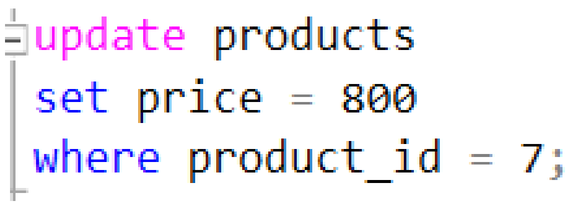
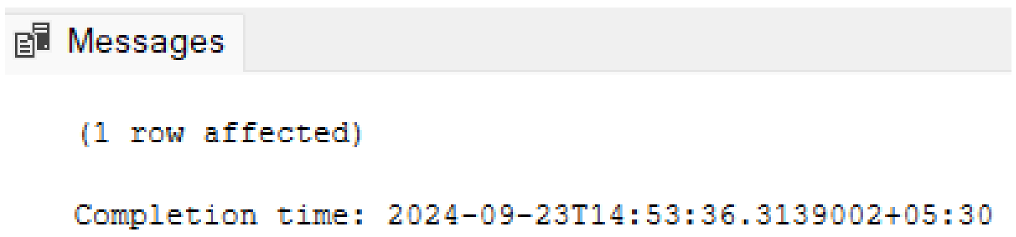
Coding Challenge
Ecommerce

By ~ **ANSHAY KHARE** *(anshaykhare27@gmail.com)*

Q1. Update refrigerator product price to 800.



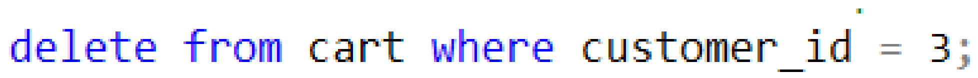
```sql
update products
set price = 800
where product_id = 7;
```



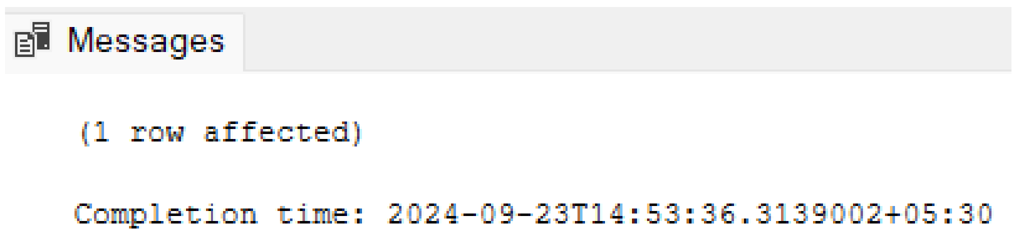
```
(1 row affected)

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

Q2. Remove all cart items for a specific customer.



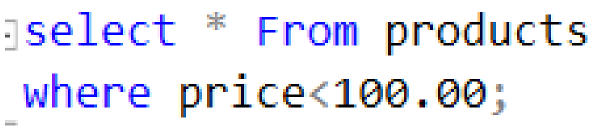
```sql
delete from cart where customer_id = 3;
```



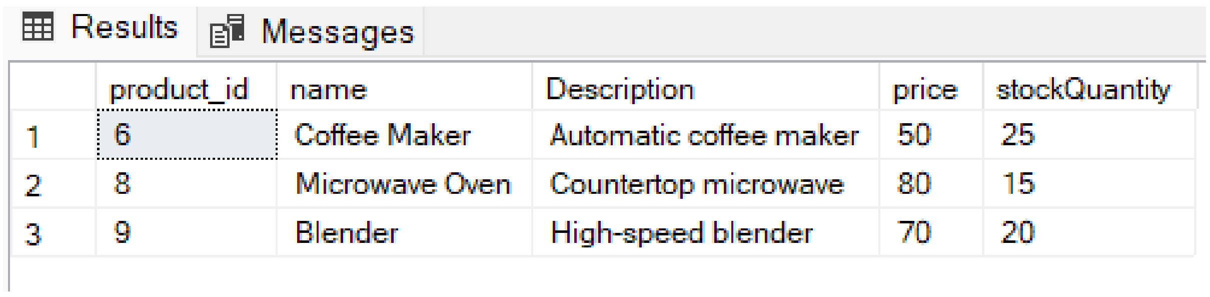
```
(1 row affected)

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

Q3. Retrieve Products Priced Below $100.



