Day-7

---1.Rank employees by their total sales

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

select e.employee\_id,

e.first\_name,

e.last\_name,

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

rank() over(order by count(o.order\_id) desc) as sales\_rank

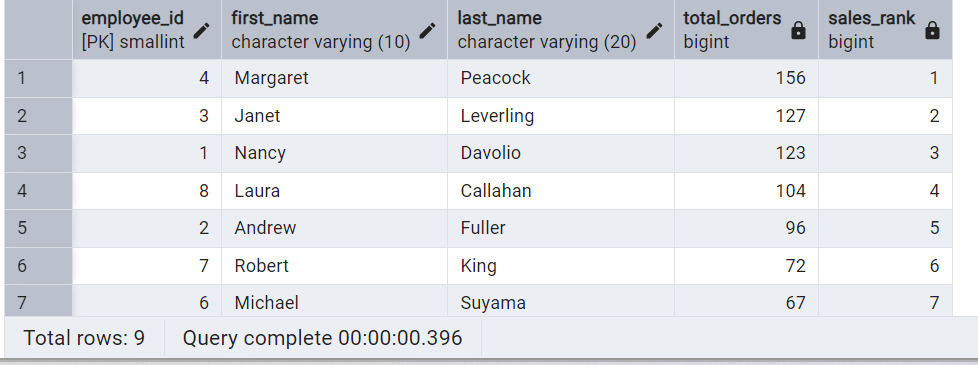
from employees e

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

GROUP BY

e.employee\_id, e.first\_name, e.last\_name;

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).

SELECT

order\_id,

customer\_id,

order\_date,

freight,

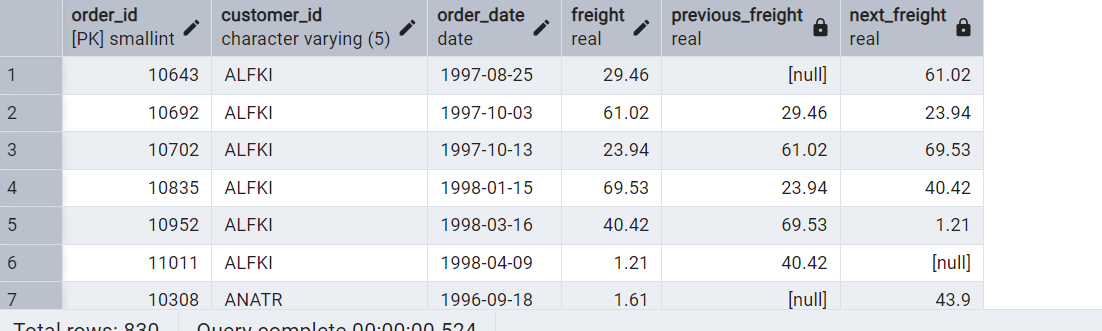
LAG(freight) OVER (PARTITION BY customer\_id ORDER BY order\_date) AS previous\_freight,

LEAD(freight) OVER (PARTITION BY customer\_id ORDER BY order\_date) AS next\_freight

FROM

orders;

Output:



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)

WITH price\_cte AS (

SELECT

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

)

SELECT

price\_category,

COUNT(\*) AS product\_count,

ROUND(AVG(unit\_price)::numeric, 2) AS avg\_price

FROM

price\_cte

GROUP BY

price\_category

ORDER BY

price\_category;

output:

