Supply Chain Order Analysis

Project Title:

Supply Chain Order Analysis

Description:

The **Supply Chain Order Analysis** project focuses on analyzing sales and canceled orders from a supply chain dataset. It uses SQL to extract insights from two CSV files- “canceled_test.csv” and “sales_test.csv” which include data such as order numbers, customer IDs, item details, and the number of units ordered and shipped. The project involves performing volume analysis, service level calculations, and ABC classification to classify items based on their contribution to total sales. Through this project, key supply chain metrics like order fulfillment, customer behavior, and cancellation patterns are analyzed, helping identify inefficiencies and trends for further optimization.

Tools Used:

- SQL
- MySQL
- PostgreSQL (or any SQL database)

Dataset:

I downloarded the dataset from Kaggle.

You can download and practice it too.

Link- https://www.kaggle.com/datasets/annelee1/supply-chain-cel-dataset

Or

Don't stress so much, I am giving you the files

canceled_test.csv

sales_test.csv

Task:

Basic:

1. How many orders were placed in January 2017 in the sales_test.csv dataset?
2. What is the total number of units ordered (NS_ORDER) in February 2017?
3. Find the number of canceled orders (NC_ORDER) for each customer in canceled_test.csv.
4. How many unique customers are there in the sales_test.csv dataset?
5. Find the average number of items ordered (NS_ORDER) per order in the sales_test.csv dataset.
6. List the top 5 items that have been ordered the most in the sales_test.csv.
7. Find the total number of successful orders (NS_ORDER) where the CUSTOMER_NO is either 1857566 or 1358538 and the DATE is in January 2017.

Intermediate

8. Find the total number of units ordered (NS_ORDER) and canceled (NC_ORDER) for each item that appears in both sales_test.csv and canceled_test.csv. Include items that have been both ordered and canceled.
9. Compare the number of canceled orders (NC_ORDER) and successful orders (NS_ORDER) for the same items.

10. Classify each order in the sales_test.csv dataset as 'High', 'Medium', or 'Low' based on the number of units ordered (NS_ORDER):
    - 'High' if NS_ORDER is greater than 50.
    - 'Medium' if NS_ORDER is between 20 and 50.
    - 'Low' if NS_ORDER is less than 20.
11. Calculate the percentage of shipped items (NS_SHIP) out of the total ordered (NS_ORDER) for each customer in sales_test.csv.
12. Find the top 3 customers with the most canceled orders in canceled_test.csv.
13. List all the items that have been canceled more than shipped in canceled_test.csv.
14. Find the customer who placed the largest number of orders in January 2017 from the sales_test.csv dataset.

Advanced:

15. For each customer, calculate the cumulative total of ordered units (NS_ORDER) over time and rank the orders by date. Show the ORDER_NO, CUSTOMER_NO, NS_ORDER, DATE, and the cumulative total of ordered units.
16. Find the top 3 customers who have the highest total number of canceled orders (NC_ORDER) from canceled_test.csv and their corresponding total sales (NS_ORDER) from sales_test.csv.
17. Find out the contribution of top 5 customers (by total NS_ORDER) to overall sales.
18. Perform an ABC classification of items in sales_test.csv, where:
    - Class A: Top 20% of items contributing to 80% of total sales.
    - Class B: Next 30% of items contributing to 15% of total sales.
    - Class C: Remaining 50% of items.

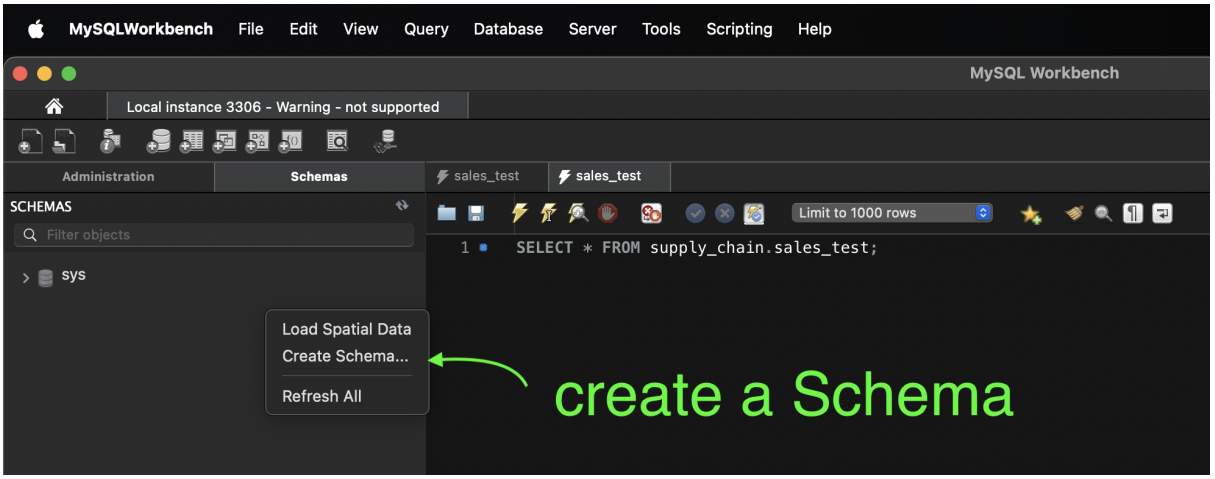
Outcome:

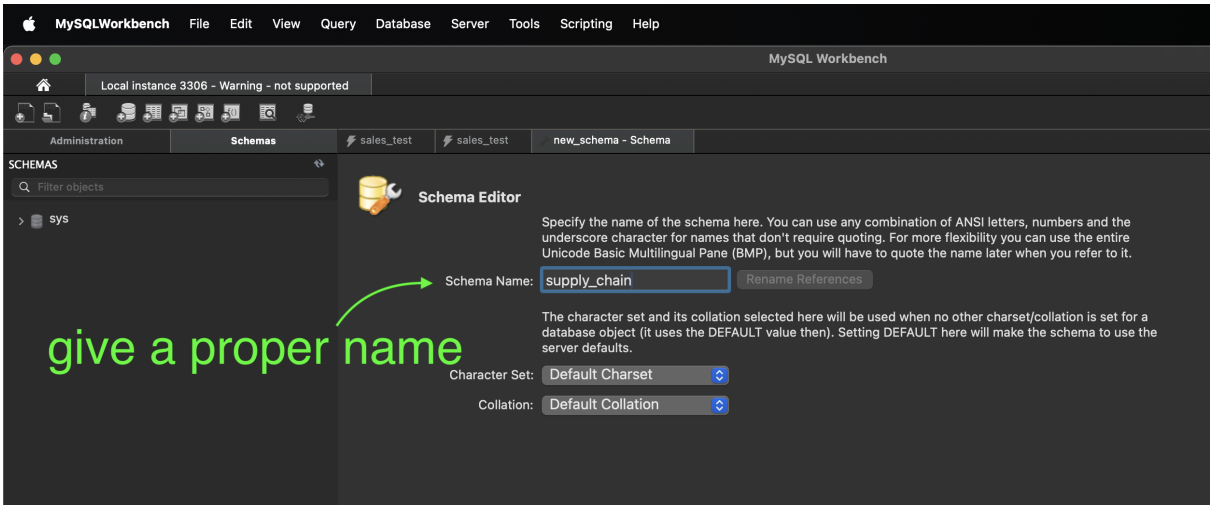
In this project, we analyzed the supply chain dataset to uncover key insights into order trends, customer behavior, and cancellation patterns. We performed volume analysis, service level assessments, and ABC classification of items. Our findings revealed the top-performing items, identified opportunities to reduce cancellations, and highlighted customer segments with the highest order values. This analysis provides actionable recommendations for optimizing inventory management and enhancing overall supply chain efficiency.

