



You're working for a company that sells motorcycle parts, and they've asked for some help in analyzing their sales data!

They operate three warehouses in the area, selling both retail and wholesale. They offer a variety of parts and accept credit cards, cash, and bank transfer as payment methods. However, each payment type incurs a different fee.

The board of directors wants to gain a better understanding of wholesale revenue by product line, and how this varies month-to-month and across warehouses. You have been tasked with calculating net revenue for each product line and grouping results by month and warehouse. The results should be filtered so that only "Wholesale" orders are included.

They have provided you with access to their database, which contains the following table called `sales`:

Sales

Column	Data type	Description
<code>order_number</code>	VARCHAR	Unique order number.
<code>date</code>	DATE	Date of the order, from June to August 2021.
<code>warehouse</code>	VARCHAR	The warehouse that the order was made from— <code>North</code> , <code>Central</code> , or <code>West</code> .
<code>client_type</code>	VARCHAR	Whether the order was <code>Retail</code> or <code>Wholesale</code> .
<code>product_line</code>	VARCHAR	Type of product ordered.
<code>quantity</code>	INT	Number of products ordered.
<code>unit_price</code>	FLOAT	Price per product (dollars).
<code>total</code>	FLOAT	Total price of the order (dollars).
<code>payment</code>	VARCHAR	Payment method— <code>Credit card</code> , <code>Transfer</code> , or <code>Cash</code> .
<code>payment_fee</code>	FLOAT	Percentage of <code>total</code> charged as a result of the <code>payment</code> method.

Your query output should be presented in the following format:

product_line	month	warehouse	net_revenue
product_one	---	---	---
product_one	---	---	---
product_one	---	---	---
product_one	---	---	---
product_one	---	---	---
product_one	---	---	---
product_two	---	---	---
...



```
-- Start coding here
SELECT product_line,
    CASE WHEN EXTRACT('month' from date) = 6 THEN 'June'
        WHEN EXTRACT('month' from date) = 7 THEN 'July'
        WHEN EXTRACT('month' from date) = 8 THEN 'August'
    END as month,
    warehouse,
    SUM(total) - SUM(payment_fee) AS net_revenue
FROM sales
WHERE client_type = 'Wholesale'
GROUP BY product_line, warehouse, month
ORDER BY product_line, month, net_revenue DESC
```

i...	...	product_line	month	wareho...	net_reve...	↑↓
0		Braking system			August			Central			3039.41			▲
1		Braking system			August			West			2500.67			
2		Braking system			August			North			1770.84			
3		Braking system			July			Central			3778.65			
4		Braking system			July			West			3060.93			
5		Braking system			July			North			2594.44			
6		Braking system			June			Central			3684.89			
7		Braking system			June			North			1487.77			
8		Braking system			June			West			1212.75			
9		Electrical system			August			North			4721.12			
10		Electrical system			August			Central			3126.43			
11		Electrical system			August			West			1241.84			
12		Electrical system			July			Central			5577.62			
13		Electrical system			July			North			1710.13			
14		Electrical system			July			West			449.46			
15		Electrical system			June			Central			2904.93			▼

Rows: 48

Expand

Projects Data DataFrame as revenue_by_product_line

```
WITH month_etc AS (
    SELECT order_number, date_part('month', date) AS month_start
    FROM sales
)
SELECT product_line,
    CASE WHEN month_start = 6 THEN 'June'
        WHEN month_start = 7 THEN 'July'
        ELSE 'August' END AS month,
    warehouse,
    SUM(total_payment_fee) AS net_revenue
FROM sales s LEFT JOIN month_etc m ON s.order_number = m.order_number
WHERE client_type = 'Wholesale'
GROUP BY product_line, month, warehouse
ORDER BY product_line ASC, month DESC, net_revenue DESC;
```

in...	...	Ξ↑	product_li...	...	↑↓	month	...	↑↓	wareh...	...	↑↓	net_rev...	...	↑↓	
	0		Braking system			June			Central			3684.89			▲
	1		Braking system			June			North			1487.77			
	2		Braking system			June			West			1212.75			
	3		Braking system			July			Central			3778.65			
	4		Braking system			July			West			3060.93			
	5		Braking system			July			North			2594.44			
	6		Braking system			August			Central			3039.41			
	7		Braking system			August			West			2500.67			
	8		Braking system			August			North			1770.84			
	9		Electrical system			June			Central			2904.93			
	10		Electrical system			June			North			2022.5			
	11		Electrical system			July			Central			5577.62			
	12		Electrical system			July			North			1710.13			
	13		Electrical system			July			West			449.46			
	14		Electrical system			August			North			4721.12			
	15		Electrical system			August			Central			3126.43			▼

Rows: 48

↗ Expand

Extended Project below

The finance team is exploring ways to reduce transaction costs and improve profitability. They've asked you to determine the most profitable payment method for each warehouse in each month. Calculate the net revenue for each payment method, grouped by warehouse and month, and identify the top payment method for each combination.

