**The National College Jayanagar (Autonomous)**

7th Block, Jayanagar, Bengaluru - 560070



Business Intelligence Project

Harshitha K R

(V Sem BCA Data science)

U03PA21S0112

CONTENTS

1.ACKNOWLEDGEMENT

2.INTRODUCTION

3.ABSTRACT

4.WORKING

5.SOURCE CODE

6.GRAPHS

7.OUTPUT

8.CONCLUSION

9.REFERENCE LINK

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who contributed to the completion of this Business Intelligence project on inventory management data analysis.

I extend my deepest gratitude to Dharani Madam for her guidance, invaluable insights, and unwavering support throughout this project. Their expertise and encouragement significantly enhanced the quality and depth of this analysis.

Furthermore, I am grateful to The National College, Jayanagar for providing the opportunities essential for the successful execution of this project.

Lastly, I acknowledge the numerous individuals and resources, including scholarly works, articles, and databases, whose contributions formed the foundation of this analysis.

Thank you all for your invaluable contributions and support.

Sincerely,

Harshitha K R

INTRODUCTION

Inventory Management Data Analysis

An Inventory Management System (IMS) is a software application that helps businesses keep track of their stocked goods. It involves managing and organizing inventory levels, orders, sales, and deliveries. The primary goal is to ensure that products are available when needed, reduce carrying costs, and prevent stockouts or overstocks.

ABSTRACT:

Any Manufacturing Company is a medium-sized manufacturing company that produces electronic components. They have a wide range of products and maintain an inventory of raw materials, work-in-progress (WIP), and finished goods. The company has been experiencing issues with inventory management, including stockouts, excess inventory, and increased carrying costs. The management team wants to conduct an inventory analysis to identify areas for improvement and optimize their inventory management practices.

**Objectives:**  
The primary objectives of the inventory analysis are as follows:

Determine the optimal inventory levels for raw materials, WIP, and finished goods.  
Identify opportunities to reduce stockouts and excess inventory.  
Analyze inventory turnover and carrying costs to optimize working capital.  
Streamline the procurement and production processes to improve efficiency.  
Develop a sustainable inventory management strategy for future growth.  
Data Available:  
The following data is available for analysis:

**Inventory records**: Detailed records of all inventory transactions, including purchases, production, sales, and adjustments.Demand data: Historical sale data for different products.  
Lead time data: Information on the time required to receive raw materials and produce finished goods.  
Cost data: Information on the cost of raw materials, production, and carrying costs.  
Tasks:  
To address the objectives, the following tasks need to be performed:

**Demand forecasting**: Analyze historical sales data to forecast future demand for different products accurately.  
ABC analysis: Classify inventory items based on their value and importance to prioritize management efforts.  
Economic Order Quantity (EOQ) analysis: Determine the optimal order quantity for raw materials to minimize ordering and carrying costs.  
Reorder point analysis: Calculate the reorder point for each product to avoid stockouts.  
**Lead time analysis:** Assess the lead time for raw materials and production to optimize inventory levels.  
Carrying cost analysis: Calculate the carrying costs associated with inventory to identify areas for cost reduction.  
**Process improvement**: Identify bottlenecks and inefficiencies in the procurement and production processes and suggest improvements.  
Inventory turnover analysis: Calculate inventory turnover ratios to assess the efficiency of inventory management.  
Deliverables:  
Based on the analysis, the following deliverables will be provided:

**Optimal inventory levels:** Recommendations for the optimal inventory levels of raw materials, WIP, and finished goods.  
Reorder point and EOQ calculations: Reorder point and economic order quantity for each product to minimize stockouts and costs.  
Process improvement recommendations: Suggestions for streamlining the procurement and production processes to improve efficiency.  
**Inventory turnover analysis:** Assessment of inventory turnover ratios and recommendations for improving working capital efficiency.  
**Inventory management strategy**: Development of a comprehensive inventory management strategy for XYZ Manufacturing Company.  
By conducting this inventory analysis, XYZ Manufacturing Company aims to improve its inventory management practices, reduce costs, and enhance customer satisfaction by ensuring the availability of products.

WORKING

Inventory management data analysis involves examining various data points related to inventory levels, demand, sales, and supply chain operations to make informed decisions.

1. Data Import:

Multiple CSV files are read into Pandas DataFrames. These include files related to purchase prices, beginning inventory, ending inventory, invoices, final purchases, and final sales.

2. Data Cleaning:

For each DataFrame, there is an inspection for missing values (`isnull().sum()`), and in some cases, the rows with missing values are dropped (`dropna()`).

3. Data Exploration:

Univariate Analysis:

Calculation of counts for the top 5 vendors based on 'VendorName' in the 'PurchasePrice' DataFrame.

Visualization of the top 5 vendors using a pie chart.

Bar plot showing the top 10 brands with the highest prices based on the 'Price' column in the 'PurchasePrice' DataFrame.

Bivariate Analysis:

Visualizations showing relationships between vendors and quantities purchased, top vendors with the highest amounts paid, top brands with the highest purchase prices, top vendors with the highest sale amounts, top brands with the highest sale amounts, brands with the highest excise tax, and top vendors with the highest sale quantities.

4. Visualizations:

Matplotlib and Seaborn libraries are used to create various visualizations such as bar plots and pie charts to explore and understand different aspects of the data, like vendor performance, brand prices, sales amounts, and quantities.

5. Insights:

The analysis provides insights into top-performing vendors, brands with high prices, sales amounts, and quantities purchased/sold.

6. Derived Features:

Additional columns like 'Total Amount' are created by summing up existing columns ('Dollars', 'Freight', etc.) to calculate total amounts.

Overall, this is a comprehensive analysis of inventory data, aiming to extract insights into vendor performance, pricing, sales, and quantities. The visualizations help in understanding the distributions and relationships between various parameters within the datasets.

