

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import ...
from scipy import stats
df = pd.read_excel('Global Superstore.xlsx')
df.head()
```

```
{"type": "dataframe", "variable_name": "df", "shape": [1000, 10], "values": [{"id": 1, "Row ID": 1, "Order ID": 1, "Customer ID": 1, "Customer Name": "Customer 1", "Segment": "Segment A", "Order Date": "2020-01-01", "Ship Date": "2020-01-01", "Ship Mode": "Standard Shipping", "Sales": 100, "Profit": 20}, {"id": 2, "Row ID": 2, "Order ID": 2, "Customer ID": 2, "Customer Name": "Customer 2", "Segment": "Segment B", "Order Date": "2020-01-02", "Ship Date": "2020-01-02", "Ship Mode": "Standard Shipping", "Sales": 150, "Profit": 30}, {"id": 3, "Row ID": 3, "Order ID": 3, "Customer ID": 3, "Customer Name": "Customer 3", "Segment": "Segment C", "Order Date": "2020-01-03", "Ship Date": "2020-01-03", "Ship Mode": "Standard Shipping", "Sales": 200, "Profit": 40}, {"id": 4, "Row ID": 4, "Order ID": 4, "Customer ID": 4, "Customer Name": "Customer 4", "Segment": "Segment D", "Order Date": "2020-01-04", "Ship Date": "2020-01-04", "Ship Mode": "Standard Shipping", "Sales": 250, "Profit": 50}, {"id": 5, "Row ID": 5, "Order ID": 5, "Customer ID": 5, "Customer Name": "Customer 5", "Segment": "Segment E", "Order Date": "2020-01-05", "Ship Date": "2020-01-05", "Ship Mode": "Standard Shipping", "Sales": 300, "Profit": 60}, {"id": 6, "Row ID": 6, "Order ID": 6, "Customer ID": 6, "Customer Name": "Customer 6", "Segment": "Segment F", "Order Date": "2020-01-06", "Ship Date": "2020-01-06", "Ship Mode": "Standard Shipping", "Sales": 350, "Profit": 70}, {"id": 7, "Row ID": 7, "Order ID": 7, "Customer ID": 7, "Customer Name": "Customer 7", "Segment": "Segment G", "Order Date": "2020-01-07", "Ship Date": "2020-01-07", "Ship Mode": "Standard Shipping", "Sales": 400, "Profit": 80}, {"id": 8, "Row ID": 8, "Order ID": 8, "Customer ID": 8, "Customer Name": "Customer 8", "Segment": "Segment H", "Order Date": "2020-01-08", "Ship Date": "2020-01-08", "Ship Mode": "Standard Shipping", "Sales": 450, "Profit": 90}, {"id": 9, "Row ID": 9, "Order ID": 9, "Customer ID": 9, "Customer Name": "Customer 9", "Segment": "Segment I", "Order Date": "2020-01-09", "Ship Date": "2020-01-09", "Ship Mode": "Standard Shipping", "Sales": 500, "Profit": 100}, {"id": 10, "Row ID": 10, "Order ID": 10, "Customer ID": 10, "Customer Name": "Customer 10", "Segment": "Segment J", "Order Date": "2020-01-10", "Ship Date": "2020-01-10", "Ship Mode": "Standard Shipping", "Sales": 550, "Profit": 110}], "missing_values": df.isnull().sum().to_dict(), "print": f"Missing values:\n{missing_values}"}
```

Missing values:

Row ID	0
Order ID	0
Order Date	0
Ship Date	0
Ship Mode	0
Customer ID	0
Customer Name	0
Segment	0
City	0
State	0
Country	0
Postal Code	0
Region	0

```
def remove_outliers(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    df = df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
    return df

df = remove_outliers(df, 'Sales')
df = remove_outliers(df, 'Profit')
print("done")
```

done

```
summary_stats = df.describe()
print(f"Summary Statistics:\n{summary_stats}")
mean_sales = df['Sales'].mean()
median_sales = df['Sales'].median()
std_sales = df['Sales'].std()
mean_profit = df['Profit'].mean()
median_profit = df['Profit'].median()
std_profit = df['Profit'].std()
print(f"Sales Mean: {mean_sales} \n{std_sales}")
print(f"Profit Mean: {mean_profit} \n{std_profit}")
```