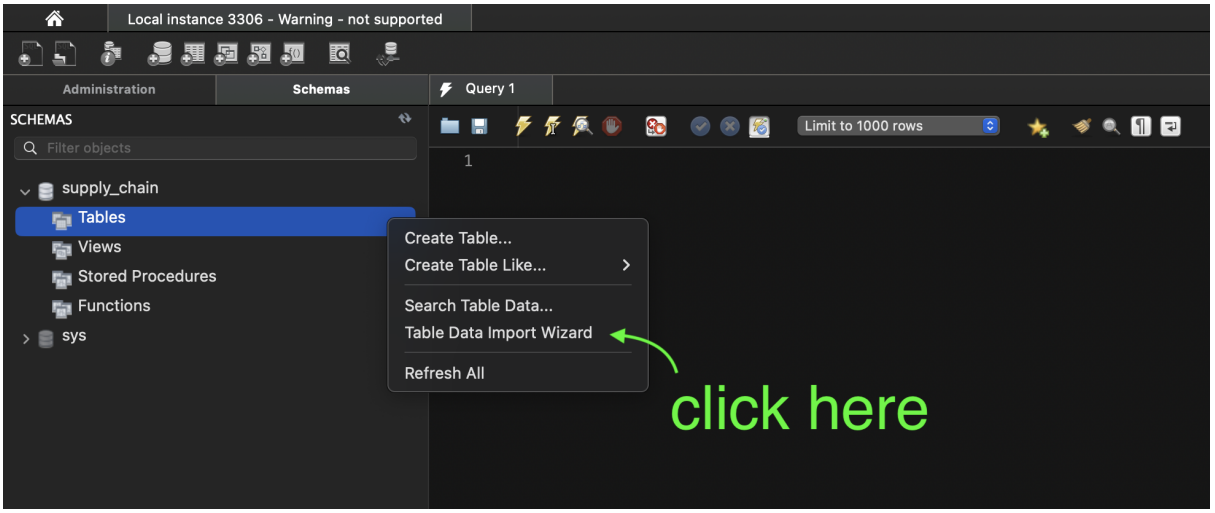
Let's start solving the problems-

At first I have to import the 2 csv files in the mysql workbench

Search
Maccy
Notion
Safari
Google Chrome
iTerm
PyCharm CE
pgAdmin 4
Prime Video
Photo Booth
QuickTime Player
MySQLWorkbench
ScreenBrush
Microsoft Excel
zoom.us
open MySQLWorkbench


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
MySQL Workbench
Local instance 3306 - Warning - not supported
Administration
Schemas
sales_test
sales_test
SCHEMAS
Filter objects
sys
Limit to 1000 rows
1 SELECT * FROM supply_chain.sales_test;
Load Spatial Data
Create Schema...
Refresh All
create a Schema

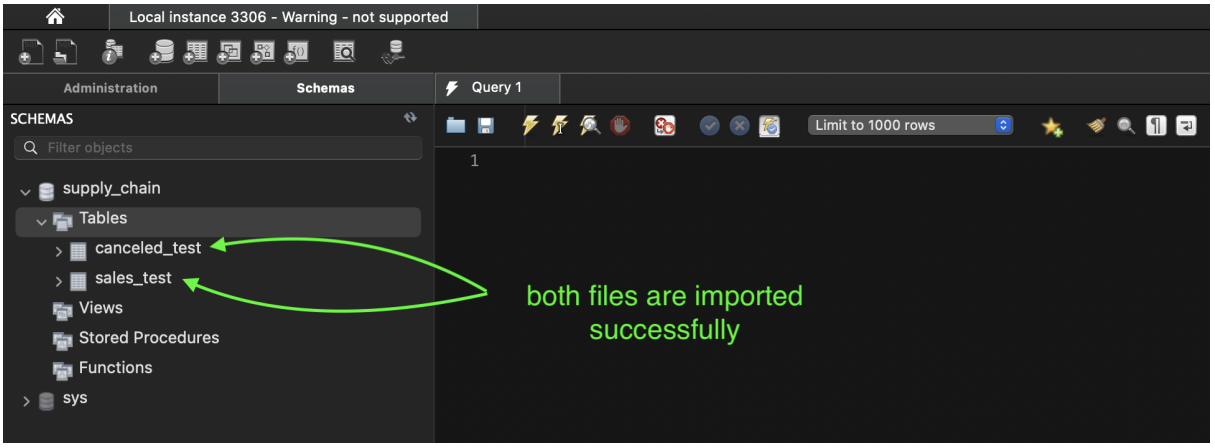

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
MySQL Workbench
Local instance 3306 - Warning - not supported
Administration
Schemas
sales_test
sales_test
new_schema - Schema
SCHEMAS
Filter objects
sys
Schema Editor
Specify the name of the schema here. You can use any combination of ANSI letters, numbers and the underscore character for names that don't require quoting. For more flexibility you can use the entire Unicode Basic Multilingual Pane (BMP), but you will have to quote the name later when you refer to it.
Schema Name:
supply_chain
Rename References
The character set and its collation selected here will be used when no other charset/collation is set for a database object (it uses the DEFAULT value then). Setting DEFAULT here will make the schema to use the server defaults.
Character Set:
Default Charset
Collation:
Default Collation
give a proper name



Then chooche the first file you want to import

Then Next > Next > Finish

Do the same fo the another file



Now we have to understand the dataset.

Understanding the Dataset

The **Supply chain CEL dataset** contains data related to orders in a supply chain system for January and February 2017. The dataset has two CSV files:

1. **canceled_test.csv**: This file contains information about orders that were canceled.
   - **Columns**:
     - `ORDER_NO`: Unique identifier for each order.

- `DATE`: The date the order was placed.
- `LINE`: The line number of the order.
- `CUSTOMER_NO`: The customer ID.
- `ITEM`: The item ID.
- `NC_ORDER`: Number of units ordered.
- `NC_SHIP`: Number of units shipped (since it's a canceled dataset, this might indicate canceled units).

```
1• SELECT * FROM canceled_test;
```

20:1

Result Grid | Filter Rows: Search | Export: | Fetch rows:

ORDER_NO	DATE	LINE	CUSTOMER_NO	ITEM	NC_ORDER	NC_SHIP
528703	Tuesday, January 3, 2017	1	1857566	10135139	1	1
528705	Tuesday, January 3, 2017	1	1857566	10135140	1	1
528706	Tuesday, January 3, 2017	2	1857566	10135138	1	1
528707	Tuesday, January 3, 2017	1	1857566	10135132	1	1
528708	Tuesday, January 3, 2017	1	1857566	10135359	1	1

2. **sales_test.csv**: This file includes information about successful orders.
   - **Columns**:
     - `ORDER_NO`: Unique identifier for each order.
     - `DATE`: The date the order was placed.
     - `LINE`: The line number of the order.
     - `CUSTOMER_NO`: The customer ID.
     - `ITEM`: The item ID.
     - `NS_ORDER`: Number of units ordered.
     - `NS_SHIP`: Number of units shipped (for fulfilled orders, this should match `NS_ORDER`).

```
1• SELECT * FROM sales_test;
```

26:1

Result Grid Filter Rows: Search Export: Fetch rows:

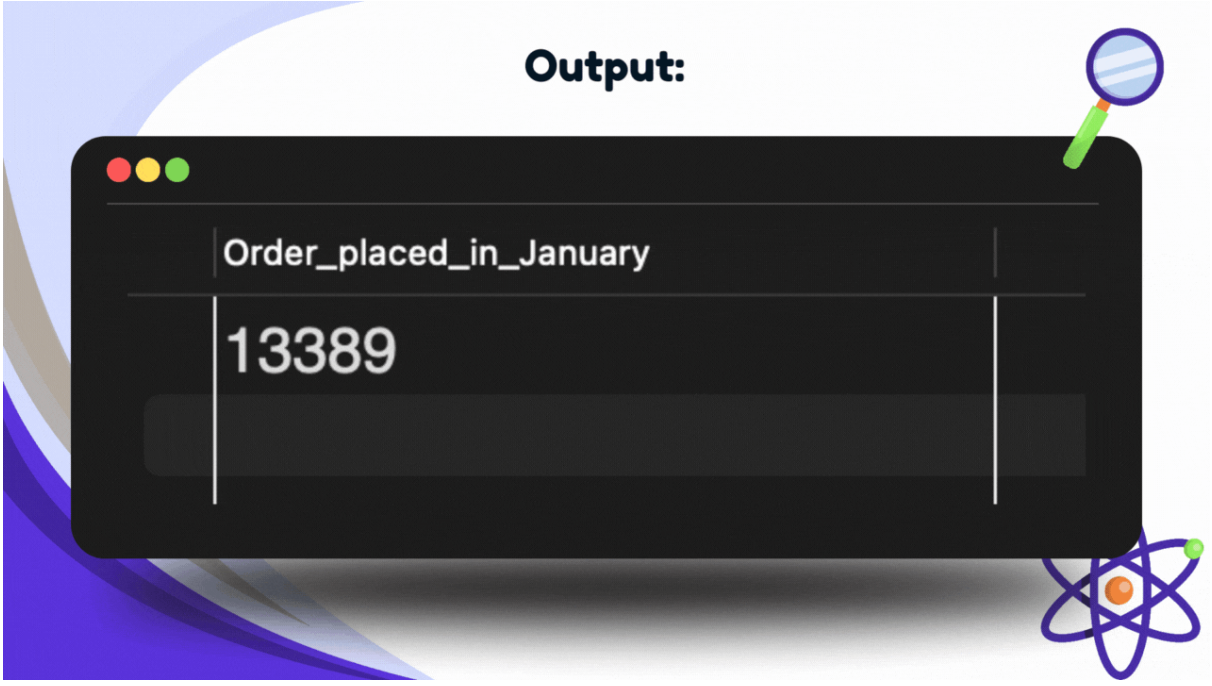
ORDER_NO	DATE	LINE	CUSTOMER_NO	ITEM	NS_ORDER	NS_SHIP
528758	Tuesday, January 3, 2017	1	1358538	111931	70	70
528791	Tuesday, January 3, 2017	1	1254798	1029071	10	10
528791	Tuesday, January 3, 2017	2	1254798	1033341	10	10
528791	Tuesday, January 3, 2017	3	1254798	1040827	5	5
528791	Tuesday, January 3, 2017	4	1254798	10106111	10	10

