1.Rank employees by their total sales

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

SELECT e.first\_name||''||e.last\_name AS Full\_name,

count(o.order\_id) AS Total\_sales,

RANK() OVER(

ORDER BY count(o.order\_id) DESC

)As sales\_rank

FROM orders o

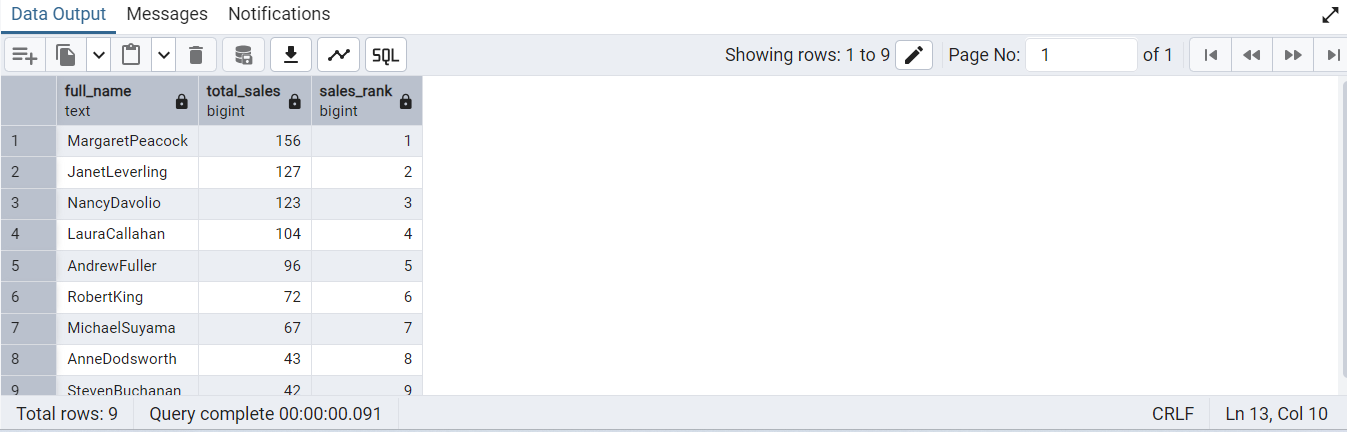
INNER JOIN

employees e

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

GROUP BY

Full\_name



==================================================

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

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

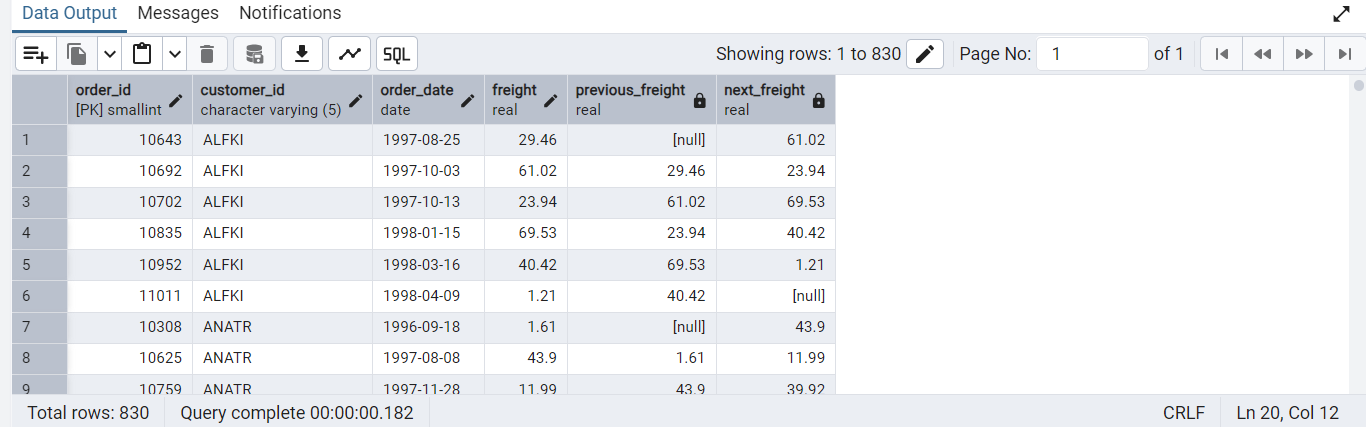
Use lead(freight) and lag(freight).

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



==================================================

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)

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

)

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;

