Project Title: "Sales Insights Using SQL: Data-Driven Business Decisions"

Introduction:

This project focuses on analyzing sales data using SQL queries to extract meaningful insights from a sales database. By utilizing SQL, we aim to uncover key trends in customer purchases, employee performance, and order fulfillment. The insights derived from this analysis can help businesses improve their decision-making, optimize operations, and enhance customer satisfaction.

QUESTIONS

- IDENTIFY THE TOTAL NO OF PRODUCTS SOLD.
- OTHER THAN COMPLETED, DISPLAY THE AVAILABLE DELIVERY STATUS'S
- DISPLAY THE ORDER ID, ORDER DATE AND PRODUCT NAME FOR ALL THE COMPLETED ORDERS.
- SORT THE ABOVE QUERY TO SHOW THE EARLIEST ORDERS AT THE TOP. ALSO, DISPLAY THE CUSTOMER WHO PURCHASED THESE ORDERS.
- DISPLAY THE TOTAL NO OF ORDERS CORRESPONDING TO EACH DELIVERY STATUS
- HOW MANY ORDERS ARE STILL NOT COMPLETED FOR ORDERS PURCHASING MORE THAN 1 ITEM?
- FIND THE TOTAL NUMBER OF ORDERS CORRESPONDING TO EACH DELIVERY STATUS-- BY IGNORING THE CASE IN THE DELIVERY STATUS. THE STATUS WITH HIGHEST NO OF ORDERS SHOULD BE AT THE TOP.

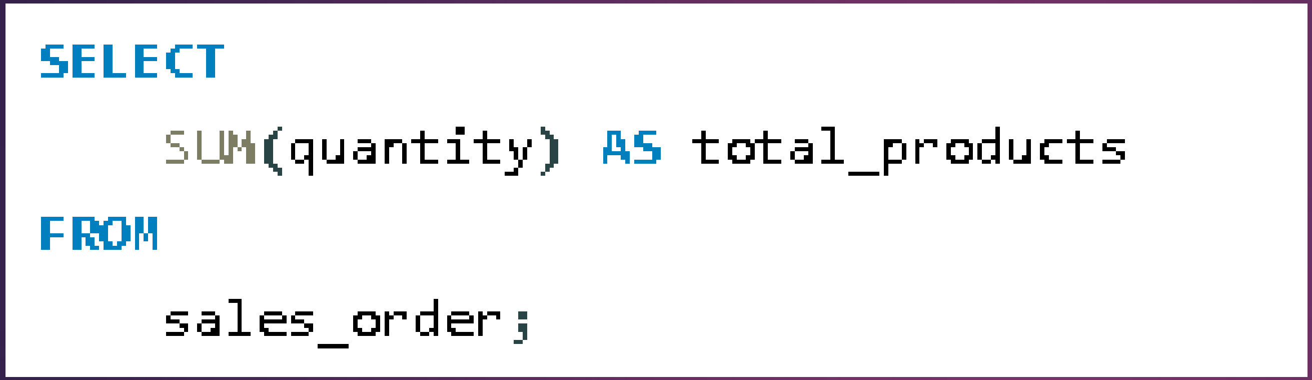
QUESTIONS

- WRITE A QUERY TO IDENTIFY THE TOTAL PRODUCTS PURCHASED BY EACH CUSTOMER .
- DISPLAY THE TOTAL SALES AND AVERAGE SALES DONE FOR EACH DAY.
- DISPLAY THE CUSTOMER NAME, EMPLOYEE NAME, AND TOTAL SALE AMOUNT OF ALL ORDERS -- WHICH ARE EITHER ON HOLD OR PENDING.
- FETCH ALL THE ORDERS WHICH WERE NEITHER COMPLETED/PENDING OR WERE HANDLED BY THE EMPLOYEE ABRAR. -- DISPLAY EMPLOYEE NAME AND ALL DETAILS OF ORDER.
- FETCH THE ORDERS WHICH COST MORE THAN 2000 BUT DID NOT INCLUDE THE MACBOOK PRO. -- PRINT THE TOTAL SALE AMOUNT AS WELL.
- IDENTIFY THE CUSTOMERS WHO HAVE NOT PURCHASED ANY PRODUCT YET.
- WRITE A QUERY TO IDENTIFY THE TOTAL PRODUCTS PURCHASED BY EACH CUSTOMER. RETURN ALL CUSTOMERS IRRESPECTIVE OF WHETHER THEY HAVE MADE A PURCHASE OR NOT. -- SORT THE RESULT WITH THE HIGHEST NO OF ORDERS AT THE TOP.

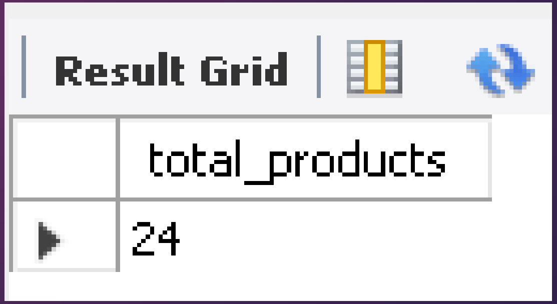
QUESTIONS

- IDENTIFY THE TOTAL NO OF PRODUCTS SOLD.
- CORRESPONDING TO EACH EMPLOYEE, DISPLAY THE TOTAL SALES THEY MADE OF ALL THE COMPLETED ORDERS. -- DISPLAY TOTAL SALES AS 0 IF AN EMPLOYEE MADE NO SALES YET.
- RE-WRITE THE ABOVE QUERY TO DISPLAY THE TOTAL SALES MADE BY EACH EMPLOYEE CORRESPONDING TO EACH CUSTOMER. IF AN EMPLOYEE HAS NOT SERVED A CUSTOMER YET THEN DISPLAY "-" UNDER THE CUSTOMER.
- RE-WRITE THE ABOVE QUERY TO DISPLAY ONLY THOSE RECORDS WHERE THE TOTAL SALES ARE ABOVE 1000.
- IDENTIFY EMPLOYEES WHO HAVE SERVED MORE THAN 2 CUSTOMERS.
- IDENTIFY THE CUSTOMERS WHO HAVE PURCHASED MORE THAN 5 PRODUCTS.
- IDENTIFY CUSTOMERS WHOSE AVERAGE PURCHASE COST EXCEEDS THE AVERAGE SALE OF ALL THE ORDERS.

1. Identify the total no of products sold?

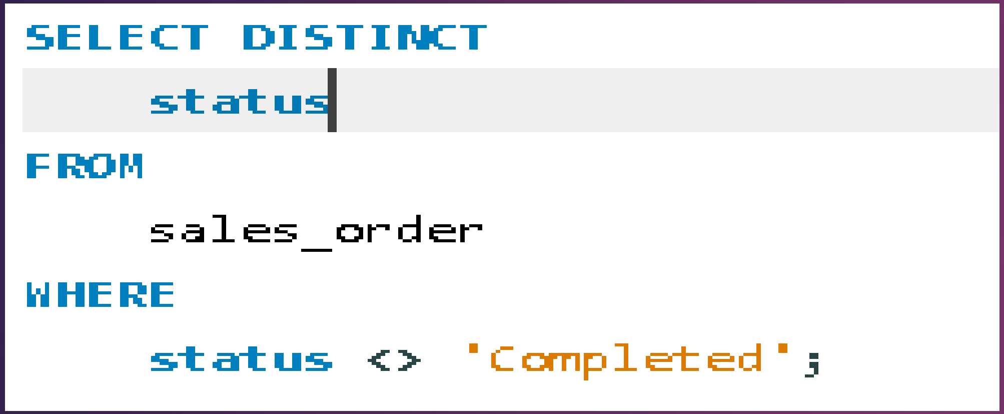


```
SELECT
    SUM(quantity) AS total_products
FROM
    sales_order;
```

