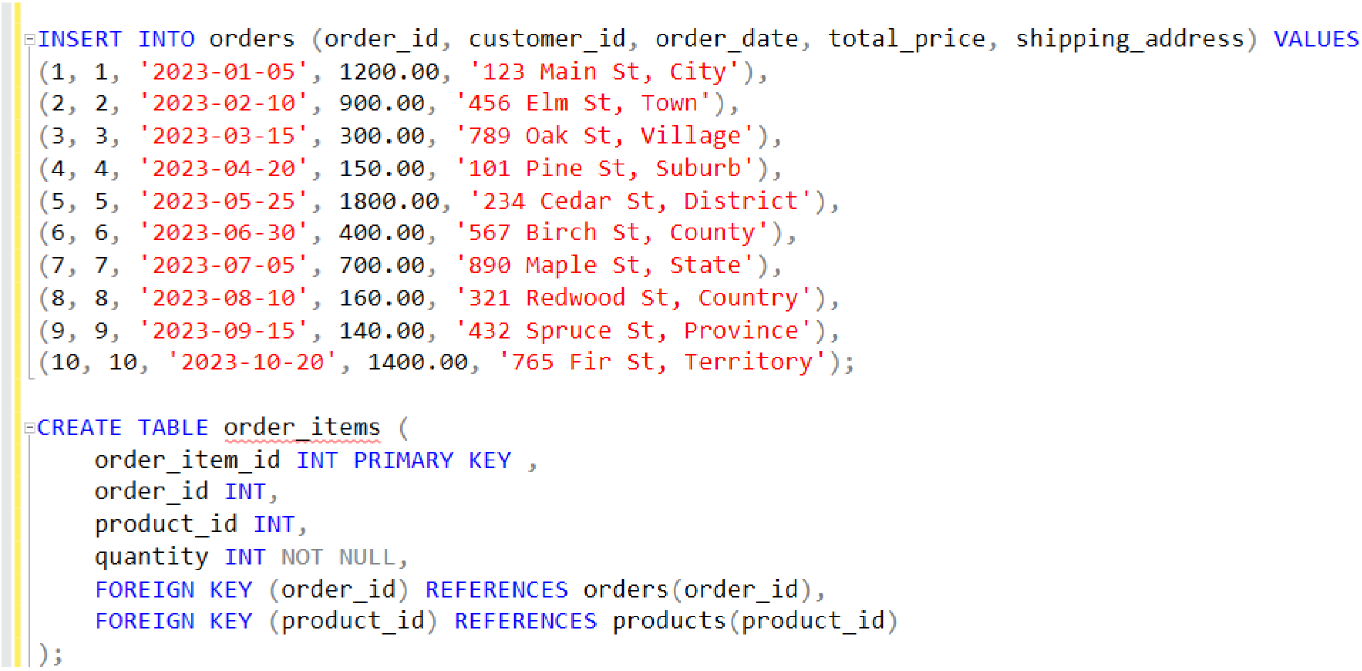
```
create table customers
(
customer_id int primary key,
name varchar(100),
email varchar(100),
password varchar(100)
)

INSERT INTO customers (customer_id, name, email, password) VALUES
(1, 'John Doe', 'johndoe@example.com', 'pass1'),
(2, 'Jane Smith', 'janesmith@example.com', 'pass2'),
(3, 'Robert Johnson', 'robert@example.com', 'pass3'),
(4, 'Sarah Brown', 'sarah@example.com', 'pass4'),
(5, 'David Lee', 'david@example.com', 'pass5'),
(6, 'Laura Hall', 'laura@example.com', 'pass6'),
(7, 'Michael Davis', 'michael@example.com', 'pass7'),
(8, 'Emma Wilson', 'emma@example.com', 'pass8'),
(9, 'William Taylor', 'william@example.com', 'pass9'),
(10, 'Olivia Adams', 'olivia@example.com', 'pass10');


CREATE TABLE products (
    product_id INT PRIMARY KEY ,
    name VARCHAR(100) NOT NULL,
    price DECIMAL(12, 2) NOT NULL,
    description varchar(255),
    stockQuantity INT NOT NULL
);
```



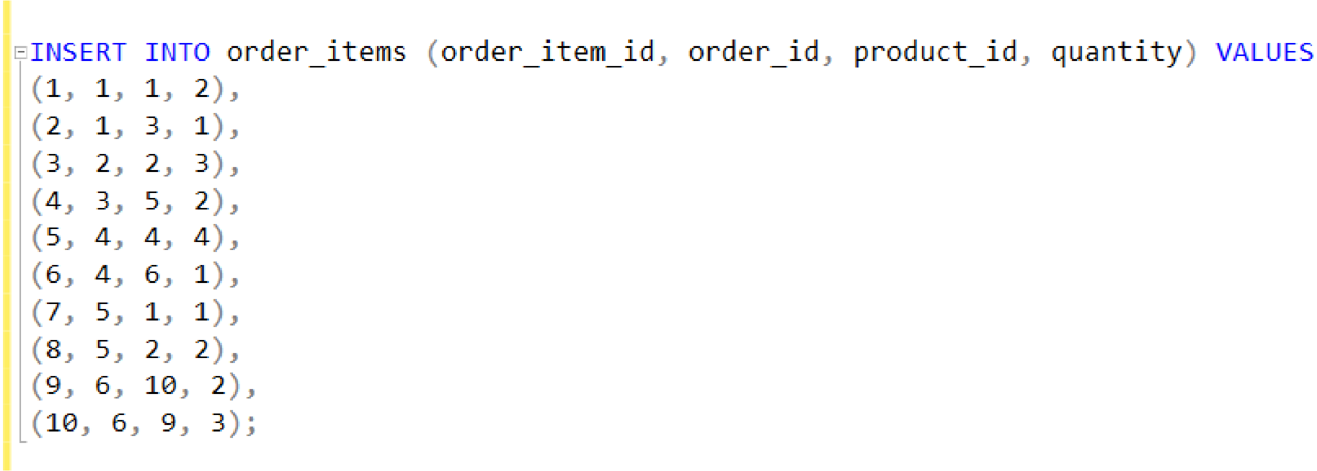
```sql
INSERT INTO products (product_id, name, description, price, stockQuantity) VALUES
(1, 'Laptop', 'High-performance laptop', 800.00, 10),
(2, 'Smartphone', 'Latest smartphone', 600.00, 15),
(3, 'Tablet', 'Portable tablet', 300.00, 20),
(4, 'Headphones', 'Noise-canceling', 150.00, 30),
(5, 'TV', '4K Smart TV', 900.00, 5),
(6, 'Coffee Maker', 'Automatic coffee maker', 50.00, 25),
(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);

CREATE TABLE cart (
    cart_id INT PRIMARY KEY ,
    customer_id INT,
    product_id INT,
    quantity INT NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);

INSERT INTO cart (cart_id, customer_id, product_id, 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, 7, 2);
```



```sql
CREATE TABLE orders (
    order_id INT PRIMARY KEY,
    customer_id INT,
    order_date date,
    total_price DECIMAL(12, 2) NOT NULL,
    shipping_address VARCHAR(255) NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);
```

