Day7\_PavithraMani\_SDET177

-- 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\_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,

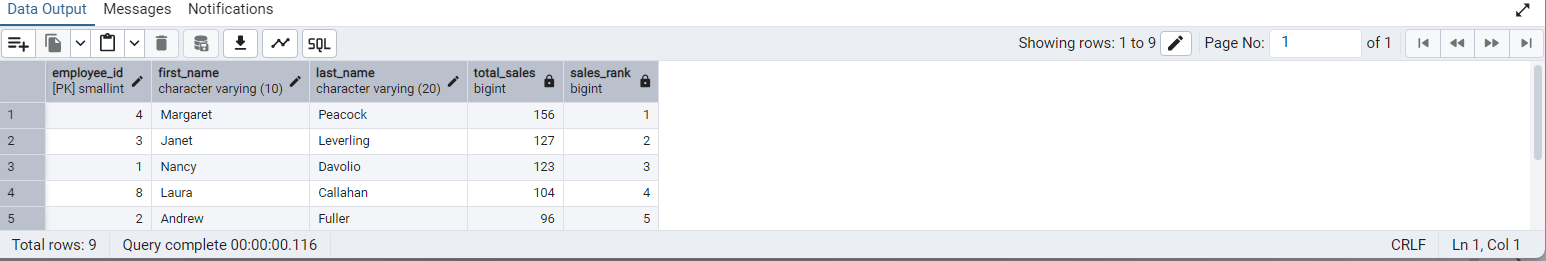
E.FIRST\_NAME,

E.LAST\_NAME

ORDER BY

TOTAL\_SALES DESC;

### OUTPUT



-- 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

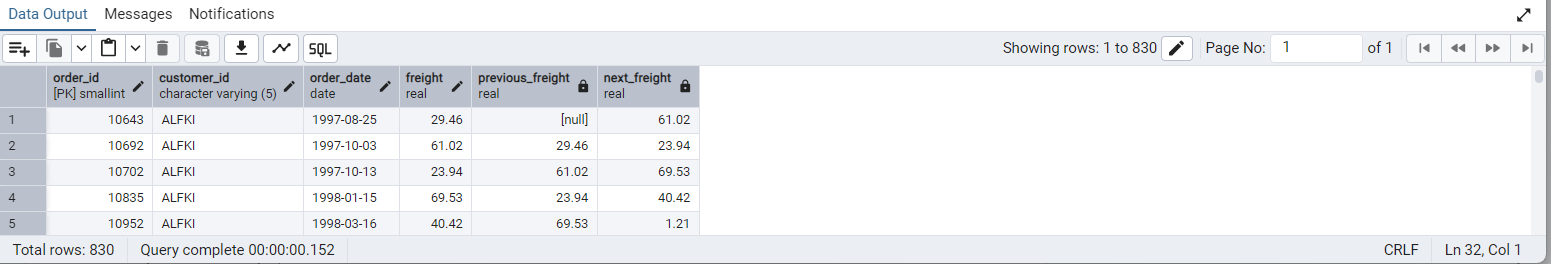
ORDERS

ORDER BY

CUSTOMER\_ID,

ORDER\_DATE;

### 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)

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;

### OUTPUT