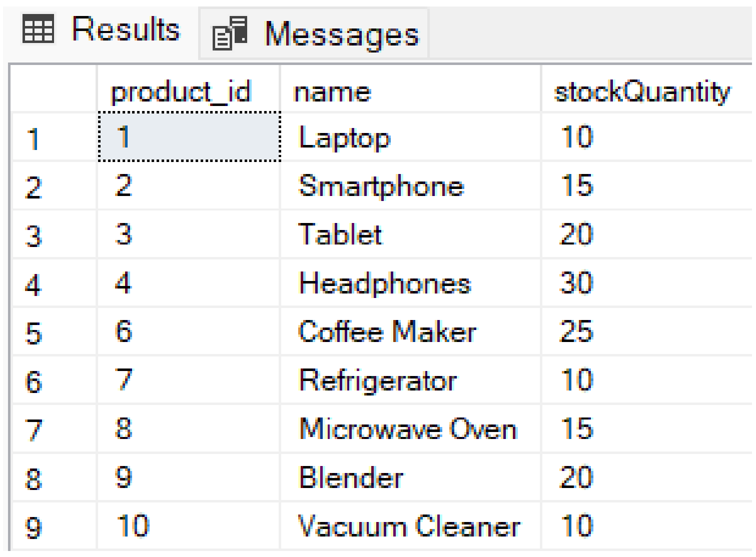
```sql
select * From products
where price<100.00;
```



	product_id	name	Description	price	stockQuantity
1	6	Coffee Maker	Automatic coffee maker	50	25
2	8	Microwave Oven	Countertop microwave	80	15
3	9	Blender	High-speed blender	70	20

Q4. Find Products with Stock Quantity Greater Than 5.

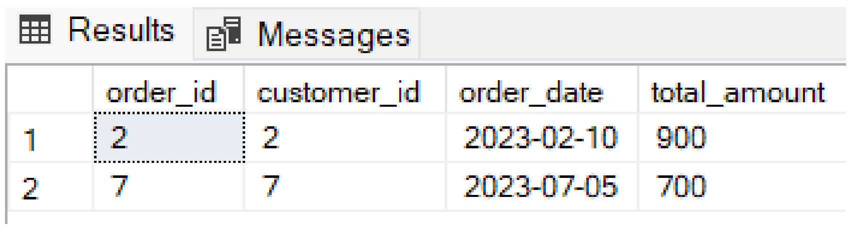
```
select product_id,name,stockQuantity From products
where stockQuantity>5;
```



	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

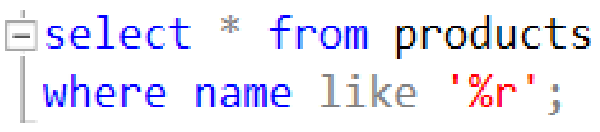
Q5. Retrieve Orders with Total Amount Between $500 and $1000.

```
select * from orders
where total_amount between 500 and 1000;
```

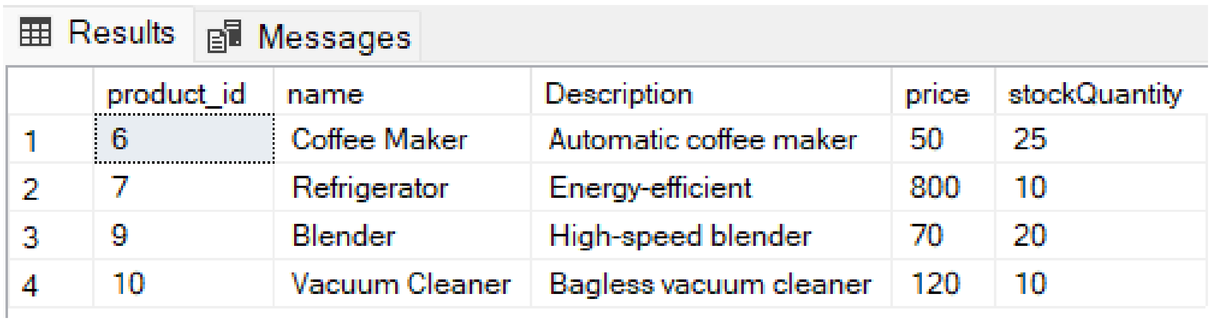


	order_id	customer_id	order_date	total_amount
1	2	2	2023-02-10	900
2	7	7	2023-07-05	700

Q6. Find Products which name end with letter ‘r’.

