SUPPLY CHAIN AND INVENTORY MANAGEMENT

Data Analysis Project

Track:
Data Analysis Specialist

Group:
GIZ1-DAT1-G4E

Contents

Meet The Team:

Aya Ayman

Marian Youssef

Mary Magdy

Reem Nasr

Sameh El-Araby

Project Summary and Workflow

1. Introduction

This project focuses on analyzing supply chain and inventory management using data-driven insights. Our main objectives are to track trends, improve processes, and understand how factors like shipping modes, product classes, and customer segments influence inventory performance.

2. Data Sources

We sourced our data from Kaggle under the title "Supply Chain and Inventory Management." The dataset includes essential information on sales, shipping modes, and product classes, all of which are crucial for our analysis.

Up next, is a screenshot from (Sale_Shipment_Data) table, imported as a CSV file on Excel.

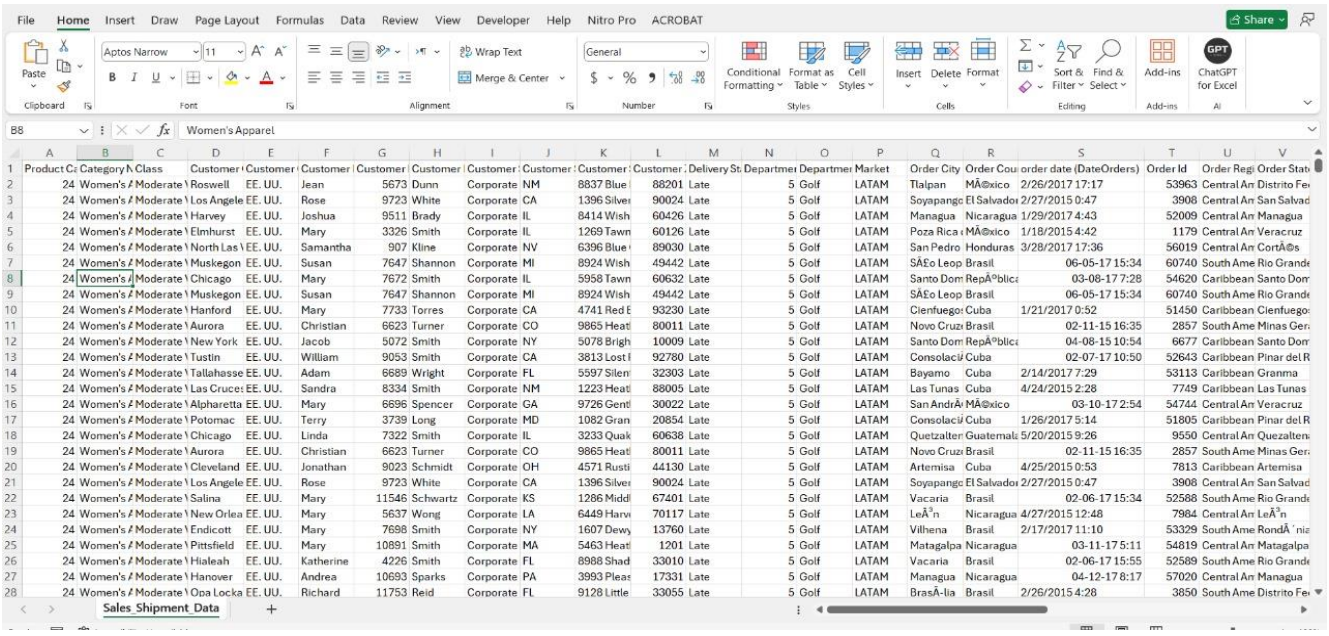
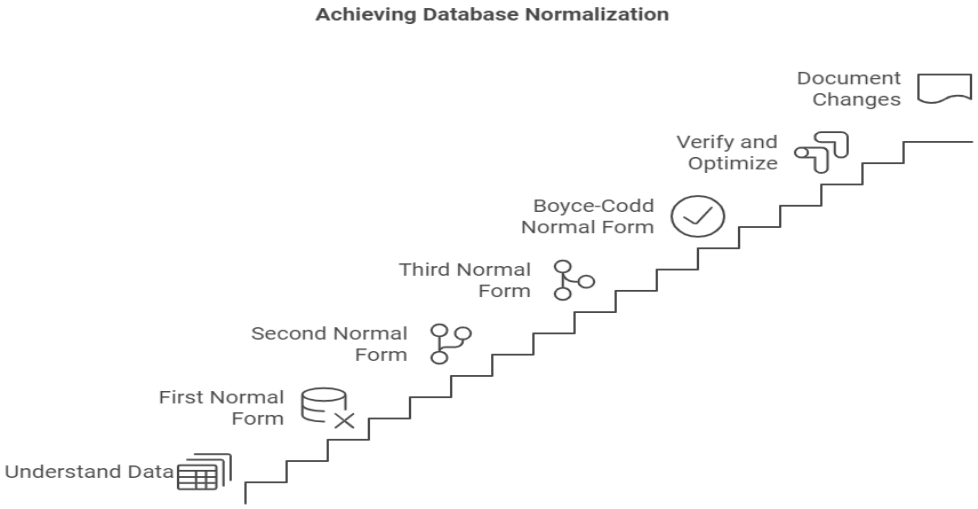


