Find information about how many product have been ordered

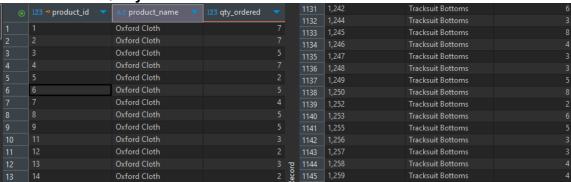
1. (10 point) Show the product IDs and product names that have been ordered more than once! Sorted by product ID

Answer here

SQL Query Syntax

```
SELECT product_id, product_name, count(order_id) as qty_ordered
FROM products
JOIN sales
USING ("product_id")
GROUP BY product_id, product_name
having count(order_id) > 1
order by product_id asc;
```

Screenshot of Query Results:



Description of Query Results:

In this query, we retrieved a total of 1145 data from the products that have been ordered more than once.

2. (10 point) From question number 1, How many products have been ordered more than once?

Answer here

SQL Query Syntax

```
WITH ordered_products AS (
        SELECT product_id, product_name, COUNT(order_id) AS qty_ordered
        FROM products
        JOIN sales USING (product_id)
        GROUP BY product_id, product_name
        HAVING COUNT(order_id) > 1
)
SELECT COUNT(*) AS total_products
FROM ordered_products;
```

Screenshot of Query Results:

```
1 1,145
```

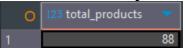
Description of Query Results:

In this query, we can conclude that there are 1145 products have been ordered more than once.

- 3. (10 point) From question number 2, How many products have only been ordered once? *Answer here*
 - SQL Query Syntax

```
WITH ordered_products AS (
        SELECT product_id, product_name, COUNT(order_id) AS qty_ordered
        FROM products
        JOIN sales USING (product_id)
        GROUP BY product_id, product_name
        HAVING COUNT(order_id) = 1
)
SELECT COUNT(*) AS total_products
FROM ordered_products;
```

Screenshot of Query Results:



Description of Query Results:

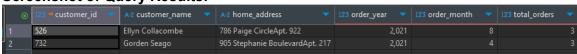
In this query, we can conclude that there are 88 products have only been ordered once.

Next, Dig deeper information into customer behavior

4. (10 point) list of customers who have placed orders more than twice in a single month. Manager need customer name and their address to give the customer special discount *Answer here*

SQL Query Syntax

Screenshot of Query Results:



Description of Query Results:

In this query, we identified 2 customers who qualified for a special discount based on their order frequency. Based on the output, Ellyn Collacombe (Cust ID: 526) from 786 Paige Circle Apt. 922 placed 3 orders in August 2021, while Gorden Seago (Cust ID: 732) from 905 Stephanie Boulevard Apt. 217 placed 3 orders in April 2021.

SQL Query Syntax

Screenshot of Query Results:

•	123 customer_id	•	Ø first_order_date ▼	
1	1	ď	2021-02-18	2023-01-15
2			2023-01-16	2023-01-16
3			2023-01-18	2023-01-18
4	4		2023-01-19	2023-01-19
5	5		2023-01-20	2023-01-20
6			2021-05-21	2021-05-21
7	10		2021-03-09	2021-03-09
8	11		2021-05-28	2021-05-28
9	12		2021-06-19	2021-06-19
10	13		2021-09-28	2021-09-28

- Description of Query Results:
- The query retrieves the first and last order dates for each customer limited to the first 10 records. Based on the output, Customer 1 had their first order on February 18, 2021, and their last order on January 15, 2023, indicating multiple purchases over time. In contrast, customers 2 to 13 (excluding customer 6, 8, and 9, who are not in the top 10 because they have not placed any orders) had the same first and last order date, suggesting they made a single purchase.
- 6. (10 point) Retrieve the top 5 customers who have spent the most amount of money on products within the "Trousers" category, including the customer's name, the quantity and total amount spent in this category. Additionally, find the total number of products sold in this category and calculate the average total price spent in this category, compare with the top 5 customers who have spent the most amount of money on products within the "Trousers" category. Finally, sort the results by the total amount spent in descending order.