By leveraging data analysis, businesses can enhance efficiency, reduce costs, improve customer satisfaction, and make strategic decisions that positively impact their bottom line.

SOURCE CODE

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

PurchasePrice = pd.read\_csv(r"C:\Users\Harshitha\Desktop\BI\2017PurchasePricesDec.csv")

BegInv = pd.read\_csv(r"C:\Users\Harshitha\Desktop\BI\BegInvFINAL12312016.csv")

EndInv = pd.read\_csv(r"C:\Users\Harshitha\Desktop\BI\EndInvFINAL12312016.csv")

Invoice = pd.read\_csv(r"C:\Users\Harshitha\Desktop\BI\InvoicePurchases12312016.csv")

Final\_Purchase = pd.read\_csv(r"C:\Users\Harshitha\Desktop\BI\PurchasesFINAL12312016.csv")

Final\_Sales = pd.read\_csv(r"C:\Users\Harshitha\Desktop\BI\SalesFINAL12312016.csv")

PurchasePrice.shape

PurchasePrice.head()

PurchasePrice.isnull().sum()

PurchasePrice[PurchasePrice['Description'].isnull()]

PurchasePrice.dropna(inplace=True)

PurchasePrice.dtypes

PurchasePrice.describe()

BegInv.shape

BegInv.head()

BegInv.isnull().sum()

BegInv.dtypes

BegInv.describe()

EndInv.shape

EndInv.head()

EndInv.isnull().sum()

EndInv[EndInv['City'].isnull()]

EndInv.dropna(inplace=True)

Invoice.shape

Invoice.head()

Invoice.isnull().sum()

Final\_Sales.shape

Final\_Sales.head()

Final\_Sales.isnull().sum()

import pandas as pd

# Load the data into a DataFrame

PurchasePrice = pd.read\_csv(r"C:\Users\Handi\Desktop\BI\2017PurchasePricesDec.csv")

top\_vendor\_counts = PurchasePrice['VendorName'].value\_counts().head(5)

print(top\_vendor\_counts)

plt.figure(figsize=(3,3))

mylabels=["MARTIGNETTI COMPANIES","ULTRA BEVERAGE COMPANY LLP" ,'M S WALKER INC','PERFECTA WINES','E & J GALLO WINERY']

plt.pie(PurchasePrice.VendorName.value\_counts()[0:5],labels=mylabels ,autopct='%0.1f%%',radius=1.25,

wedgeprops={'edgecolor':'white'}

,textprops={'size':10,},shadow=True)

plt.title('Top 5 vendors\n')

plt.show()

plt.figure(figsize=(15,5))

vc = PurchasePrice.groupby(['Description'])['Price'].max().sort\_values(ascending=False)[:10]

g = sns.barplot(x= vc.index , y = vc.values , data = PurchasePrice)

for i in range(10):

value = vc[i]

g.text(y = value -2, x= i+0.25, s= value, color='black', ha = 'center', fontsize= 10)

plt.title ('Top 10 Brands with heighest Price tags')

plt.xlabel('Brands')

plt.xticks(rotation= 45)

plt.show()

Invoice.head(1)

Invoice['Total Amount'] = Invoice['Dollars']+ Invoice['Freight']

Invoice.head(2)

plt.figure(figsize=(12,5))

vc = Invoice.groupby(['VendorName'])['Quantity'].max().sort\_values(ascending=False)[:10]

g = sns.barplot(x=vc.index, y= vc.values, data = Invoice, palette='magma')

for i in range(10):

value = vc[i]

g.text(y= value-2 , x= i, s= value, color='black', ha = 'center', fontsize =10)

plt.title('Vendors vs Quantity')

plt.xlabel('Vendor Names')

plt.xticks(rotation= 35)

plt.show()

plt.figure(figsize=(12,5))

vc = Invoice.groupby(['VendorName'])['Total Amount'].max().sort\_values(ascending=False)[:10]

g = sns.barplot(x= vc.index , y= vc.values , data = Invoice)

for i in range(10):

value = vc[i]

g.text(y= value -2 , x= i+0.125, s= value, color = 'black', ha='center',fontsize=8)

plt.title('Top 10 vendors with heighest amount paid')

plt.xlabel('Vendor Names')

plt.xticks(rotation = 35)

plt.show()

plt.figure(figsize=(15,5))

vc = Final\_Purchase.groupby(['Description'])['PurchasePrice'].max().sort\_values(ascending=False)[:10]

g= sns.barplot(x=vc.index, y= vc.values , data = Final\_Purchase)

for i in range(10):

value = vc[i]

g.text(x=i , y= value , s = value , ha='center', color='black', fontsize=10)

plt.title('Top 10 Brands with heighest Purchase Price')

plt.xlabel('Brands')

plt.xticks(rotation=35)

plt.show()

Final\_Sales.head(2)

Final\_Sales['Total Amount'] = Final\_Sales['SalesDollars']+Final\_Sales['ExciseTax']