***Figure 1 - (Sales_Shipment_Data) Table*

Normalization Process

We began by examining the "Sales and Shipping Data" table, which contained redundant columns. We identified primary and foreign keys and split the data into five distinct tables: Orders, Order Items, Products, Customers, and Shipping.



***As per the diagram shown above, illustrating the normalization process.*

We created the separated tables using SQL, then transferred the data from the unified table to the new columns in the new table.

```
Use [Supply_Chain];

--Creating the customers Table

CREATE TABLE Customers (
    Customer_Id INT PRIMARY KEY,
    Customer_Fname NVARCHAR(50),
    Customer_Lname NVARCHAR(50),
    Customer_Segment NVARCHAR(50),
    Customer_City NVARCHAR(50),
    Customer_State NVARCHAR(50),
    Customer_Country NVARCHAR(50),
    Customer_Zipcode INT,
    Customer_Street NVARCHAR(50)
);

Alter Table Customers Add Sales_per_customer Float;
```

Figure 2 Creating the Customers Table

```
--To only transfer one record for each Customer_Id to Customers Table

WITH UniqueCustomers AS (
    SELECT *,
           ROW_NUMBER() OVER (PARTITION BY Customer_Id ORDER BY (SELECT NULL)) AS rn
    FROM [dbo].[Sales_Shipment_Data]
)
INSERT INTO [dbo].[Customers]
(Customer_Id, Customer_Fname,Customer_Lname,Customer_Segment,Customer_City,Customer_State,Customer_Country,Customer_Zipcode,Customer_Street,Sales_per_customer)
SELECT
Customer_Id, Customer_Fname,Customer_Lname,Customer_Segment,Customer_City,Customer_State,Customer_Country,Customer_Zipcode,Customer_Street, Sales_per_customer, Sales_per_customer
FROM UniqueCustomers
WHERE rn = 1;
```

***Figure 3 Transferring the data from the unified table to the "customers" table*

Applied the previous steps to all the intended new tables.

