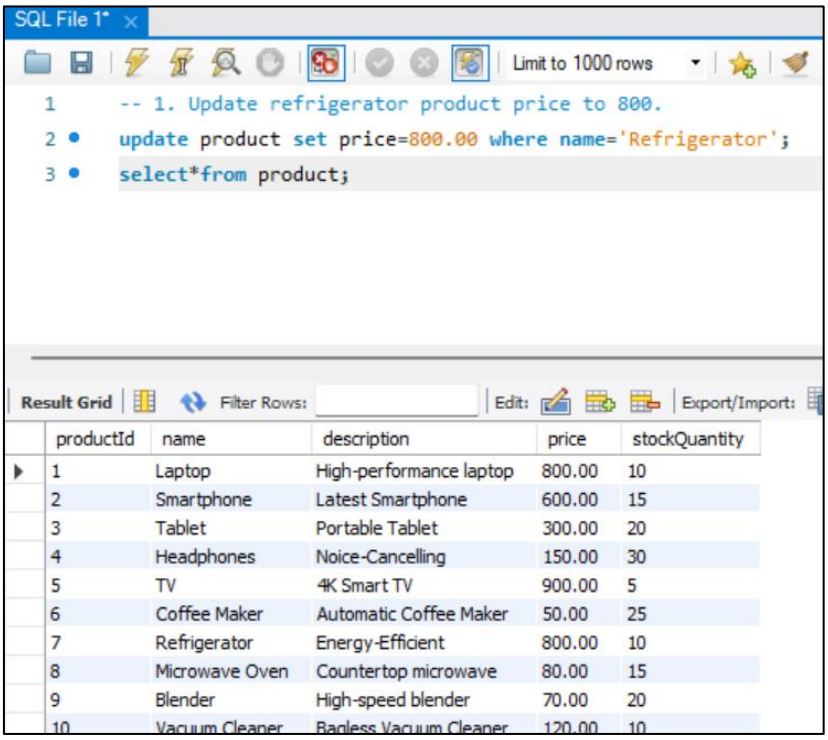
SQL - Coding Challenge – Ecom

1. Update refrigerator product price to 800.

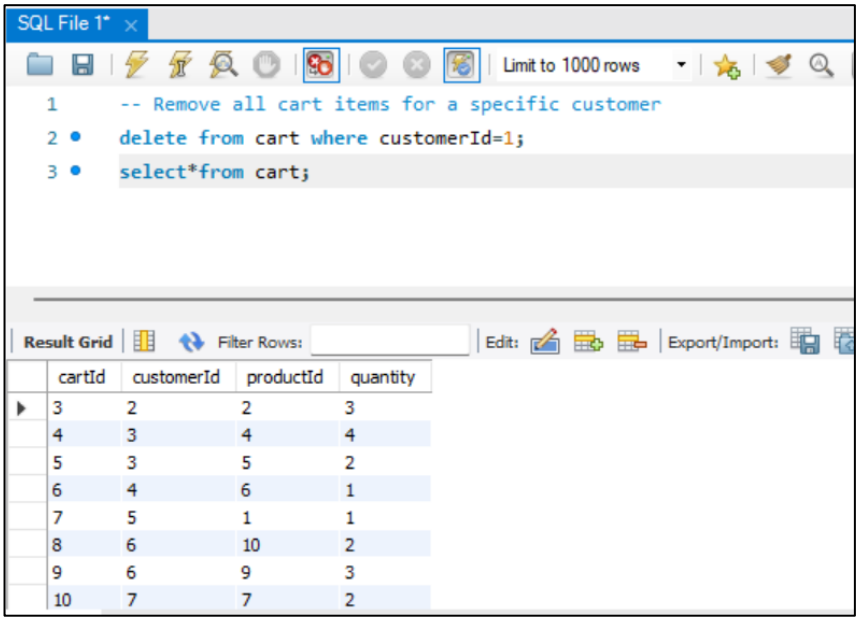
update product set price=800.00 where name='Refrigerator';



productId	name	description	price	stockQuantity
1	Laptop	High-performance laptop	800.00	10
2	Smartphone	Latest Smartphone	600.00	15
3	Tablet	Portable Tablet	300.00	20
4	Headphones	Noice-Cancelling	150.00	30
5	TV	4K Smart TV	900.00	5
6	Coffee Maker	Automatic Coffee Maker	50.00	25
7	Refrigerator	Energy-Efficient	800.00	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