🟢

Basic:

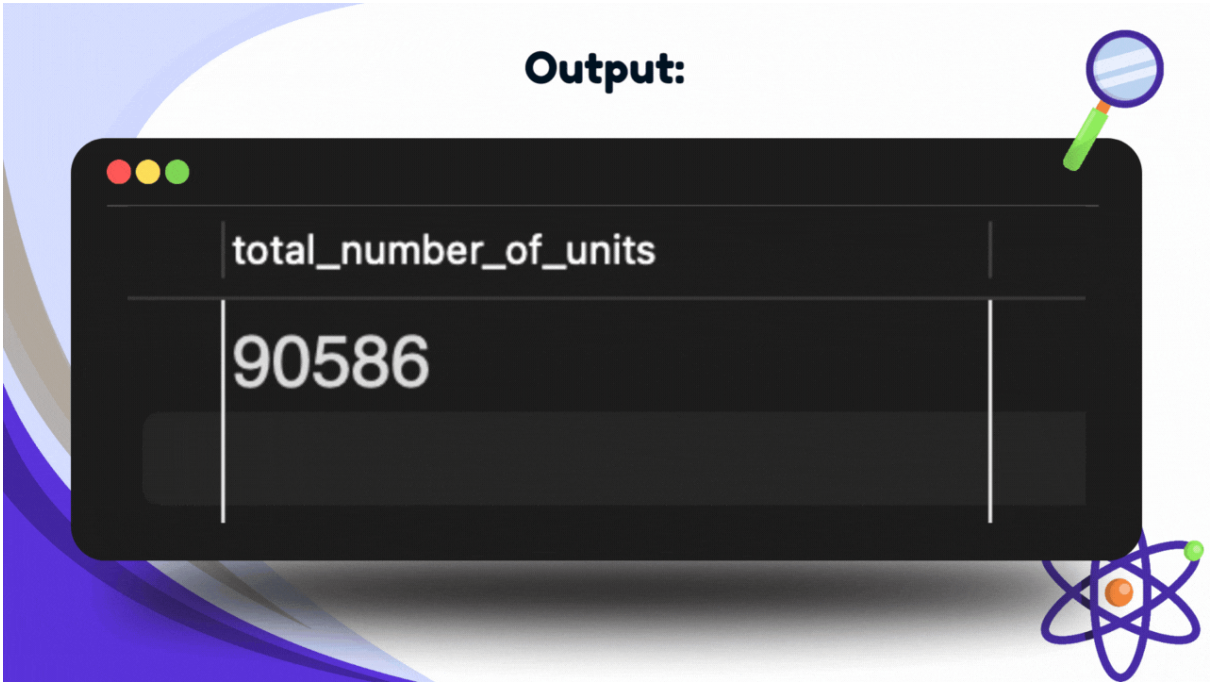
1. How many orders were placed in January 2017 in the sales_test.csv dataset?

```
SELECT COUNT(ORDER_NO) as Order_placed_in_January
FROM sales_test
WHERE DATE LIKE '%January%';
```



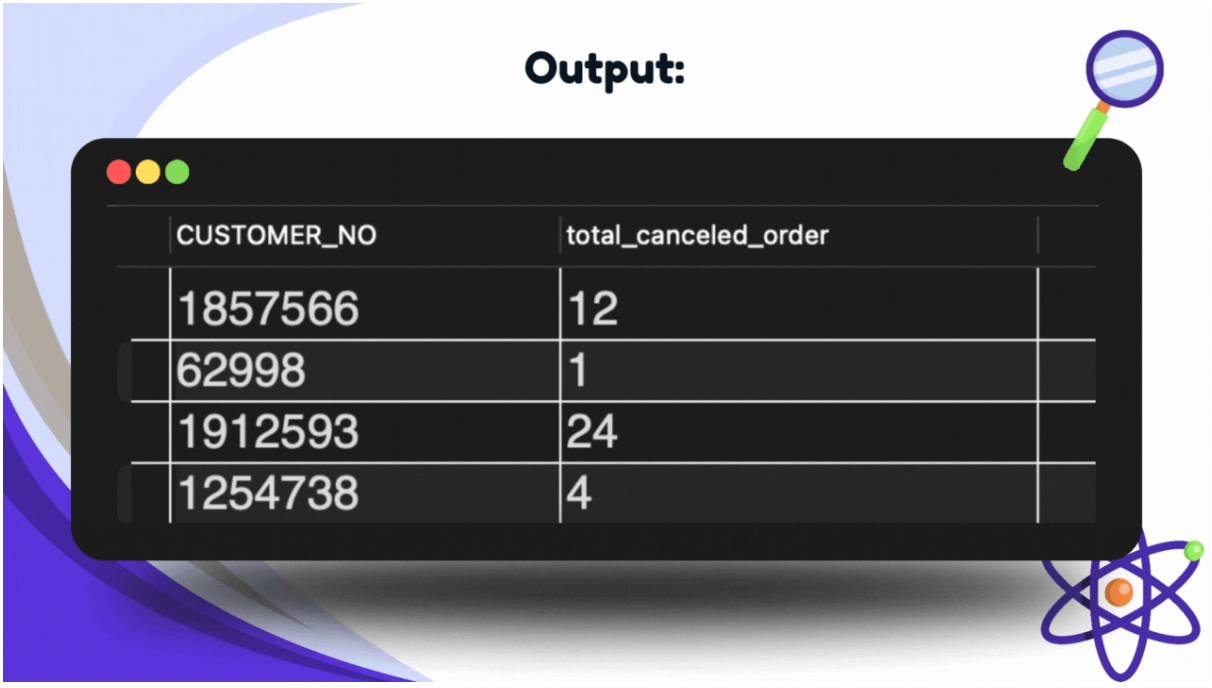
2. What is the total number of units ordered (NS_ORDER) in February 2017?

```
SELECT SUM(NS_ORDER) as total_number_of_units
FROM sales_test
WHERE DATE LIKE '%February%';
```



3. Find the number of canceled orders (NC_ORDER) for each customer in canceled_test.csv.

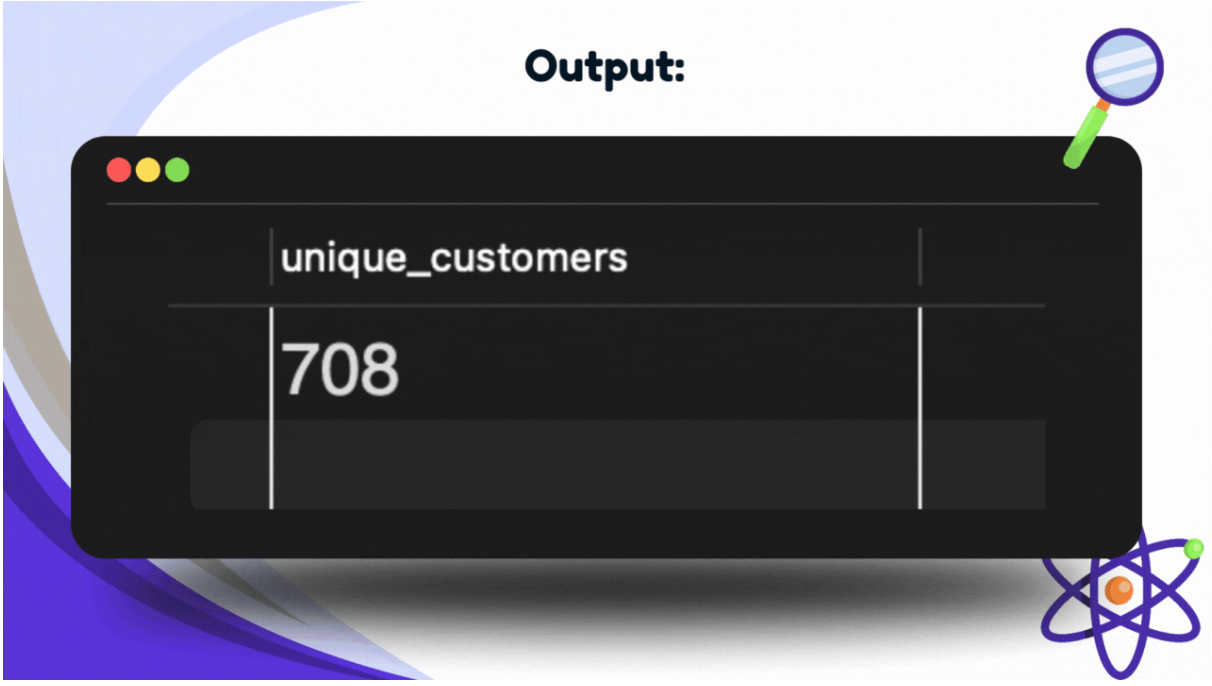
```
SELECT CUSTOMER_NO, SUM(NC_ORDER) as total_canceled_order
FROM canceled_test
GROUP BY CUSTOMER_NO;
```