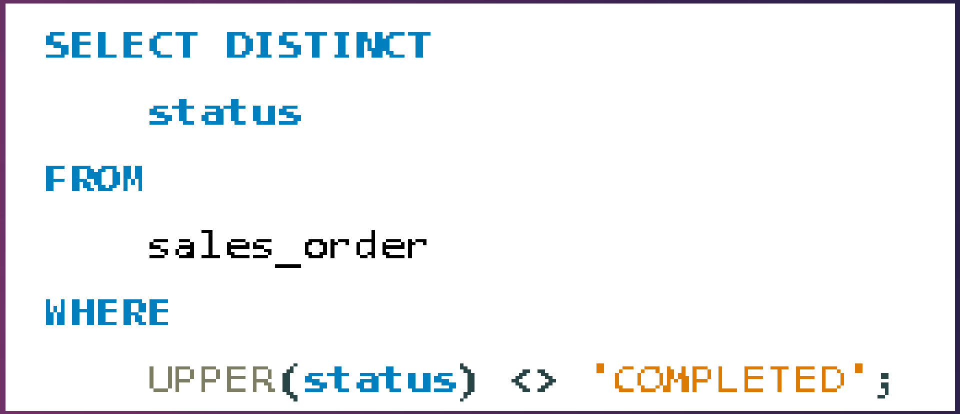


```
total_products
24
```

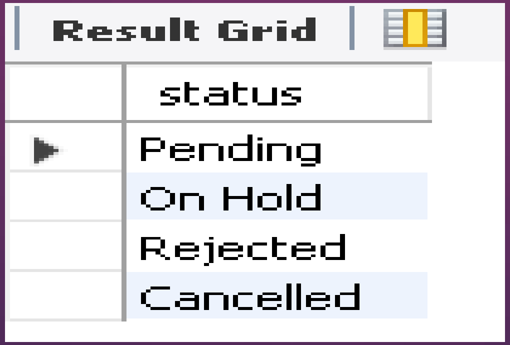
2. Other than Completed, display the available delivery status's?



```
SELECT DISTINCT
    status
FROM
    sales_order
WHERE
    status <> 'Completed';
```



```
SELECT DISTINCT
    status
FROM
    sales_order
WHERE
    UPPER(status) <> 'COMPLETED';
```

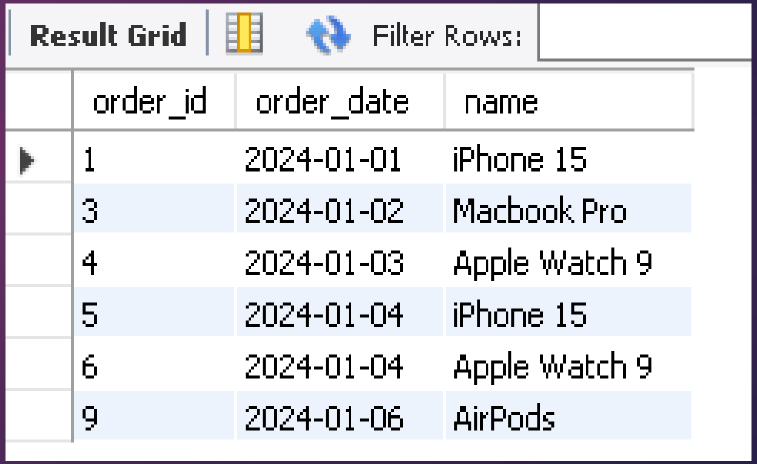


Result Grid

status
Pending
On Hold
Rejected
Cancelled

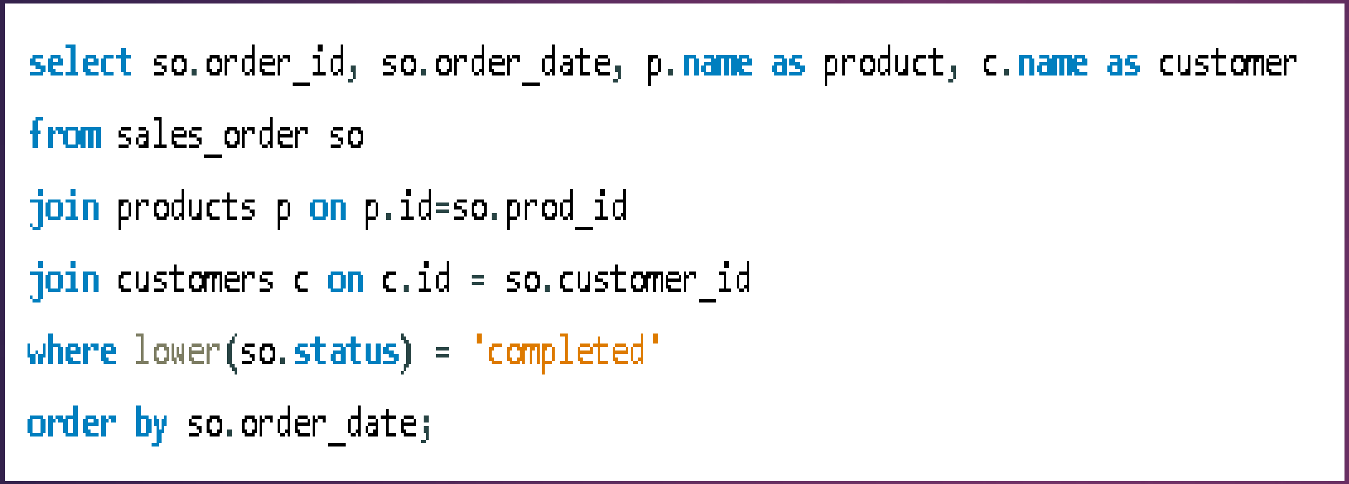
3. Display the order id, order_date and product_name for all the completed orders?

```
SELECT
    order_id, order_date, name
FROM
    sales_order
        JOIN
    products ON sales_order.prod_id = products.id
WHERE
    status = 'Completed';
```

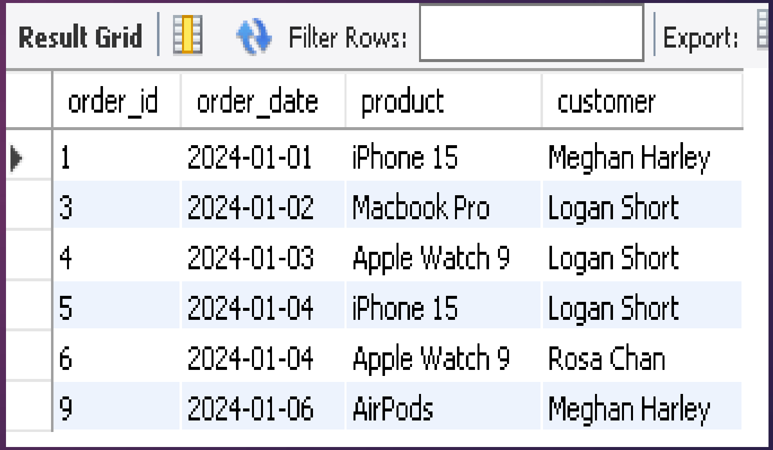


order_id	order_date	name
1	2024-01-01	iPhone 15
3	2024-01-02	Macbook Pro
4	2024-01-03	Apple Watch 9
5	2024-01-04	iPhone 15
6	2024-01-04	Apple Watch 9
9	2024-01-06	AirPods

4. Sort the above query to show the earliest orders at the top. Also, display the customer who purchasedthese orders?



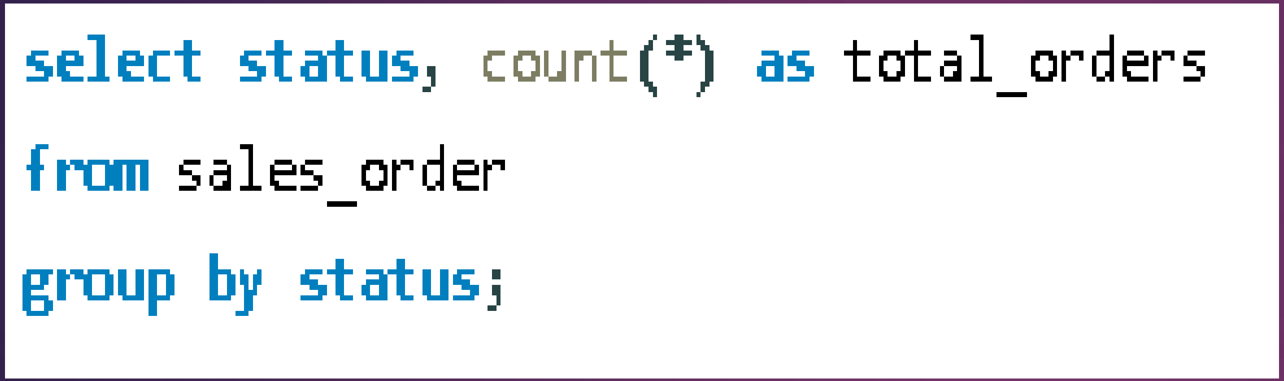
```
select so.order_id, so.order_date, p.name as product, c.name as customer
from sales_order so
join products p on p.id=so.prod_id
join customers c on c.id = so.customer_id
where lower(so.status) = 'completed'
order by so.order_date;
```



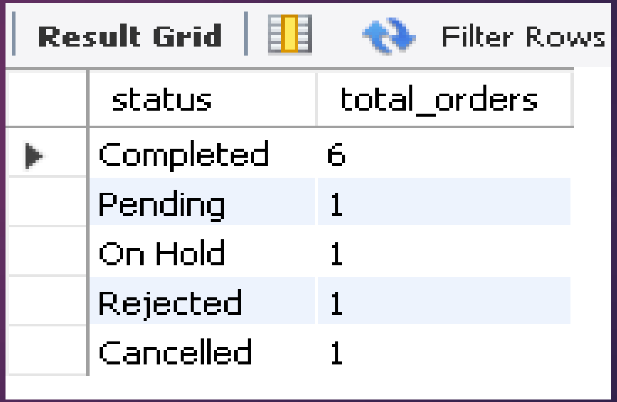
Result Grid | Filter Rows: | Export:

order_id	order_date	product	customer
1	2024-01-01	iPhone 15	Meghan Harley
3	2024-01-02	Macbook Pro	Logan Short
4	2024-01-03	Apple Watch 9	Logan Short
5	2024-01-04	iPhone 15	Logan Short
6	2024-01-04	Apple Watch 9	Rosa Chan
9	2024-01-06	AirPods	Meghan Harley