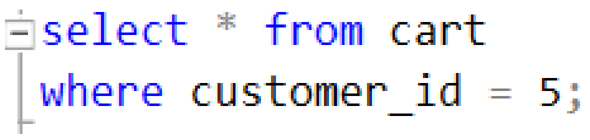


```sql
select * from products
where name like '%r';
```

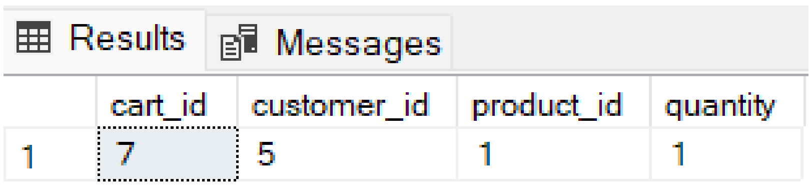


	product_id	name	Description	price	stockQuantity
1	6	Coffee Maker	Automatic coffee maker	50	25
2	7	Refrigerator	Energy-efficient	800	10
3	9	Blender	High-speed blender	70	20
4	10	Vacuum Cleaner	Bagless vacuum cleaner	120	10

Q7. Retrieve Cart Items for Customer 5.

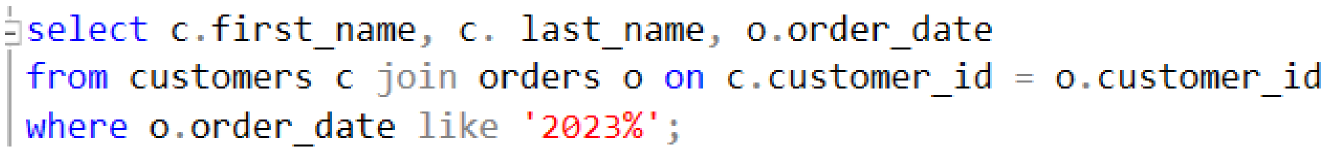


```sql
select * from cart
where customer_id = 5;
```

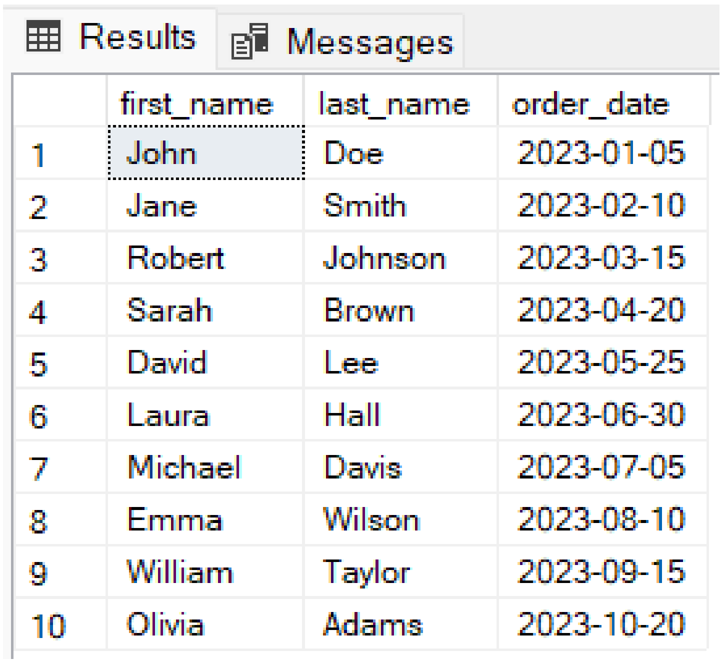


	cart_id	customer_id	product_id	quantity
1	7	5	1	1

Q8. Find Customers Who Placed Orders in 2023.



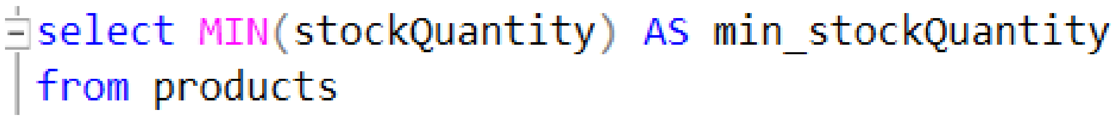
```sql
select c.first_name, c. last_name, o.order_date
from customers c join orders o on c.customer_id = o.customer_id
where o.order_date like '2023%';
```