Final\_Sales.head(2)

plt.figure(figsize=(12,5))

vc= Final\_Sales.groupby(['VendorName'])['Total Amount'].sum().sort\_values(ascending=False)[:10]

g = sns.barplot(x=vc.index , y= vc.values , data = Final\_Sales, palette="Set2")

for i in range(10):

value = vc[i]

g.text(y=value-2 , x= i+0.125 , s = value , ha='center', color='black', fontsize=10)

plt.title('Top 10 vendors with heighest sale amount')

plt.xlabel('Brands')

plt.xticks(rotation = 25)

plt.show()

plt.figure(figsize=(12,5))

vc= Final\_Sales.groupby(['Description'])['Total Amount'].max().sort\_values(ascending=False)[:10]

g = sns.barplot(x=vc.index , y= vc.values , data = Final\_Sales, palette="Set3")

for i in range(10):

value = vc[i]

g.text(y=value-2 , x= i+0.125 , s = value , ha='center', color='black', fontsize=10)

plt.title('Top 10 Brands with heighest sale amount')

plt.xlabel('Brands')

plt.xticks(rotation = 25)

plt.show()

plt.figure(figsize=(12,5))

vc= Final\_Sales.groupby(['Description'])['ExciseTax'].max().sort\_values(ascending=False)[:10]

g = sns.barplot(x=vc.index , y= vc.values , data = Final\_Sales, palette="Set1")

for i in range(10):

value = vc[i]

g.text(y=value-2 , x= i+0.125 , s = value , ha='center', color='black', fontsize=10)

plt.title('top 10 Brands with heighest Excise Tax')

plt.xlabel('Brands')

plt.xticks(rotation = 25)

plt.show()

plt.figure(figsize=(12,5))

vc= Final\_Sales.groupby(['VendorName'])['SalesQuantity'].max().sort\_values(ascending=False)[:10]

g = sns.barplot(x=vc.index , y= vc.values , data = Final\_Sales, palette="Set3")

for i in range(10):

value = vc[i]

g.text(y=value-2 , x= i+0.125 , s = value , ha='center', color='black', fontsize=10)

plt.title('Top 10 vendors with heighest Sale Quantity')

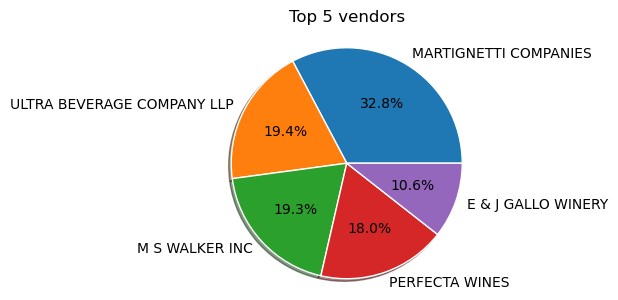
plt.xlabel('Vendors')

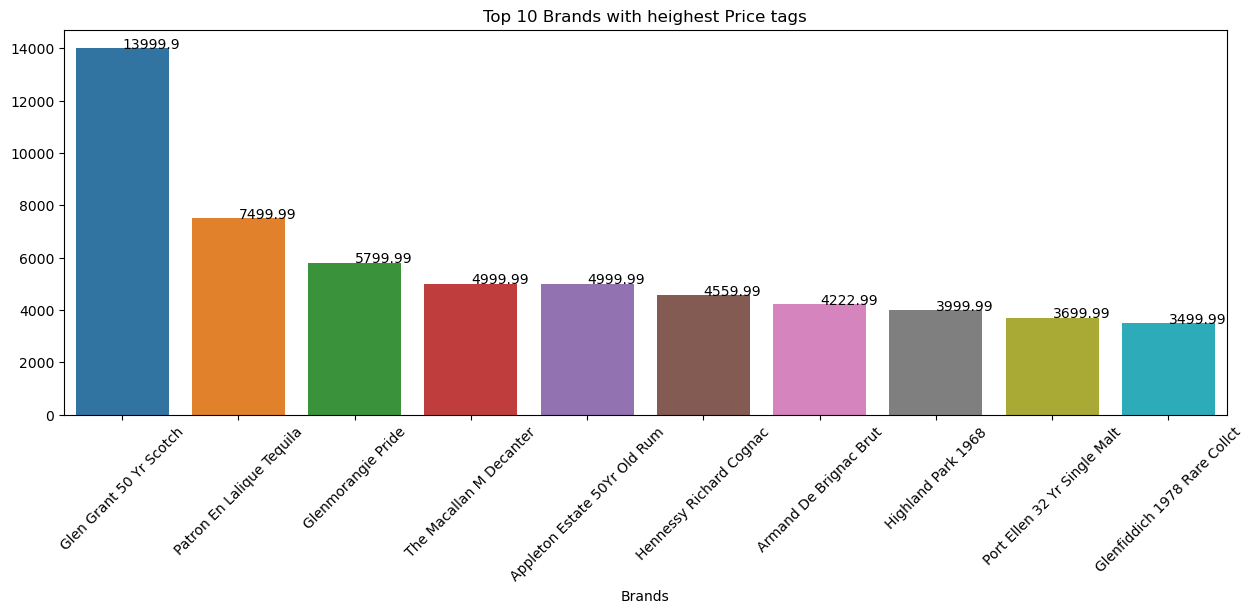
plt.xticks(rotation = 25)

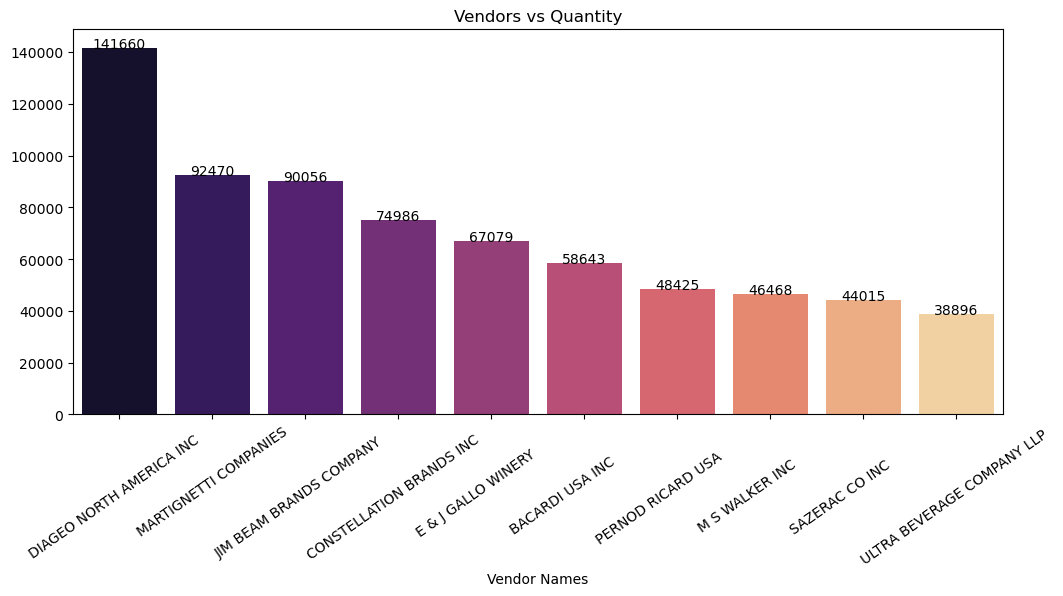
plt.show()

