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

RANK() OVER(ORDER BY COUNT(o.order\_id) DESC) AS sales\_rank

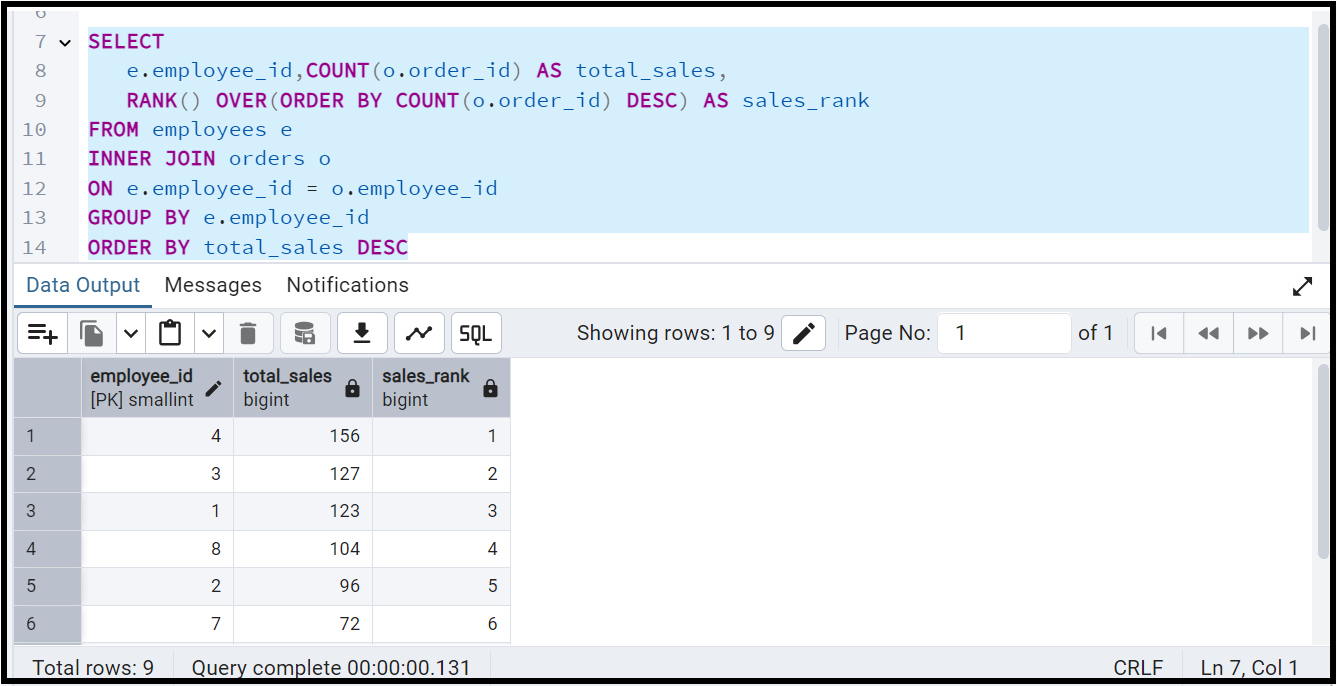
FROM employees e

INNER JOIN orders o

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

GROUP BY e.employee\_id

ORDER BY total\_sales DESC



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

order\_id,customer\_id,order\_date,freight,

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

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

FROM orders;



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 CTE1 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

)

SELECT price\_category,

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

Count(product\_id) AS product\_count

from CTE1

GROUP BY price\_category

ORDER BY avg\_price;