5. Display the total no of orders corresponding to each delivery status?

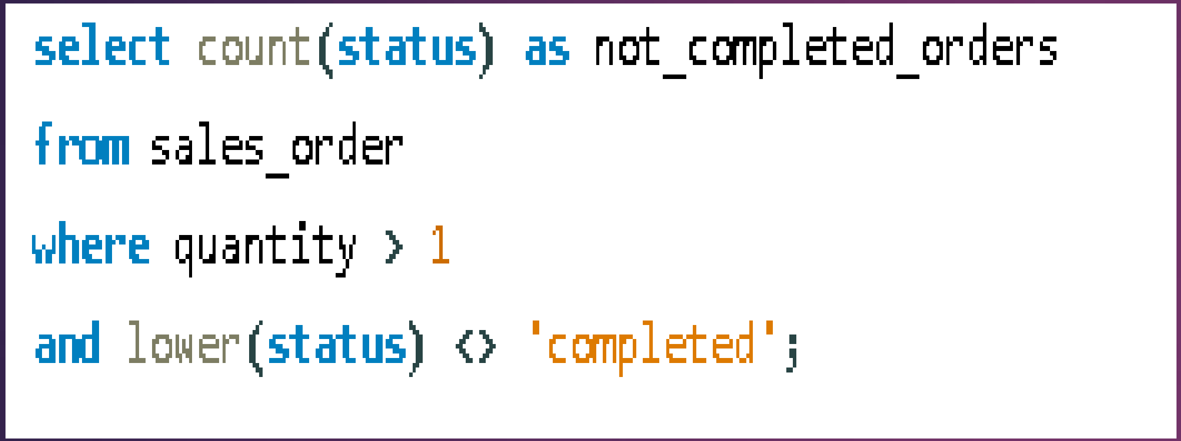


```
select status, count(*) as total_orders
from sales_order
group by status;
```

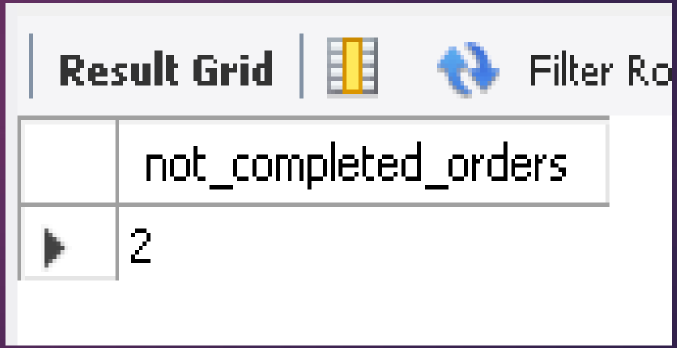


status	total_orders
Completed	6
Pending	1
On Hold	1
Rejected	1
Cancelled	1

6. How many orders are still not completed for orders purchasing more than 1 item?

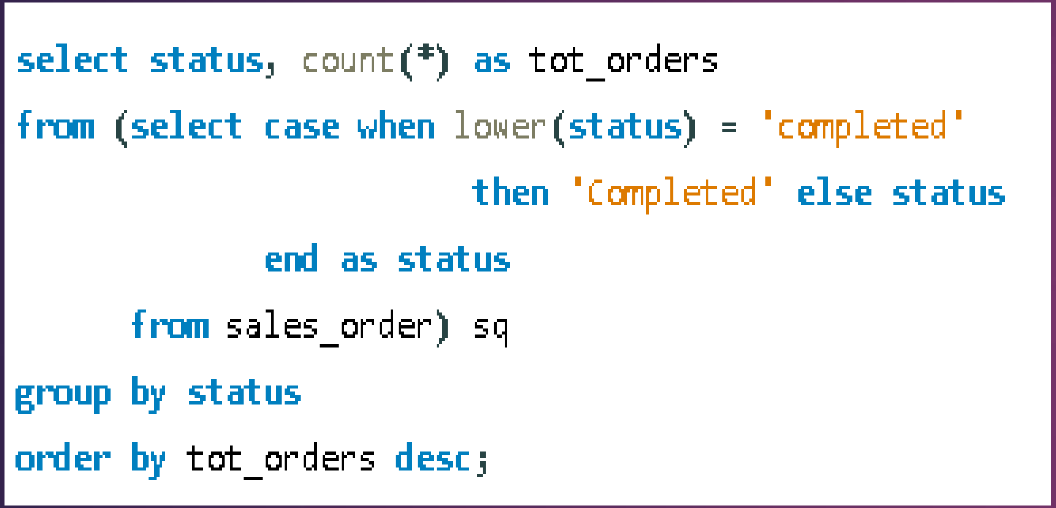


```
select count(status) as not_completed_orders
from sales_order
where quantity > 1
and lower(status) <> 'completed';
```

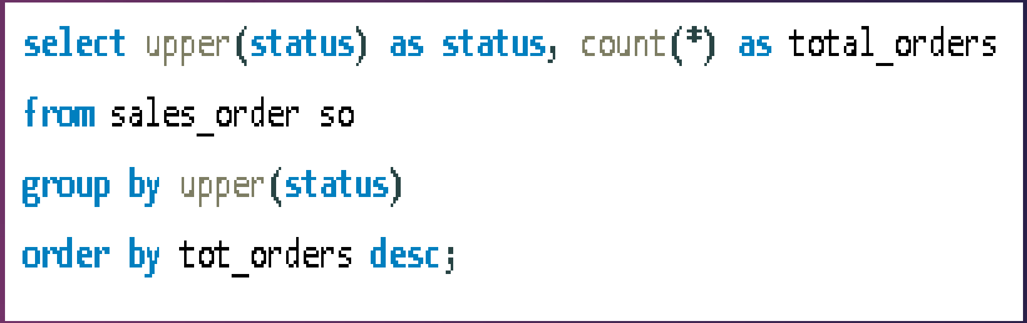


```
not_completed_orders
2
```

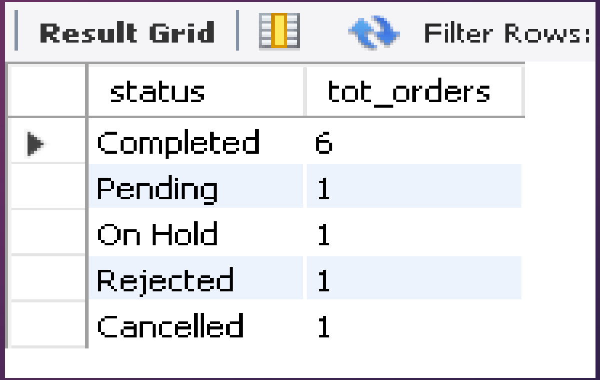
7. Find the total number of orders corresponding to each delivery status-- by ignoring the case in the delivery status. The status with highest no of orders should be at the top?



```
select status, count(*) as tot_orders
from (select case when lower(status) = 'completed'
                                 then 'Completed' else status
                    end as status
        from sales_order) sq
group by status
order by tot_orders desc;
```

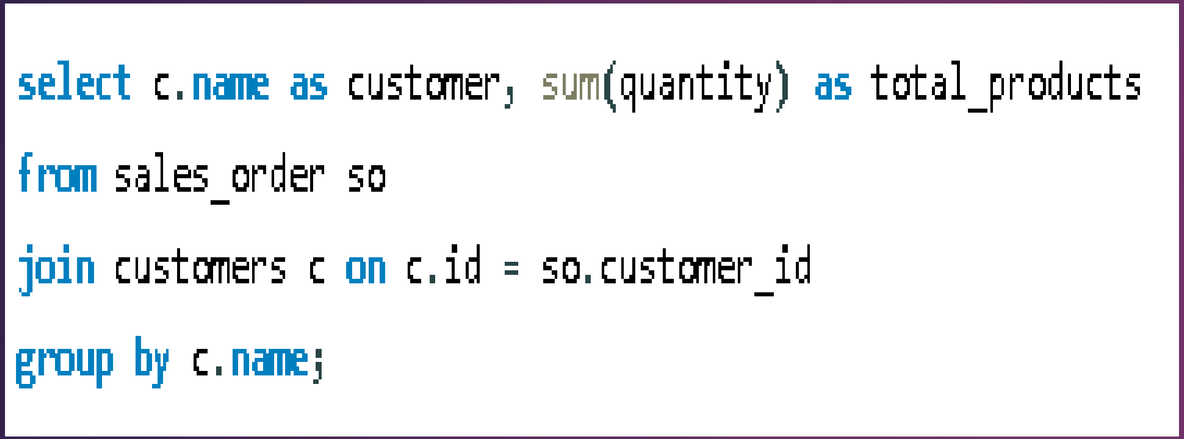


```
select upper(status) as status, count(*) as total_orders
from sales_order so
group by upper(status)
order by tot_orders desc;
```