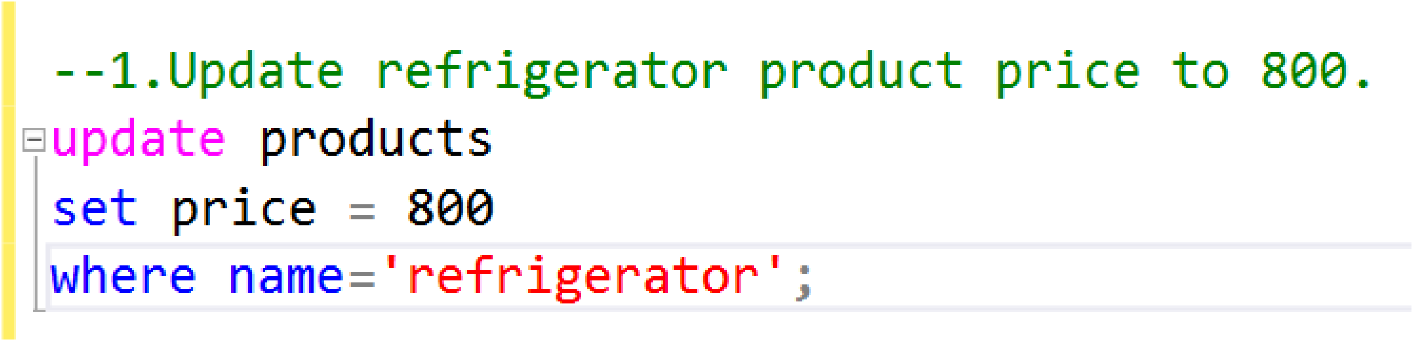


```sql
INSERT INTO orders (order_id, customer_id, order_date, total_price, shipping_address) VALUES
(1, 1, '2023-01-05', 1200.00, '123 Main St, City'),
(2, 2, '2023-02-10', 900.00, '456 Elm St, Town'),
(3, 3, '2023-03-15', 300.00, '789 Oak St, Village'),
(4, 4, '2023-04-20', 150.00, '101 Pine St, Suburb'),
(5, 5, '2023-05-25', 1800.00, '234 Cedar St, District'),
(6, 6, '2023-06-30', 400.00, '567 Birch St, County'),
(7, 7, '2023-07-05', 700.00, '890 Maple St, State'),
(8, 8, '2023-08-10', 160.00, '321 Redwood St, Country'),
(9, 9, '2023-09-15', 140.00, '432 Spruce St, Province'),
(10, 10, '2023-10-20', 1400.00, '765 Fir St, Territory');

CREATE TABLE order_items (
    order_item_id INT PRIMARY KEY ,
    order_id INT,
    product_id INT,
    quantity INT NOT NULL,
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);
```



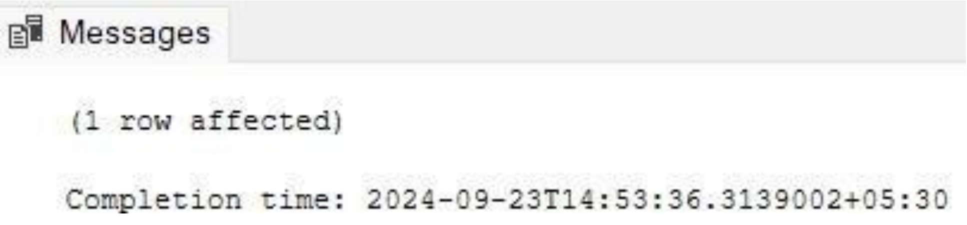
```sql
INSERT INTO order_items (order_item_id, order_id, product_id, quantity) VALUES
(1, 1, 1, 2),
(2, 1, 3, 1),
(3, 2, 2, 3),
(4, 3, 5, 2),
(5, 4, 4, 4),
(6, 4, 6, 1),
(7, 5, 1, 1),
(8, 5, 2, 2),
(9, 6, 10, 2),
(10, 6, 9, 3);
```

1.



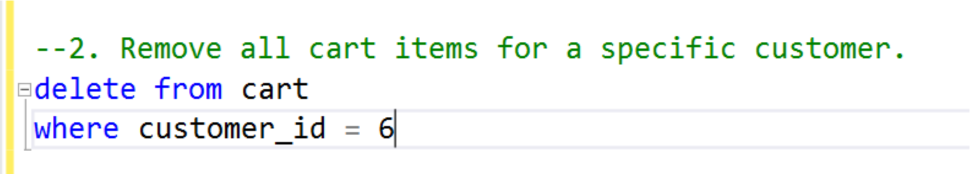
```sql
--1.Update refrigerator product price to 800.
update products
set price = 800
where name='refrigerator';
```

OUTPUT



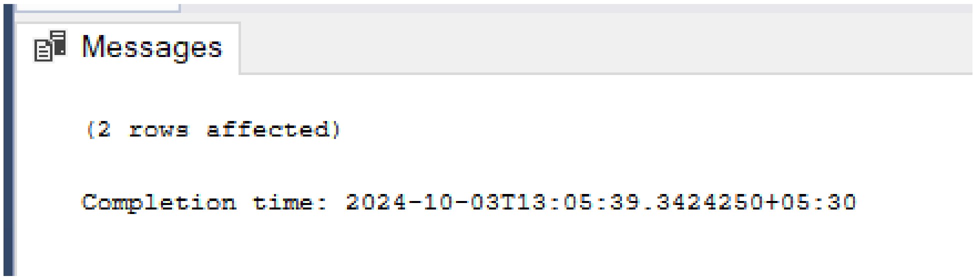
```
(1 row affected)

Completion time: 2024-09-23T14:53:36.3139002+05:30
```

2.



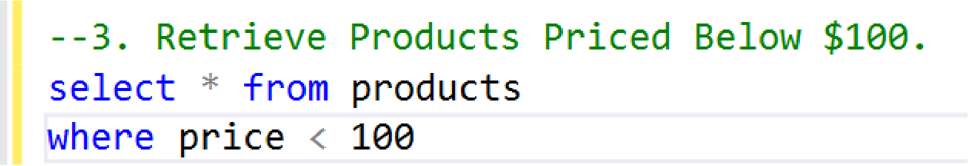
```
--2. Remove all cart items for a specific customer.
delete from cart
where customer_id = 6
```

OUTPUT



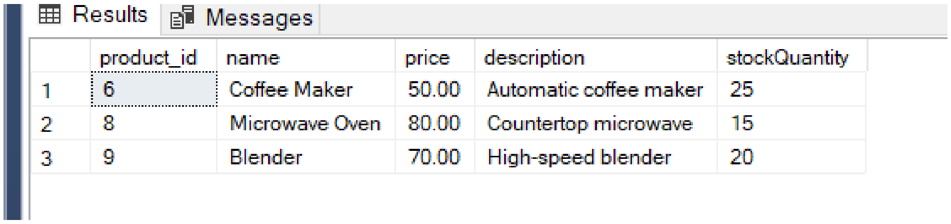
```
(2 rows affected)

Completion time: 2024-10-03T13:05:39.3424250+05:30
```

3.