```
mean           2.959353      0.126369  
min            1.000000      0.000000  
25%           2.000000      0.000000  
50%           2.000000      0.000000  
75%           4.000000      0.200000  
max           14.000000      0.800000  
std            1.931421      0.201431  
Sales Mean: 59.329159581560944, Median: 53.22619362813223  
Profit Mean: 8.36607933768157, Median: 13.62697052083758
```

```
correlation_matrix = df.drop(columns=['Order Date', 'Ship Date', 'Customer ID', 'State', 'Country', 'Market', 'Region', 'Sub-Category', 'Product Name', 'Postal Code'])  
print(f"Correlation Matrix:\n{correlation_matrix.corr().round(2)}")  
plt.figure(figsize=(8, 6))  
sns.heatmap(correlation_matrix, annot=True, fmt='.2f', linewidths=0.5)  
plt.title('Correlation Heatmap')  
plt.show()
```

Correlation Matrix:

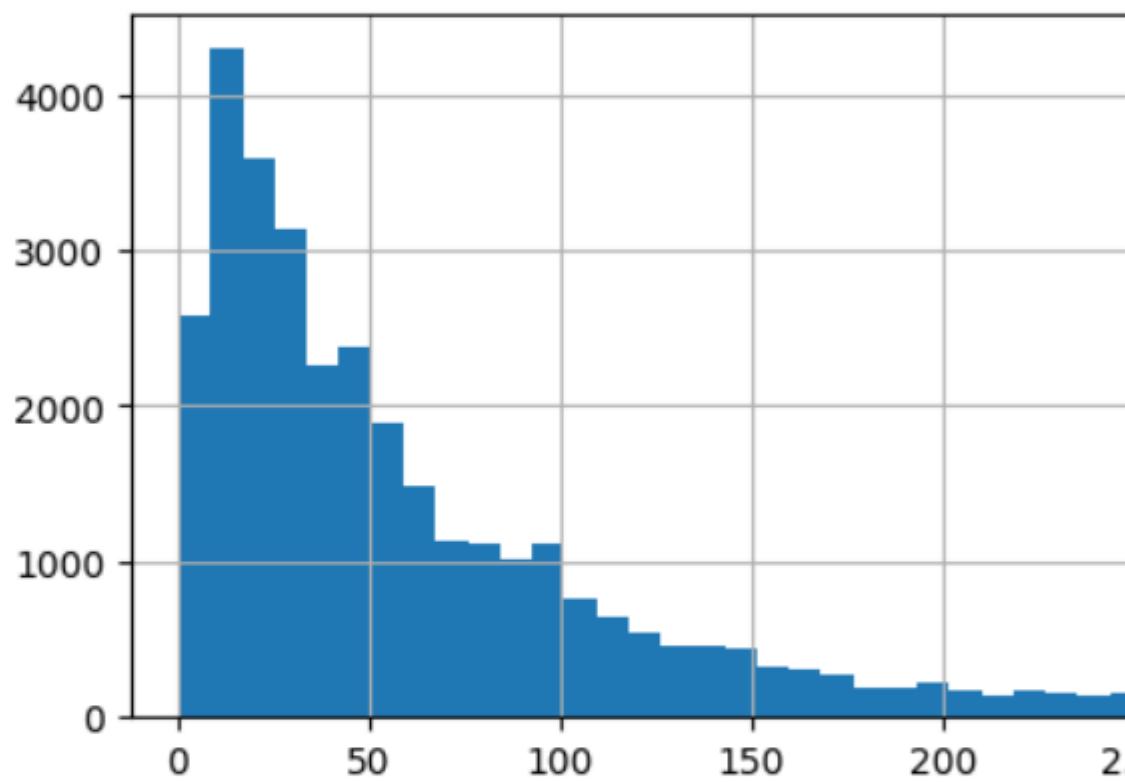