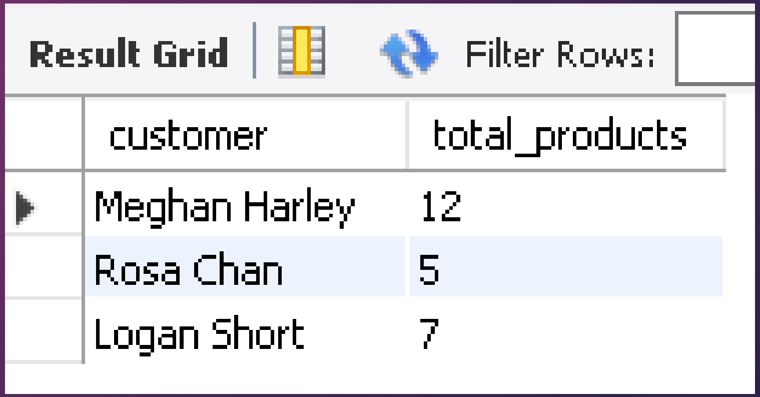


status	tot_orders
Completed	6
Pending	1
On Hold	1
Rejected	1
Cancelled	1

8. Write a query to identify the total products purchased by each customer?

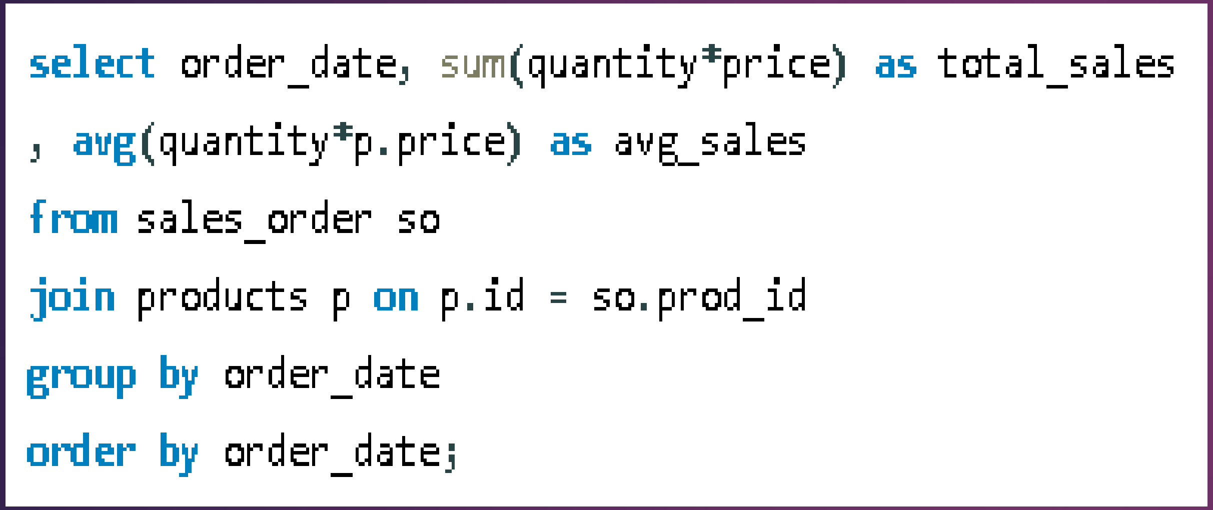


```
select c.name as customer, sum(quantity) as total_products
from sales_order so
join customers c on c.id = so.customer_id
group by c.name;
```

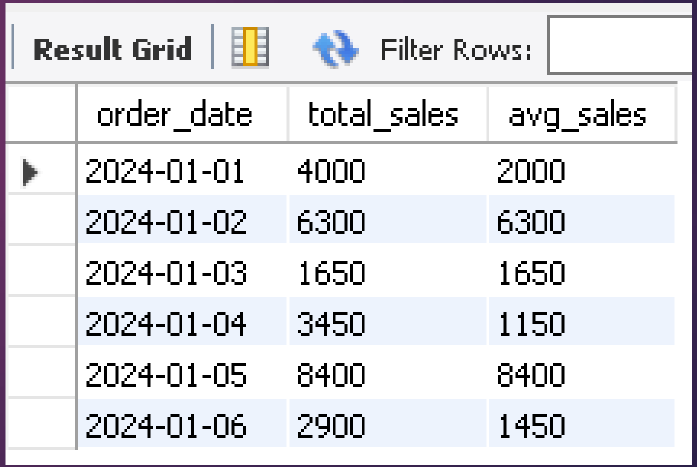


```
customer        total_products
Meghan Harley   12
Rosa Chan       5
Logan Short     7
```

9. Display the total sales and average sales done for each day?

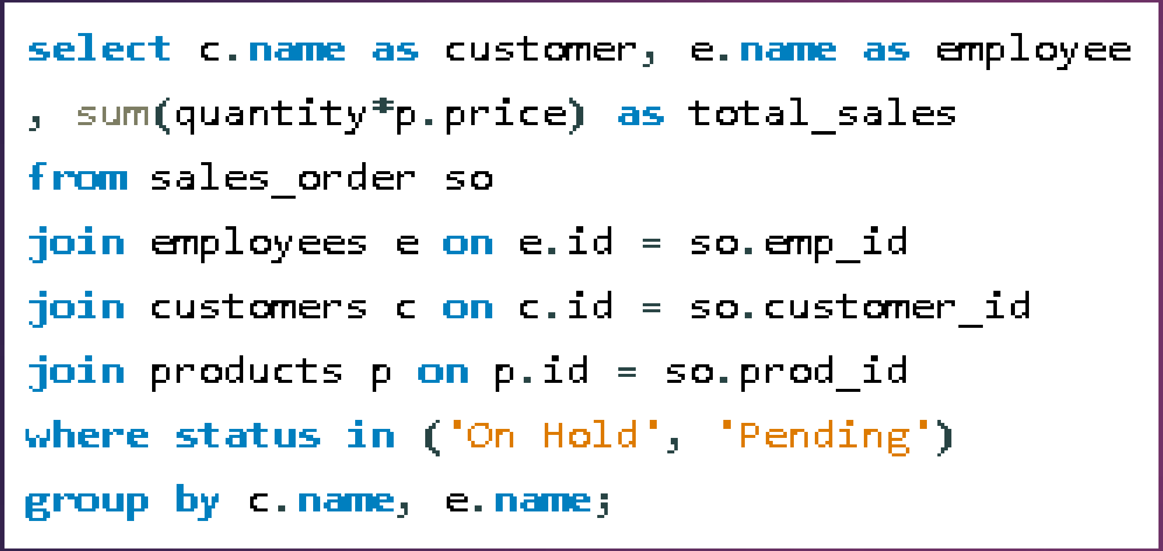


```
select order_date, sum(quantity*price) as total_sales
, avg(quantity*p.price) as avg_sales
from sales_order so
join products p on p.id = so.prod_id
group by order_date
order by order_date;
```

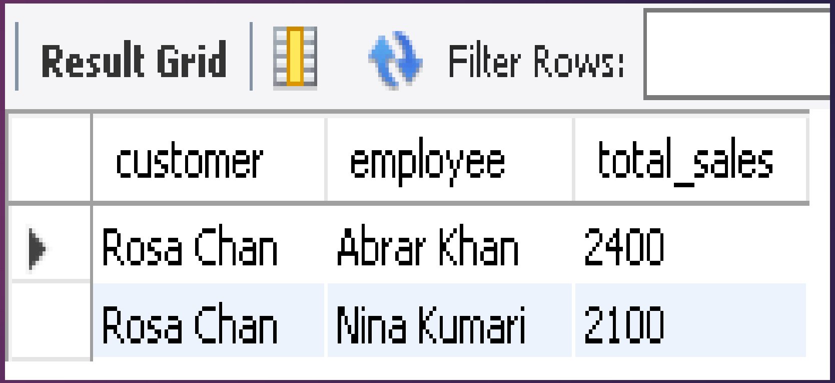


order_date	total_sales	avg_sales
2024-01-01	4000	2000
2024-01-02	6300	6300
2024-01-03	1650	1650
2024-01-04	3450	1150
2024-01-05	8400	8400
2024-01-06	2900	1450