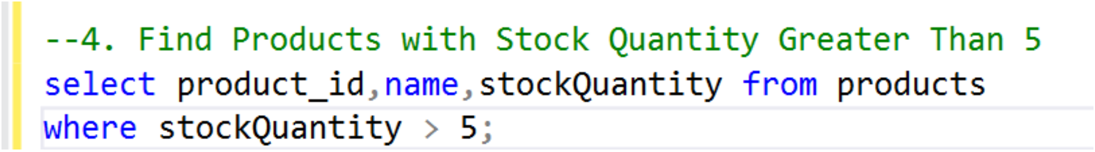
```
--3. Retrieve Products Priced Below $100.
select * from products
where price < 100
```

OUTPUT



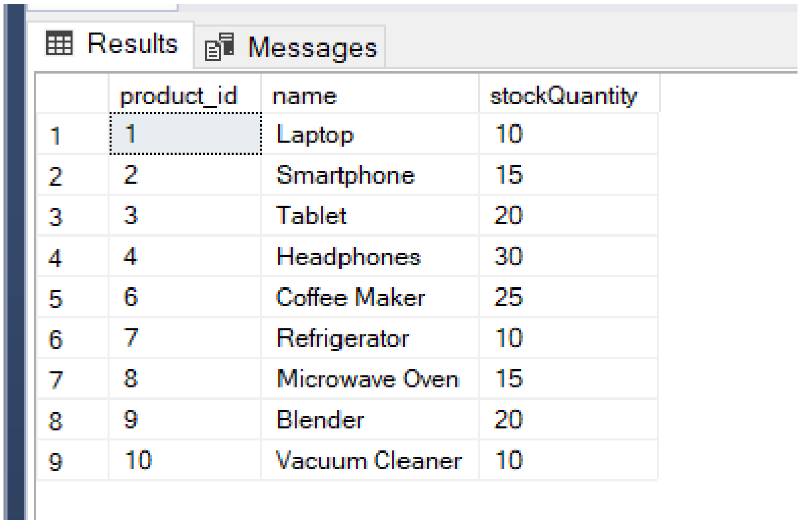
	product_id	name	price	description	stockQuantity
1	6	Coffee Maker	50.00	Automatic coffee maker	25
2	8	Microwave Oven	80.00	Countertop microwave	15
3	9	Blender	70.00	High-speed blender	20

4.



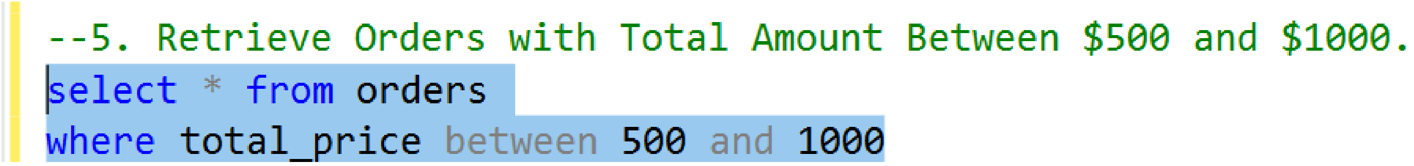
```
--4. Find Products with Stock Quantity Greater Than 5
select product_id,name,stockQuantity from products
where stockQuantity > 5;
```

OUTPUT



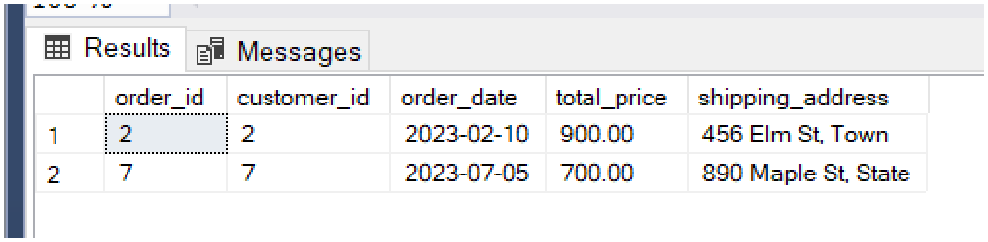
	product_id	name	stockQuantity
1	1	Laptop	10
2	2	Smartphone	15
3	3	Tablet	20
4	4	Headphones	30
5	6	Coffee Maker	25
6	7	Refrigerator	10
7	8	Microwave Oven	15
8	9	Blender	20
9	10	Vacuum Cleaner	10

5.



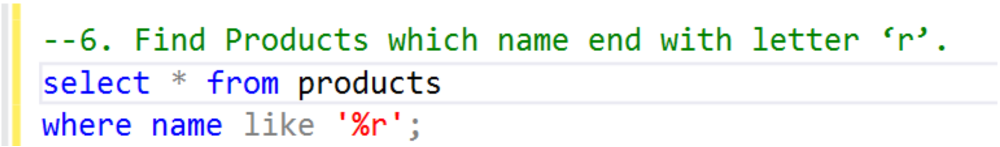
```
--5. Retrieve Orders with Total Amount Between $500 and $1000.
select * from orders
where total_price between 500 and 1000
```

OUTPUT



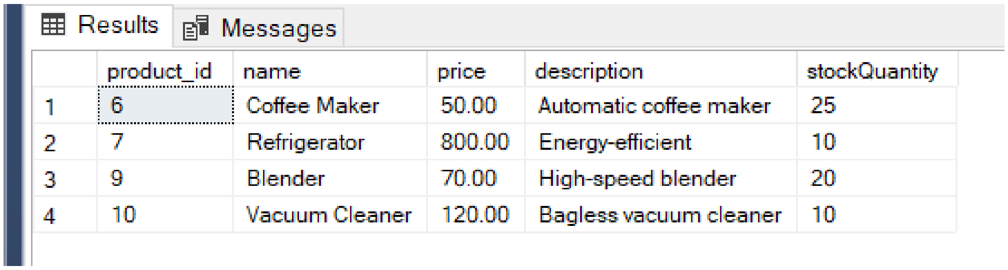
	order_id	customer_id	order_date	total_price	shipping_address
1	2	2	2023-02-10	900.00	456 Elm St, Town
2	7	7	2023-07-05	700.00	890 Maple St, State

6.



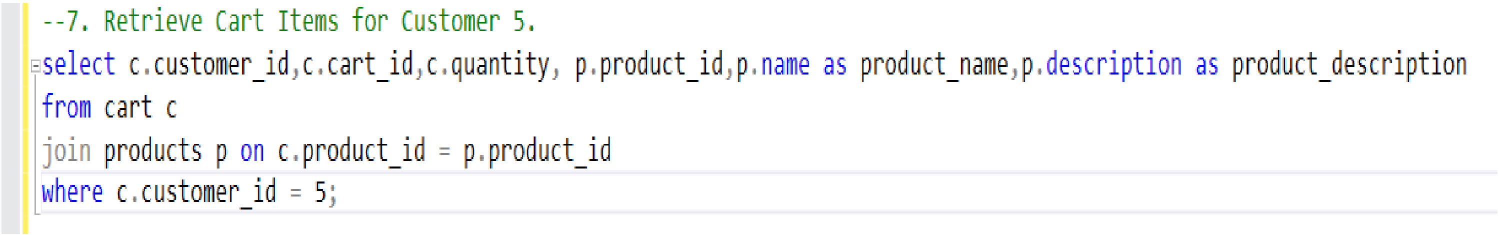
```sql
--6. Find Products which name end with letter ‘r’.
select * from products
where name like '%r';
```

OUTPUT



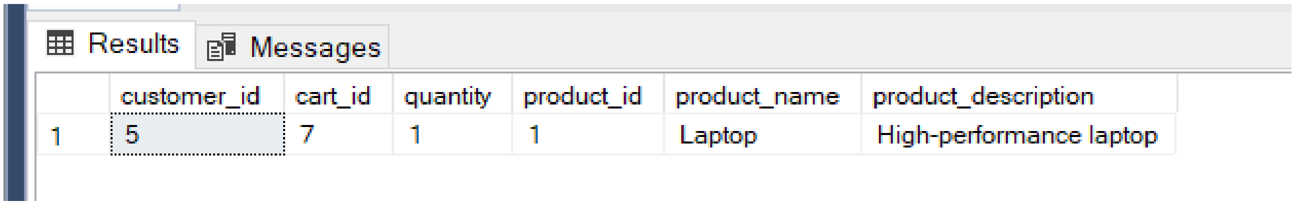
	product_id	name	price	description	stockQuantity
1	6	Coffee Maker	50.00	Automatic coffee maker	25
2	7	Refrigerator	800.00	Energy-efficient	10
3	9	Blender	70.00	High-speed blender	20
4	10	Vacuum Cleaner	120.00	Bagless vacuum cleaner	10

7.