GRAPHS

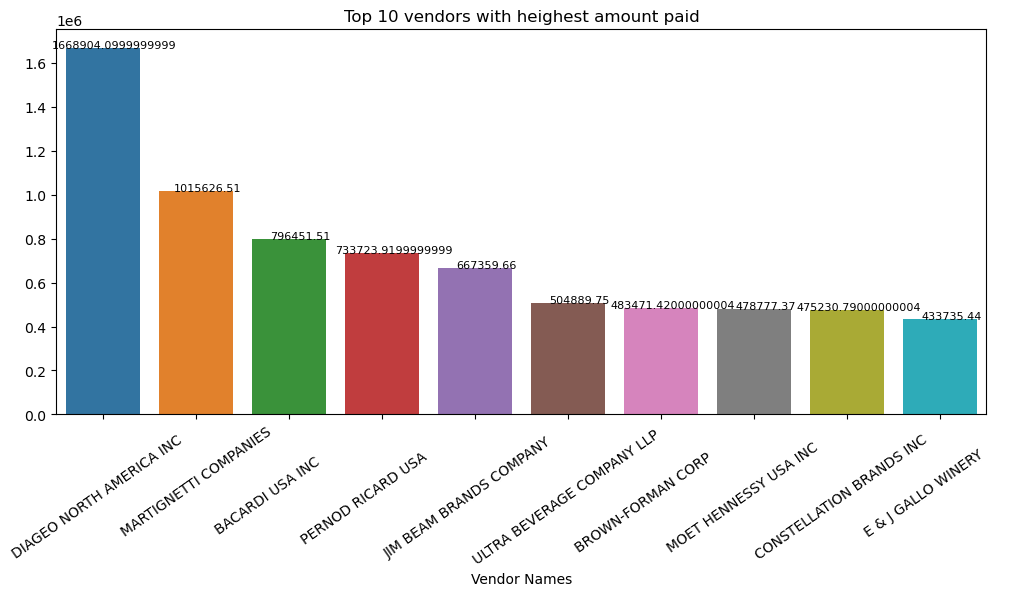
1.This pie chart gives a visualization of vendors who are on the top 5.



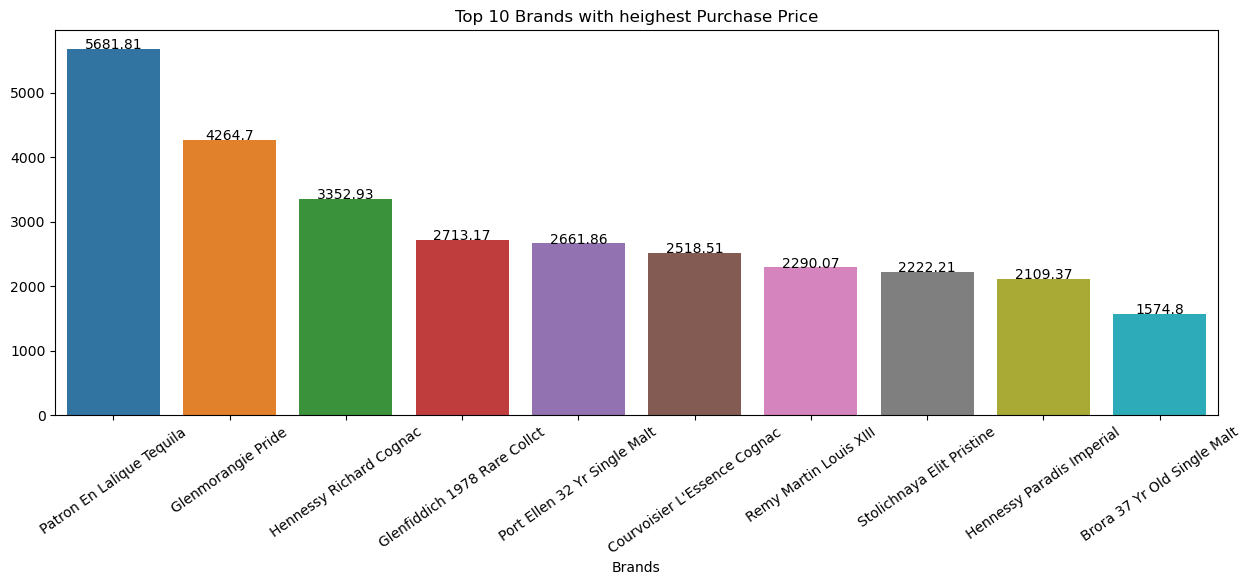
2. This bar graph gives visualization on the top 10 brands with the highest prices.

3. 