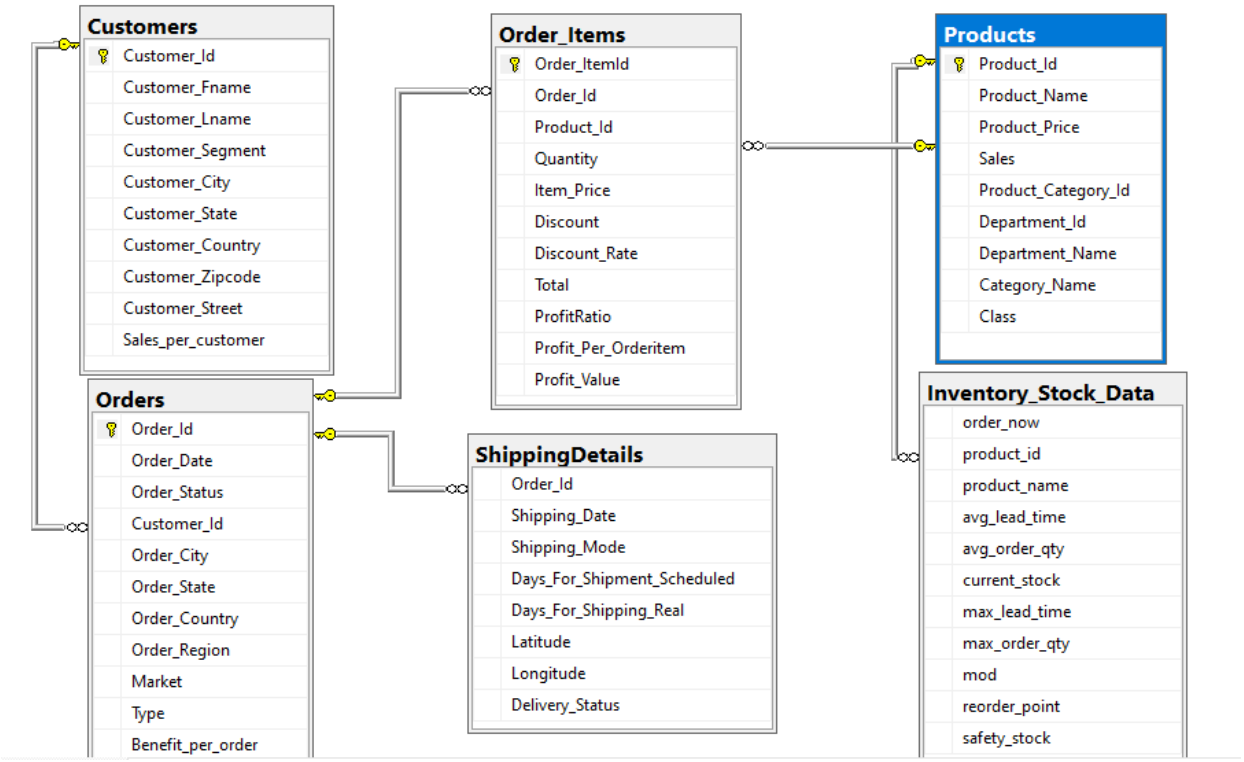
Now, we have 5 new tables and the (Inventory_Stock_Data) table.

Changes to Data Types

As we created the tables, we adjusted data types to handle decimal values and larger data types, ensuring the integrity of our data structure.

Database Diagram

We developed a database diagram by “SQL Server Management Studio” to illustrate the relationships and keys (Primary and Foreign) between the tables, facilitating a clearer understanding of our data structure.



3. Data Exploration

To be able to proceed with the data exploration, analysis and visualization, we created 2 views on SQL: (Customers_Orders_Shippings) & (Inventory_Product).

Then we explored the 2 views one at a time to check for any changes or any calculated columns to be added before proceeding with our analysis.

We connected the views to “Python”

```
import pandas as pd
import pyodbc
connection_string = 'DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-JLFVPGL;DATABASE=Supply_Chain;Trusted_Connection=yes'
conn = pyodbc.connect(connection_string)
df_customer_sales = pd.read_sql('SELECT * FROM Vw_Customers_Orders_Shipping', conn)
df_inventory_product_details = pd.read_sql('SELECT * FROM Vw_Product_Inventory', conn)
conn.close()
```

- Examples for some transformation applied:

  Columns that return the year, month and day from the order date column.

4. Analysis Focus Areas

Our initial exploration revealed critical insights regarding reorder points, lead times, current stock levels, and safety stock. We focused on identifying products that required immediate reordering due to low stock levels. Additionally, we analyzed sales aspects that impact inventory and stock management, including sales volumes and seasonality. Understanding these factors allowed us to better predict demand fluctuations and optimize inventory levels throughout the year.

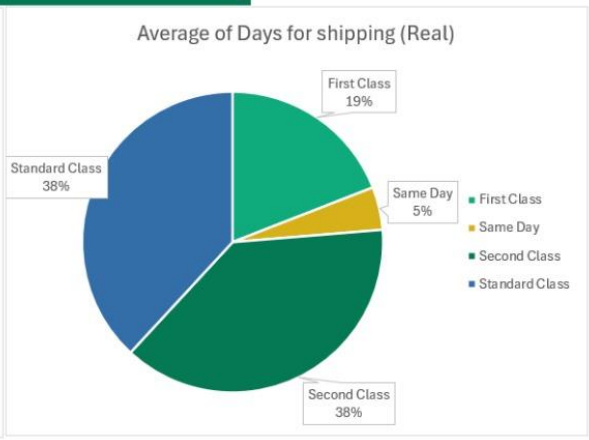
Key areas of analysis included:

- Calculate high level metrics like, total sale value, total sale units, inventory value, inventory quantity, profit value, number of distinct products, number of distinct categories, number of distinct products
- Top 10 Most ordered products
- Sales units/value by time (by each week, month, year, quarter)
- Product Order quantity trend by time (by each week, month, year, quarter)
- Which shipping mode is more efficient in terms of not delaying?
- Number of orders, sales, quantity by order status.
- Inventory by class.