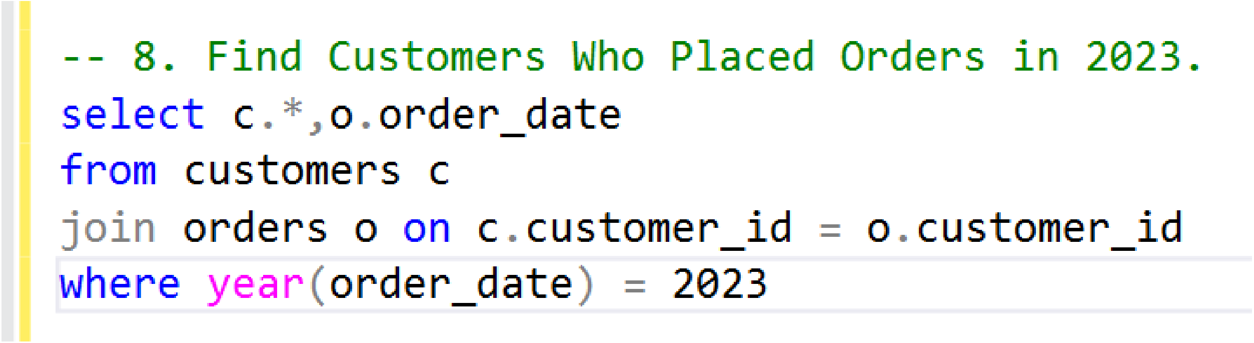
```sql
--7. Retrieve Cart Items for Customer 5.
select c.customer_id,c.cart_id,c.quantity, p.product_id,p.name as product_name,p.description as product_description
from cart c
join products p on c.product_id = p.product_id
where c.customer_id = 5;
```

OUTPUT



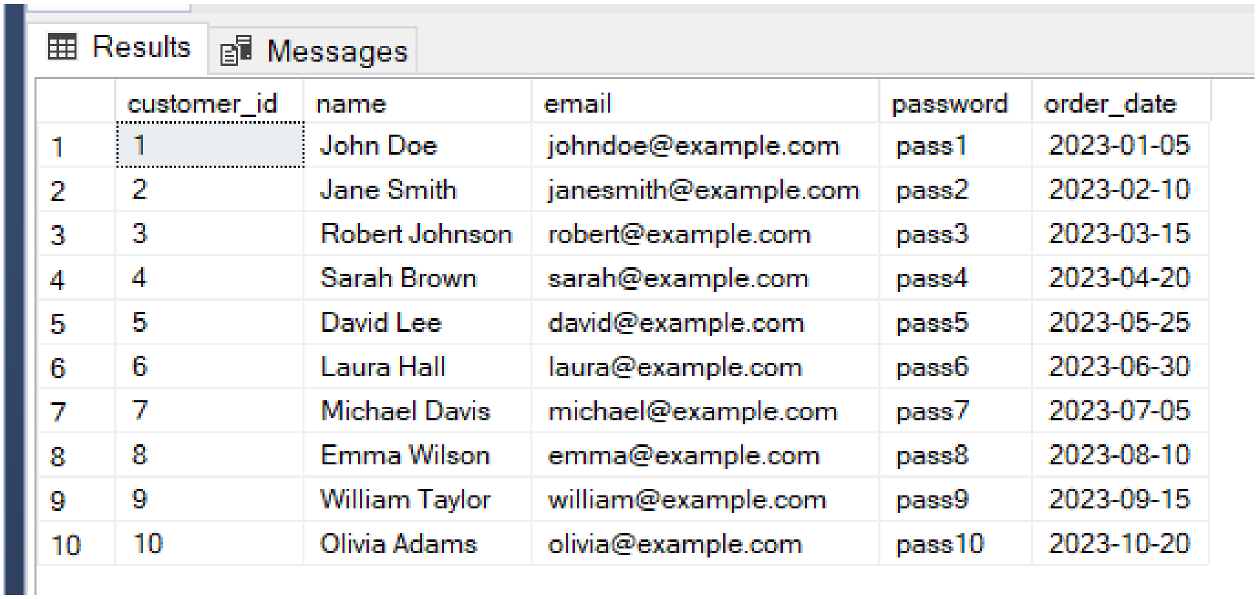
	customer_id	cart_id	quantity	product_id	product_name	product_description
1	5	7	1	1	Laptop	High-performance laptop

8.



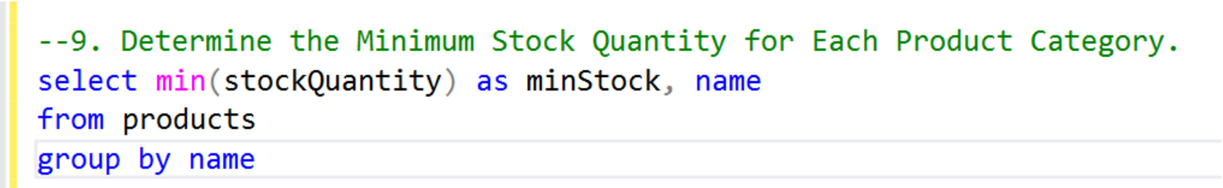
```sql
-- 8. Find Customers Who Placed Orders in 2023.
select c.*,o.order_date
from customers c
join orders o on c.customer_id = o.customer_id
where year(order_date) = 2023
```

OUTPUT



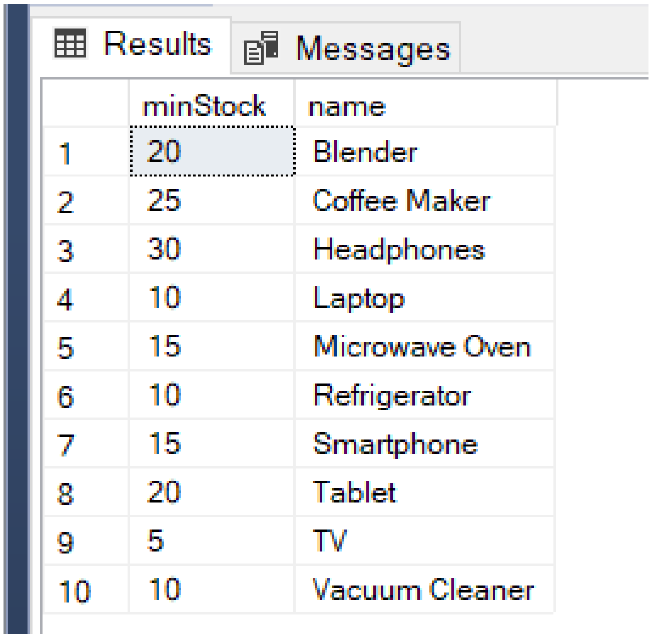
	customer_id	name	email	password	order_date
1	1	John Doe	johndoe@example.com	pass1	2023-01-05
2	2	Jane Smith	janesmith@example.com	pass2	2023-02-10
3	3	Robert Johnson	robert@example.com	pass3	2023-03-15
4	4	Sarah Brown	sarah@example.com	pass4	2023-04-20
5	5	David Lee	david@example.com	pass5	2023-05-25
6	6	Laura Hall	laura@example.com	pass6	2023-06-30
7	7	Michael Davis	michael@example.com	pass7	2023-07-05
8	8	Emma Wilson	emma@example.com	pass8	2023-08-10
9	9	William Taylor	william@example.com	pass9	2023-09-15
10	10	Olivia Adams	olivia@example.com	pass10	2023-10-20

9.