CUSTOMER_NO	total_canceled_order
1857566	12
62998	1
1912593	24
1254738	4

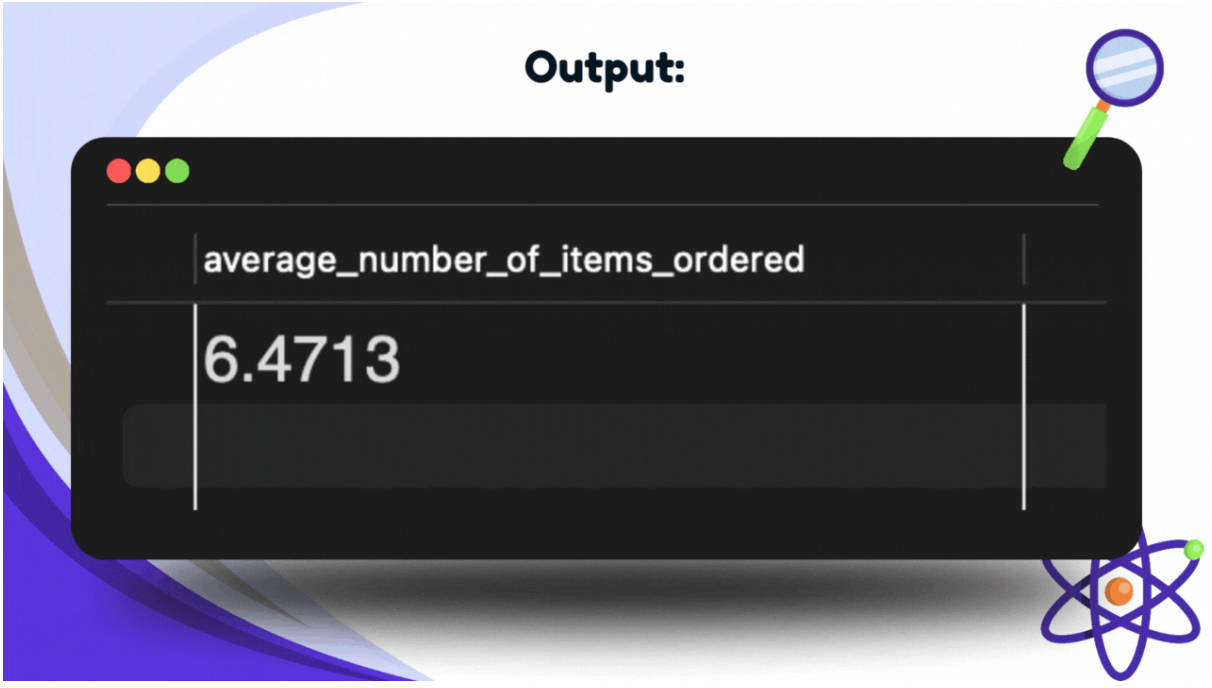
4. How many unique customers are there in the sales_test.csv dataset?

```
SELECT COUNT(DISTINCT CUSTOMER_NO) as unique_customers
FROM sales_test;
```



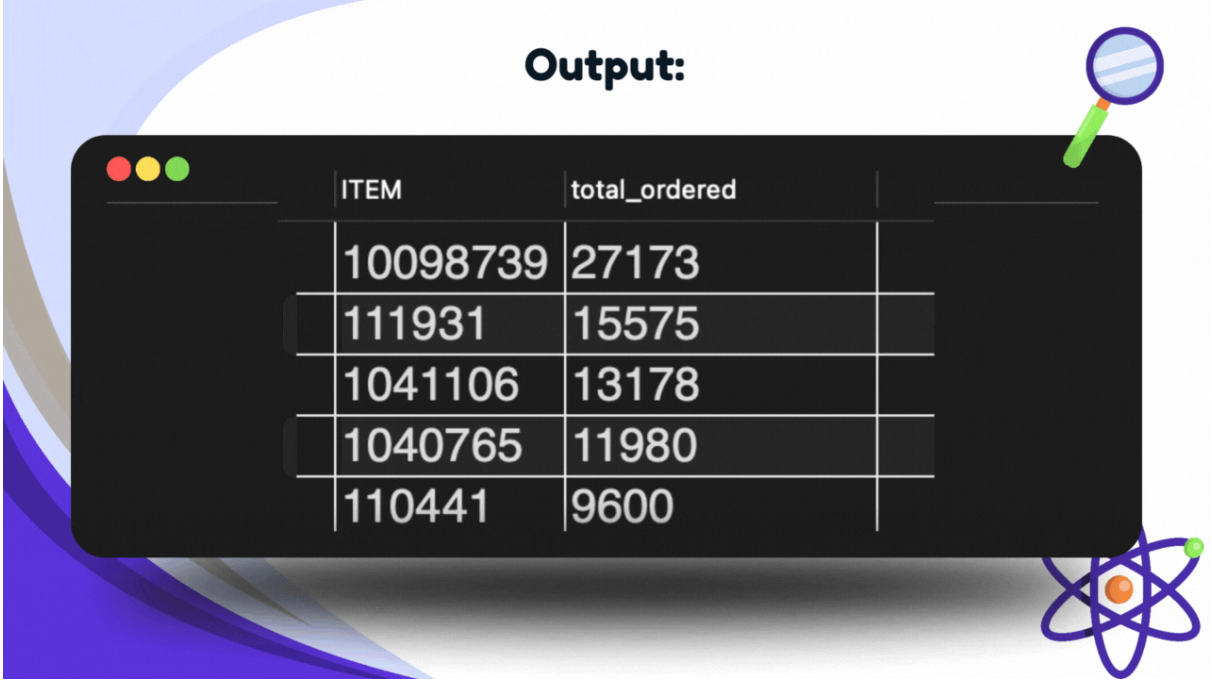
5. Find the average number of items ordered (NS_ORDER) per order in the sales_test.csv dataset.

```
SELECT AVG(NS_ORDER) as average_number_of_items_ordered
FROM sales_test;
```



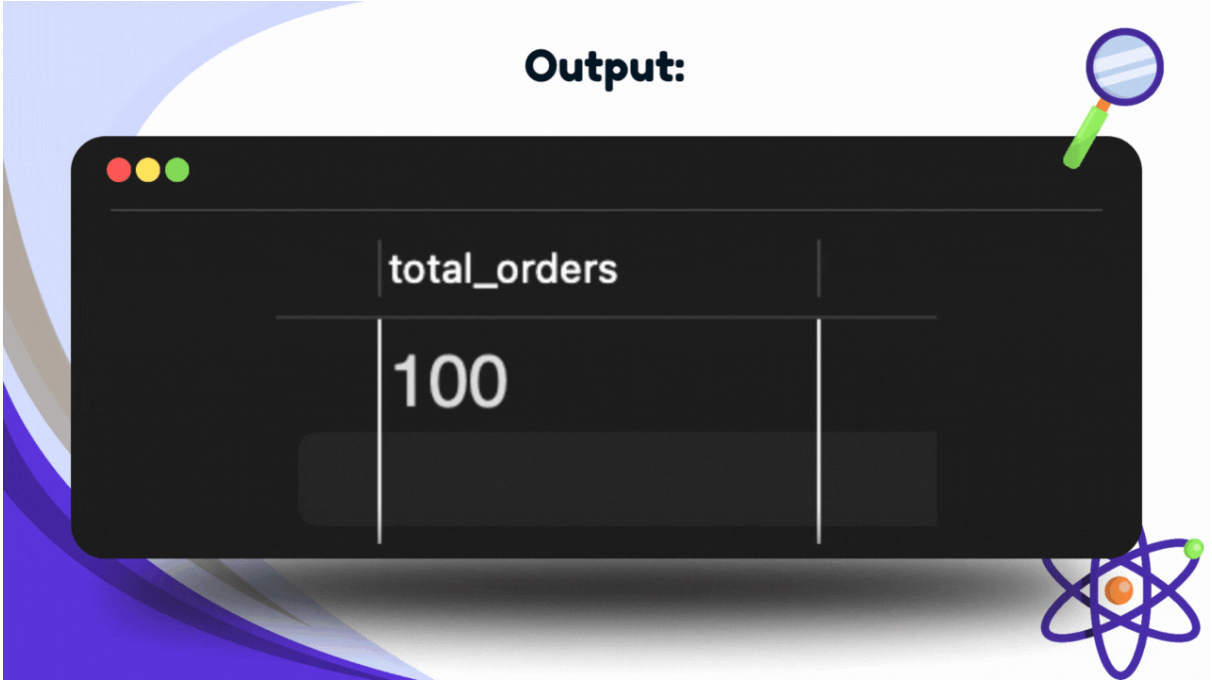
6. List the top 5 items that have been ordered the most in the sales_test.csv.

```
SELECT ITEM, SUM(NS_ORDER) as total_ordered
FROM sales_test
GROUP BY ITEM
ORDER BY total_ordered DESC
LIMIT 5;
```