10. Display the customer name, employee name, and total sale amount of all orders, which are either on hold or pending?



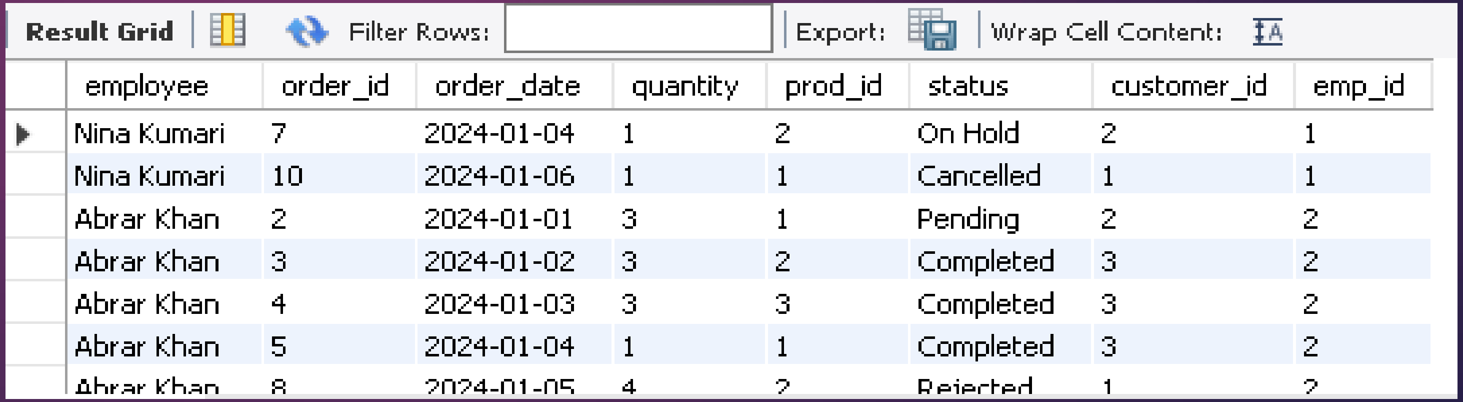
```
select c.name as customer, e.name as employee
, sum(quantity*p.price) as total_sales
from sales_order so
join employees e on e.id = so.emp_id
join customers c on c.id = so.customer_id
join products p on p.id = so.prod_id
where status in ('On Hold', 'Pending')
group by c.name, e.name;
```



customer	employee	total_sales
Rosa Chan	Abrar Khan	2400
Rosa Chan	Nina Kumari	2100

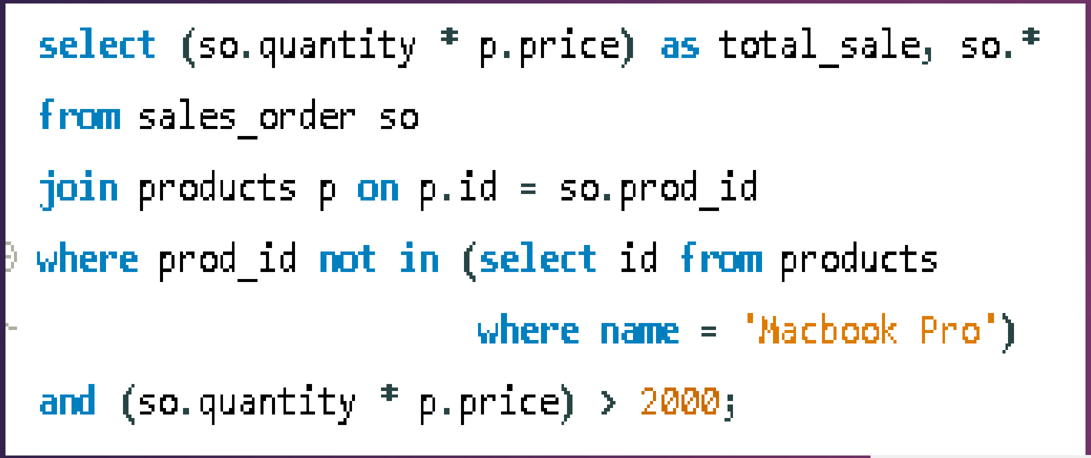
11. Fetch all the orders which were neither completed/pending or were handled by the employee Abrar. Display employee name and all details of order?

```sql
select e.name as employee, so.*
from sales_order so
join employees e on e.id = so.emp_id
where lower(status) not in ('completed', 'pending')
or lower(e.name) like '%abrar%';
```

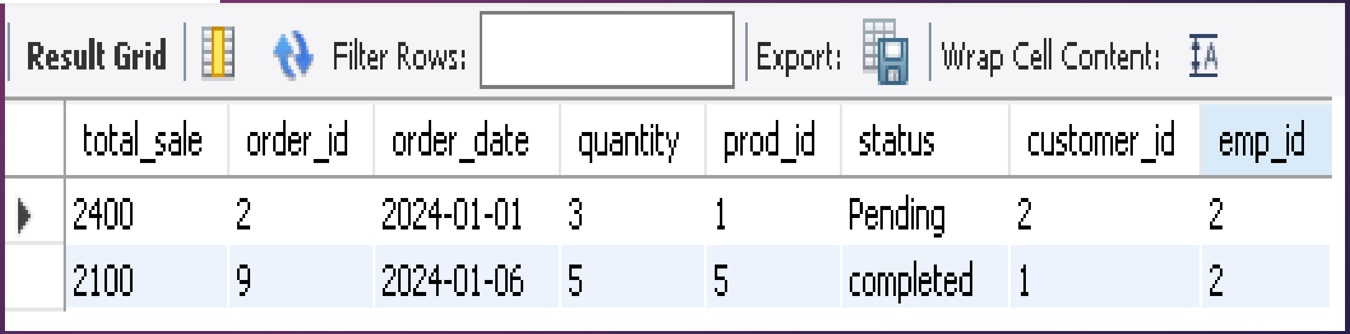


employee	order_id	order_date	quantity	prod_id	status	customer_id	emp_id
Nina Kumari	7	2024-01-04	1	2	On Hold	2	1
Nina Kumari	10	2024-01-06	1	1	Cancelled	1	1
Abrar Khan	2	2024-01-01	3	1	Pending	2	2
Abrar Khan	3	2024-01-02	3	2	Completed	3	2
Abrar Khan	4	2024-01-03	3	3	Completed	3	2
Abrar Khan	5	2024-01-04	1	1	Completed	3	2
Abrar Khan	8	2024-01-05	4	2	Rejected	1	2

12. Fetch the orders which cost more than 2000 but did not include the MacBook Pro. Print the total sale amount as well?

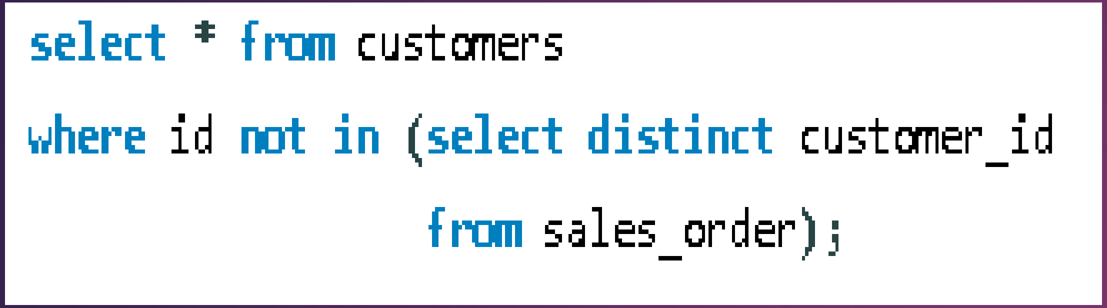


```
select (so.quantity * p.price) as total_sale, so.*
from sales_order so
join products p on p.id = so.prod_id
where prod_id not in (select id from products
                      where name = 'Macbook Pro')
and (so.quantity * p.price) > 2000;
```

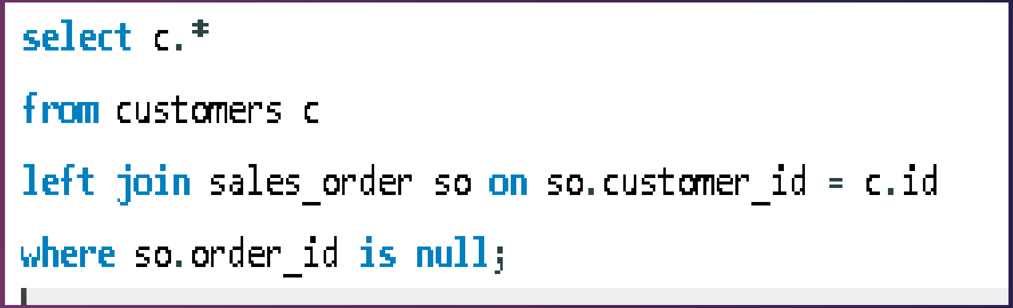


total_sale	order_id	order_date	quantity	prod_id	status	customer_id	emp_id
2400	2	2024-01-01	3	1	Pending	2	2
2100	9	2024-01-06	5	5	completed	1	2