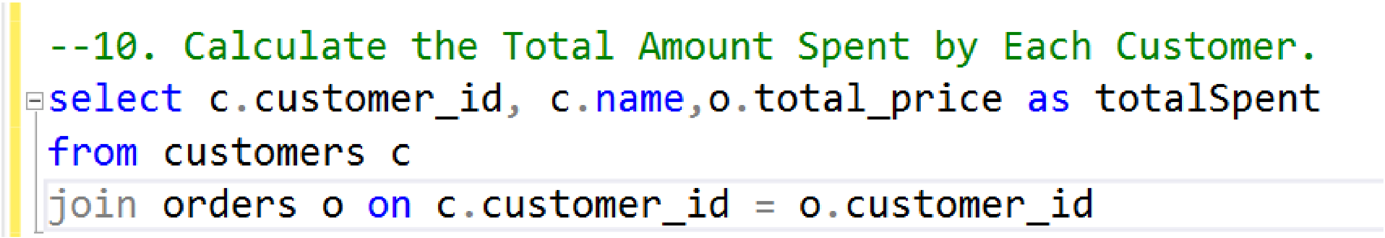
```
--9. Determine the Minimum Stock Quantity for Each Product Category.
select min(stockQuantity) as minStock, name
from products
group by name
```

OUTPUT



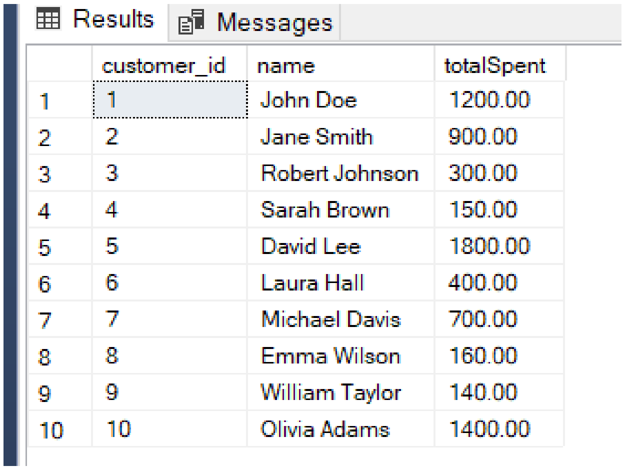
	minStock	name
1	20	Blender
2	25	Coffee Maker
3	30	Headphones
4	10	Laptop
5	15	Microwave Oven
6	10	Refrigerator
7	15	Smartphone
8	20	Tablet
9	5	TV
10	10	Vacuum Cleaner

10.



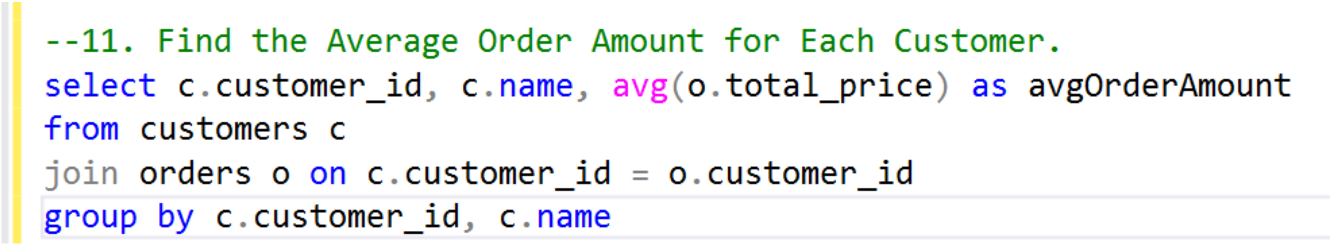
```
--10. Calculate the Total Amount Spent by Each Customer.
select c.customer_id, c.name,o.total_price as totalSpent
from customers c
join orders o on c.customer_id = o.customer_id
```

OUTPUT



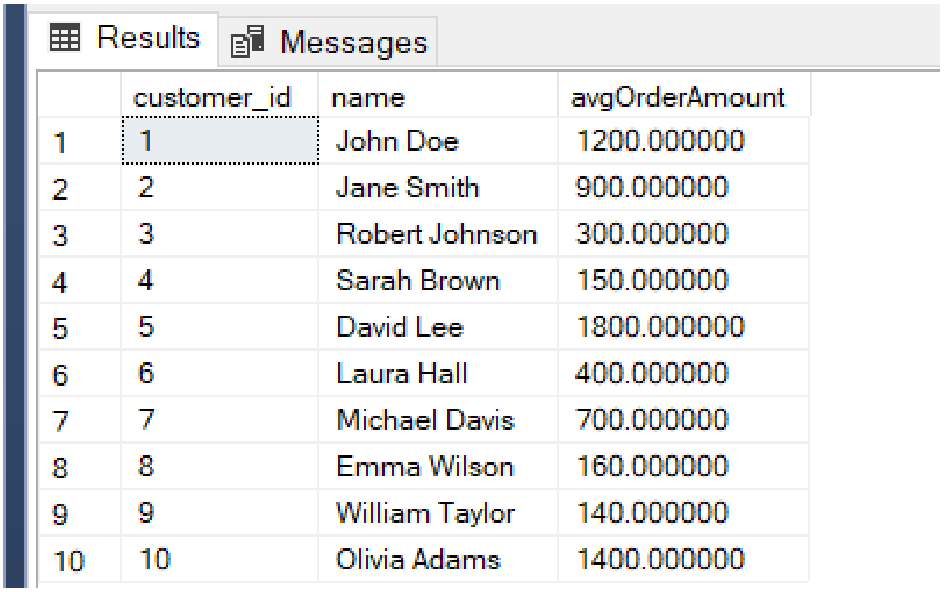
	customer_id	name	totalSpent
1	1	John Doe	1200.00
2	2	Jane Smith	900.00
3	3	Robert Johnson	300.00
4	4	Sarah Brown	150.00
5	5	David Lee	1800.00
6	6	Laura Hall	400.00
7	7	Michael Davis	700.00
8	8	Emma Wilson	160.00
9	9	William Taylor	140.00
10	10	Olivia Adams	1400.00

11.