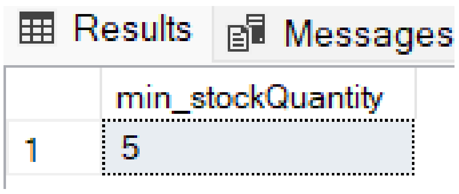
Results | Messages

	first_name	last_name	order_date
1	John	Doe	2023-01-05
2	Jane	Smith	2023-02-10
3	Robert	Johnson	2023-03-15
4	Sarah	Brown	2023-04-20
5	David	Lee	2023-05-25
6	Laura	Hall	2023-06-30
7	Michael	Davis	2023-07-05
8	Emma	Wilson	2023-08-10
9	William	Taylor	2023-09-15
10	Olivia	Adams	2023-10-20

Q9. Determine the Minimum Stock Quantity for Each Product Category.



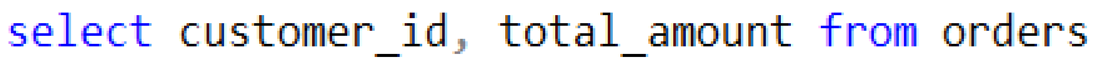
```
select MIN(stockQuantity) AS min_stockQuantity
from products
```



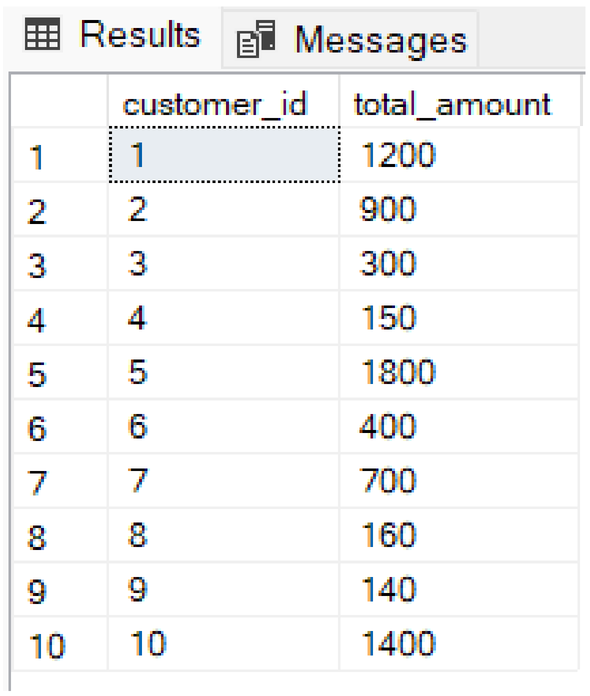
Results | Messages

	min_stockQuantity
1	5

Q10. Calculate the Total Amount Spent by Each Customer.



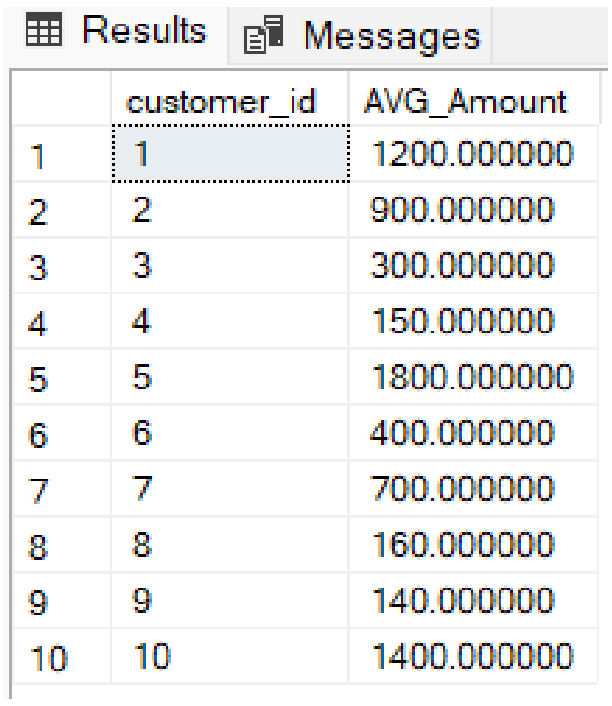
```
select customer_id, total_amount from orders
```


Results | Messages

	customer_id	total_amount
1	1	1200
2	2	900
3	3	300
4	4	150
5	5	1800
6	6	400
7	7	700
8	8	160
9	9	140
10	10	1400