2. Remove all cart items for a specific customer.

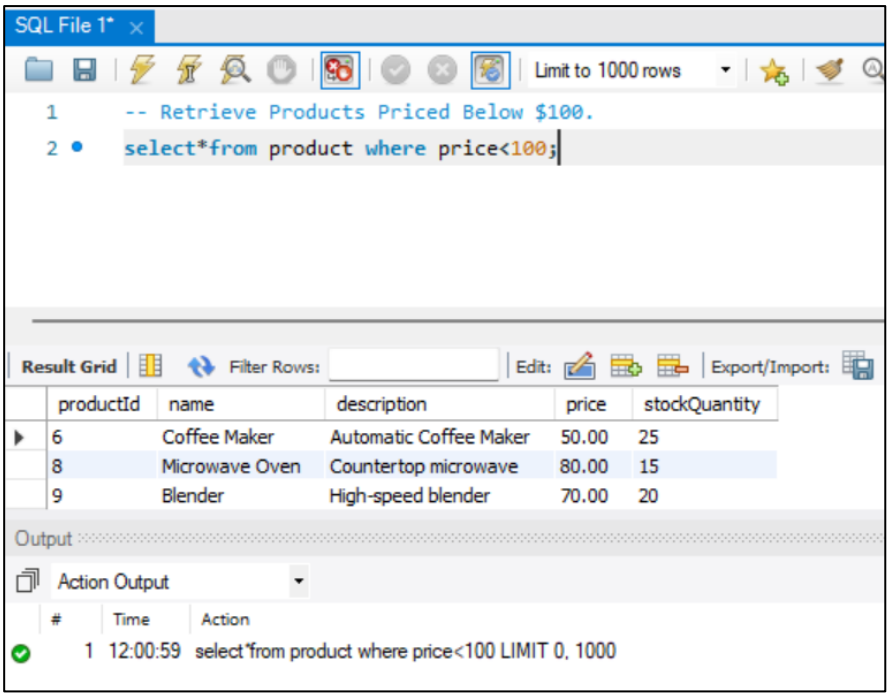
delete from cart where customerId=1;



cartId	customerId	productId	quantity
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

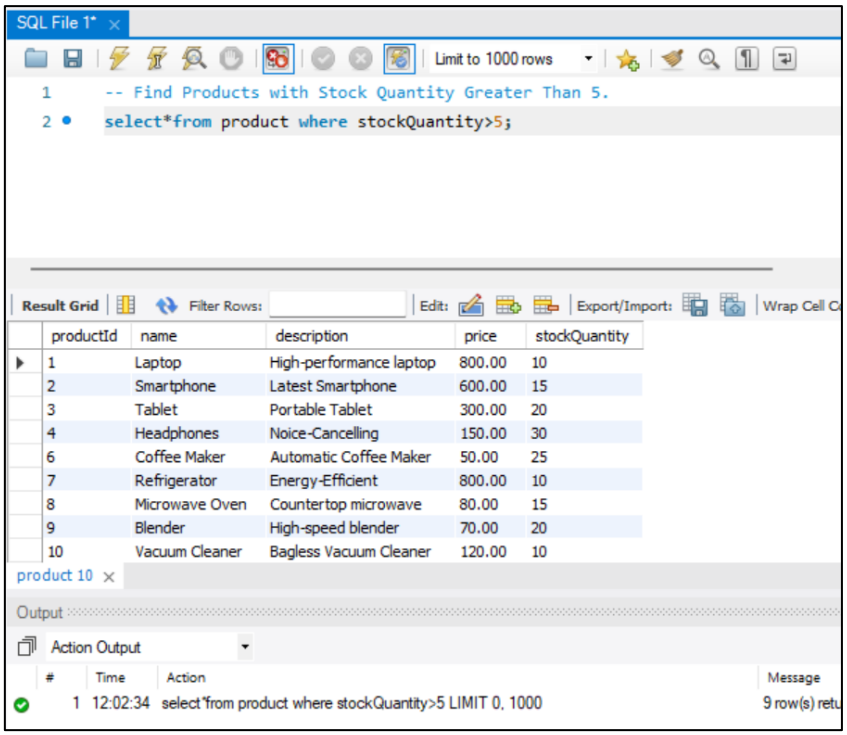
3. Retrieve Products Priced Below $100.