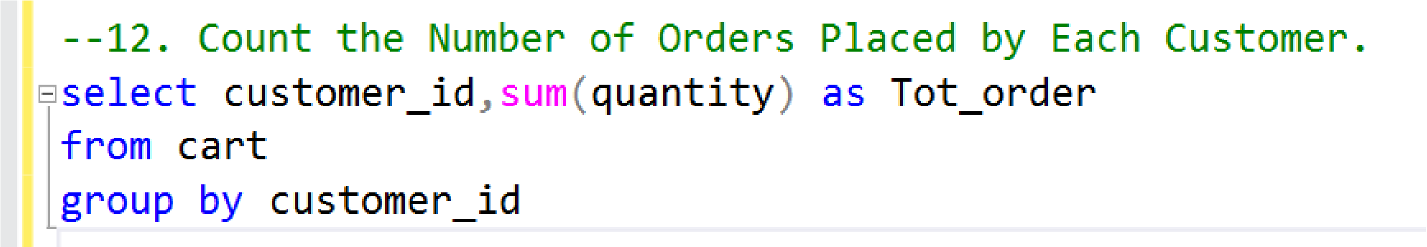
```
--11. Find the Average Order Amount for Each Customer.
select c.customer_id, c.name, avg(o.total_price) as avgOrderAmount
from customers c
join orders o on c.customer_id = o.customer_id
group by c.customer_id, c.name
```

OUTPUT



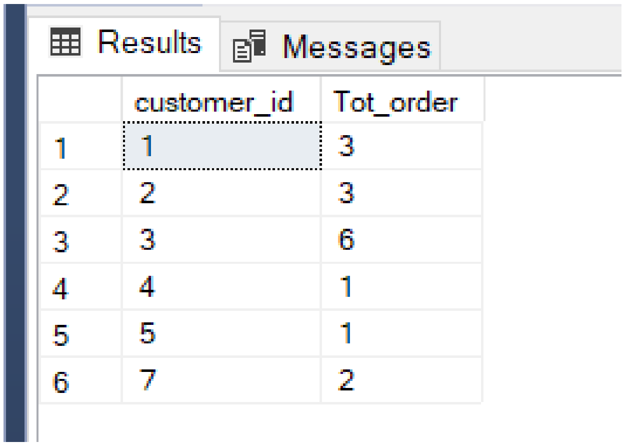
	customer_id	name	avgOrderAmount
1	1	John Doe	1200.000000
2	2	Jane Smith	900.000000
3	3	Robert Johnson	300.000000
4	4	Sarah Brown	150.000000
5	5	David Lee	1800.000000
6	6	Laura Hall	400.000000
7	7	Michael Davis	700.000000
8	8	Emma Wilson	160.000000
9	9	William Taylor	140.000000
10	10	Olivia Adams	1400.000000

12.



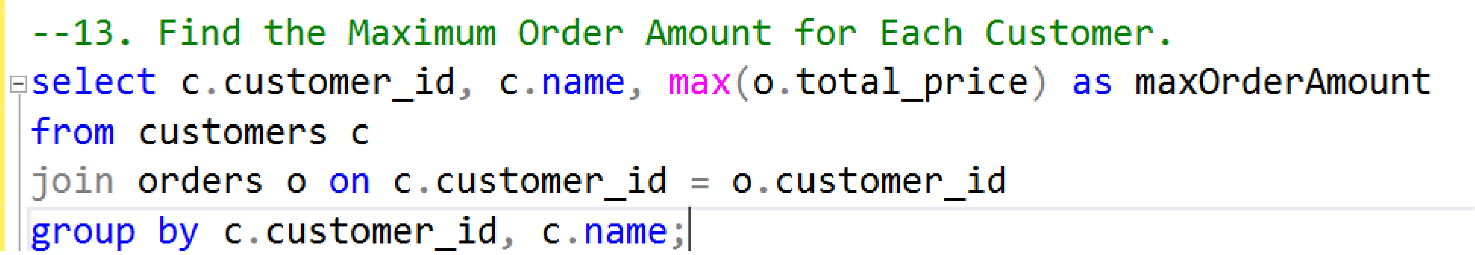
```
--12. Count the Number of Orders Placed by Each Customer.
select customer_id,sum(quantity) as Tot_order
from cart
group by customer_id
```

OUTPUT



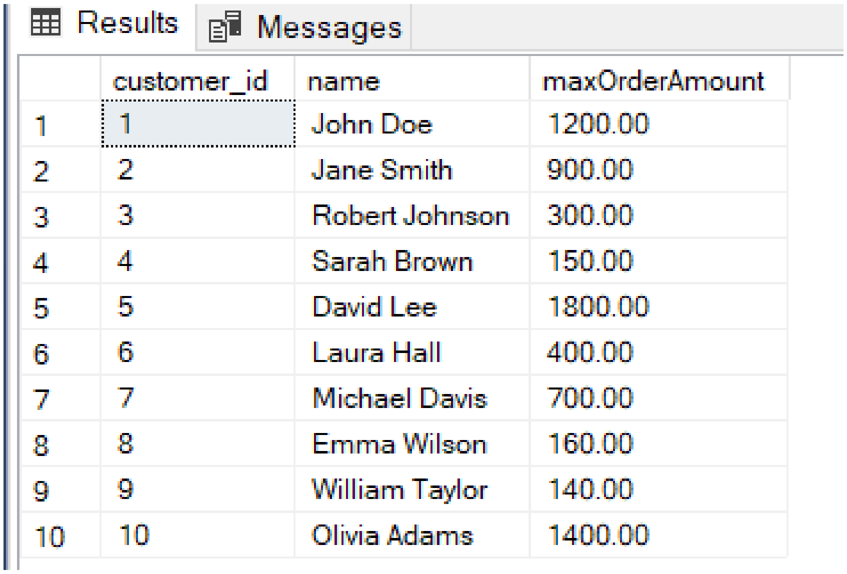
	customer_id	Tot_order
1	1	3
2	2	3
3	3	6
4	4	1
5	5	1
6	7	2

13.



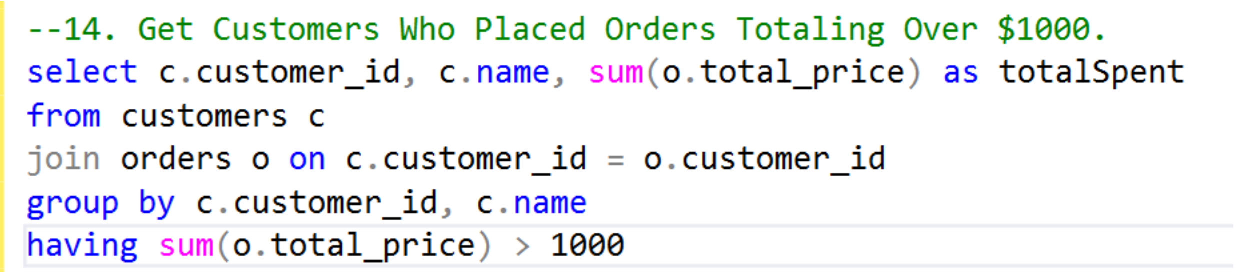
```sql
--13. Find the Maximum Order Amount for Each Customer.
select c.customer_id, c.name, max(o.total_price) as maxOrderAmount
from customers c
join orders o on c.customer_id = o.customer_id
group by c.customer_id, c.name;
```

OUTPUT



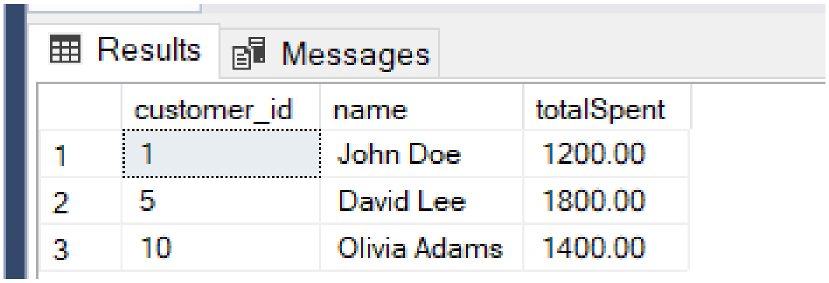
	customer_id	name	maxOrderAmount
1	1	John Doe	1200.00
2	2	Jane Smith	900.00
3	3	Robert Johnson	300.00
4	4	Sarah Brown	150.00
5	5	David Lee	1800.00
6	6	Laura Hall	400.00
7	7	Michael Davis	700.00
8	8	Emma Wilson	160.00
9	9	William Taylor	140.00
10	10	Olivia Adams	1400.00

14.