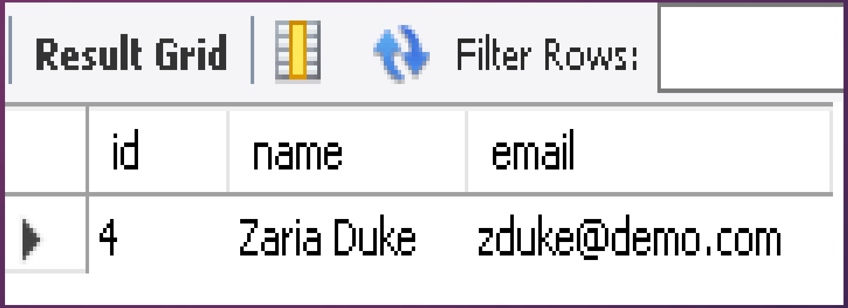
13. Identify the customers who have not purchased any product yet?



```
select * from customers
where id not in (select distinct customer_id
                 from sales_order);
```

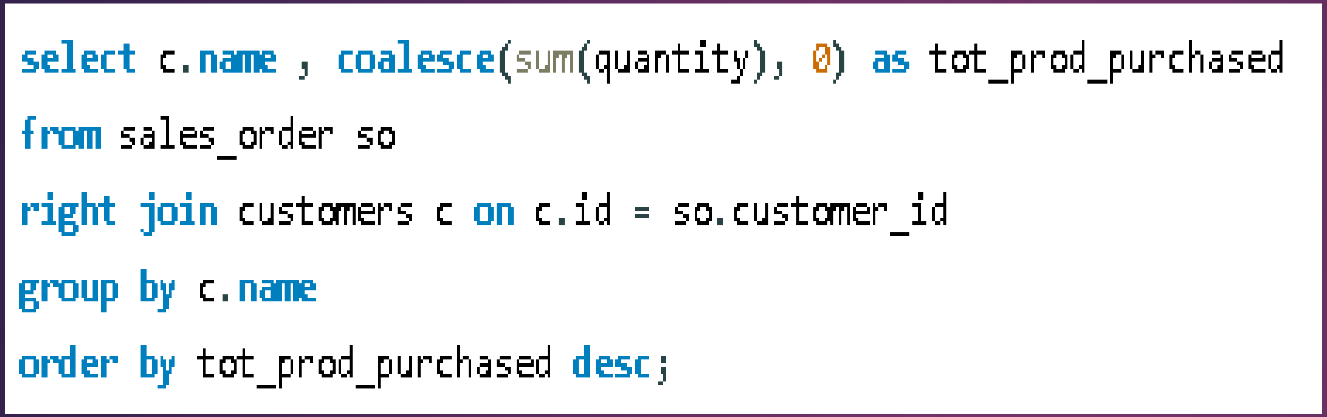


```
select c.*
from customers c
left join sales_order so on so.customer_id = c.id
where so.order_id is null;
```

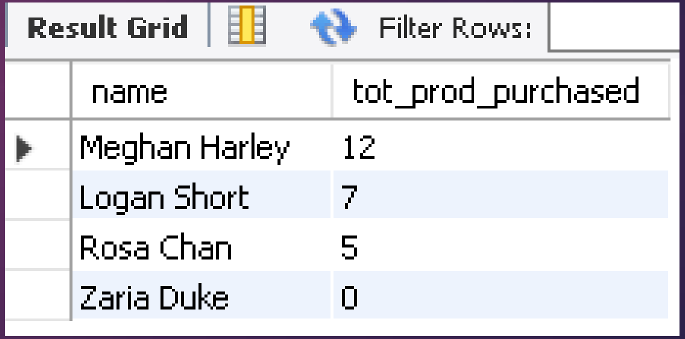


id	name	email
4	Zaria Duke	zduke@demo.com

14. Write a query to identify the total products purchased by each customer. Return all customers irrespective of whether they have made a purchase or not. Sort the result with the highest no of orders at the top?

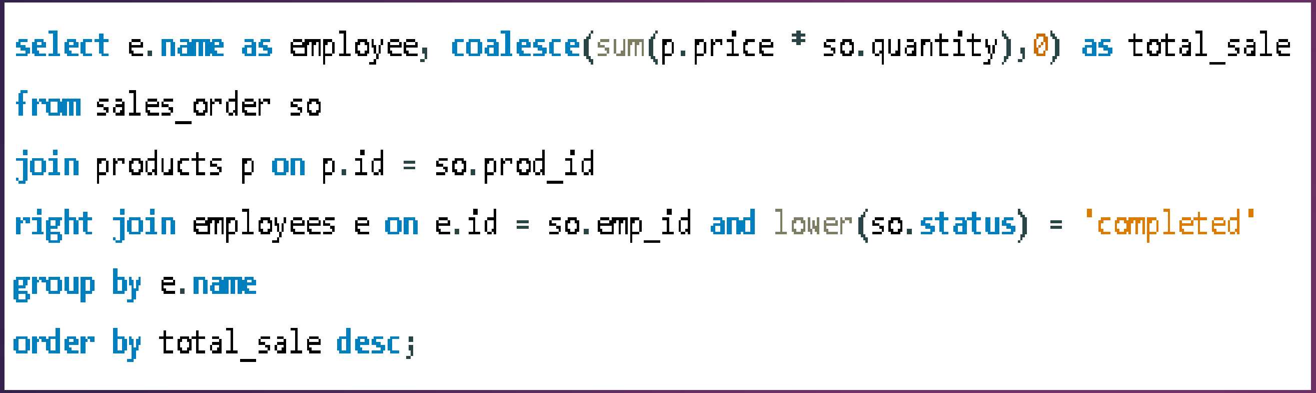


```sql
select c.name , coalesce(sum(quantity), 0) as tot_prod_purchased
from sales_order so
right join customers c on c.id = so.customer_id
group by c.name
order by tot_prod_purchased desc;
```

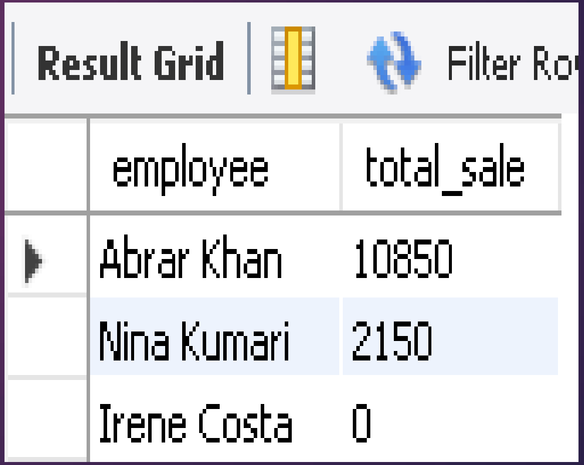


name	tot_prod_purchased
Meghan Harley	12
Logan Short	7
Rosa Chan	5
Zaria Duke	0

15. Corresponding to each employee, display the total sales they made of all the completed orders. Display total sales as 0 if an employee made no sales yet?

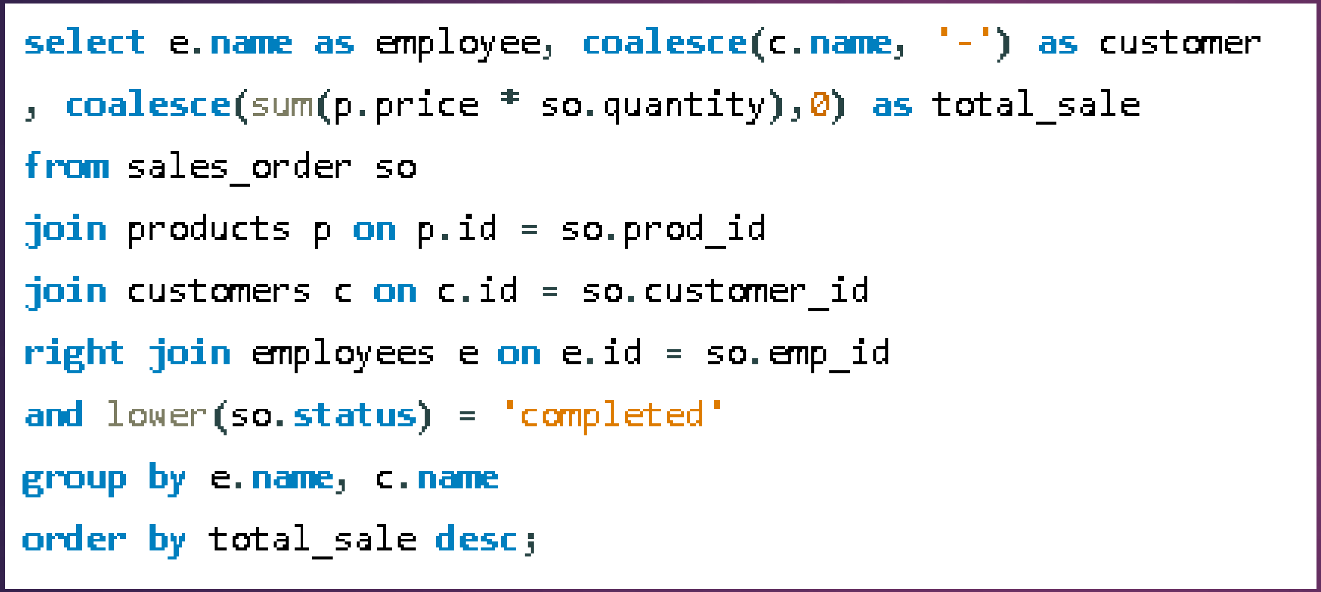


```
select e.name as employee, coalesce(sum(p.price * so.quantity),0) as total_sale
from sales_order so
join products p on p.id = so.prod_id
right join employees e on e.id = so.emp_id and lower(so.status) = 'completed'
group by e.name
order by total_sale desc;
```

