

Welcome! You are now in DataLab.

You successfully completed your project and are looking for some additional related challenges. This DataLab workbook contains the official solution from our curriculum staff, along with Additional Challenges at the bottom. If you would like a quick overview of DataLab, please refer to the help menu. You can easily share your project with your friends and colleagues when you're done.

Good luck with your additional challenges!

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— North, Central, or West .
<code>client_type</code>	VARCHAR	Whether the order was Retail or Wholesale .
<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— Credit card , Transfer , or Cash .
<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	---	---	---
...

Projects Data DataFrame as DataCamp-demo_revenue_by_product

```
-- 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
```

index	product_line	month	warehouse
0	Braking system	August	Central
1	Braking system	August	Western
2	Braking system	August	Northern
3	Braking system	July	Central
4	Braking system	July	Western
5	Braking system	July	Northern
6	Braking system	June	Central
7	Braking system	June	Northern
8	Braking system	June	Western
9	Electrical system	August	Northern
10	Electrical system	August	Central
11	Electrical system	August	Western
12	Electrical system	July	Central
13	Electrical system	July	Northern
14	Electrical system	July	Western
15	Electrical system	June	Central

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;
```

index	product_line	month	warehouse
0	Braking system	June	Central
1	Braking system	June	North
2	Braking system	June	West
3	Braking system	July	Central
4	Braking system	July	West
5	Braking system	July	North
6	Braking system	August	Central
7	Braking system	August	West
8	Braking system	August	North
9	Electrical system	June	Central
10	Electrical system	June	North
11	Electrical system	July	Central
12	Electrical system	July	North
13	Electrical system	July	West
14	Electrical system	August	North
15	Electrical system	August	Central

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 t

```
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 = '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

...	↑↓	...	↑↓	...	↑↓	w	...	↑↓	net...	...	↑↓
0	Transfer	June		Central			23453.08				
1	Transfer	June		North			17000.12				
2	Transfer	June		West			8645.98				
3	Transfer	July		Central			23893.59				
4	Transfer	July		North			17585.25				
5	Transfer	July		West			7606.51				
6	Transfer	August		Central			31509				
7	Transfer	August		North			23480.13				
8	Transfer	August		West			6466.42				

Rows: 9

Expand

Projects Data DataFrame as n

```
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
```

...	↑↓	pa...	...	↑↓	...	↑↓	w	...	↑↓	net...	...	↑↓
0	Transfer	June		Central			23453.08					
1	Credit card	June		Central			18888.12					
2	Cash	June		Central			1784.08					
3	Transfer	June		North			17000.12					
4	Credit card	June		North			13180.04					
5	Cash	June		North			3135.73					
6	Transfer	June		West			8645.98					
7	Credit card	June		West			6924.45					
8	Cash	June		West			2300.8					
9	Transfer	July		Central			23893.59					
10	Credit card	July		Central			21241.77					
11	Cash	July		Central			3130.38					
12	Transfer	July		North			17585.25					
13	Credit card	July		North			10338.81					
14	Cash	July		North			1196.57					
15	Transfer	July		West			7606.51					

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 t

```

(
    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

```

...	↑↓	clie...	...	↑↓	product_line	...	↑↓	tot...	...	↑↓
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

```

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

```

...	↑↓	clie...	...	↑↓	product_line	...	↑↓	tot...	...	↑↓
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