7. Find the total number of successful orders (NS_ORDER) where the CUSTOMER_NO is either 1857566 or 1358538 and the DATE is in January 2017.

```
SELECT SUM(NS_ORDER) AS total_orders
FROM sales_test
WHERE (CUSTOMER_NO = 1857566 OR CUSTOMER_NO = 1358538)
AND DATE LIKE '%January%';
```

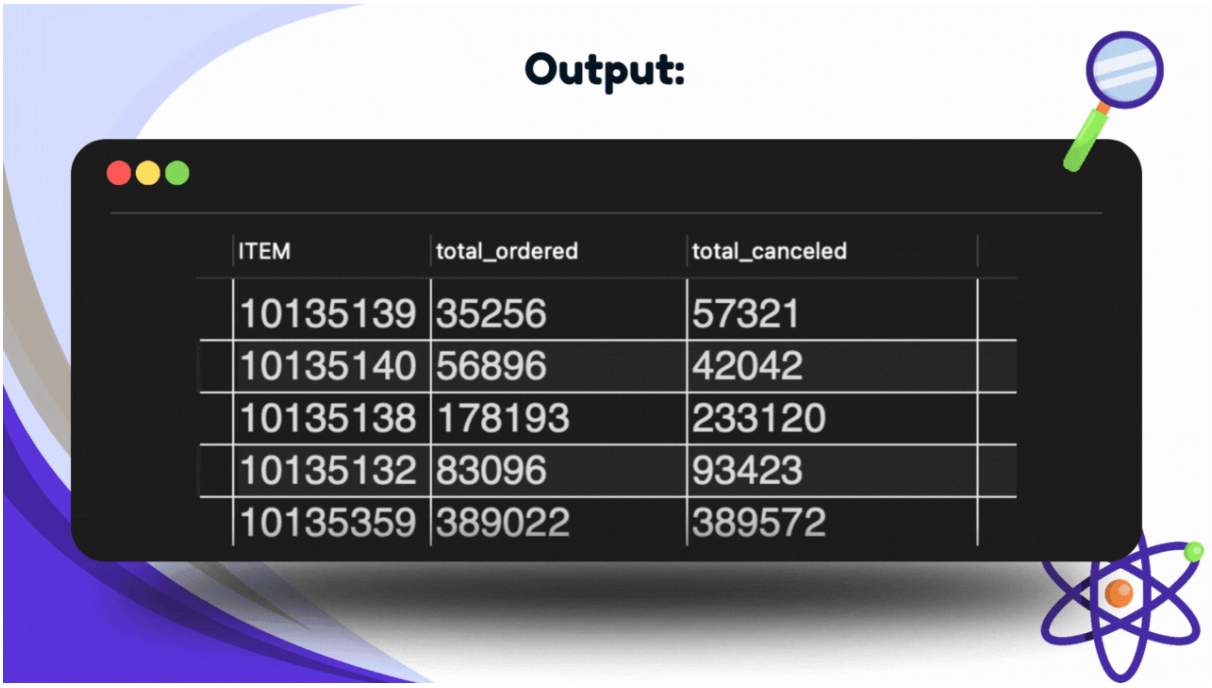


🟠 Intermediate:

8. Find the total number of units ordered (NS_ORDER) and canceled (NC_ORDER) for each item that appears in both sales_test.csv and canceled_test.csv. Include items that have been both ordered and canceled.

```
SELECT  s.ITEM,
                SUM(s.NS_ORDER) as total_ordered,
                SUM(c.NC_ORDER) as total_canceled
FROM sales_test s
INNER JOIN canceled_test c
```

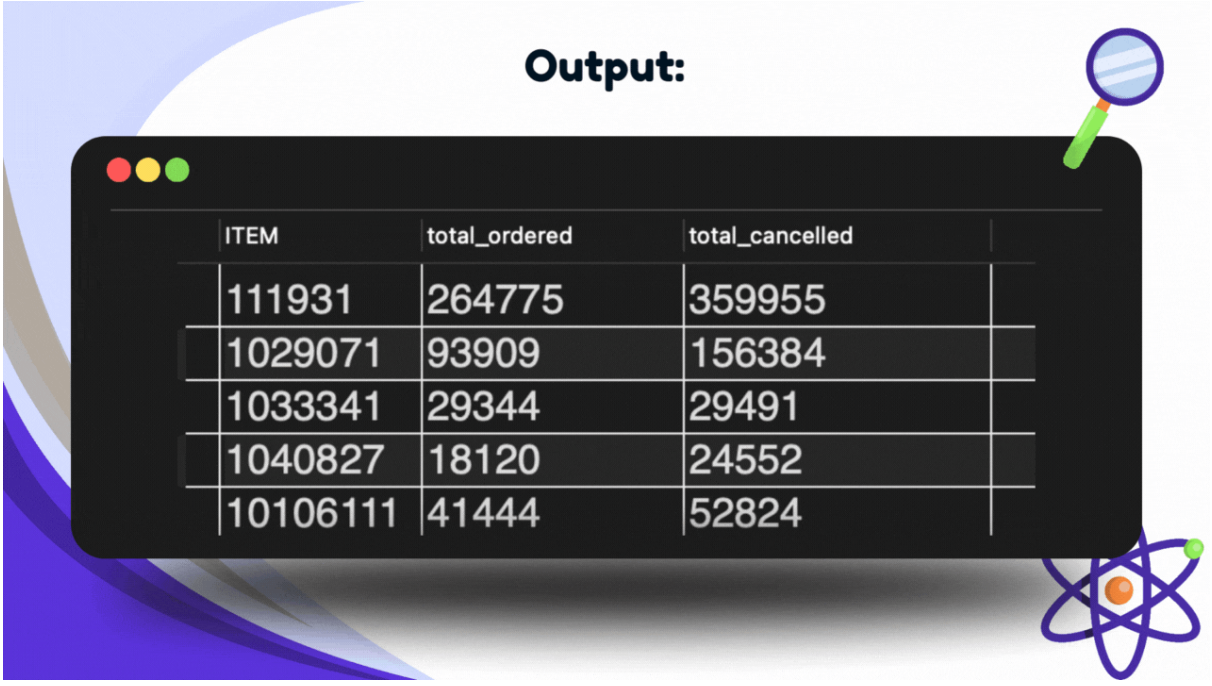
```
ON s.ITEM = c.ITEM
GROUP BY s.ITEM;
```



ITEM	total_ordered	total_canceled
10135139	35256	57321
10135140	56896	42042
10135138	178193	233120
10135132	83096	93423
10135359	389022	389572

9. Compare the number of canceled orders (NC_ORDER) and successful orders (NS_ORDER) for the same items.

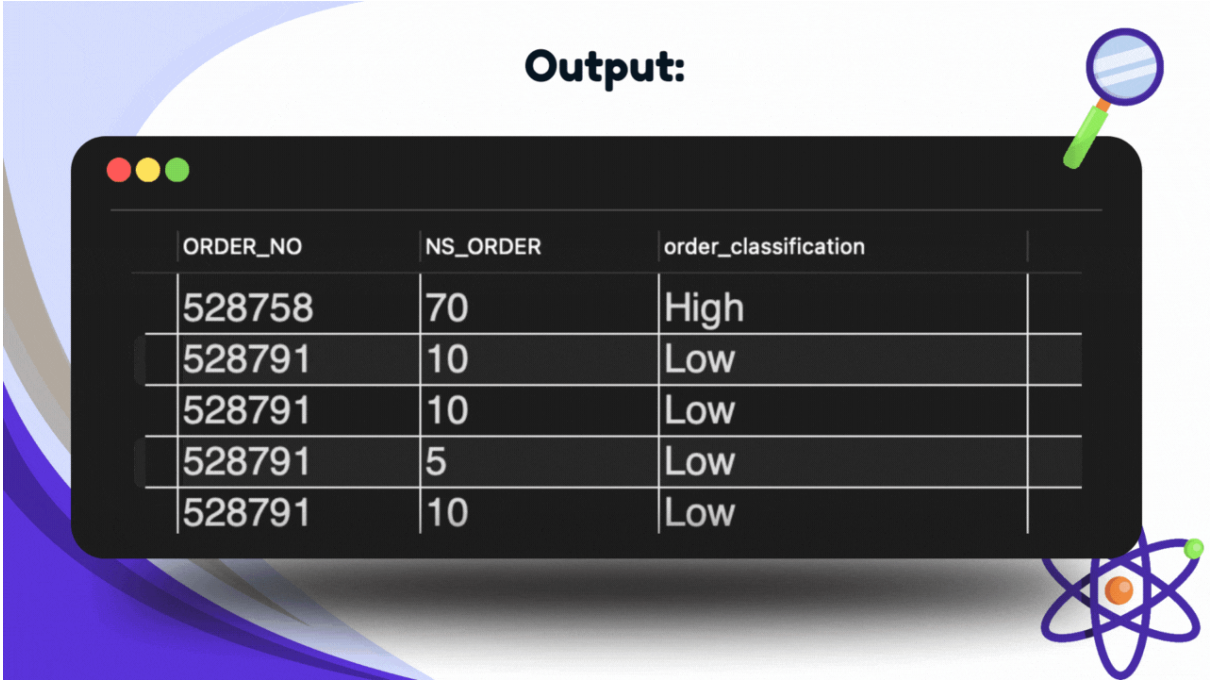
```
SELECT  s.ITEM,
                SUM(s.NS_ORDER) as total_ordered,
        SUM(c.NC_ORDER) as total_cancelled
FROM sales_test s
LEFT JOIN canceled_test c
ON s.ITEM=c.ITEM
GROUP BY s.ITEM;
```