select*from product where price<100;



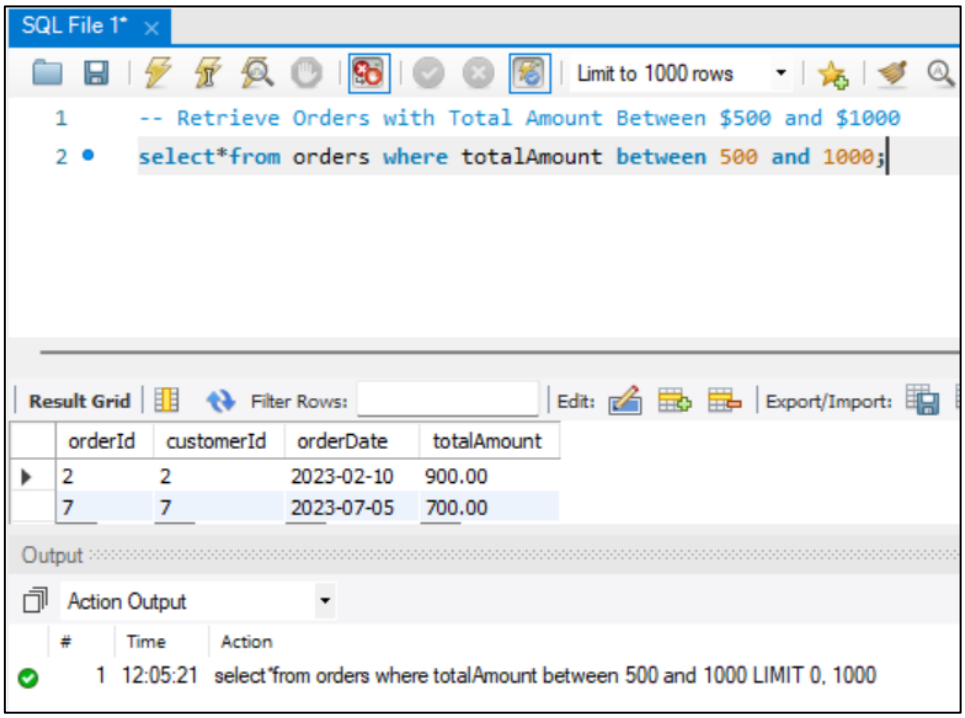
4. Find Products with Stock Quantity Greater Than 5.

select*from product where stockQuantity>5;



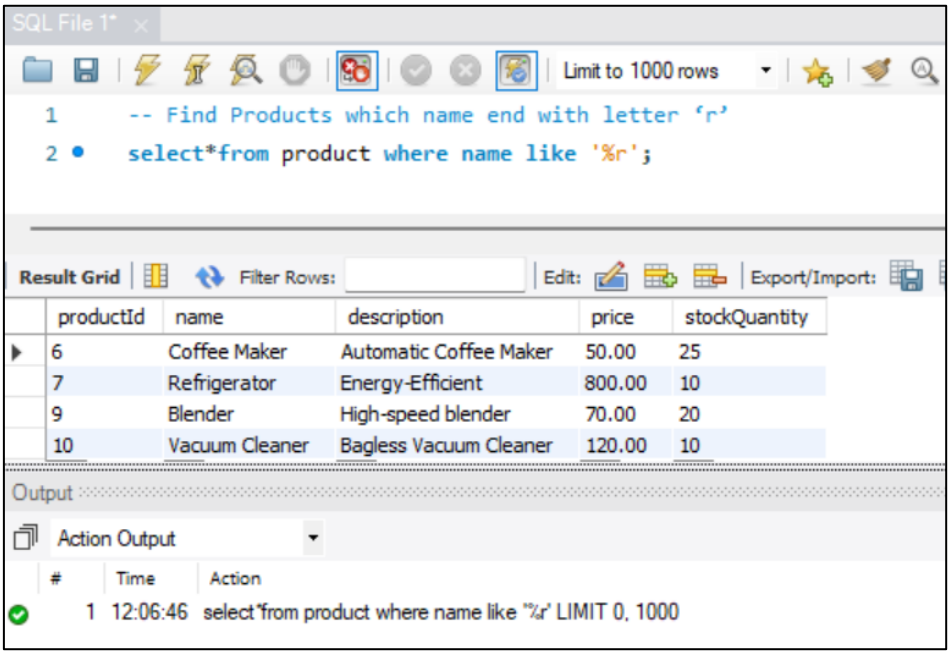
5. Retrieve Orders with Total Amount Between $500 and $1000.