 Projects Data DataFrame as `top_payment_by_warehouse_month`

```
WITH payment_summary AS (
  SELECT payment,
    CASE
      WHEN EXTRACT('month' FROM date) = 6 THEN 'June'
      WHEN EXTRACT('month' FROM date) = 7 THEN 'July'
      ELSE 'August' END AS month,
    warehouse,
    SUM(total-payment_fee) AS net_revenue
  FROM sales
  GROUP BY payment, month, warehouse)
```

```
( SELECT * FROM payment_summary
  WHERE month = 'June' AND warehouse = 'Central'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'June' AND warehouse = 'North'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'June' AND warehouse = 'West'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'July' AND warehouse = 'Central'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'July' AND warehouse = 'North'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'August' AND warehouse = 'Central'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'August' AND warehouse = 'North'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'August' AND warehouse = 'West'
  ORDER BY net_revenue DESC
  LIMIT 1
)
```

```
( SELECT * FROM payment_summary
  WHERE month = 'July' AND warehouse = 'West'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'August' AND warehouse = 'Central'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'August' AND warehouse = 'North'
  ORDER BY net_revenue DESC
  LIMIT 1
)
UNION
(
  SELECT * FROM payment_summary
  WHERE month = 'August' AND warehouse = 'West'
  ORDER BY net_revenue DESC
  LIMIT 1
)
ORDER BY month DESC, warehouse
```

in...	...	↑↓	payment	...	↑↓	month	...	↑↓	warehouse	...	↑↓	net_reve...	...
0	Transfer			June		Central			23453				
1	Transfer			June		North			17000				
2	Transfer			June		West			8645				
3	Transfer			July		Central			23893				
4	Transfer			July		North			17585				
5	Transfer			July		West			7606				
6	Transfer			August		Central			315				
7	Transfer			August		North			23480				
8	Transfer			August		West			6466				

Rows: 9

Expand



Projects Data DataFrame as net_revenue_by_warehouse_month

```

SELECT payment,
CASE
WHEN EXTRACT('month' FROM date) = 6 THEN 'June'
WHEN EXTRACT('month' FROM date) = 7 THEN 'July'
ELSE 'August' END AS month,
warehouse,
SUM(total-payment_fee) AS net_revenue
FROM sales
GROUP BY payment, month, warehouse
ORDER BY month DESC, warehouse, net_revenue DESC
    
```

i..	...	↑↓	payment	...	↑↓	month	...	↑↓	warehouse	...	↑↓	net_reve...	...	↑↓
0	Transfer			June		Central			23453					
1	Credit card			June		Central			18888					
2	Cash			June		Central			1784					
3	Transfer			June		North			17000					
4	Credit card			June		North			13180					
5	Cash			June		North			3135					
6	Transfer			June		West			8645					
7	Credit card			June		West			6924					
8	Cash			June		West			230					
9	Transfer			July		Central			23893					
10	Credit card			July		Central			21241					
11	Cash			July		Central			3130					
12	Transfer			July		North			17585					
13	Credit card			July		North			10338					
14	Cash			July		North			1196					
15	Transfer			July		West			7606					

Rows: 27

Expand

The marketing team is planning a targeted campaign and wants to know the most popular product lines for retail and wholesale customers.

They have given you the task to find the top 3 most ordered product lines for each client type.

```
Projects Data DataFrame as top_3_most_ordered_product_line:  
(  
    SELECT client_type, product_line, COUNT(order_number) AS total_order  
    FROM sales  
    WHERE client_type = 'Retail'  
    GROUP BY client_type, product_line  
    ORDER BY client_type, total_order DESC  
    LIMIT 3  
)  
UNION  
(  
    SELECT client_type, product_line, COUNT(order_number) AS total_order  
    FROM sales  
    WHERE client_type = 'Wholesale'  
    GROUP BY client_type, product_line  
    ORDER BY client_type, total_order DESC  
    LIMIT 3  
)  
ORDER BY client_type, total_order DESC
```

in...	...	↑↓	client_type	...	↑↓	product_line	...	↑↓	total_or...	...	↑↓
	0		Retail			Suspension & traction			177		
	1		Retail			Braking system			175		
	2		Retail			Electrical system			155		
	3		Wholesale			Braking system			55		
	4		Wholesale			Suspension & traction			51		
	5		Wholesale			Electrical system			38		

Rows: 6

 Expand



Projects Data DataFrame as order_quanity_by_product_line_c

```
SELECT client_type, product_line, COUNT(order_number) AS total_order
FROM sales
GROUP BY client_type, product_line
ORDER BY client_type, total_order DESC
```

i...	...	Ξ↑	client_type	...	↑↓	product_line	...	↑↓	total_order	...	↑↓
0	Retail		Suspension & traction						177		
1	Retail		Braking system						175		
2	Retail		Electrical system						155		
3	Retail		Frame & body						128		
4	Retail		Miscellaneous						92		
5	Retail		Engine						48		
6	Wholesale		Braking system						55		
7	Wholesale		Suspension & traction						51		
8	Wholesale		Frame & body						38		
9	Wholesale		Electrical system						38		
10	Wholesale		Miscellaneous						30		
11	Wholesale		Engine						13		

Rows: 12

Expand