10. Classify each order in the sales_test.csv dataset as 'High', 'Medium', or 'Low' based on the number of units ordered (NS_ORDER):

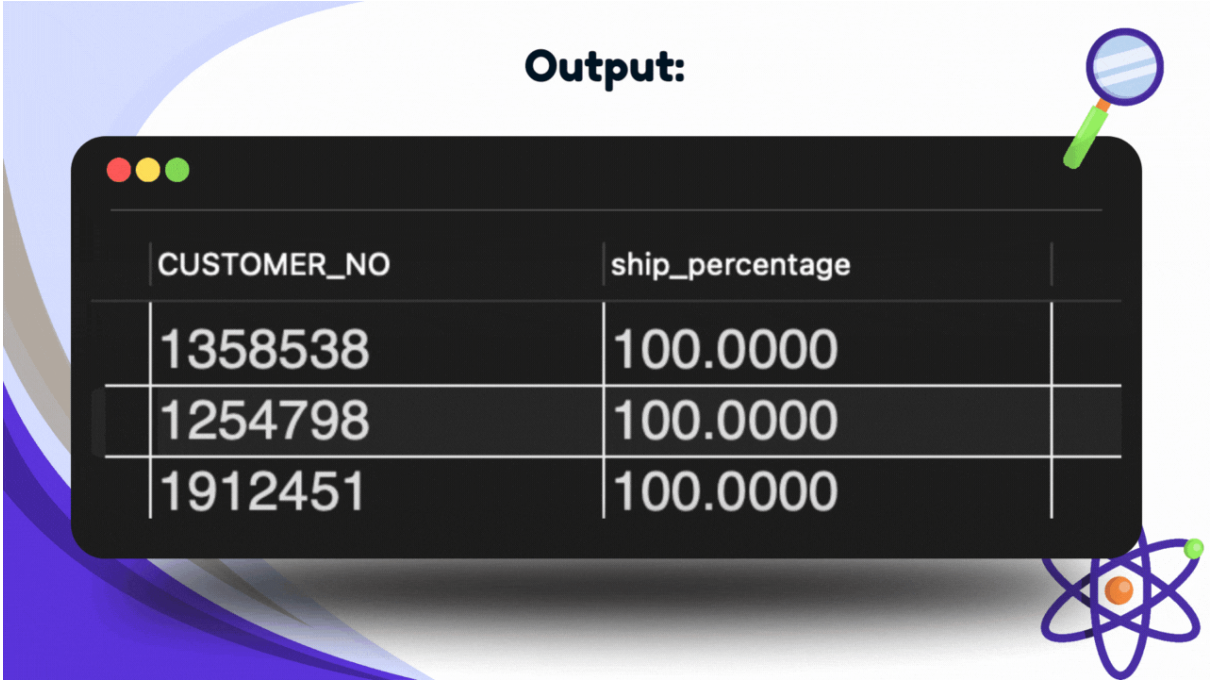
- **'High' if NS_ORDER is greater than 50.**
- **'Medium' if NS_ORDER is between 20 and 50.**
- **'Low' if NS_ORDER is less than 20.**

```
SELECT  ORDER_NO,
                NS_ORDER,
        CASE
        WHEN NS_ORDER > 50 THEN 'High'
        WHEN NS_ORDER BETWEEN 20 AND 50 THEN 'Medium'
        ELSE 'Low'
        END as order_classification
FROM sales_test;
```



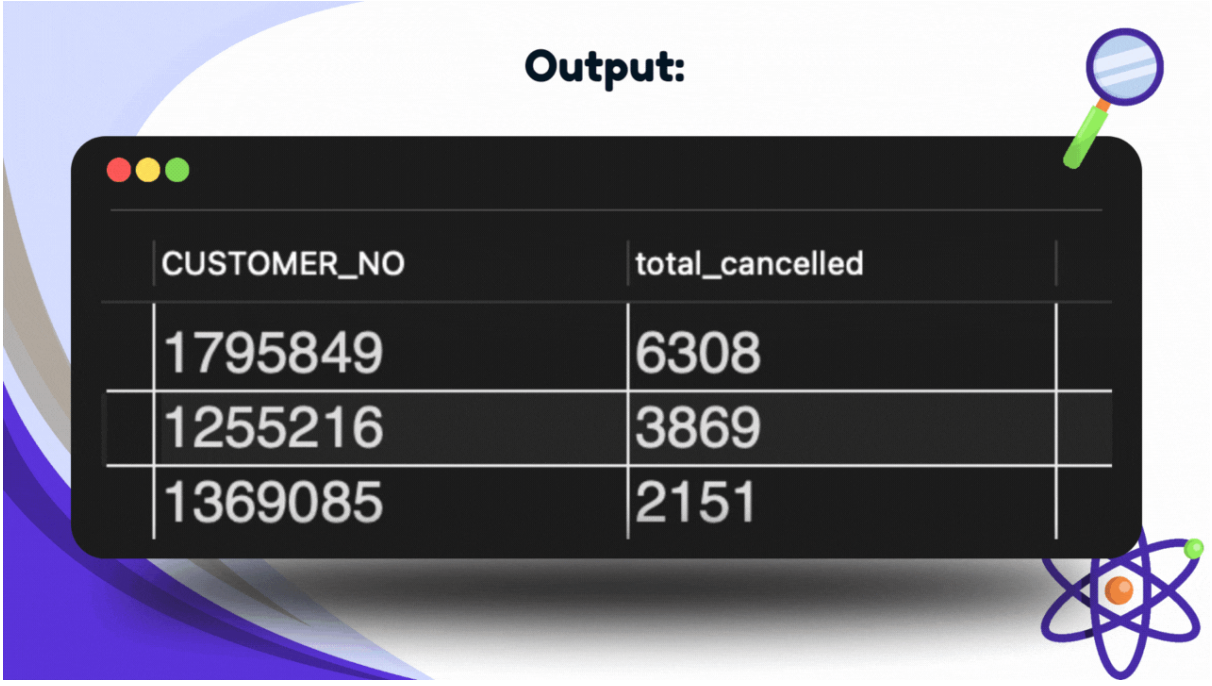
11. Calculate the percentage of shipped items (NS_SHIP) out of the total ordered (NS_ORDER) for each customer in sales_test.csv.

```
SELECT  CUSTOMER_NO,
        SUM(NS_SHIP) / SUM(NS_ORDER) * 100 as ship_percentage
FROM sales_test
GROUP BY CUSTOMER_NO;
```



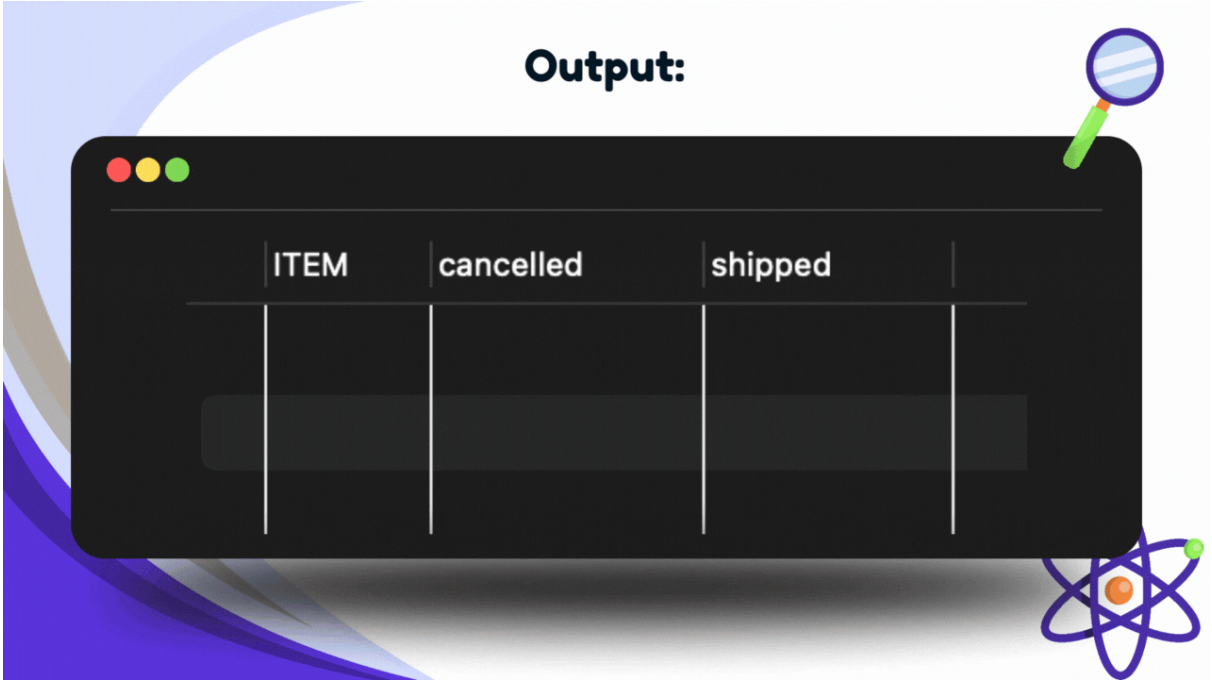
12. Find the top 3 customers with the most canceled orders in canceled_test.csv.

```
SELECT  CUSTOMER_NO,
                SUM(NC_ORDER) as total_cancelled
FROM canceled_test
GROUP BY CUSTOMER_NO
ORDER BY total_cancelled DESC
LIMIT 3;
```