select*from orders where totalAmount between 500 and 1000;



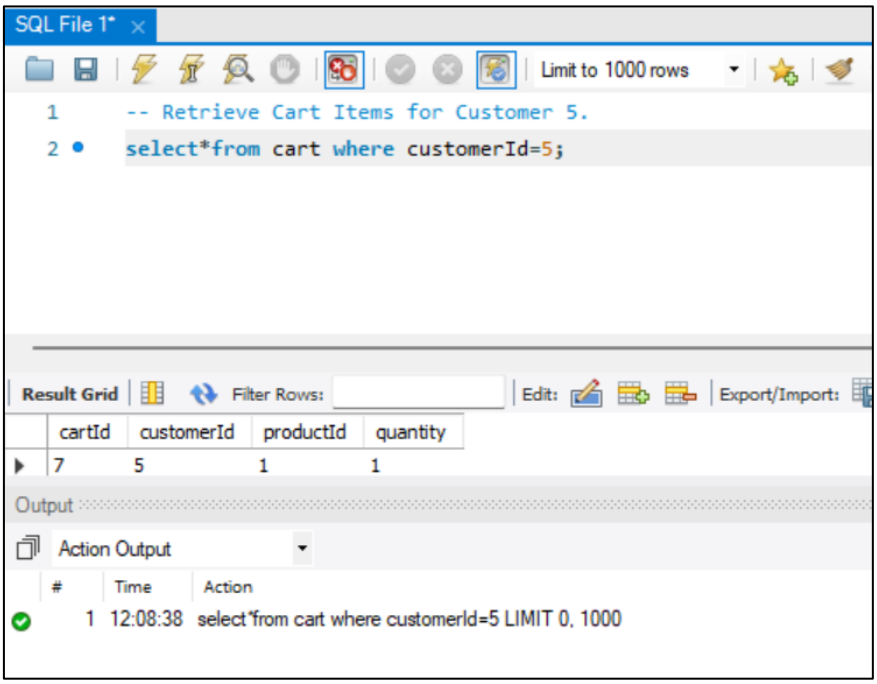
6. Find Products which name end with letter 'r'.

select*from product where name like '%r';



7. Retrieve Cart Items for Customer 5.

select*from cart where customerId=5;



8. Find Customers Who Placed Orders in 2023

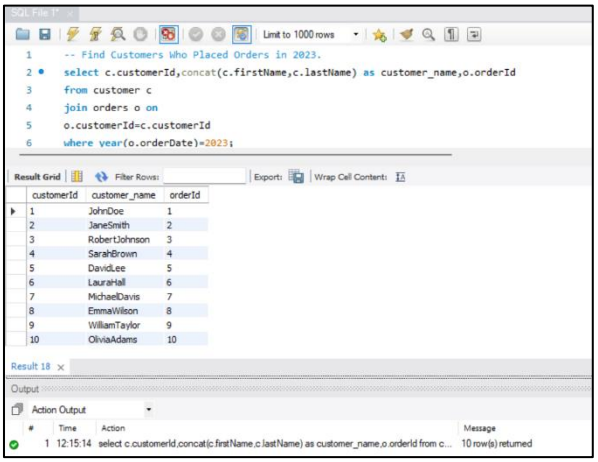
select c.customerId,concat(c.firstName,c.lastName) as customer_name,o.orderId