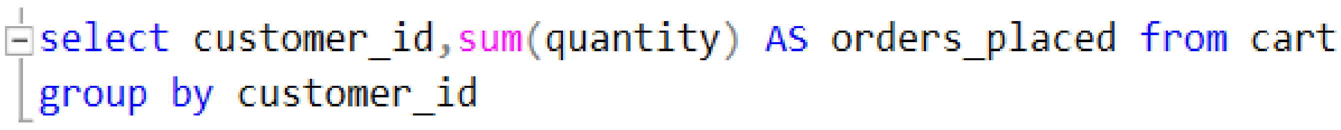
Q11. Find the Average Order Amount for Each Customer.

```
select customer_id, avg(total_amount) AS AVG_Amount
from orders group by customer_id;
```

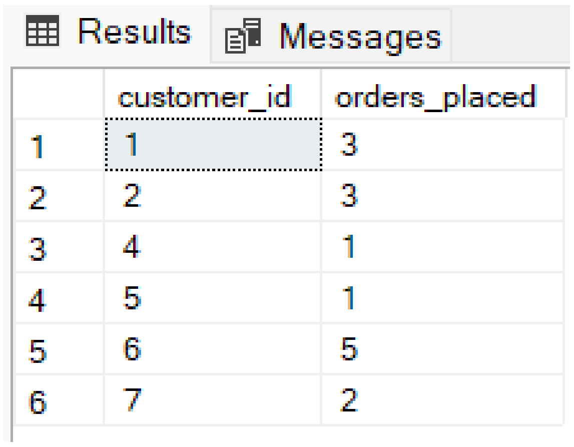

Results | Messages

	customer_id	AVG_Amount
1	1	1200.000000
2	2	900.000000
3	3	300.000000
4	4	150.000000
5	5	1800.000000
6	6	400.000000
7	7	700.000000
8	8	160.000000
9	9	140.000000
10	10	1400.000000

Q12. Count the Number of Orders Placed by Each Customer.

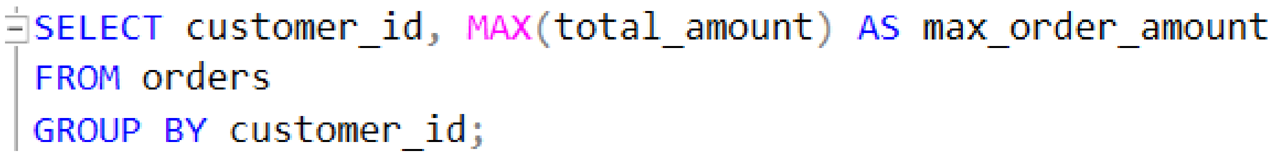


```sql
select customer_id,sum(quantity) AS orders_placed from cart
group by customer_id
```

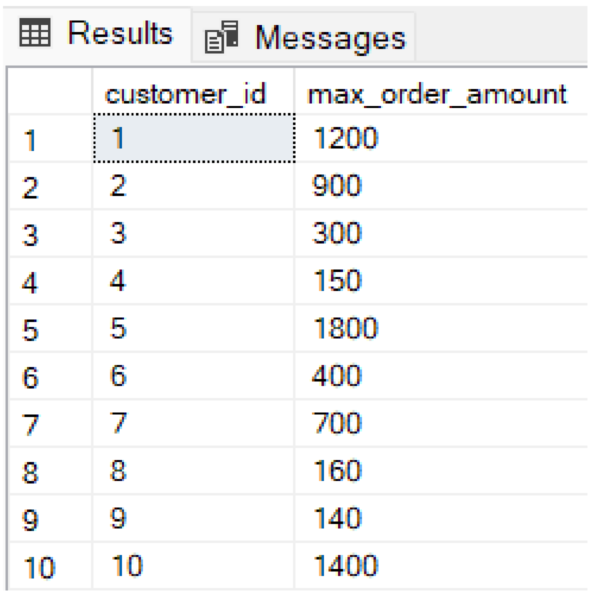


	customer_id	orders_placed
1	1	3
2	2	3
3	4	1
4	5	1
5	6	5
6	7	2

Q13. Find the Maximum Order Amount for Each Customer.