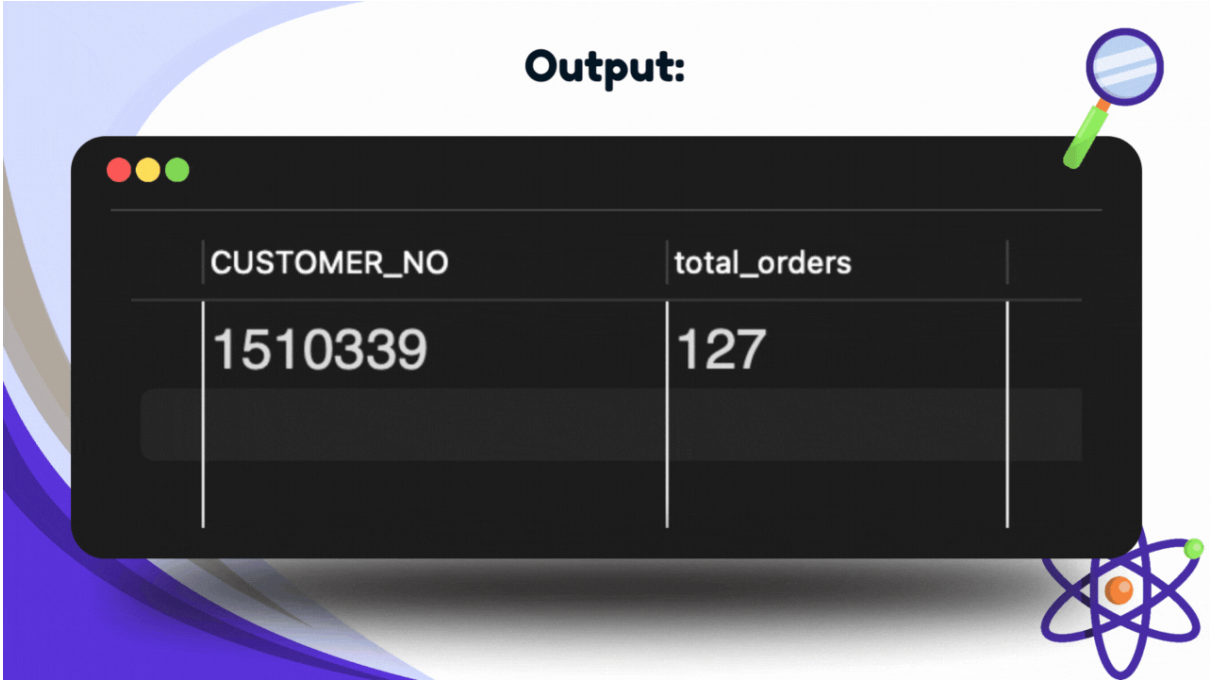
13. List all the items that have been canceled more than shipped in canceled_test.csv.

```
SELECT  ITEM,
                SUM(NC_ORDER) as cancelled,
        SUM(NC_SHIP) as shipped
FROM canceled_test
GROUP BY ITEM
HAVING cancelled > shipped;
```



14. Find the customer who placed the largest number of orders in January 2017 from the sales_test.csv dataset.

```
SELECT CUSTOMER_NO, total_orders
FROM (
                SELECT  CUSTOMER_NO,
                COUNT(NS_ORDER) as total_orders
                FROM sales_test
        WHERE DATE LIKE '%January%'
        GROUP BY CUSTOMER_NO
        )
        as customer_orders
ORDER BY total_orders DESC
LIMIT 1;
```



🔴 Advanced:

15. For each customer, calculate the cumulative total of ordered units (NS_ORDER) over time and rank the orders by date. Show the ORDER_NO, CUSTOMER_NO, NS_ORDER, DATE, and the cumulative total of ordered units.

```
SELECT
    ORDER_NO,
    CUSTOMER_NO,
    NS_ORDER,
    DATE,
    SUM(NS_ORDER) OVER (PARTITION BY CUSTOMER_NO ORDER BY DAT
    ROW_NUMBER() OVER (PARTITION BY CUSTOMER_NO ORDER BY DATE
```

```
FROM
    sales_test
ORDER BY
    CUSTOMER_NO, DATE;
```

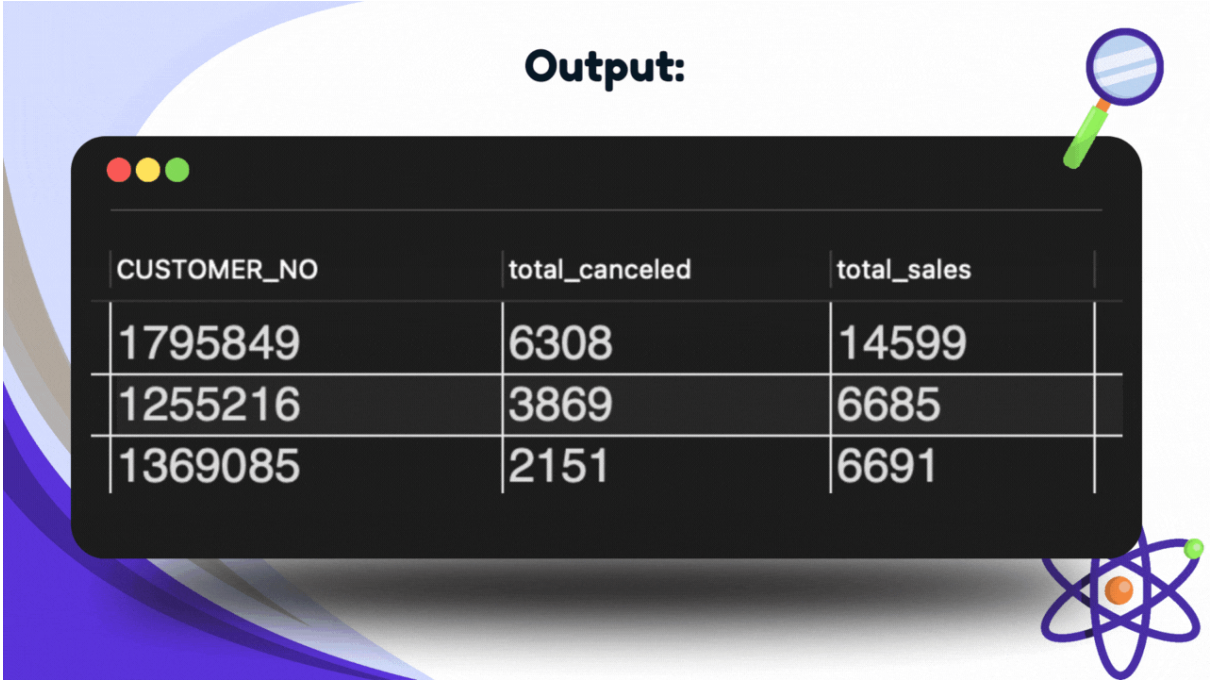
Output:

ORDER_NO	CUSTOMER_NO	NS_ORDER	DATE	cumulative_ordered_units	order_rank
538885	1254612	5	Friday, February 10, 2017	179	1
538887	1254612	2	Friday, February 10, 2017	179	2
538887	1254612	2	Friday, February 10, 2017	179	3
538887	1254612	1	Friday, February 10, 2017	179	4
538887	1254612	1	Friday, February 10, 2017	179	5
538887	1254612	4	Friday, February 10, 2017	179	6
538887	1254612	2	Friday, February 10, 2017	179	7
538886	1254612	1	Friday, February 10, 2017	179	8

16. Find the top 3 customers who have the highest total number of canceled orders (NC_ORDER) from canceled_test.csv and their corresponding total sales (NS_ORDER) from sales_test.csv.