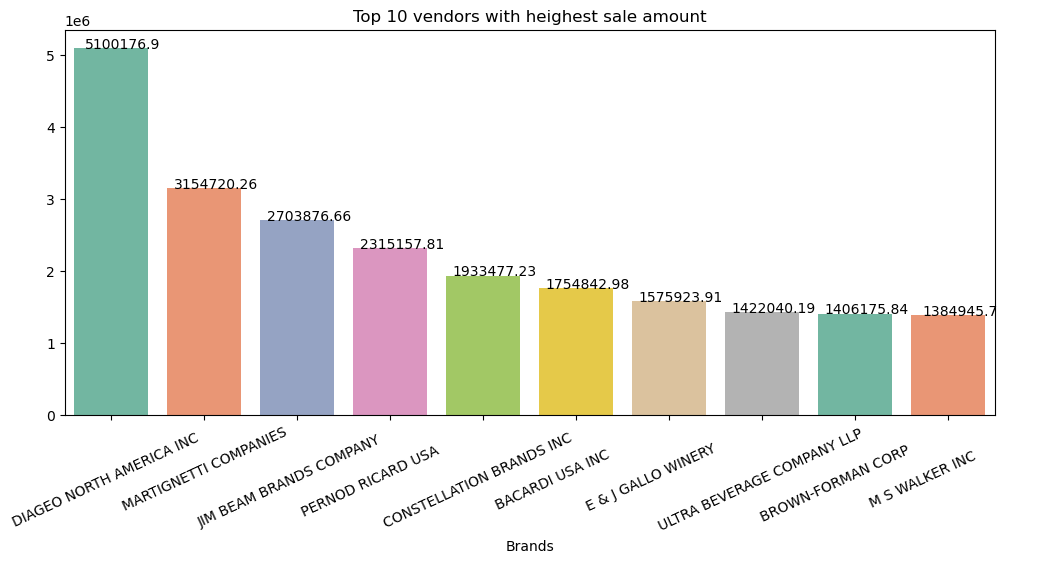
The above bar graph provides visualization about the quantity that each vendor is providing.

4. 

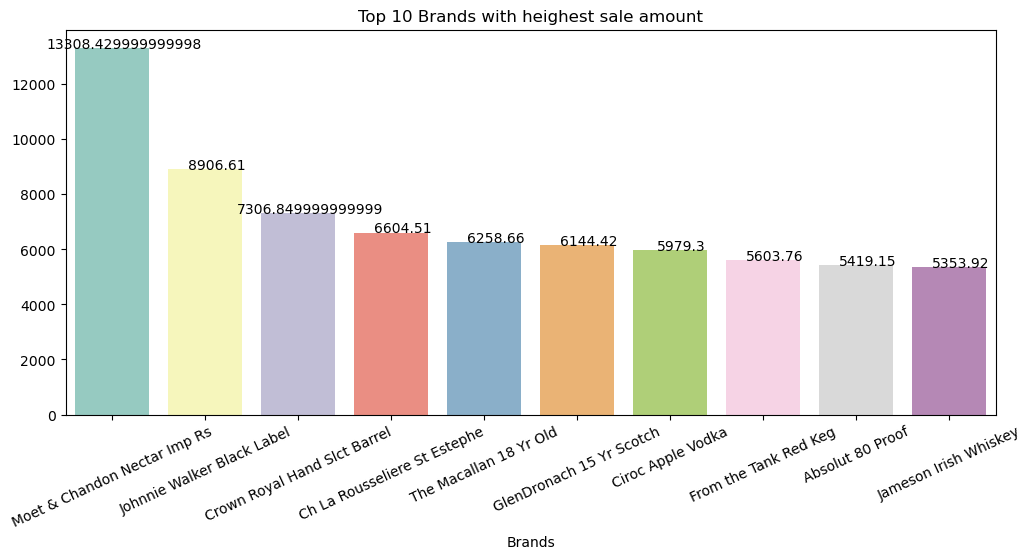
The above graph provides insights on the top 10 vendors who is getting paid the highest amount.

5. 

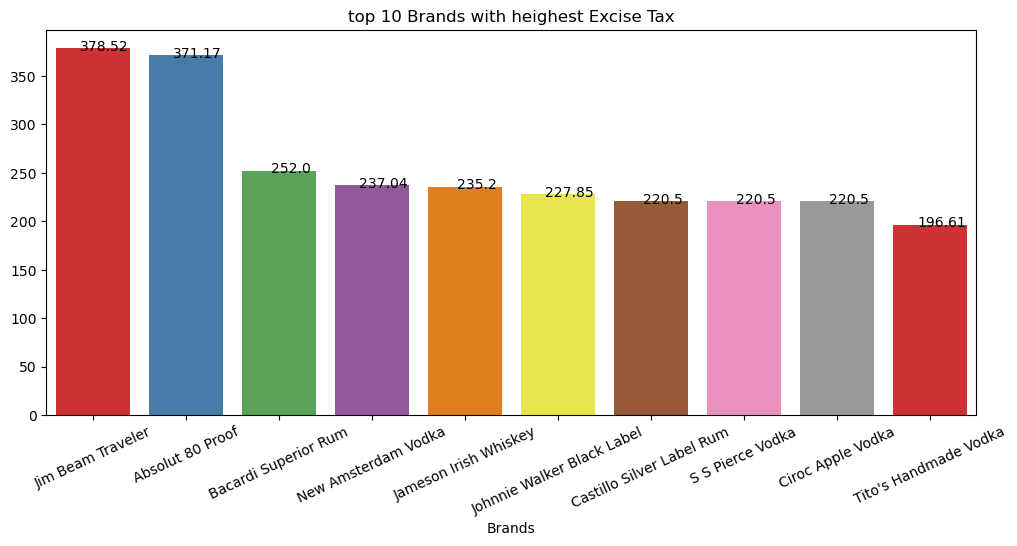
The above bar graph provides insights on the top 10 brands that is highest purchased by the customers.

6. 