from customer c

join orders o on

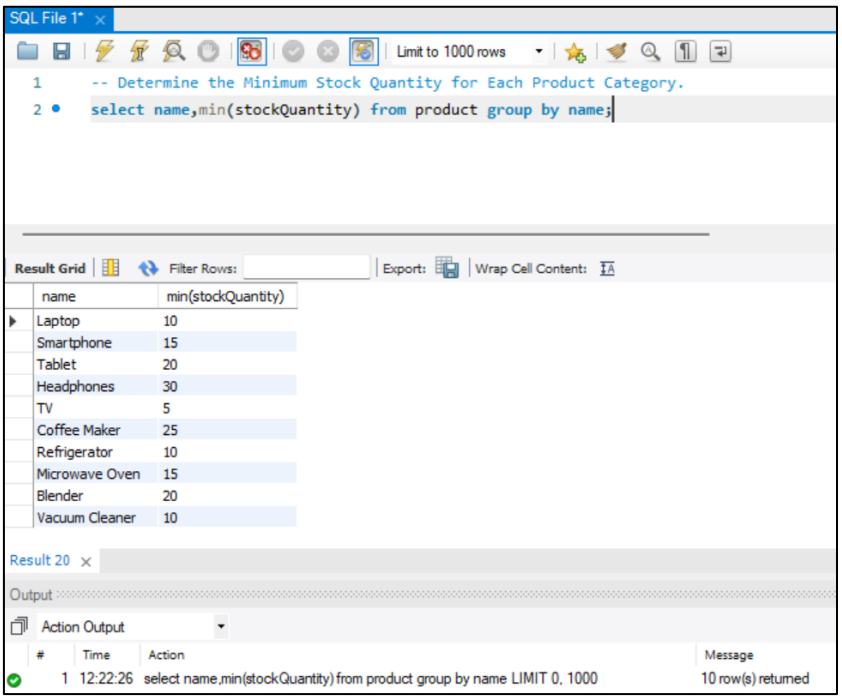
o.customerId=c.customerId

where year(o.orderDate)=2023;



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

select name,min(stockQuantity) from product group by name;



10. Calculate the Total Amount Spent by Each Customer.

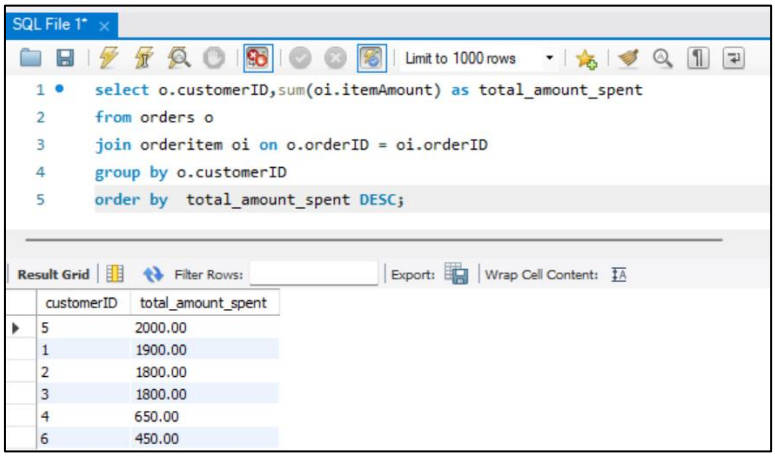
select o.customerID,sum(oi.itemAmount) as total_amount_spent