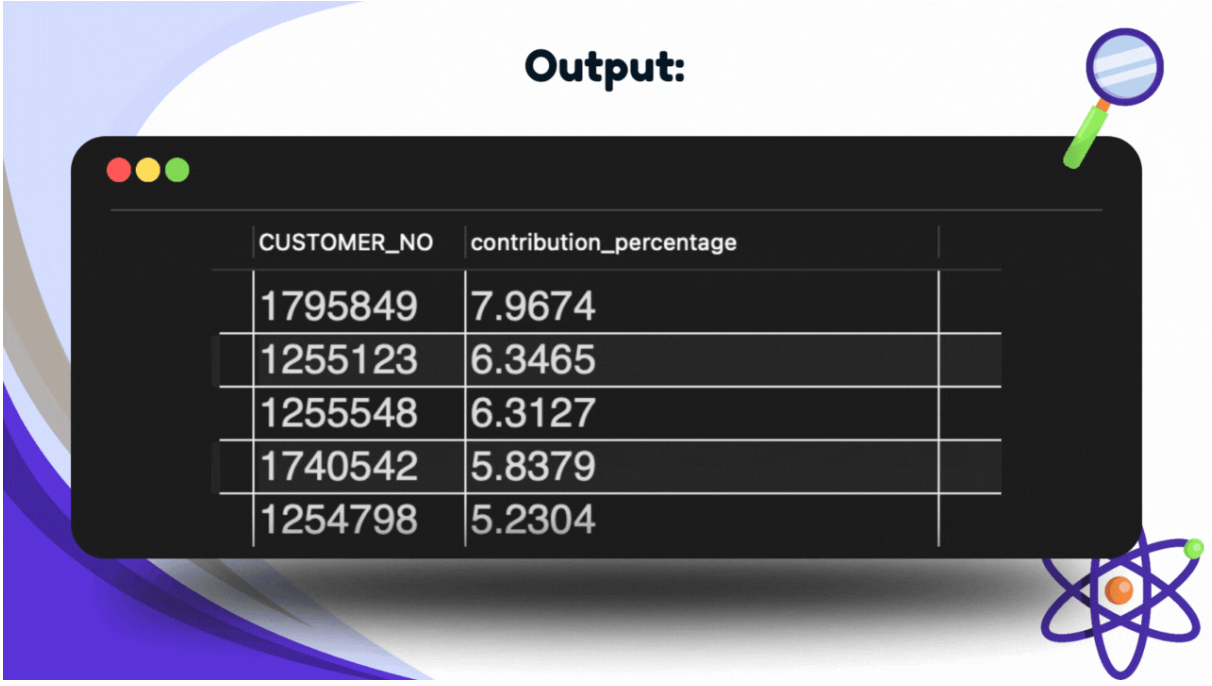
```
-- CTE to calculate total canceled orders for each customer
WITH canceled_orders AS (
    SELECT CUSTOMER_NO, SUM(NC_ORDER) AS total_canceled
    FROM canceled_test
    GROUP BY CUSTOMER_NO
)

-- Query to join the CTE with the sales data
SELECT c.CUSTOMER_NO, c.total_canceled, COALESCE(SUM(s.NS_ORD
FROM canceled_orders c
LEFT JOIN sales_test s ON c.CUSTOMER_NO = s.CUSTOMER_NO
```

```
GROUP BY c.CUSTOMER_NO, c.total_canceled
ORDER BY c.total_canceled DESC
LIMIT 3;
```



17. Find out the contribution of top 5 customers (by total NS_ORDER) to overall sales.

```
WITH total_sales AS (
  SELECT SUM(NS_ORDER) AS total_sales FROM sales_test
),
top_customers AS (
  SELECT CUSTOMER_NO, SUM(NS_ORDER) AS customer_sales
  FROM sales_test
  GROUP BY CUSTOMER_NO
  ORDER BY customer_sales DESC
  LIMIT 5
)
SELECT CUSTOMER_NO, customer_sales / (SELECT total_sales FROM
FROM top_customers;
```

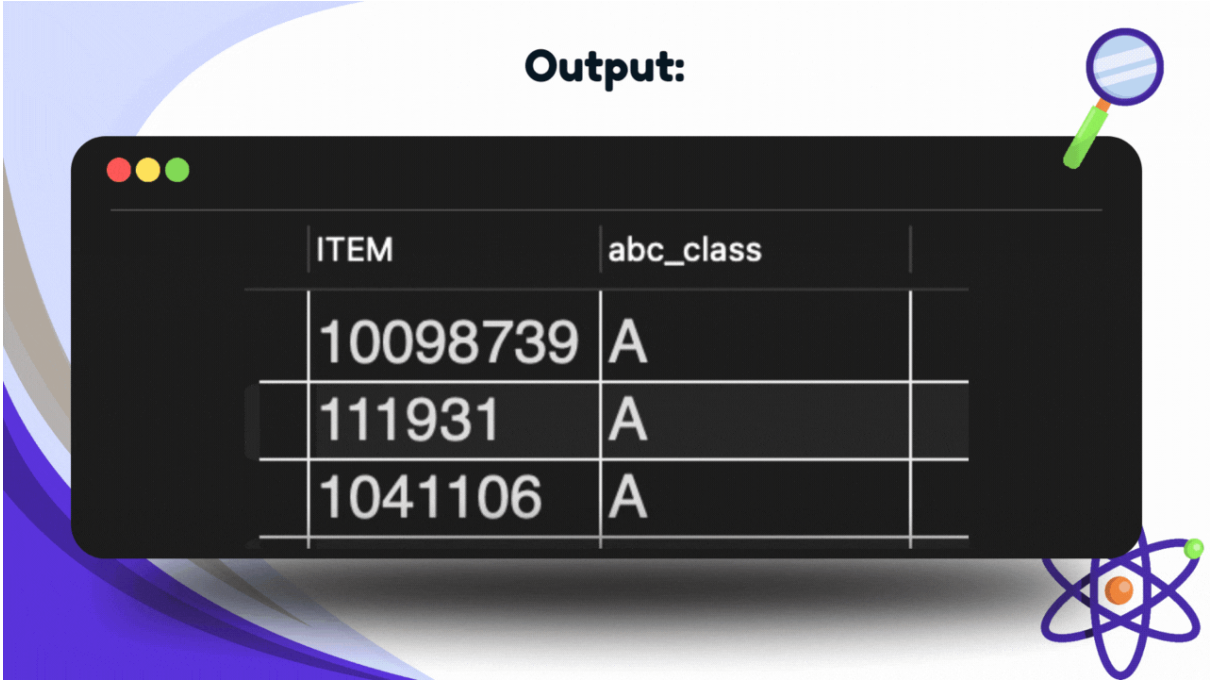


18. Perform an ABC classification of items in sales_test.csv, where:

- **Class A: Top 20% of items contributing to 80% of total sales.**
- **Class B: Next 30% of items contributing to 15% of total sales.**
- **Class C: Remaining 50% of items.**

```
WITH total_sales AS (
  SELECT ITEM, SUM(NS_ORDER) AS item_sales
  FROM sales_test
  GROUP BY ITEM
),
ranked_sales AS (
  SELECT ITEM, item_sales,
         NTILE(100) OVER (ORDER BY item_sales DESC) AS percen
  FROM total_sales
)
SELECT ITEM,
       CASE
         WHEN percentile <= 20 THEN 'A'
         WHEN percentile <= 50 THEN 'B'
```

```
          ELSE 'C'
        END AS abc_class
FROM ranked_sales;
```



Conclusion of the Supply Chain CEL Dataset Project

In this project, we conducted a detailed analysis of the Supply Chain CEL dataset, focusing on the two key files: `canceled_test.csv` and `sales_test.csv`. Our primary objectives were to understand order patterns, assess service levels, and classify items based on their sales performance.

1. **Volume Analysis**:
    - We examined the volume of orders placed, comparing successful sales with canceled orders. This analysis highlighted significant trends in customer behavior and order fulfillment.
2. **Cancellation Patterns**:
    - By analyzing canceled orders, we identified specific days of the week that had higher cancellation rates. Understanding these patterns can

help businesses implement strategies to reduce cancellations, such as improving communication with customers or addressing potential issues before they lead to order cancellations.

3. **Service Level Evaluation**:
   - We calculated the service levels for each customer, determining the percentage of shipped items compared to ordered items. This metric is crucial for assessing customer satisfaction and operational efficiency within the supply chain.
4. **ABC Classification**:
   - We performed an ABC classification of items based on their contribution to sales, enabling the business to prioritize inventory management strategies for high-impact items. This classification aids in optimizing stock levels, reducing holding costs, and ensuring that popular items are always available to meet customer demand.
5. **Insights and Recommendations**:
   - The findings suggest potential areas for operational improvement, including enhancing order processing and fulfillment strategies and refining customer engagement practices. By focusing on high-impact items and monitoring cancellation trends, businesses can improve service levels and overall customer satisfaction.

In summary, this project demonstrated the value of data analytics in optimizing supply chain operations. The insights gained can help inform strategic decisions, improve inventory management, and enhance customer service in the dynamic environment of supply chain management. Future work may include a more in-depth investigation into the factors driving cancellations and a predictive analysis of customer ordering behaviors.