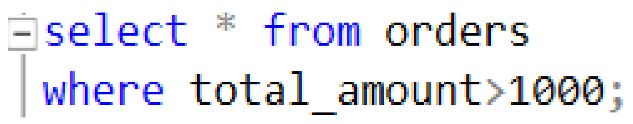


```sql
SELECT customer_id, MAX(total_amount) AS max_order_amount
FROM orders
GROUP BY customer_id;
```

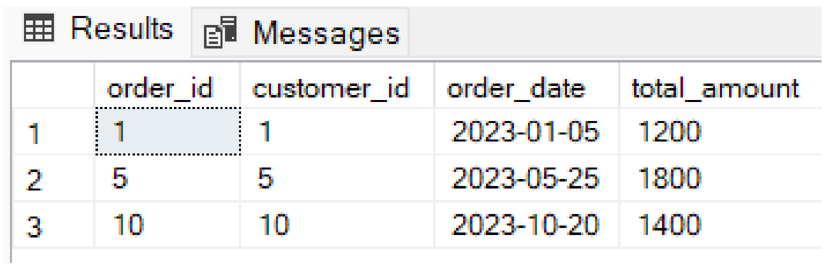


	customer_id	max_order_amount
1	1	1200
2	2	900
3	3	300
4	4	150
5	5	1800
6	6	400
7	7	700
8	8	160
9	9	140
10	10	1400

Q14. Get Customers Who Placed Orders Totaling Over $1000.



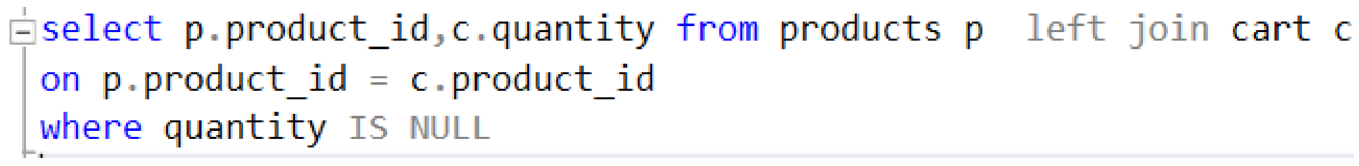
```sql
select * from orders
where total_amount>1000;
```



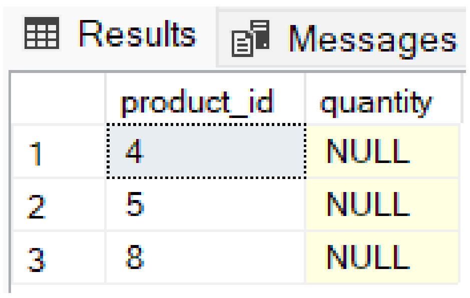
Results | Messages

	order_id	customer_id	order_date	total_amount
1	1	1	2023-01-05	1200
2	5	5	2023-05-25	1800
3	10	10	2023-10-20	1400

Q15. Subquery to Find Products Not in the Cart.



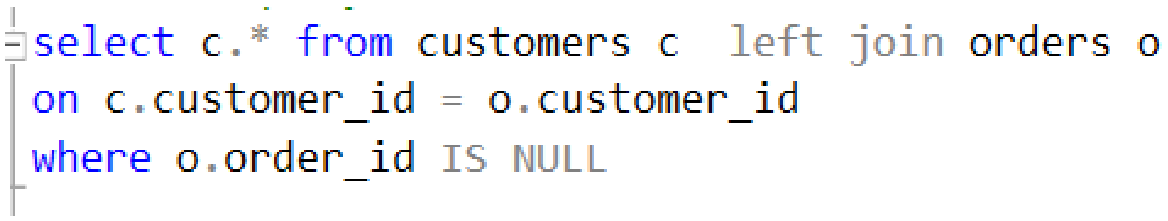
```sql
select p.product_id,c.quantity from products p  left join cart c
on p.product_id = c.product_id
where quantity IS NULL
```



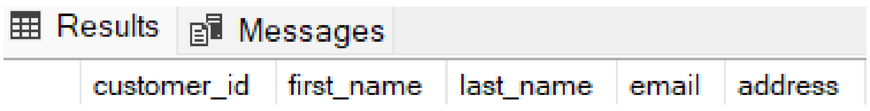
Results | Messages

	product_id	quantity
1	4	NULL
2	5	NULL
3	8	NULL

Q16. Subquery to Find Customers Who Haven't Placed Orders.