from orders o

left join orderitem oi on o.orderID = oi.orderID

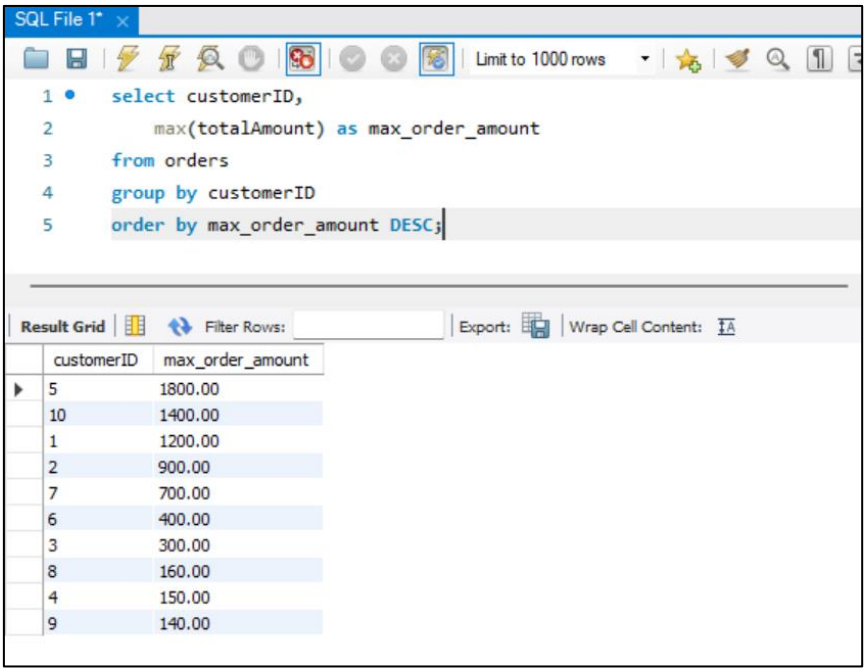
group by o.customerID

order by total_amount_spent DESC;



11. Find the Average Order Amount for Each Customer.

```
select customerID,
max(totalAmount) as max_order_amount
from orders
group by customerID
order by max_order_amount DESC;
```



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

```
select c.customerId,concat(c.firstName,c.lastName) as
customer_name,count(o.orderId) as noof_orders_placed
from customer c
join orders o on o.customerId=c.customerId
group by customerid
order by noof_orders_placed;
```

```sql
-- Count the Number of Orders Placed by Each Customer.
select c.customerId,concat(c.firstname,c.lastName) as customer_name,count(o.orderId) as noof_orders_placed
from customer c
join orders o on o.customerId=c.customerId
group by customerId
order by noof_orders_placed;
```

customerId	customer_name	noof_orders_placed
1	JohnDoe	1
2	JaneSmith	1
3	RobertJohnson	1
4	SarahBrown	1
5	DavidLee	1
6	LauraHall	1
7	MichaelDavis	1
8	EmmaWilson	1
9	WilliamTaylor	1
10	OliviaAdams	1

Result 23

Output

Action Output

#	Time	Action	Message
1	12:37:28	select c.customerId,concat(c.firstname,c.lastName) as customer_name,count(o.orderId) ...	10 row(s) returned