Answer here

SQL Query Syntax

Screenshot of Query Results:

•	123 · customer_id 🔻	A-Z customer_name	123 total_spent 🔻	123 qty_order 🔻	123 qty_trousers 🔻	123 avg_price_trousers
1	571	Kristofor Roos	2,827	28	3,360	202.7177658942
2	348	Thorny Nornable	2,802		3,360	202.7177658942
3	163	Harrietta Burchatt	2,426	24	3,360	202.7177658942
4	282	Wren Helgass	2,272	22	3,360	202.7177658942
5	465	Mallory Castellani	2,126	21	3,360	202.7177658942

Description of Query Results:

In this query, we identified the top 5 customers who have spent the most money on Trousers, while also providing insights into the total quantity sold and the average total price spent in this category.

From the results:

- Kristofor Roos (Cust ID: 571) spent the most at \$2,827, purchasing 28 trousers.
- Thorny Nornable (Cust ID: 348) followed closely, spending \$2,802 on 28 trousers.
- Harrietta Burchatt (Cust ID: 163) spent \$2,426 for 24 trousers.
- Wren Helgass (Cust ID: 282) spent \$2,272 for 22 trousers.
- Mallory Castellani Cust (ID: 465) spent \$2,126 for 21 trousers.

In comparison, across all customers, the total number of trousers sold was 3,360, and the average total price of trousers was \$202.72.

The results indicate that the top spenders purchased significantly more than the average, reinforcing their status as high-value customers for this product category.

7. (10 point) Find the top-selling (Top 1) product for each month. You want to know the product with the highest **total quantity sold** in each month. If there are products that have the same total quantity sold, choose the smallest product ID. Return the product name, the corresponding month, and the total quantity sold for each month's top-selling product. Sorted by month

Answer here

SQL Query Syntax

Screenshot of Query Results:

		,			
•	123 sale_month 🔻	123 sale_year 🔻	123 ⁴ product_id ▼	A-Z product_name	123 total_quantity 🔻
1	1	2,021	1,084	Joggers	10
2	2	2,021	920	Drawstring	9
3		2,021		Oxford Cloth	10
4	4	2,021	850	Chinos	8
5	5	2,021	650	Coach	11
6		2,021	1,139	Cargo Pants	10
7	7	2,021	383	Henley	7
8	8	2,021	125	Denim	7
9	9	2,021	28	Oxford Cloth	9
10	10	2,021	1,177	High-Waisted	9

Description of Query Results:

From this query, we can identified that the dataset captured orders during the period between January 2021 till October 2021 and here are the results that we can concluded :

- In January 2021, Joggers (Prod ID: 1084) was the top seller with 10 units sold.
- In February 2021, Drawstring (Prod ID: 920) was the top seller with 9 units sold.
- In March 2021, Oxford Cloth (Prod ID: 1) was the top seller with 10 units sold.
- In May 2021, Coach (Prod ID: 650) had the highest sales with 11 units sold, making it the highest monthly quantity sold in the dataset.
- In June 2021, Cargo Pants (Prod ID: 1139) was the top seller with 10 units sold.
- In July 2021, Henley (Prod ID: 383) was the top seller with 7 units sold.
- In August 2021, Denim was the top seller with 7 units sold.
- In September 2021, Oxford Cloth was the top seller with 9 units sold.
- In October 2021, High-Waisted was the top seller with 9 units sold.
- 8. (10 point) Create a view to store a query for calculating monthly total payment. Answer here
 - SQL Query Syntax

```
CREATE VIEW monthlytotalpayment AS

SELECT

EXTRACT(YEAR FROM CAST(order_date AS DATE)) AS sale_year,

EXTRACT(MONTH FROM CAST(order_date AS DATE)) AS sale_month,

SUM(payment) AS total_transaction_amount

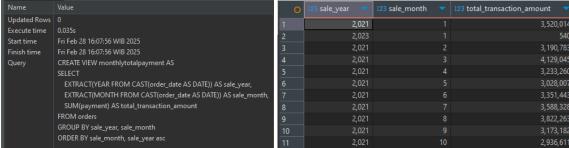
FROM orders

GROUP BY sale_year, sale_month

ORDER BY sale_month, sale_year asc;
```