	Row ID	Postal Code
Profit	\	

Corr

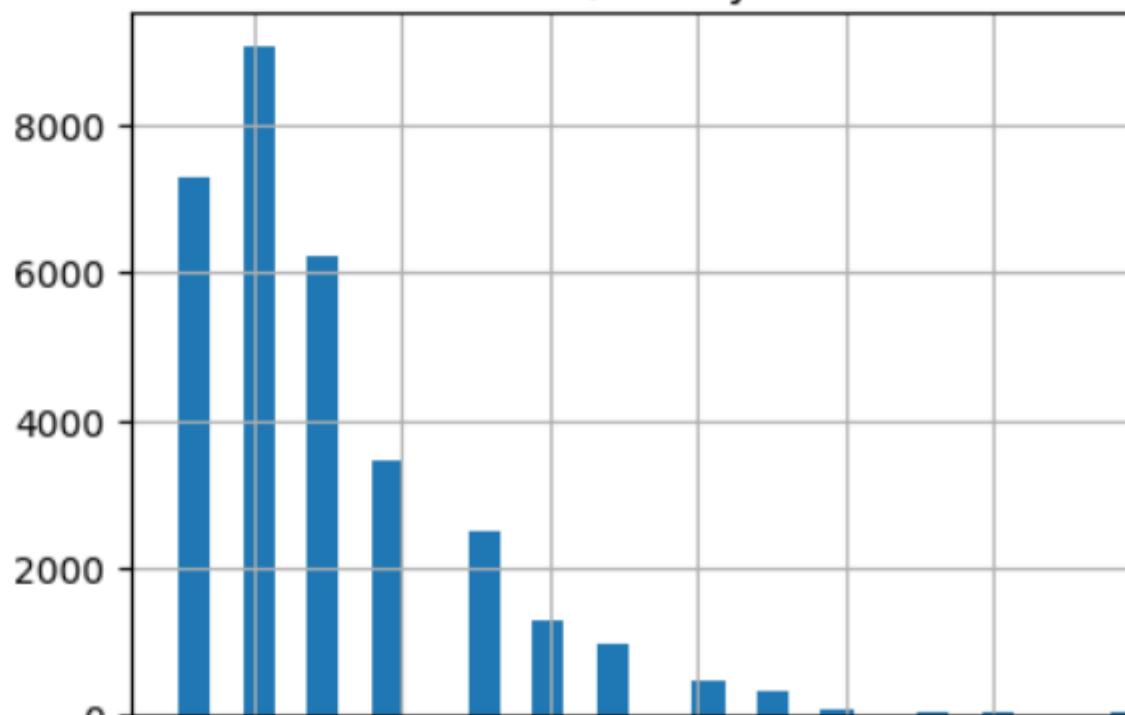
	Row ID	Postal Code	Sales
Row ID	1.00	0.02	-0.13
Postal Code	0.02	1.00	0.04
Sales	-0.13	0.04	1.00
Quantity	-0.19	0.02	0.26
Discount	0.07	0.04	-0.22
Profit	-0.05	-0.02	0.45
Shipping Cost	-0.10	0.03	0.72

## Distribution

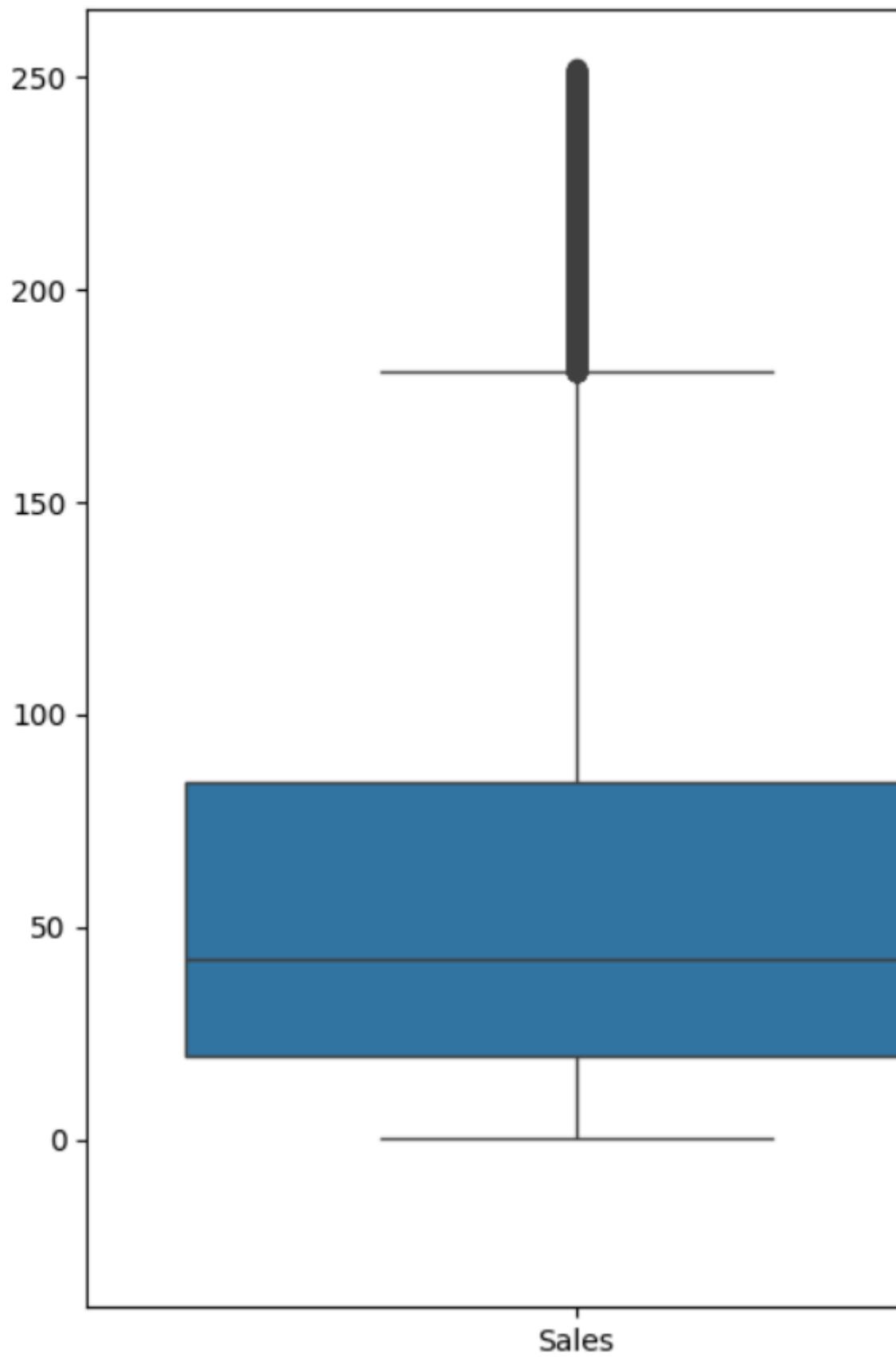
Sales

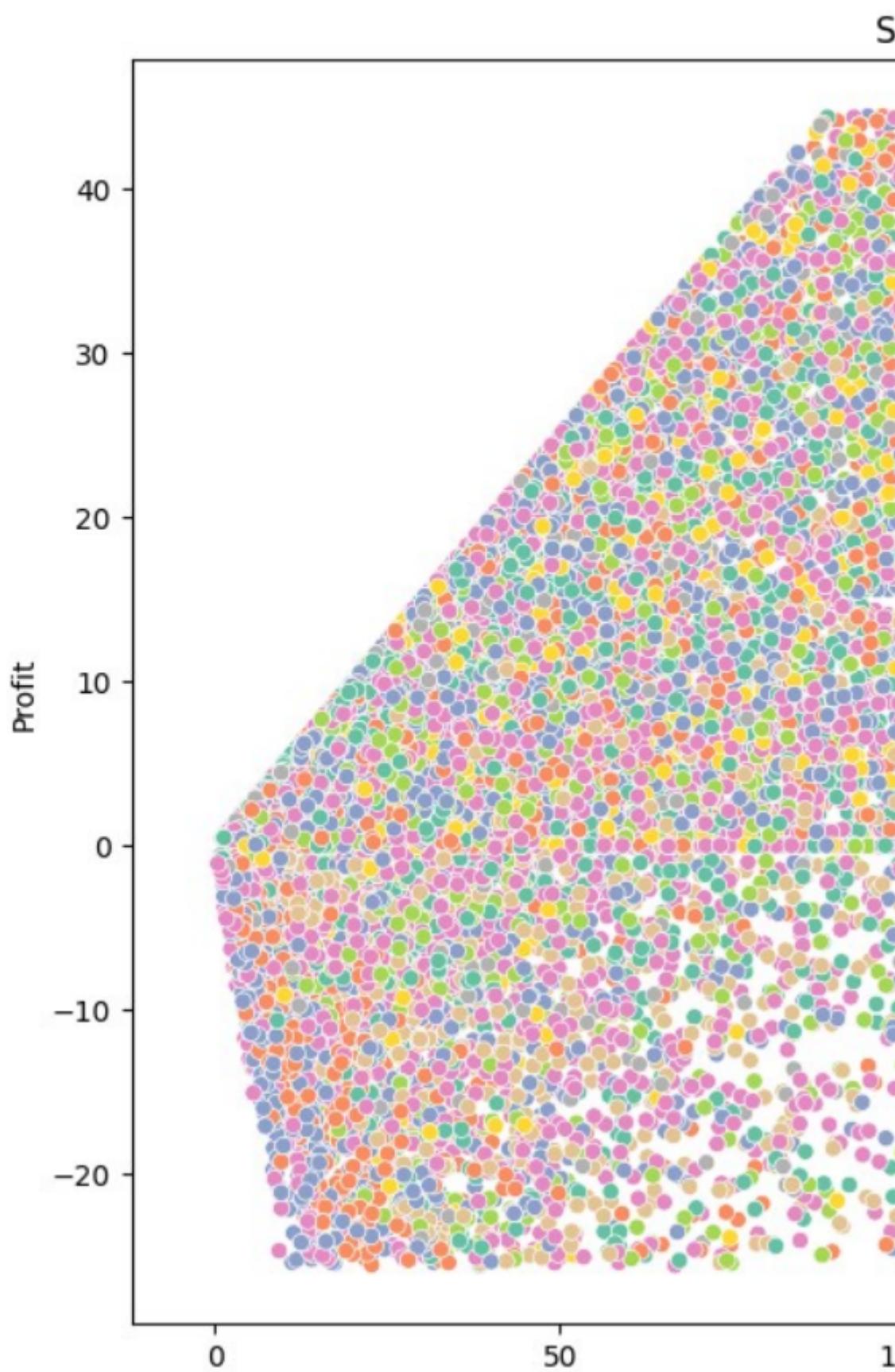


Quantity



Box





