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 as employeename,

count (o.order\_id) as ordersplaced,

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

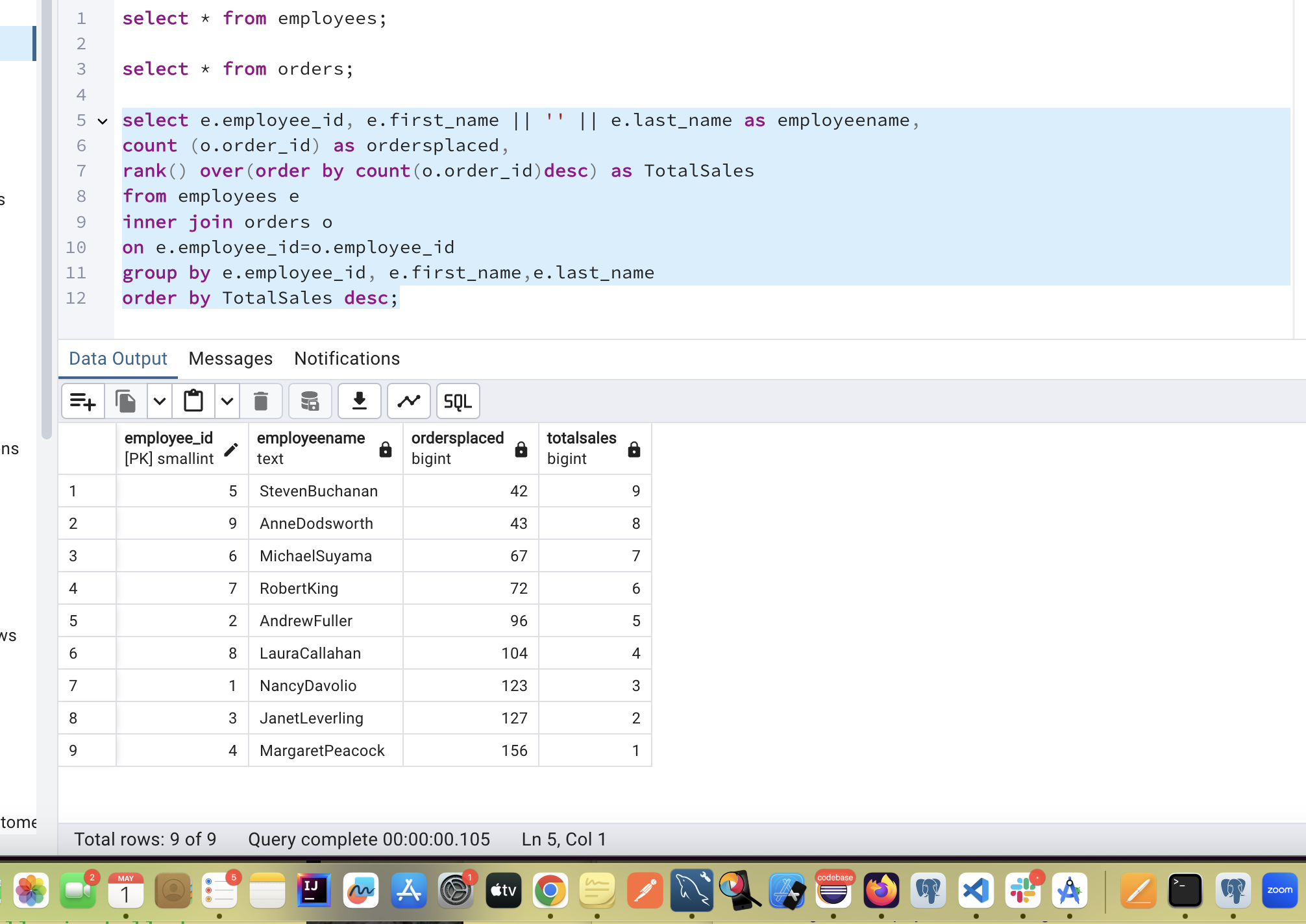
from employees e

inner join orders o

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

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

order by TotalSales 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).