13. Find the Maximum Order Amount for Each Customer

select o.customerId,max(o.totalAmount) as max_order-amount

from orders o

group by o.customerId

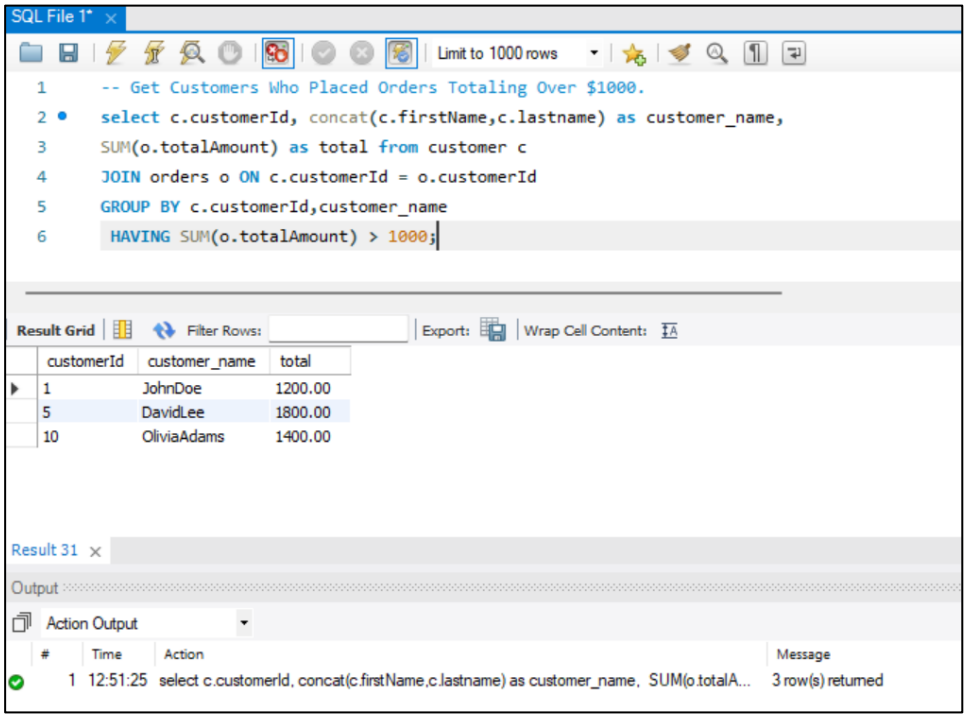
order by max_order_amount DESC;

```sql
-- Find the Maximum Order Amount for Each Customer.
select o.customerID,max(o.totalAmount) as max_order_amount
from orders o
group by o.customerID
ORDER BY max_order_amount DESC ,customerId ASC;
```

customerID	max_order_amount
5	1800.00
10	1400.00
1	1200.00
2	900.00
7	700.00
6	400.00
3	300.00
8	160.00
4	150.00
9	140.00

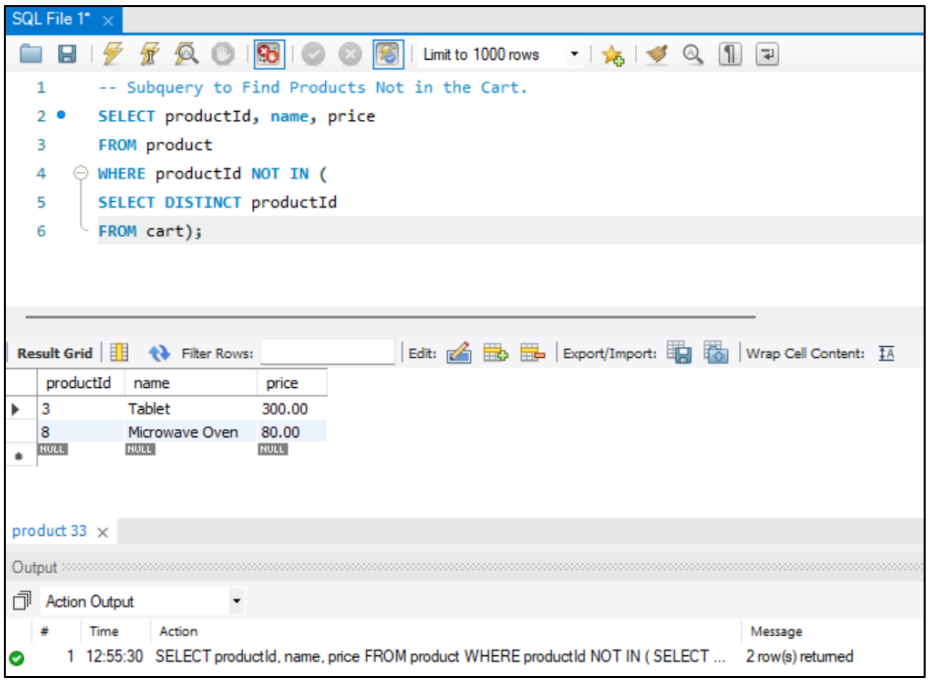
14. Get Customers Who Placed Orders Totaling Over $1000.

```
select c.customerId,concat(c.firstName,c.lastName) as customer_name,
sum(o.totalAmount) as total from customer c
join orders o on c.customerId = o.customerId
group by c.customerId,customer_name
having sum(o.totalAmount)>1000;
```



15. Subquery to Find Products Not in the Cart.

```
select productid,name,price
from product
where productid not in(
select distinct productId from cart);
```