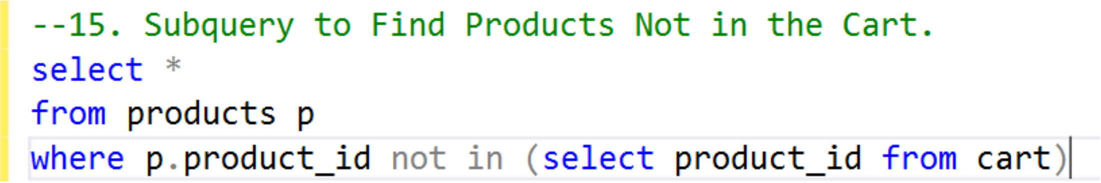
```sql
--14. Get Customers Who Placed Orders Totaling Over $1000.
select c.customer_id, c.name, sum(o.total_price) as totalSpent
from customers c
join orders o on c.customer_id = o.customer_id
group by c.customer_id, c.name
having sum(o.total_price) > 1000
```

OUTPUT



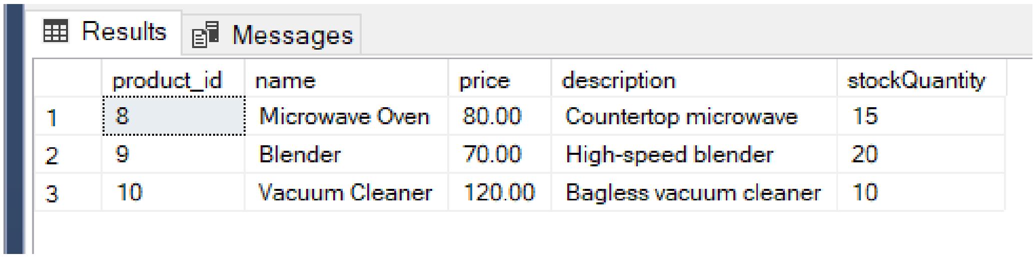
	customer_id	name	totalSpent
1	1	John Doe	1200.00
2	5	David Lee	1800.00
3	10	Olivia Adams	1400.00

15.



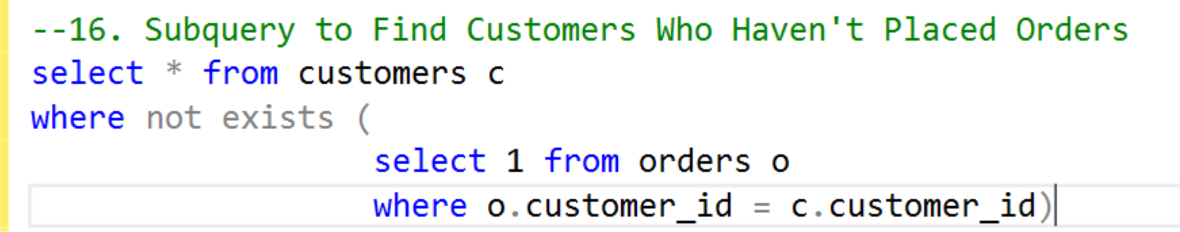
```
--15. Subquery to Find Products Not in the Cart.
select *
from products p
where p.product_id not in (select product_id from cart)
```

OUTPUT



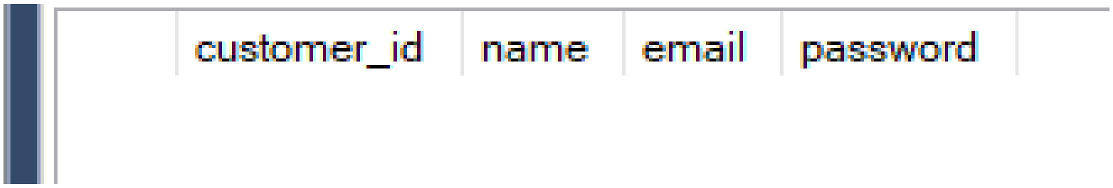
	product_id	name	price	description	stockQuantity
1	8	Microwave Oven	80.00	Countertop microwave	15
2	9	Blender	70.00	High-speed blender	20
3	10	Vacuum Cleaner	120.00	Bagless vacuum cleaner	10

16.



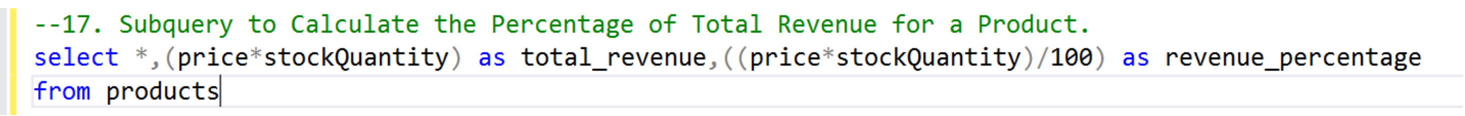
```
--16. Subquery to Find Customers Who Haven't Placed Orders
select * from customers c
where not exists (
                  select 1 from orders o
                  where o.customer_id = c.customer_id)
```

OUTPUT



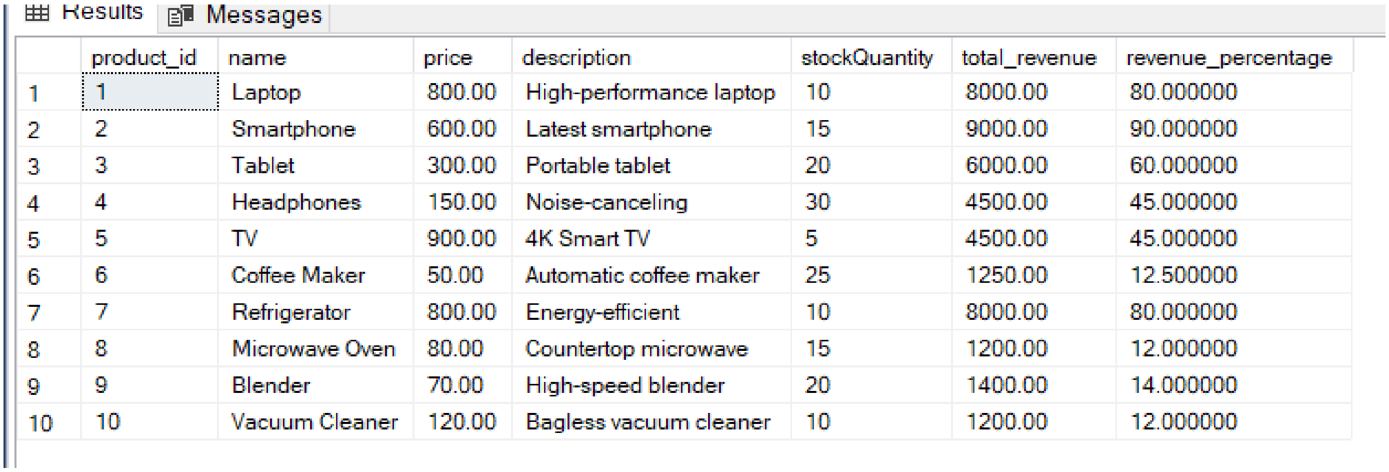
	customer_id	name	email	password

17.