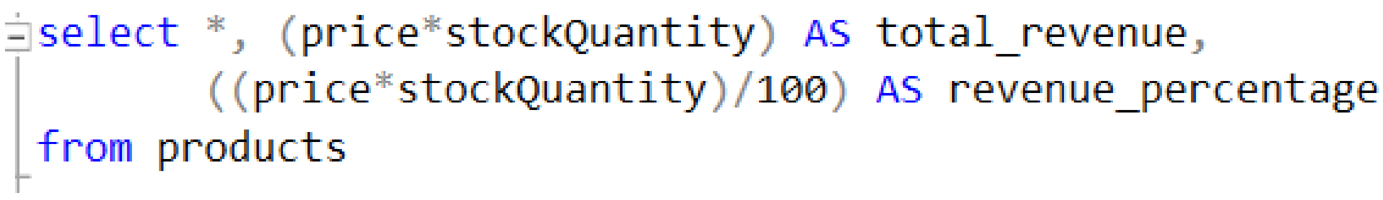
```sql
select c.* from customers c  left join orders o
on c.customer_id = o.customer_id
where o.order_id IS NULL
```



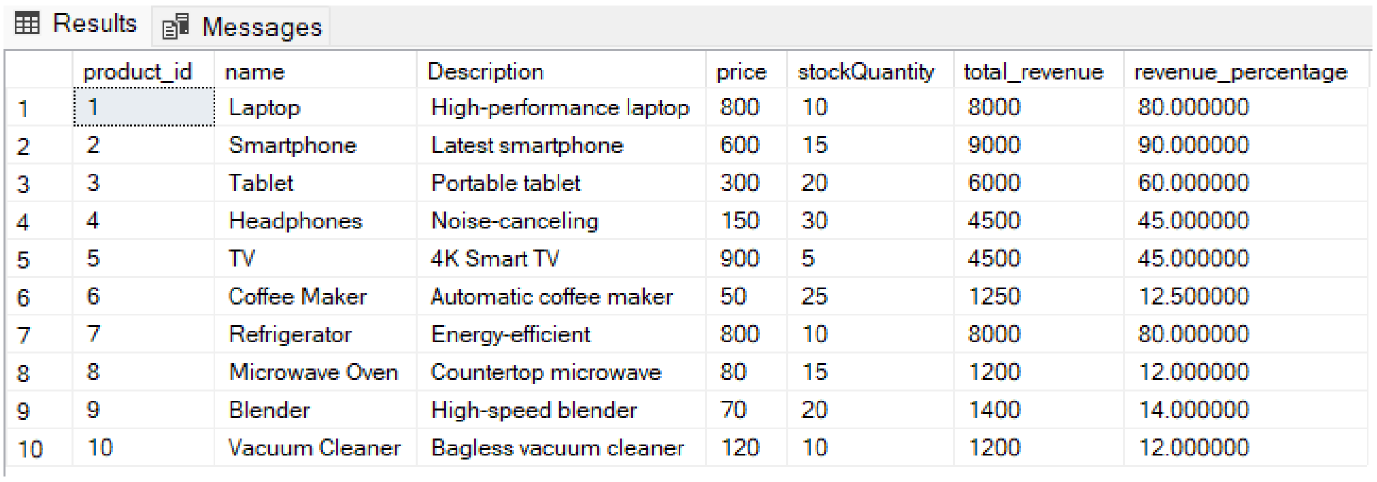
customer_id	first_name	last_name	email	address

(Empty Record as all customers placed order)

Q17. Subquery to Calculate the Percentage of Total Revenue for a Product.

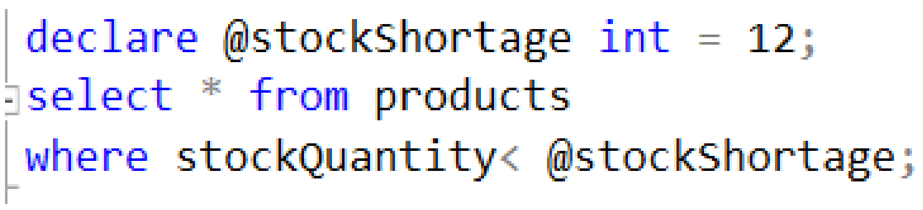


```
select *, (price*stockQuantity) AS total_revenue,
        ((price*stockQuantity)/100) AS revenue_percentage
from products
```