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

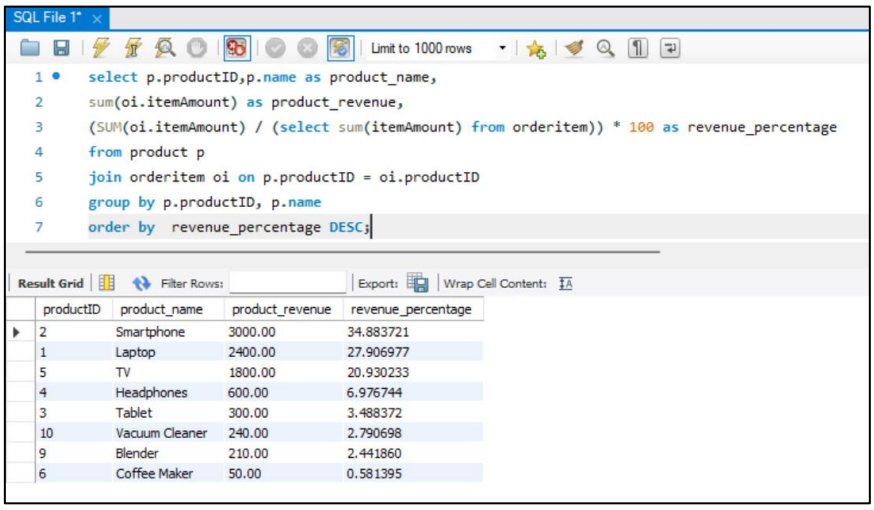
select p.productid,name as product_name,sum(oi.itemAmount) as product_revenue,(sum(oi.itemAmount)/(select sum(itemAmount) from orderitem))*100 as revenue_percentage

from product p

join orderitem oi on p.productId=oi.productId

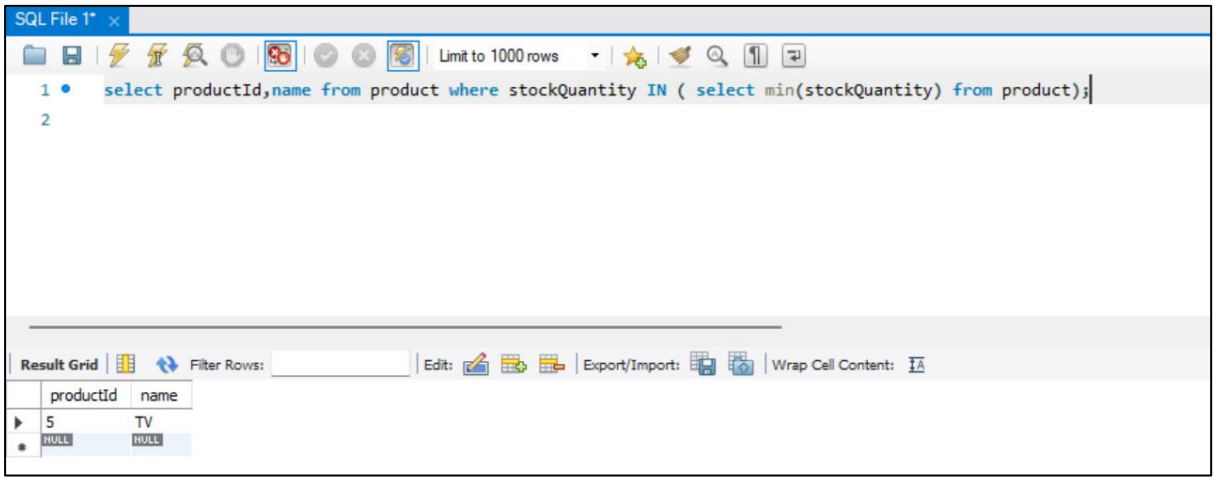
group by p.productId,p.name

order by revenue_percentage DESC;



18. Subquery to Find Products with Low Stock.

select productId,name from product where stockQuantity in(select min(stockQuantity) from product);



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

select firstName,lastName,email,address from customer where customerId in

(select customerId from orders where totalAmount=(select max(totalAmount) from orders));