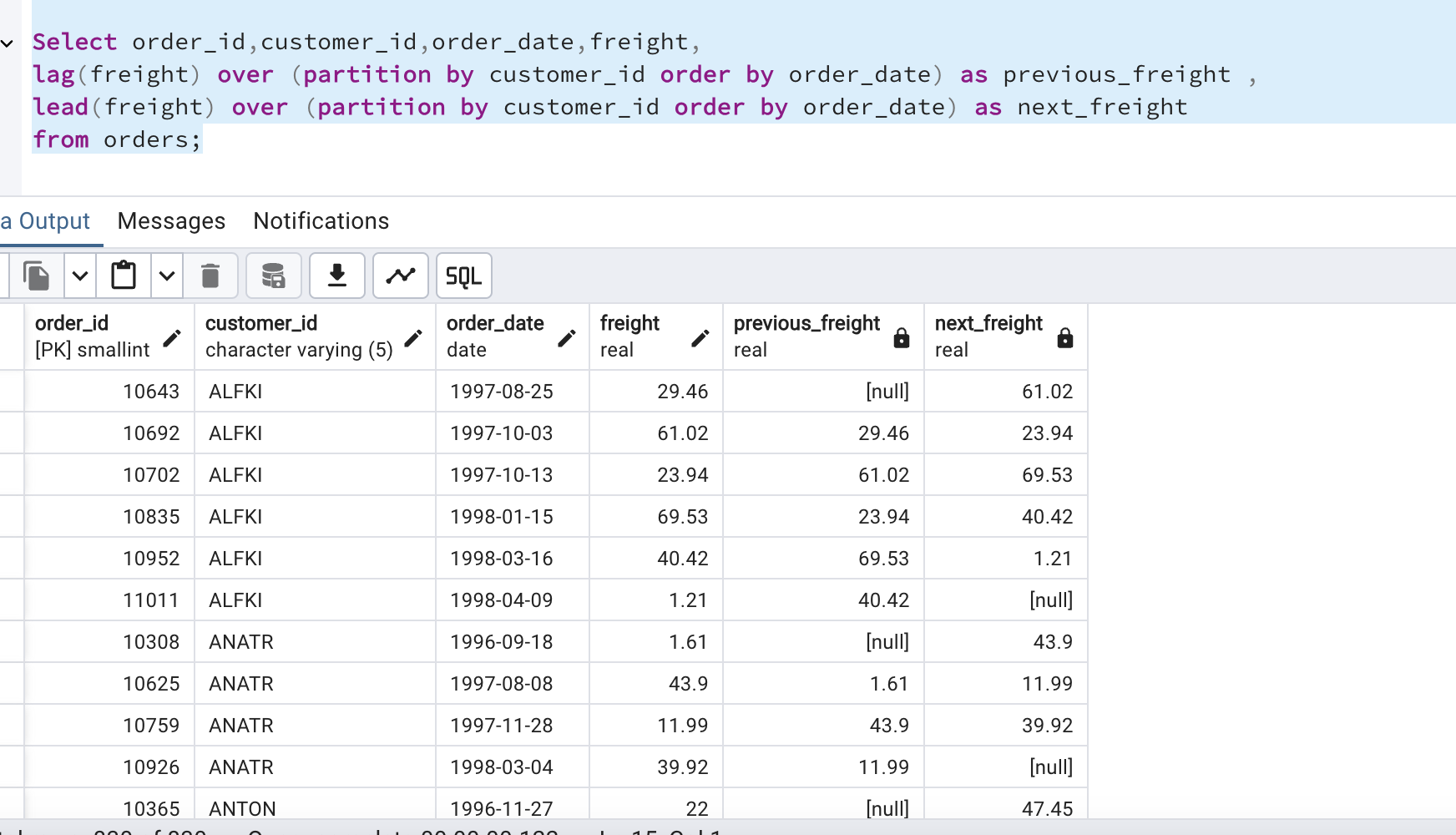
Select order\_id,customer\_id,order\_date,freight,

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:

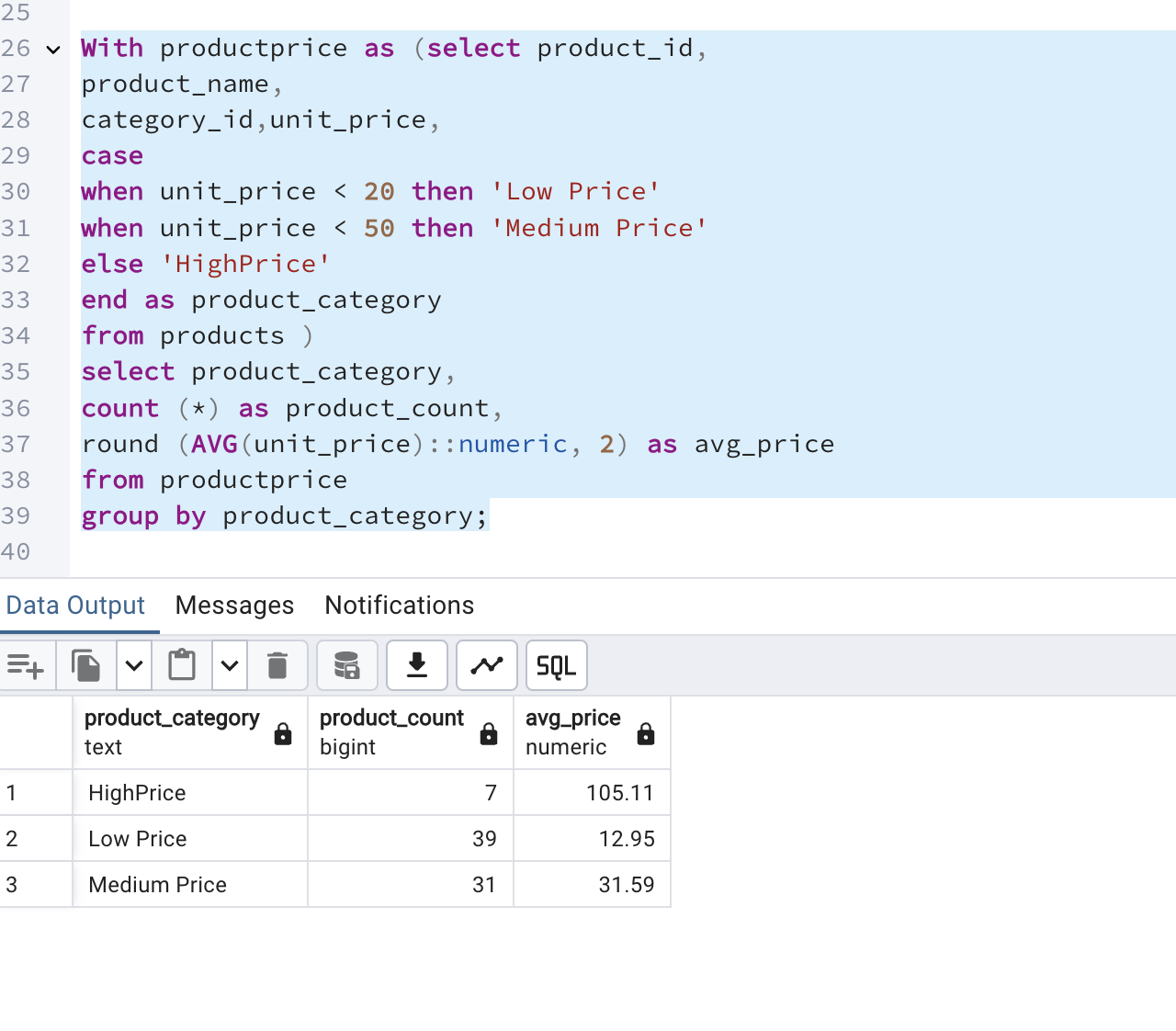
· Create a CTE which should have price\_category definition:

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 productprice as (select product\_id,

product\_name,

category\_id,unit\_price,

case

when unit\_price < 20 then 'Low Price'

when unit\_price < 50 then 'Medium Price'

else 'HighPrice'

end as product\_category

from products )

select product\_category,

count (\*) as product\_count,

round (AVG(unit\_price)::numeric, 2) as avg\_price

from productprice

group by product\_category;