SELECT * FROM monthlytotalpayment;

Screenshot of Query Results:



Description of Query Results:

From this query and view, we can identified results below:

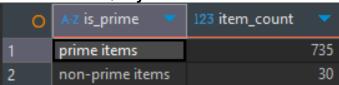
- January 2021 recorded the highest total payment at 3,520,014, whereas January 2023 had a significantly lower total of 540, indicating fewer transactions in that period.
- March 2021 had the highest total payment of 4,129,045, suggesting peak sales activity.
- Throughout 2021, the monthly total payments varied, with notable highs in August (3,822,263) and July (3,588,328).
- The total payment values show fluctuations, potentially indicating seasonal trends or changes in customer purchasing behavior.
- 9. (10 point) As a warehouse manager responsible for stock management in your company's warehouse, you oversee a warehouse with a total area of 600,000 sq ft. There are two types of items: prime items and non-prime items. These items come in various sizes, with priority given to prime items. Your task is to determine the maximum number of prime and non-prime items that can be stored in the warehouse
 - Prime and non-prime items are stored in their respective containers. For example, In the database, there are 15 non-prime items and 35 prime items. Each prime container must contain 35 prime items, and each non-prime container must contain 15 non-prime items
 - Non-prime items must always be available in stock to meet customer demand, so the non-prime item count should never be zero.

Answer Here

SQL Query Syntax:

```
-- Mengecek hasil dalam temporary table
SELECT * FROM item prime;
create temp table item_nonprime as
             sum(item_size_sqft) as nonprime_container_size,
select
      count(item_size_sqft) as qty_nonprime
from item
where is prime is false;
SELECT * FROM item nonprime;
create temp table cont prime as
select floor(600000.0 / prime_container_size) as qty_container_prime,
      atv prime
from item_prime;
create temp table item_summary_prime as
select qty_container_prime * qty_prime as item_count_prime
from cont prime;
create temp table cont_nonprime as
with sisa_luas_warehouse as (
select (600000.0 - qty_container_prime * prime_container_size) as
sisa_luas warehouse
from item_prime
cross join cont_prime
select floor(sisa_luas_warehouse / nonprime_container_size) as
qty_container_nonprime,
      qty_nonprime
from sisa_luas_warehouse
cross join item_nonprime;
create temp table item_summary_nonprime as
select qty_container_nonprime * qty_nonprime as item_count_nonprime
from cont nonprime;
select 'prime items' as is_prime, item_count_prime as item_count
from item_summary_prime
union all
select 'non-prime items' as is_prime, qty_container_nonprime * qty_nonprime as
from cont nonprime;
```

Screenshot of Query Results:



Description of Query Results:

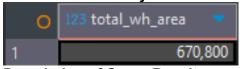
From this query, the output shows that 735 prime items are allocated more space due to their priority, while 30 non-prime items are stored to ensure continuous availability. The query determines the maximum number of prime and non-prime items that can be stored within a 600,000 sq ft warehouse while prioritizing prime items. It calculates the total storage occupied by prime and non-prime items, then determines how many containers of each type can fit within the available space. Prime items are given priority, so their containers are allocated first. The remaining warehouse space is then used for non-prime items, ensuring that at least one non-prime container is always available. By calculating storage distribution efficiently, this query helps warehouse managers optimize space utilization while maintaining minimum stock levels for non-prime items.

10. (10 point) The warehouse manager is planning to find a new warehouse to store their products. The warehouse is expected to accommodate 20 containers for each prime and non-prime item. What is the minimum required size for the warehouse?

SQL Query Syntax:

```
-- new warehouse for 20 container prime and 20 container non prime
-- total new warehouse area
select 20 * prime_container_size + 20 * nonprime_container_size as
total_wh_area
from item prime
cross join item nonprime;
```

Screenshot of Query Results:



Description of Query Results:

From this query we can determined the total warehouse area required. The final output, 670,800 sq ft, represents the necessary warehouse space, ensuring adequate capacity for storing 20 prime containers and 20 non-prime containers.