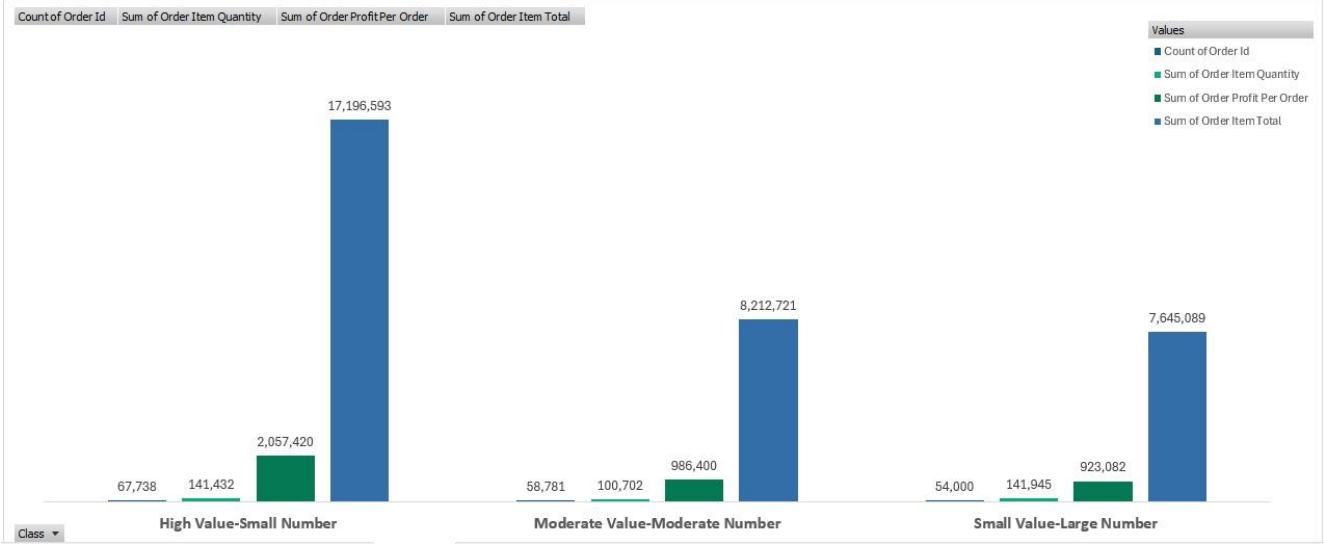
- As mentioned before, we worked on analyzing and exploring the data, using different tools. Here is some work done on excel.

Avearga Days of Shipping for each Class

Row Labels	Average of Days for shipping (Real)	Average of Days for shipment (Scheduled)
First Class	2	1
Same Day	0	0
Second Class	4	2
Standard Class	4	4
Grand Total	**4**	**3**



Row Labels	Count of Order Id	Sum of Order Item Quantity	Sum of Order Profit Per Order	Sum of Order Item Total
High Value-Small Number	67,738	141,432	2,057,420	17,196,593
Moderate Value-Moderate Number	58,781	100,702	986,400	8,212,721
Small Value-Large Number	54,000	141,945	923,082	7,645,089
Grand Total	**180,519**	**384,079**	**3,966,903**	**33,054,402**



- Moreover, we made some data analysis queries on “SQL Server Management Studio”

```sql
---Analysis of Shipping Modes
SELECT
    S.Shipping_Mode,
    COUNT(S.Order_Id) AS TotalShipments,
    AVG(DATEDIFF(day, O.[Order_Date], S.[Shipping_Date])) AS AvgDaysToShip
FROM
    [dbo].[ShippingDetails] S
JOIN
    Orders O ON O.Order_Id = S.Order_Id
GROUP BY
    S.Shipping_Mode
ORDER BY
    AvgDaysToShip;
```