```
plt.figure(figsize=(10, 6))
df.groupby('ORDERDATE')['SALES']
plt.title("Sales Trend Over Time")
plt.xlabel("Date")
plt.ylabel("Total Sales")
plt.grid(True)
plt.show()
plt.figure(figsize=(8, 6))
sns.scatterplot(x='QUANTITYORDERED',
plt.title("Price Each vs Quantity")
plt.xlabel("Quantity Ordered")
plt.ylabel("Price Each")
plt.grid(True)
plt.show()
plt.figure(figsize=(8, 6))
df.groupby('COUNTRY')['SALES'].sum()
plt.title("Sales by Country")
plt.xlabel("Country")
plt.ylabel("Total Sales")
plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(8, 6))
df.groupby('PRODUCTLINE')
['SALES'].sum().sort_values().plot()
plt.title("Sales by Product Line")
```

```
print(f"Quantity Ordered Coefficients: {model.coef_}")

print("\nRecommendations:")
if model.coef_[0] > 0:
    print("Increasing Price Each high-margin products.")
else:
    print("Increasing Price Each low-margin products. Consider other strategies.")

if model.coef_[1] > 0:
    print("Increasing Quantity Order. However, be mindful of profit margins")
else:
    print("Increasing Quantity Order. However, be mindful of profit margins and increase Sales. Re-evaluate discount strategy")

best_country = df.groupby('COUNTRY').sum()
best_productline = df.groupby('PRODUCTLINE').sum()

print(f"\nTop performing country: {best_country['SALES'].idxmax()}")
print(f"Top performing product line: {best_productline['SALES'].idxmax()}")
```