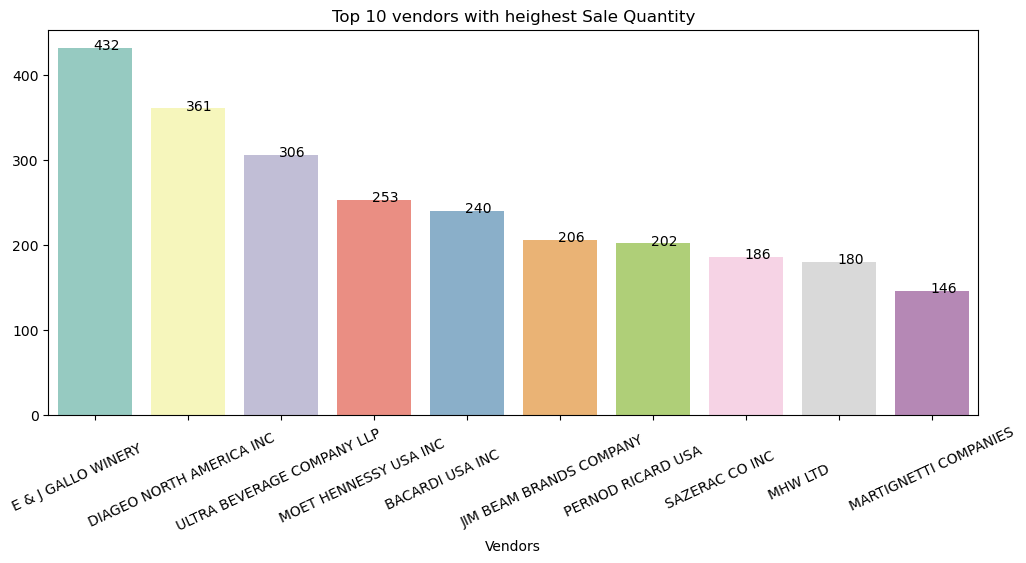
The above graph provides insights on the 10 vendors who hast the highest sales of their products.

7. 

The above graph provides insights on the top 10 brands which has highest sales.

8. 

The above graph provides insights on the top 10 brands that pays the highest excise tax.

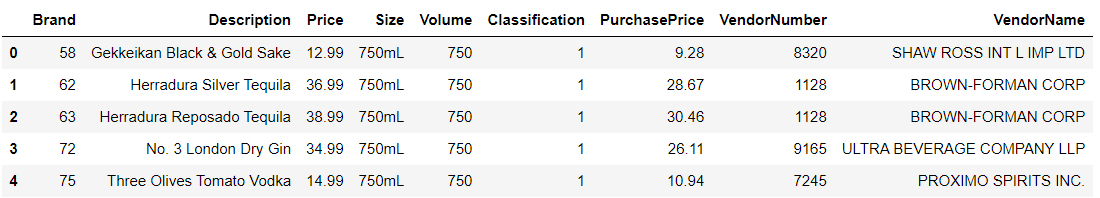
9. 

The above bar graph provides insights on the top 10 vendors with the highest sale quantity.

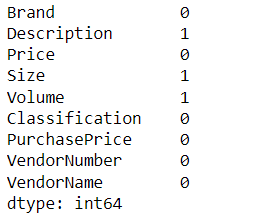
OUTPUT

1.(12261, 9)

The output of `PurchasePrice.shape` is a tuple representing the shape of the DataFrame `PurchasePrice`, indicating it has 12,261 rows and 9 columns.

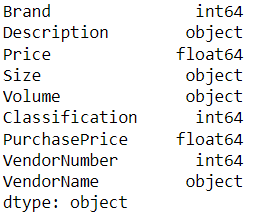
2. The output of `PurchasePrice.head()` displays the first five rows of the DataFrame `PurchasePrice`, showcasing the initial data and structure within those rows.

3. The output of `PurchasePrice.isnull().sum()` is a Series that shows the count of null values in each column of the `PurchasePrice` DataFrame.

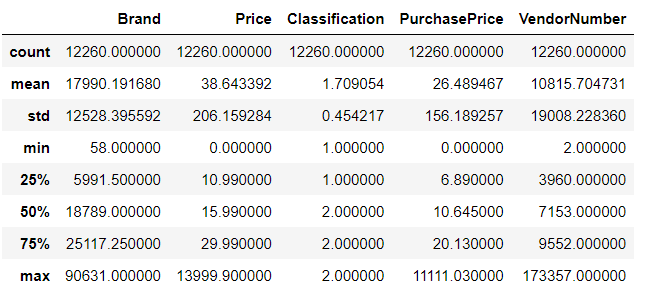


4. The code `PurchasePrice[PurchasePrice['Description'].isnull()]` identifies and displays rows in the 'PurchasePrice' dataset where the 'Description' column is missing or null.

5. The output of `PurchasePrice.dtypes` provides the data types of each column in the 'PurchasePrice' DataFrame, showcasing the specific data type assigned to each column, similar to the earlier code used to determine column data types.

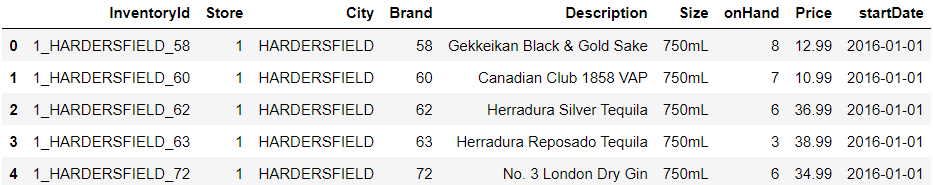


6. The output of `PurchasePrice.describe()` generates summary statistics for numerical columns in the 'PurchasePrice' DataFrame, including count, mean, standard deviation, minimum, quartiles, and maximum values for each numerical column, akin to the earlier code used to display statistical information about the dataset.