100 %

Results | Messages

	Shipping_Mode	TotalShipments	AvgDaysToShip
1	Same Day	3571	0
2	First Class	10079	2
3	Standard Class	39324	3
4	Second Class	12778	4

```sql
---Total Orders per region
SELECT
    O.Order_Region,
    COUNT(S.Order_Id) AS TotalOrders
FROM
[dbo].[ShippingDetails] s
JOIN
    Orders O ON S.Order_Id = O.Order_Id
GROUP BY
    O.Order_Region
ORDER BY
    TotalOrders Desc;
```

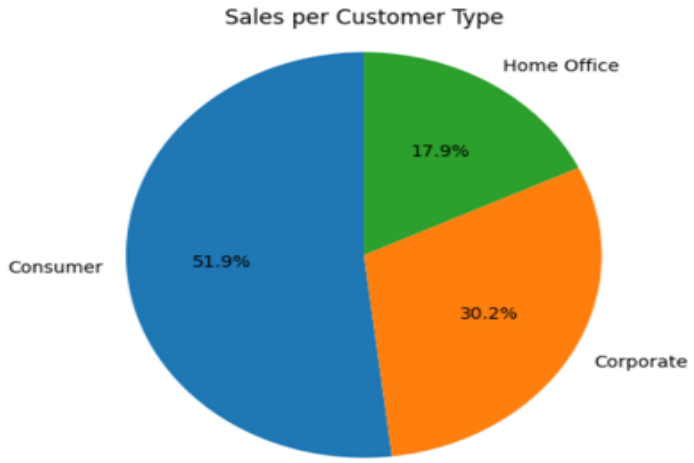
100 %

Results | Messages

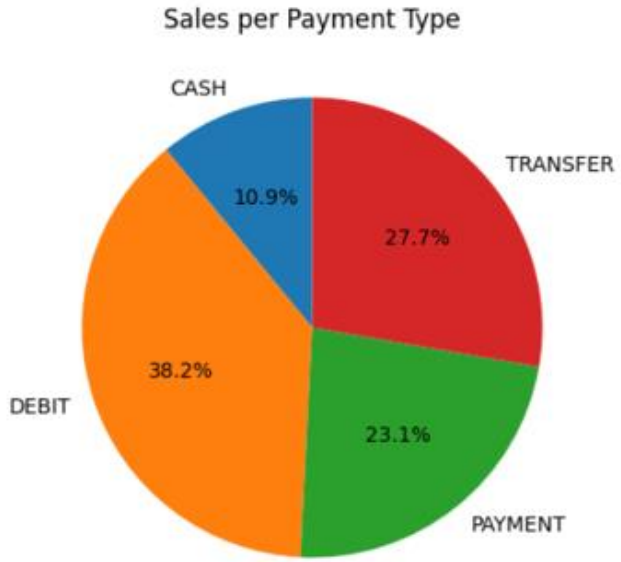
	Order_Region	TotalOrders
1	Western Europe	10010
2	Central America	9396
3	South America	4979
4	Oceania	4362
5	Southeast Asia	4356
6	Northern Europe	3716
7	Southern Europe	3543
8	South Asia	3335
9	Eastern Asia	3318
10	Caribbean	2806

➢ And using “Python” as well.

```
sales_by_customer_type = df_customer_sales.groupby('Customer_Segment')['Order_ID'].nunique().reset_index(name='Distinct_Order_Count')
plt.figure(figsize=(5, 5))
plt.pie(sales_by_customer_type['Distinct_Order_Count'],
        labels=sales_by_customer_type['Customer_Segment'],
        autopct='%1.1f%%',
        startangle=90)
plt.title('Sales per Customer Type')
plt.axis('equal')
plt.show()
```

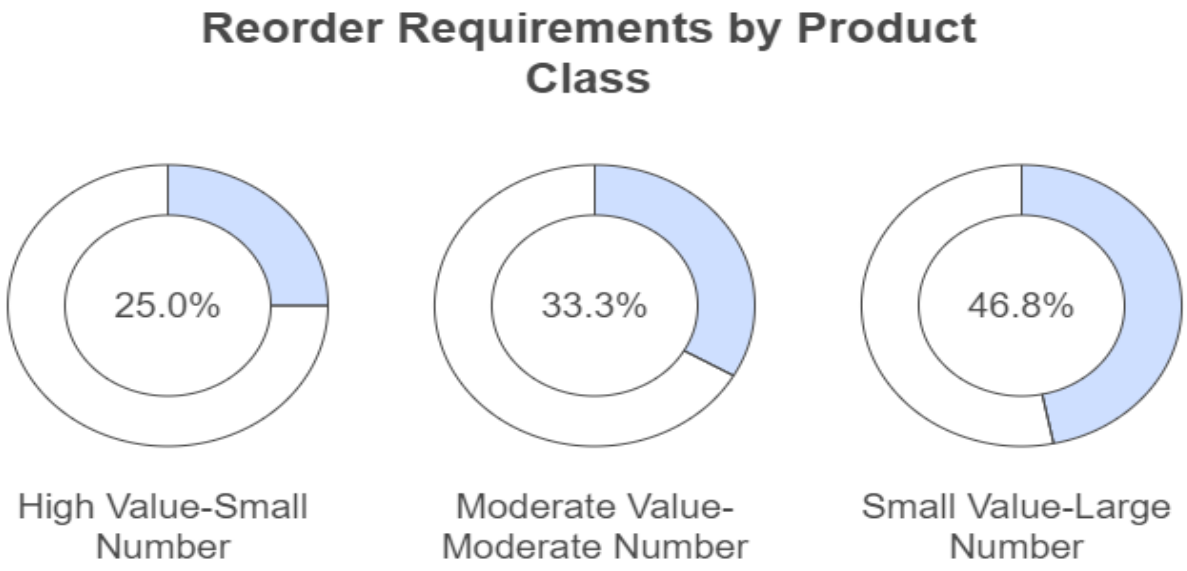


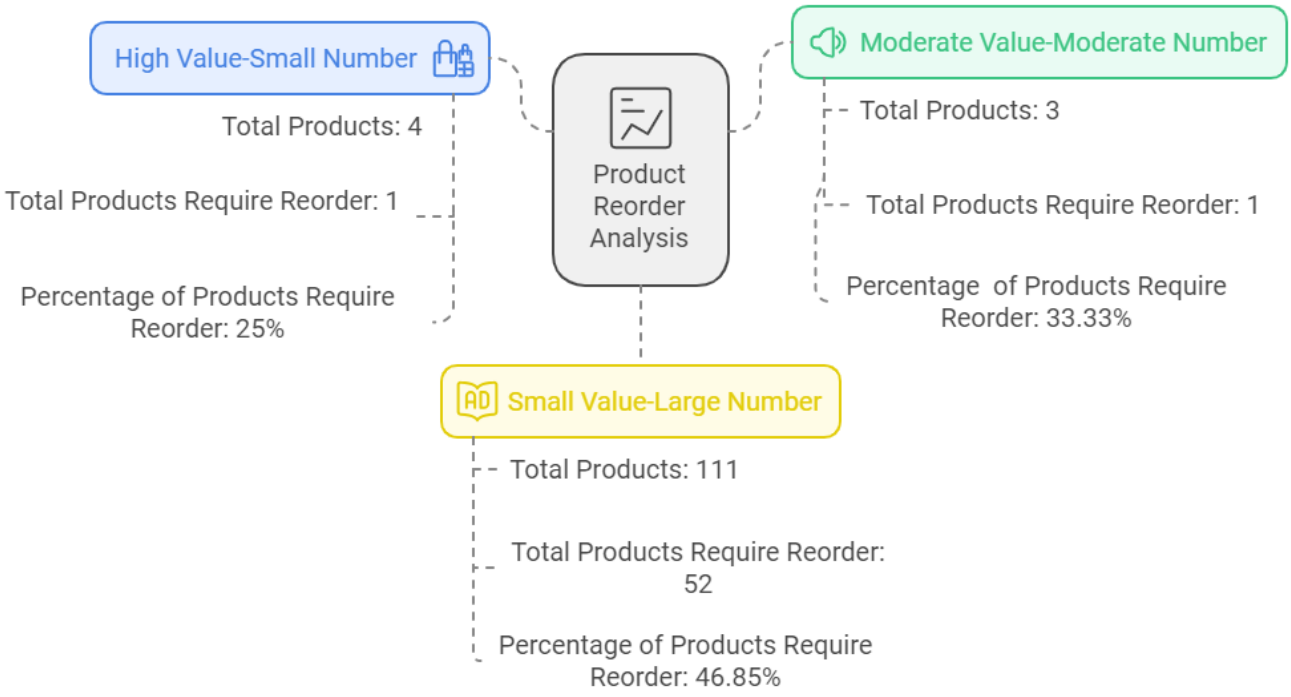

```
sales_by_Payment_type = df_customer_sales.groupby('Type_of_Payment')['Total'].sum()
plt.figure(figsize=(5, 5))
sales_by_Payment_type.plot(kind='pie', autopct='%1.1f%%', startangle=90)
plt.title('Sales per Payment Type')
plt.ylabel('')
plt.show()
```