```
Shape of the dataset: (2823, 24)
Data types:
    ORDERNUMBER           int64
```

Missing values:

ORDERNUMBER	0
QUANTITYORDERED	0
PRICEEACH	0
ORDERLINENUMBER	0
SALES	0
ORDERDATE	0
STATUS	0
QTR_ID	0
MONTH_ID	0
YEAR_ID	0
PRODUCTLINE	0
MSRP	0
PRODUCTCODE	0
CUSTOMERNAME	0
PHONE	0
ADDRESSLINE1	0
ADDRESSLINE2	2521
CITY	0
STATE	1486
POSTALCODE	76
COUNTRY	0
TERRITORY	1074
CONTACTLASTNAME	0
CONTACTFIRSTNAME	0

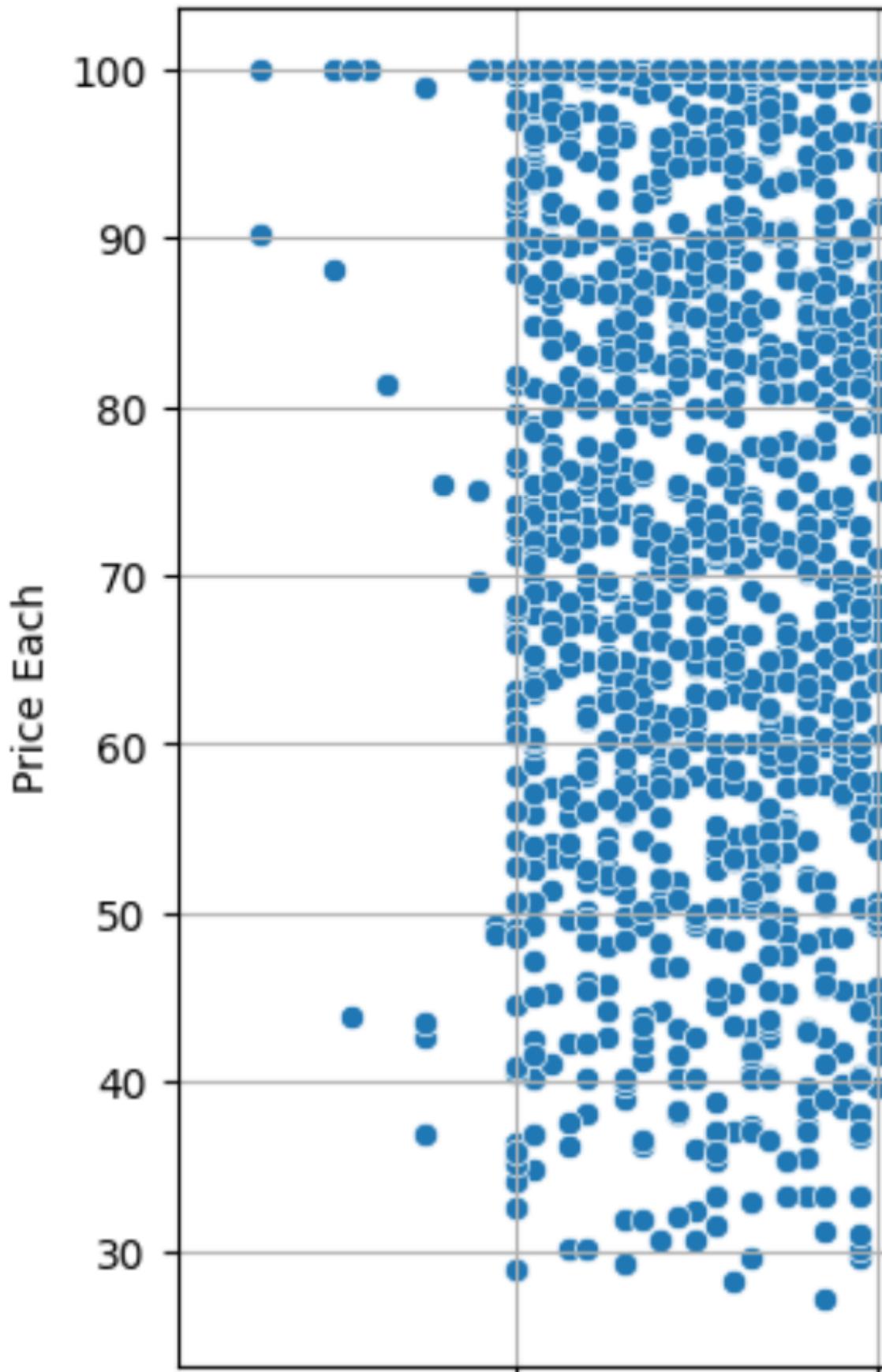
