**Day 7**

1. Rank employees by their total sales

(Total sales = Total no of orders handled, JOIN employees and orders table)

Query:

select e.employee\_id, e.first\_name || '\_' || e.last\_name AS employee\_name,

Count(o.order\_id) as total\_orders,

Rank() Over (Order by count(o.order\_id) desc) as sales\_rank

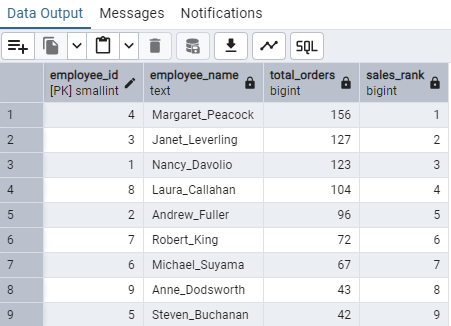
from employees e

join orders o ON e.employee\_id = o.employee\_id

group by e.employee\_id, e.first\_name, e.last\_name

order by sales\_rank

Output:



2. Compare current order's freight with previous and next order for each customer.

(Display order\_id, customer\_id, order\_date, freight,

Use lead(freight) and lag(freight).

Query:

select o.order\_id,

c.customer\_id,

o.order\_date,

o.freight,

lag(o.freight)over(partition by c.customer\_id order by o.order\_date) as previous\_order,

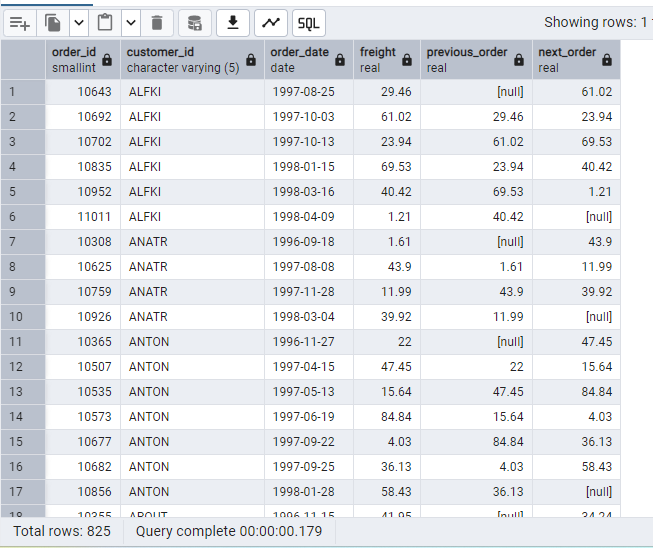
lead(o.freight) over(partition by c.customer\_id order by o.order\_date) as next\_order

from customers c

join orders o ON c.customer\_id = o.customer\_id

order by c.customer\_id,o.order\_id

Output:



3. Show products and their price categories, product count in each category, avg price:

(HINT:Create a CTE which should have price\_category definition:

WHEN unit\_price < 20 THEN 'Low Price'

WHEN unit\_price < 50 THEN 'Medium Price'

ELSE 'High Price'

· In the main query display: price\_category, product\_count in each price\_category, ROUND(AVG(unit\_price)::numeric, 2) as avg\_price)

Query:

with price\_category\_cte as(

select product\_id,product\_name,unit\_price,

case

when unit\_price< 20 then 'LOW Price'

when unit\_price < 50 THEN 'Medium Price'

Else 'High Price' end as Price\_category

from products

)

--main query--

select price\_category, count(product\_id) as product\_count, ROUND(AVG(unit\_price)::numeric, 2) AS avg\_price

from price\_category\_cte

group by price\_category

order by price\_category

Output:

