**Day-7 Assignment**

1.Rank employees by their total sales (Total sales = Total no of orders handled, JOIN employees and orders table).

**QUERY:1**

select \* from employees

select \* from orders

SELECT

e.employee\_id,

e.first\_name || ' ' || e.last\_name AS employee\_name,

COUNT(o.order\_id) AS total\_sales,

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

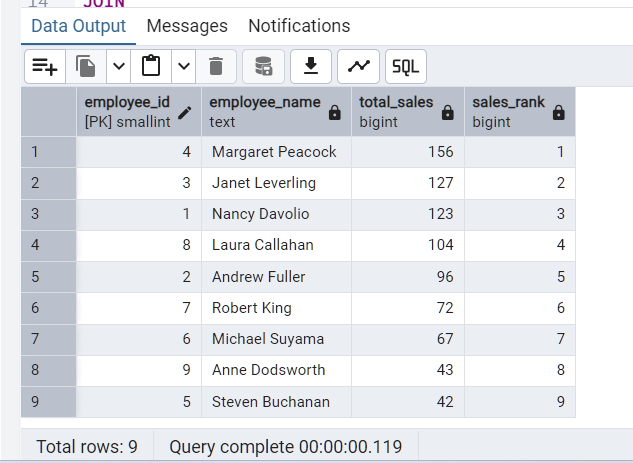
GROUP BY

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

ORDER BY

total\_sales DESC;

**OUTPUT:1**

****

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:2**

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

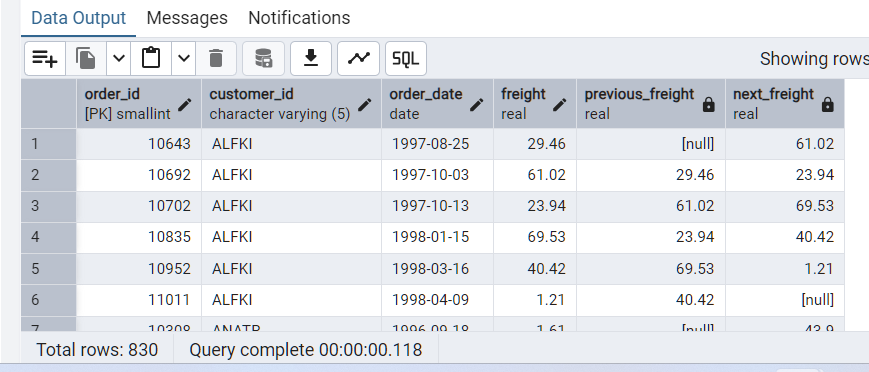
orders

order by

customer\_id,

order\_date;

**OUTPUT:2**

****

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:3**

-- Create cte

WITH price\_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(\*) AS product\_count,

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

FROM

price\_cte

GROUP BY

price\_category

ORDER BY

price\_category;

**OUTPUT:3**