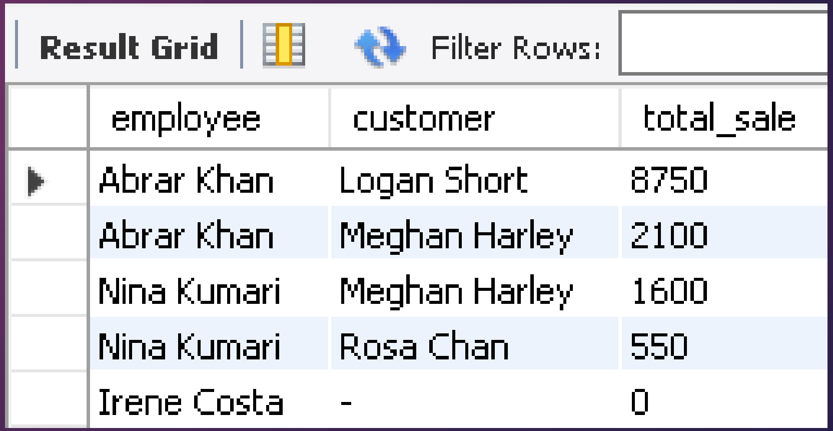


employee	total_sale
Abrar Khan	10850
Nina Kumari	2150
Irene Costa	0

16. Re-write the above query to display the total sales made by each employee corresponding to each customer. If an employee has not served a customer yet then display "-" under the customer?

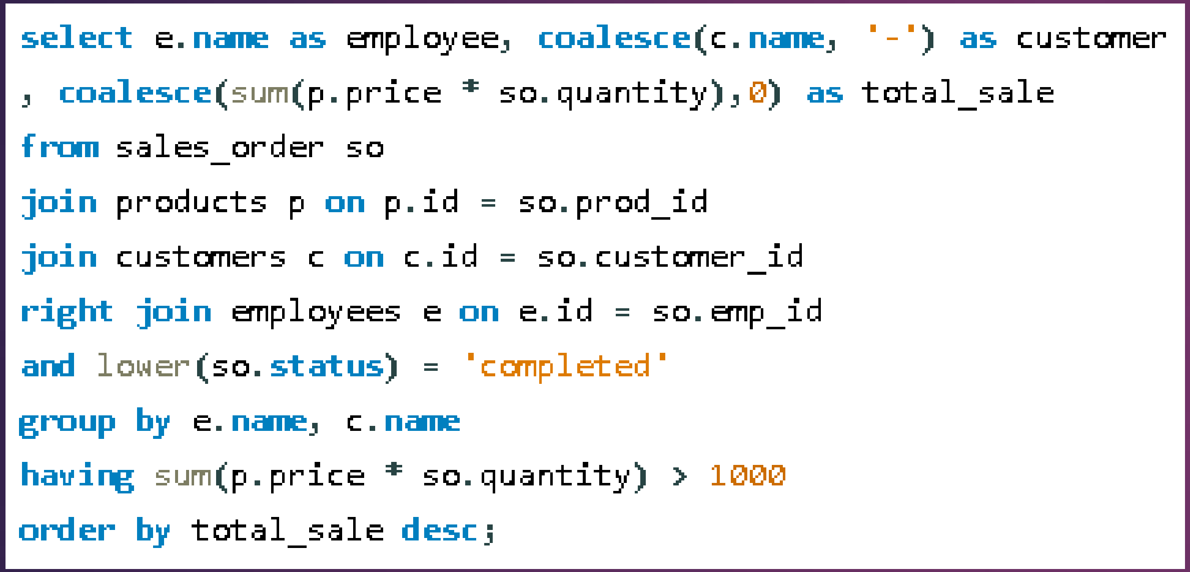


```
select e.name as employee, coalesce(c.name, '-') as customer
, coalesce(sum(p.price * so.quantity),0) as total_sale
from sales_order so
join products p on p.id = so.prod_id
join customers c on c.id = so.customer_id
right join employees e on e.id = so.emp_id
and lower(so.status) = 'completed'
group by e.name, c.name
order by total_sale desc;
```

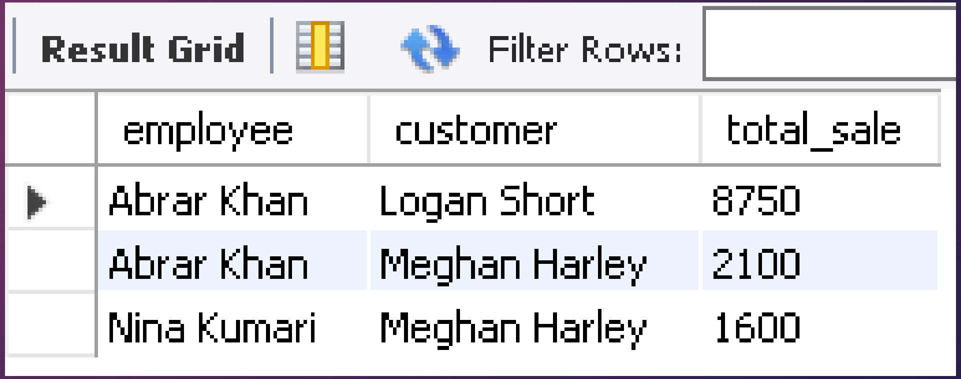


employee	customer	total_sale
Abrar Khan	Logan Short	8750
Abrar Khan	Meghan Harley	2100
Nina Kumari	Meghan Harley	1600
Nina Kumari	Rosa Chan	550
Irene Costa	-	0

17. Re-write the above query to display only those records where the total sales are above 1000?

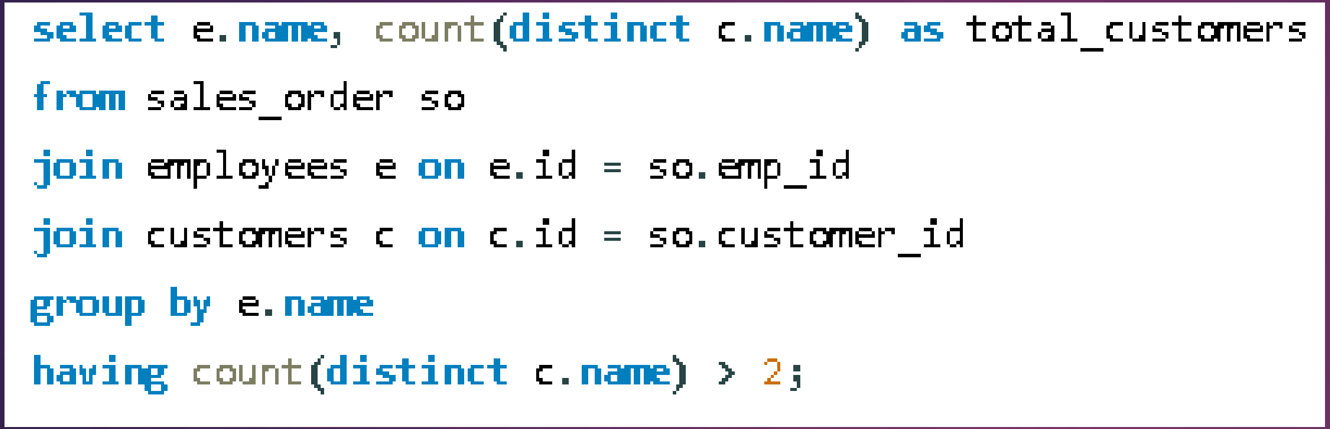


```
select e.name as employee, coalesce(c.name, '-') as customer
, coalesce(sum(p.price * so.quantity),0) as total_sale
from sales_order so
join products p on p.id = so.prod_id
join customers c on c.id = so.customer_id
right join employees e on e.id = so.emp_id
and lower(so.status) = 'completed'
group by e.name, c.name
having sum(p.price * so.quantity) > 1000
order by total_sale desc;
```