5. Key Findings

We discovered significant trends related to products class, their sales volumes and their impact on inventory management accordingly. Visuals such as charts and graphs support our findings.





For more consolidated visuals we created dashboards on Tableau.

These dashboards included all the insights correlated to the supply chain and inventory management process.

6. Challenges and Limitations

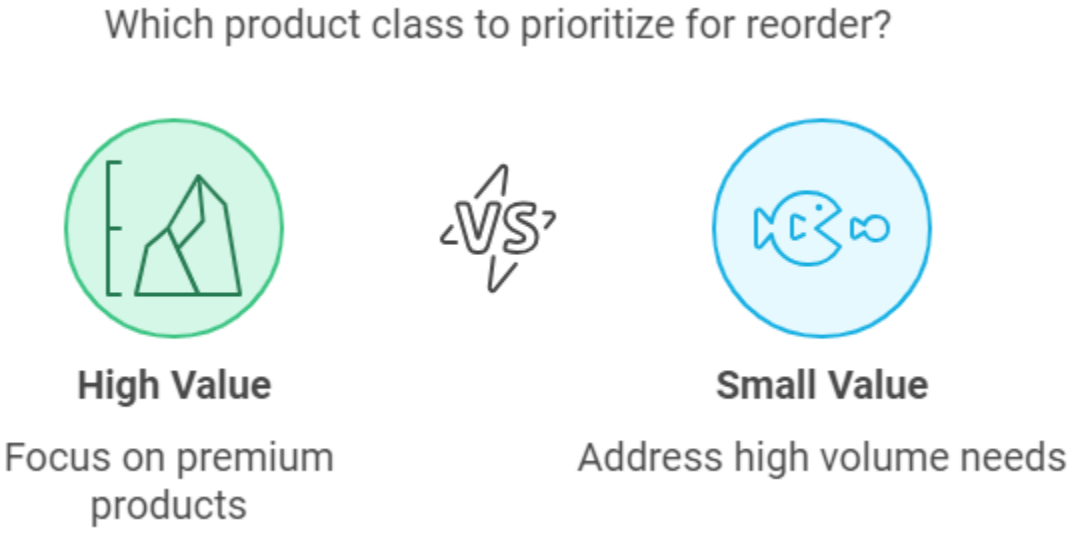
Throughout the analysis, we faced challenges related to data completeness. Additional data on:

- freight costs.
- shipping methods.
- Stockouts history.
- Inventory holding cost.

The presence of such data would have enhanced our evaluation of suppliers and the inventory process.

7. Recommendations and Actionable insights

As per the products classification, there are some that return high profit and others of smaller value but get ordered the most.



➢ In the dataset, there is a column "Class" and here is a walkthrough what it represents as it is crucial to the supply chain and inventory performance:

1. High Value - Small Number (A-Class Items)

   - These are the most expensive or high-value items but are usually few in number (i.e., low volume).
   - These items contribute the most to the total inventory value.
   - Usually about 10-20% of the items, but they account for about 70-80% of the value.
   - Require tight control because of their high cost and criticality to the business.

2. Moderate Value - Moderate Number (B-Class Items)

   - These are items with moderate value and quantity. They are of medium importance.
   - Usually about 30% of the items, contributing to about 15-25% of the total value.
   - Less critical than A-class items but still need efficient inventory control.

3. Small Value - Large Number (C-Class Items)

- These items are low in value but are high in quantity and volume.
- About 50-60% of the items, but they only account for around 5-10% of the total value.
- They are often less critical but still necessary for operations.

➢ One more thing noticed in the data is the sales seasonality that should be considered in the inventory management to avoid stockout during high seasons.