4	10/10/2003	0:00	Shipped
6505551386			

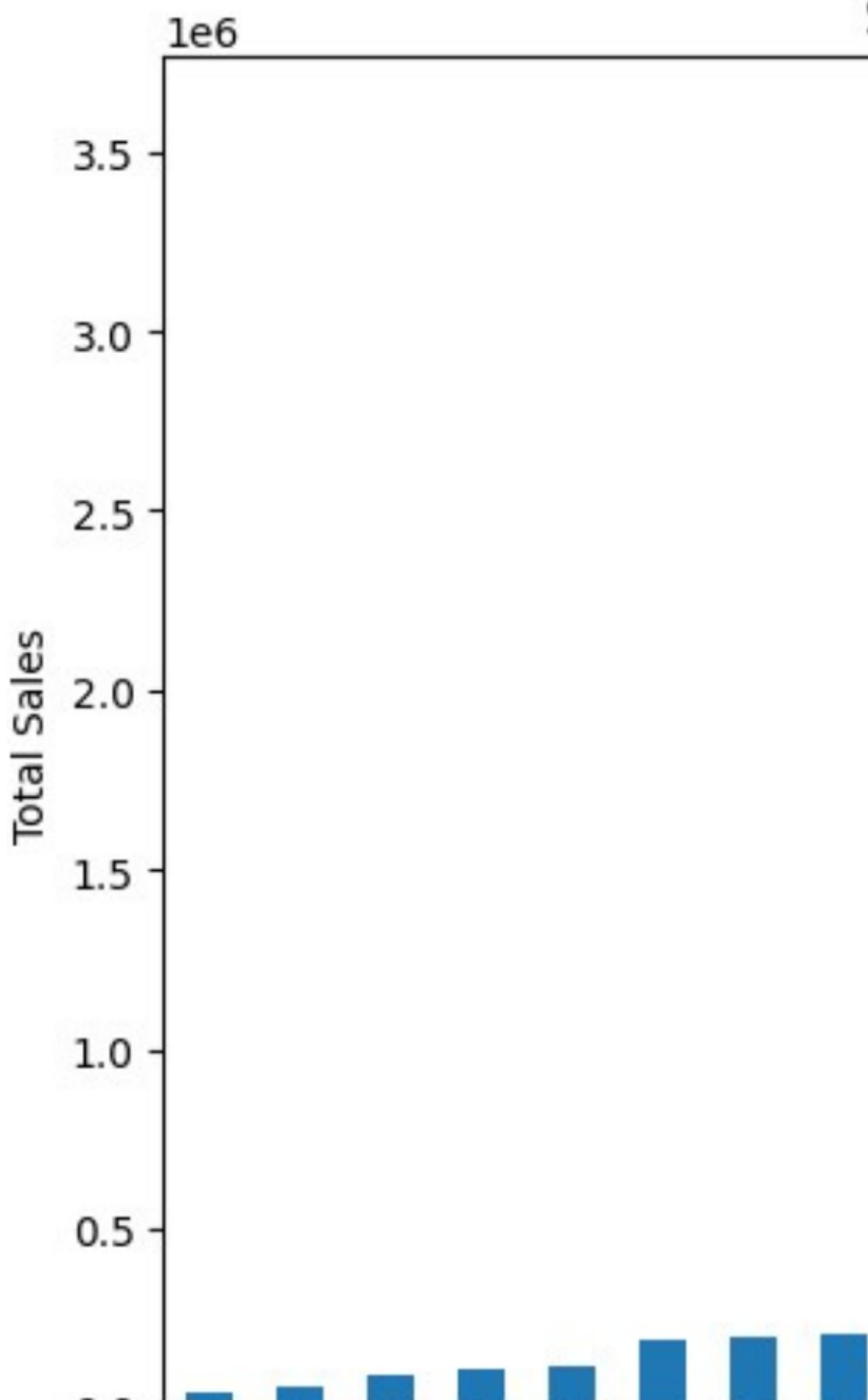
	ADDRESSLINE1		
POSTALCODE	\	0	897 Long Airport Avenue
10022		1	59 rue de l'Abbaye
51100		2	27 rue du Colonel Pierre Avia
75508		3	78934 Hillside Dr.
90003		4	7734 Strong St.
NaN			

	COUNTRY	TERRITORY	CONTACT
0	United States		NaN
1	France		EMEA
2	France		EMEA
3	USA		NaN
4	United States		NaN

[5 rows x 24 columns]

Price Each





Sa