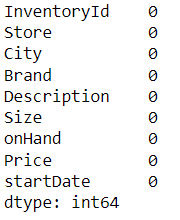


7. The output of `BegInv.shape` gives a tuple that represents the dimensions of the DataFrame 'BegInv', indicating the number of rows and columns present in that DataFrame, similar to previously used code to determine the shape of a DataFrame.

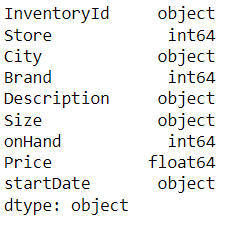
(206529, 9)

8. The output of `BegInv.head()` displays the first five rows of the DataFrame 'BegInv', presenting the initial data and structure contained within those rows, similar to the previously used code to show the beginning rows of a DataFrame.

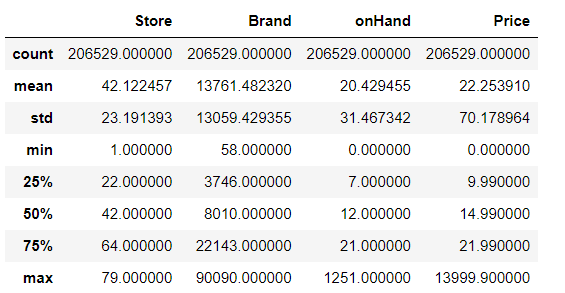
9. The output of `BegInv.isnull().sum()` is a Series that shows the count of null values in each column of the 'BegInv' DataFrame, similar to the previously used code to check for null values within a DataFrame.



10. The output of `BegInv.dtypes` provides the data types of each column in the 'BegInv' DataFrame, showcasing the specific data type assigned to each column, akin to the earlier code used to determine column data types.

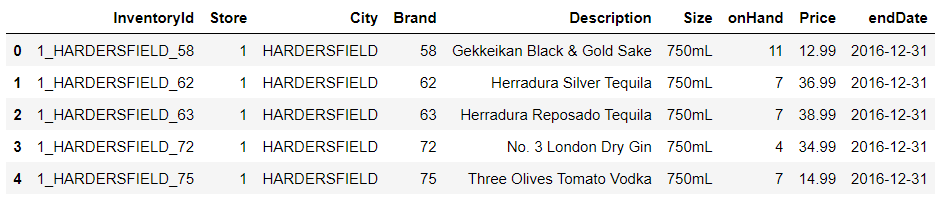


11. The output of `BegInv.describe()` generates summary statistics for numerical columns in the 'BegInv' DataFrame, including count, mean, standard deviation, minimum, quartiles, and maximum values for each numerical column, similar to the earlier code used to display statistical information about the dataset.

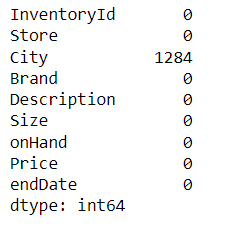


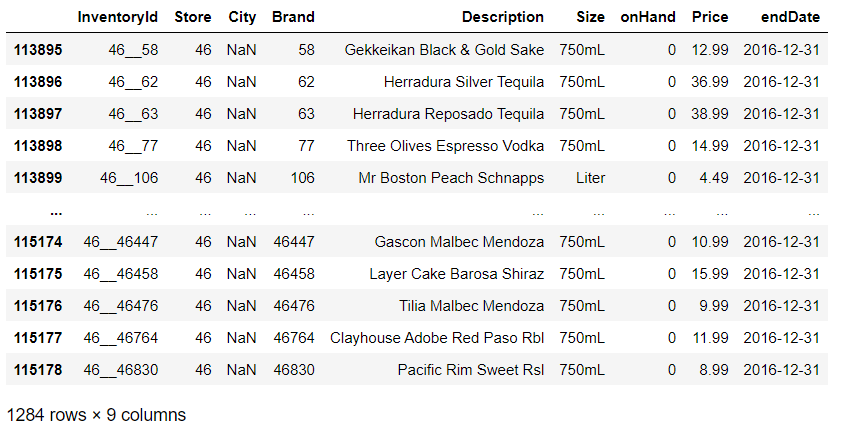
12. The `BegInv.shape` command provides the dimensions of the DataFrame 'BegInv', representing the number of rows and columns present in that particular DataFrame.

(224489, 9)

13. The output of `BegInv.head()` presents the initial five rows of the 'BegInv' DataFrame, showcasing the first entries and their corresponding details within those rows.

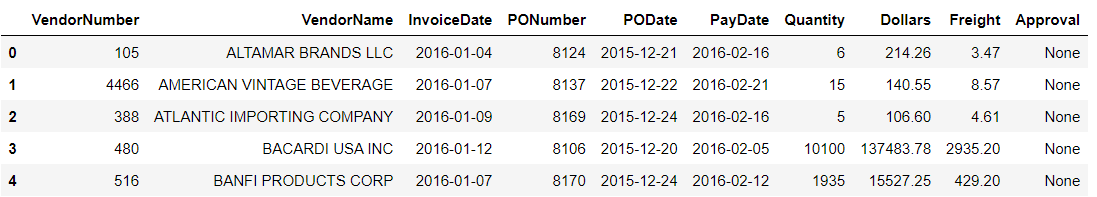
14. The output of `BegInv.isnull().sum()` provides a count of null or missing values in each column of the 'BegInv' DataFrame. This code is used to identify the number of missing values present in each column, similar to a previous command used to check for null values in a DataFrame.



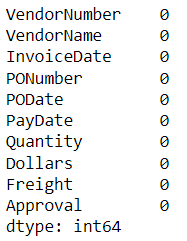
15. The output of `BegInv.dtypes` lists the data types of each column in the 'BegInv' DataFrame, showing the specific data type assigned to each column, just like a previous command used to display the data types of DataFrame columns.