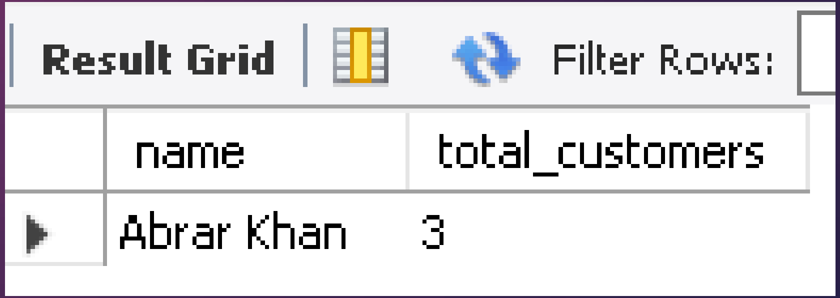


employee	customer	total_sale
Abrar Khan	Logan Short	8750
Abrar Khan	Meghan Harley	2100
Nina Kumari	Meghan Harley	1600

18. Identify employees who have served more than 2 customers?

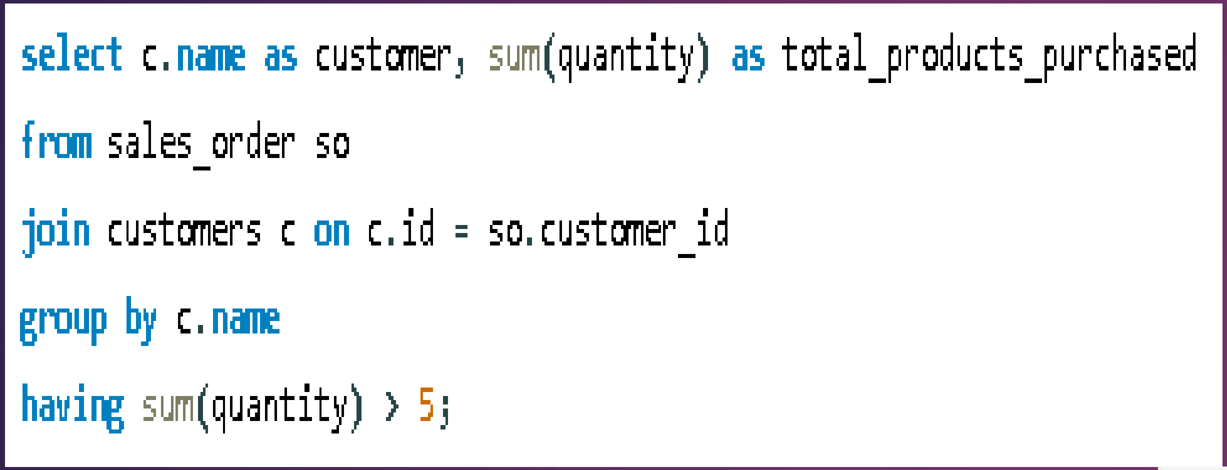


```
select e.name, count(distinct c.name) as total_customers
from sales_order so
join employees e on e.id = so.emp_id
join customers c on c.id = so.customer_id
group by e.name
having count(distinct c.name) > 2;
```

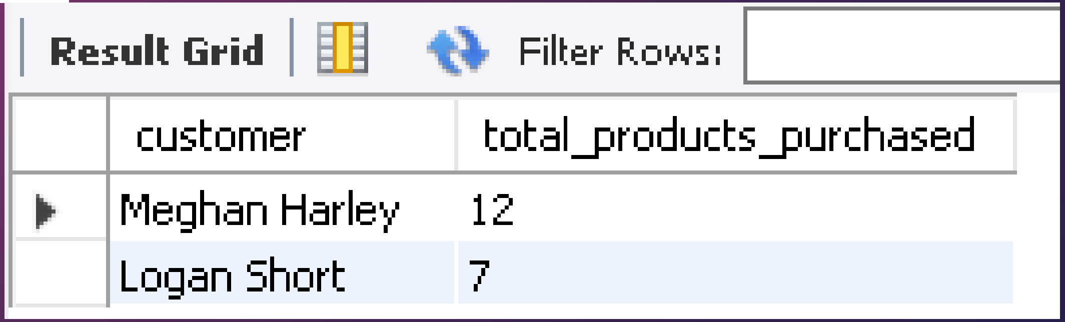


name	total_customers
Abrar Khan	3

19. Identify the customers who have purchased more than 5 products?

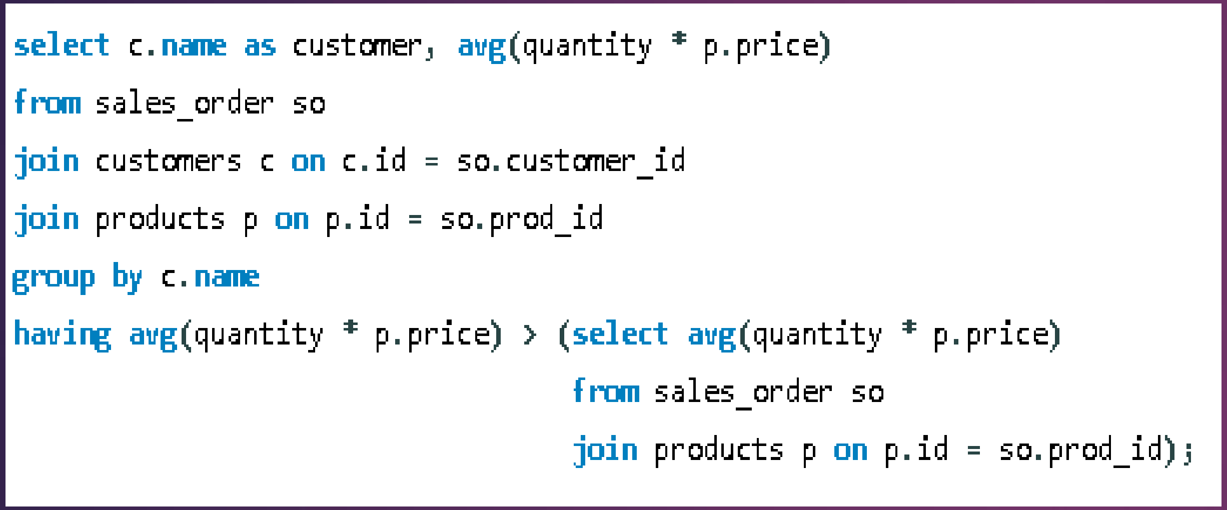


```
select c.name as customer, sum(quantity) as total_products_purchased
from sales_order so
join customers c on c.id = so.customer_id
group by c.name
having sum(quantity) > 5;
```

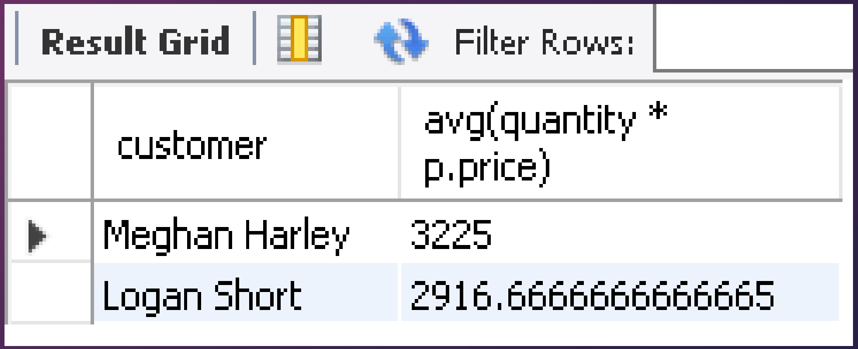


customer	total_products_purchased
Meghan Harley	12
Logan Short	7

20. Identify customers whose average purchase cost exceeds the average sale of all the orders?



```sql
select c.name as customer, avg(quantity * p.price)
from sales_order so
join customers c on c.id = so.customer_id
join products p on p.id = so.prod_id
group by c.name
having avg(quantity * p.price) > (select avg(quantity * p.price)
                                  from sales_order so
                                  join products p on p.id = so.prod_id);
```



customer	avg(quantity * p.price)
Meghan Harley	3225
Logan Short	2916.666666666665

Key Insights:

Total Sales and Product Performance

Identified the total number of products sold.

Analyzed daily sales trends, including total and average sales.

Order Status and Delivery Trends

Extracted all unique order statuses apart from "Completed."

Counted the total number of orders under each delivery status.

Ranked statuses based on the highest number of orders.

Customer Purchase Behavior

Identified customers who made the most purchases.

Listed customers who have not made any purchases.

Found customers whose average purchase cost exceeds the overall average sale.

Employee Performance Metrics

Displayed total sales made by each employee.

Identified employees who served more than two customers.

Highlighted employees with total sales above 1000.

Product and Order Analysis

Filtered orders that exceeded a total cost of 2000 but did not include MacBook Pro.

Displayed orders containing more than one item that were still not completed.

Thanks

Gaurav

9711407364

PANWARGAURAV813@GMAIL.COM