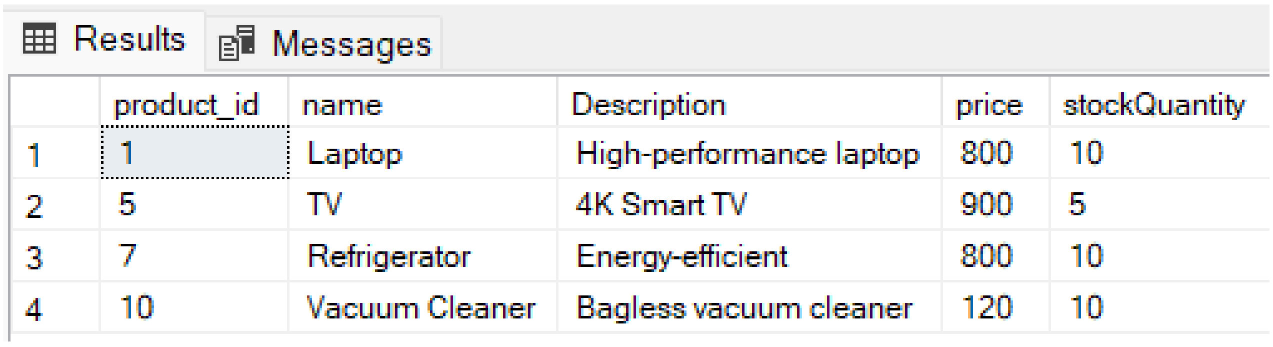


	product_id	name	Description	price	stockQuantity	total_revenue	revenue_percentage
1	1	Laptop	High-performance laptop	800	10	8000	80.000000
2	2	Smartphone	Latest smartphone	600	15	9000	90.000000
3	3	Tablet	Portable tablet	300	20	6000	60.000000
4	4	Headphones	Noise-canceling	150	30	4500	45.000000
5	5	TV	4K Smart TV	900	5	4500	45.000000
6	6	Coffee Maker	Automatic coffee maker	50	25	1250	12.500000
7	7	Refrigerator	Energy-efficient	800	10	8000	80.000000
8	8	Microwave Oven	Countertop microwave	80	15	1200	12.000000
9	9	Blender	High-speed blender	70	20	1400	14.000000
10	10	Vacuum Cleaner	Bagless vacuum cleaner	120	10	1200	12.000000

Q18. Subquery to Find Products with Low Stock.

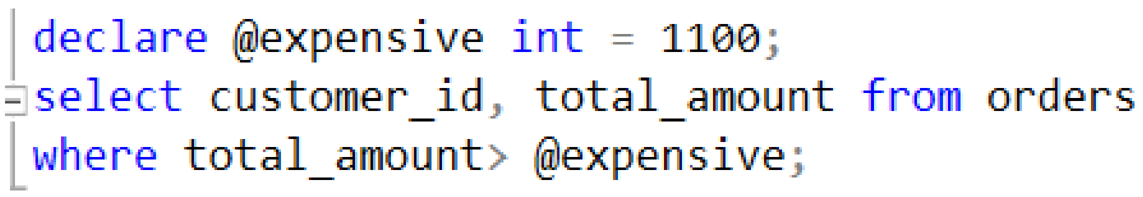


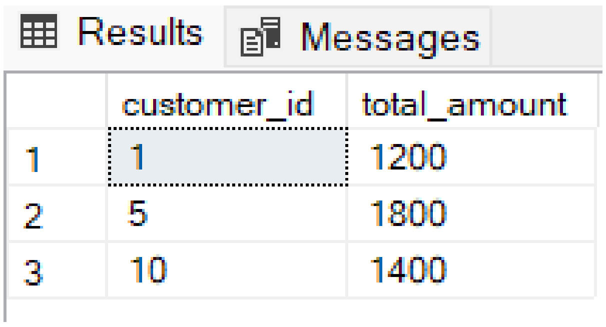
```
declare @stockShortage int = 12;
select * from products
where stockQuantity< @stockShortage;
```


Results | Messages

	product_id	name	Description	price	stockQuantity
1	1	Laptop	High-performance laptop	800	10
2	5	TV	4K Smart TV	900	5
3	7	Refrigerator	Energy-efficient	800	10
4	10	Vacuum Cleaner	Bagless vacuum cleaner	120	10

Q19. Subquery to Find Customers Who Placed High-Value Orders.


```sql
declare @expensive int = 1100;
select customer_id, total_amount from orders
where total_amount> @expensive;
```


Results | Messages

	customer_id	total_amount
1	1	1200
2	5	1800
3	10	1400

-----------------------------x-----------------------------