16. The output of `Invoice.shape` represents the shape or dimensions of the 'Invoice' DataFrame. It typically displays a tuple indicating the number of rows and columns, respectively, within the DataFrame. For instance, if the output is (5543, 10), it means there are 5543 rows and 10 columns in the 'Invoice' DataFrame.

(5543, 10)

17. The output of `Invoice.head()` displays the first five rows of the DataFrame 'Invoice', presenting the initial data and structure contained within those rows, similar to a previous command used to showcase the beginning rows of a DataFrame.

18. The output of `Invoice.isnull().sum()` provides a count of null or missing values in each column of the 'Invoice' DataFrame. This code is used to identify the number of missing values present in each column, akin to a previous command used to check for null values in a DataFrame.

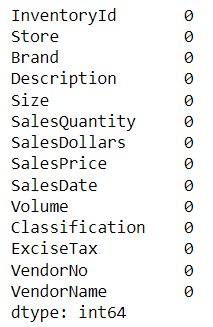


19. The command `Final\_Sales.shape` provides the dimensions of the DataFrame 'Final\_Sales', indicating the number of rows and columns present in that DataFrame.

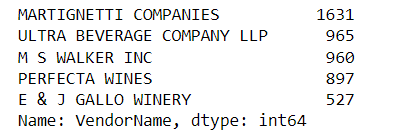
(1048575, 14)

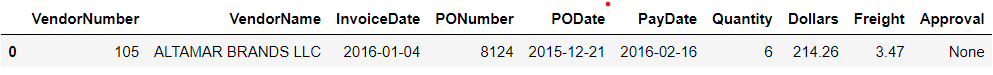
20. The output of `Final\_Sales.head()` displays the first five rows of the DataFrame 'Final\_Sales', presenting the initial data and structure contained within those rows, similar to a previous command used to showcase the beginning rows of a DataFrame.

21. The output of `Final\_Sales.isnull().sum()` provides a count of null or missing values in each column of the 'Final\_Sales' DataFrame. This command helps identify the number of missing values present in each column, similar to a previous command used to check for null values in a DataFrame.

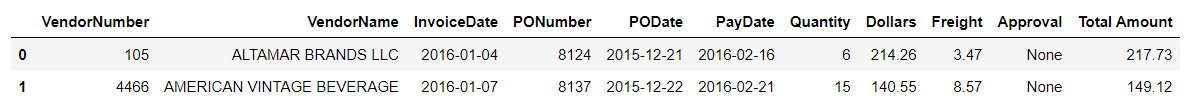


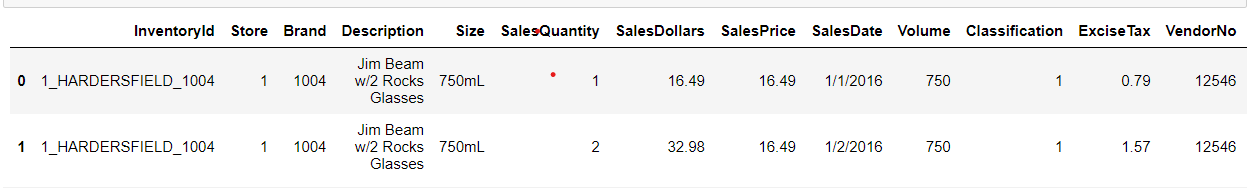
22. The code snippet calculates and displays the top five vendors along with their respective counts from the 'VendorName' column in the 'PurchasePrice' DataFrame.



23. The output of `Invoice.head(1)` displays the first row of the 'Invoice' DataFrame, showing the initial data and structure of that single row.

24. The code snippet assigns a new column 'Total Amount' by summing the 'Dollars' and 'Freight' columns in the 'Invoice' DataFrame and then displays the first two rows of the modified DataFrame, all in a single line:



25. The command `Final\_Sales.head(2)` displays the first two rows of the DataFrame 'Final\_Sales', showing the initial data and structure contained within those rows.

CONCLUSION

1. Vendor Performance:

Top Vendors: "MARTIGNETTI COMPANIES," "ULTRA BEVERAGE COMPANY LLP," and "M S WALKER INC" were among the top vendors based on the frequency of purchases.

Highest Sales Quantity: "Diageo North America inc" made the highest purchases in terms of quantity.

2. Product Insights:

High-Priced Brands: "Glan grant scotch" and "PEL Tequilla" were identified as the brands with the highest prices.

High-Priced Purchases: "PEL Tequilla" and "G. Pride" were the brands with the highest purchase prices.

3.Sales Analysis:

Top Sale Amounts: "Diageo North America" and "Martignetti Companies" had the highest sale amounts.

High Sale Quantities: "Diageo North America" was observed to have the highest sale quantities among vendors.

4.Financial Insights:

Total Amount Calculations: Additional columns were created to calculate total amounts by combining columns like 'Dollars,' 'Freight,' and 'ExciseTax' for better financial analysis.

5.Data Quality:

- The analysis involved handling missing values by dropping rows with null values in specific columns like 'Description,' 'Size,' 'Volume,' and 'City' to ensure data integrity.

6. Visualization Insights:

Visualizations such as bar plots and pie charts aided in understanding the distribution and relationships between vendors, brands, sales amounts, and quantities.

This analysis provides a comprehensive view of vendor performance, product pricing, sales trends, and financial aspects related to inventory management. It highlights key performers and valuable insights to aid decision-making for inventory stocking, vendor management, and financial planning within the observed dataset.

LINKS

[Inventory Data Analysis | Kaggle](https://www.kaggle.com/code/reyanshbhardwaj12/inventory-data-analysis) DATA ALSO WAS USED FROM THE SAME LINK.