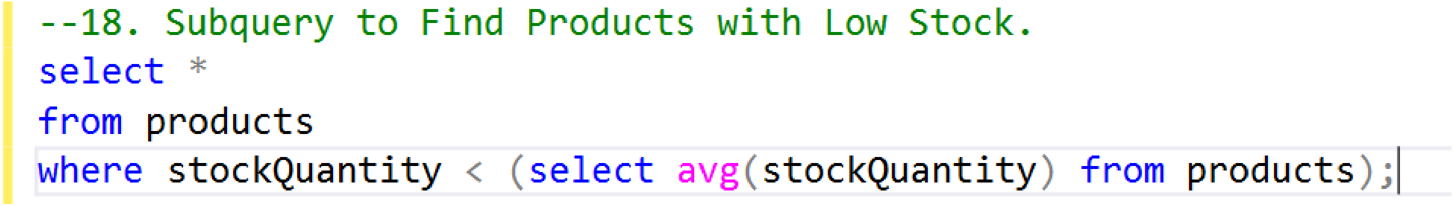
```sql
--17. Subquery to Calculate the Percentage of Total Revenue for a Product.
select *,(price*stockQuantity) as total_revenue,((price*stockQuantity)/100) as revenue_percentage
from products
```

OUTPUT



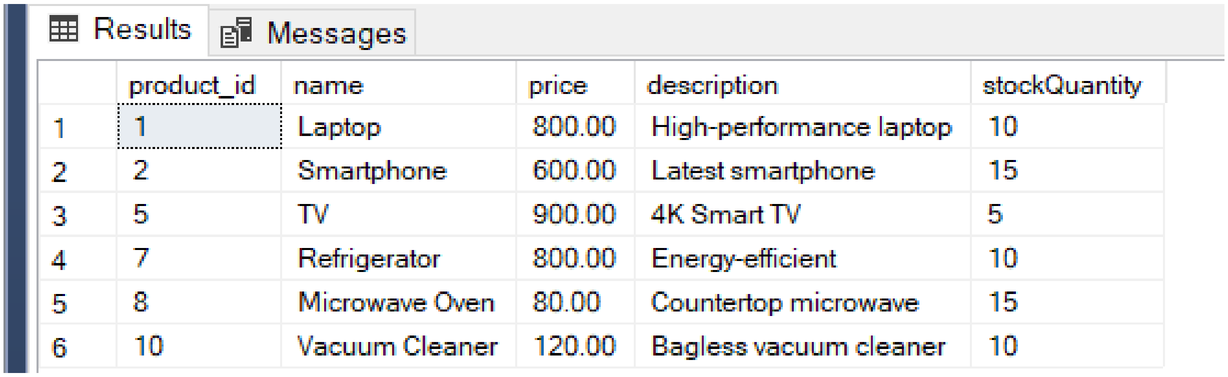
	product_id	name	price	description	stockQuantity	total_revenue	revenue_percentage
1	1	Laptop	800.00	High-performance laptop	10	8000.00	80.000000
2	2	Smartphone	600.00	Latest smartphone	15	9000.00	90.000000
3	3	Tablet	300.00	Portable tablet	20	6000.00	60.000000
4	4	Headphones	150.00	Noise-canceling	30	4500.00	45.000000
5	5	TV	900.00	4K Smart TV	5	4500.00	45.000000
6	6	Coffee Maker	50.00	Automatic coffee maker	25	1250.00	12.500000
7	7	Refrigerator	800.00	Energy-efficient	10	8000.00	80.000000
8	8	Microwave Oven	80.00	Countertop microwave	15	1200.00	12.000000
9	9	Blender	70.00	High-speed blender	20	1400.00	14.000000
10	10	Vacuum Cleaner	120.00	Bagless vacuum cleaner	10	1200.00	12.000000

18.



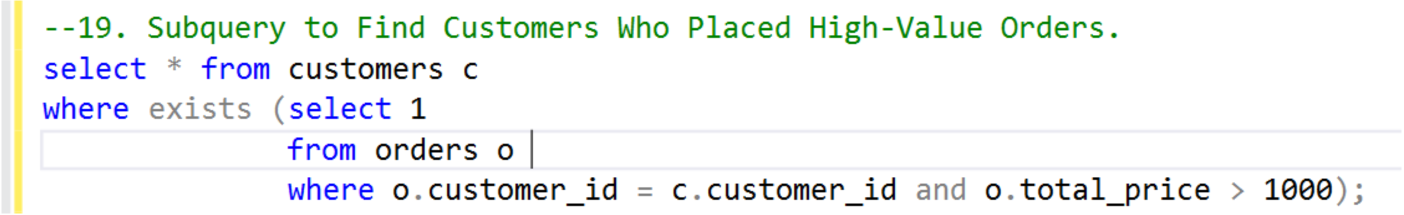
```sql
--18. Subquery to Find Products with Low Stock.
select *
from products
where stockQuantity < (select avg(stockQuantity) from products);
```

OUTPUT



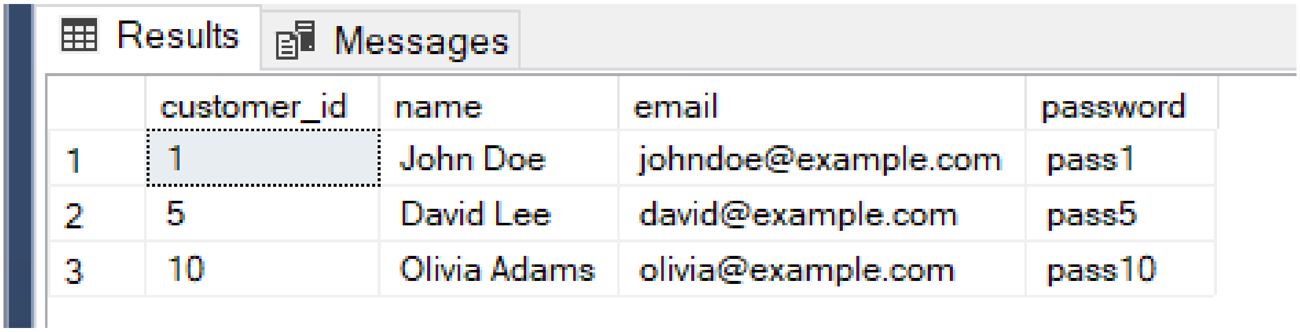
	product_id	name	price	description	stockQuantity
1	1	Laptop	800.00	High-performance laptop	10
2	2	Smartphone	600.00	Latest smartphone	15
3	5	TV	900.00	4K Smart TV	5
4	7	Refrigerator	800.00	Energy-efficient	10
5	8	Microwave Oven	80.00	Countertop microwave	15
6	10	Vacuum Cleaner	120.00	Bagless vacuum cleaner	10

19.



```
--19. Subquery to Find Customers Who Placed High-Value Orders.
select * from customers c
where exists (select 1
              from orders o
              where o.customer_id = c.customer_id and o.total_price > 1000);
```

OUTPUT



	customer_id	name	email	password
1	1	John Doe	johndoe@example.com	pass1
2	5	David Lee	david@example.com	pass5
3	10	Olivia Adams	olivia@example.com	pass10