This report analyzes Amazon's sales performance, focusing on key trends, product categories, and revenue growth. It examines quarterly and annual data, identifies top-performing regions, and evaluates customer behavior. The report highlights factors like seasonal trends and marketing strategies, providing insights and recommendations to optimize sales and drive business growth.

AMAZON SALES REPORT

Data-Driven Insights for Optimizing Sales and Enhancing Business Strategies

welcome

1.Executive Summary

This report analyzes Amazon sales data to identify key trends and insights related to customer behavior, sales patterns, product performance, and fulfillment methods. Using various visualization techniques, such as charts and graphs, the analysis highlights important relationships and trends within the data. The goal is to help the business optimize sales strategies, enhance customer satisfaction, and improve operational efficiency by making data-driven decisions. In essence, the findings aim to provide actionable insights that support better business decision-making and overall growth.

2. Problem Statement

2.1. Objective

The goal of this analysis is to examine Amazon's sales data and extract actionable insights to support business decision-making. Key objectives include understanding sales performance, identifying product trends, evaluating fulfillment methods, and exploring customer segmentation.

2.2 . Problem Description

The dataset includes detailed information about sales transactions, including order ID, date, status, fulfillment method, sales channel, product category, size, quantity, amount, shipping details, and more. The analysis aims to provide insights into product preferences, geographic sales distribution, and customer behavior.

2.3 Dataset

[Download Amazon Sales]

..\Downloads\Amazon Sale Report.csv

3. Data Import and Preprocessing

3.1. Import Libraries and Load Data

We begin by importing the necessary libraries and loading the Amazon sales dataset. This helps us understand the structure of the data and prepare it for analysis.

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import math


# Load the dataset

df=pd.read_csv(r'C:\Users\welcome\Desktop\Amazon_Sale_Report.csv', encoding='ISO-8859-1')

df.
```

Output:

amazon sales report

/notebooks/amazon%20sales%20report.ipynb?

jupyter amazon sales report Last Checkpoint: yesterday

Edit View Run Kernel Settings Help Not Trusted

Code JupyterLab Python 3 (ipykernel)

[2]:
```
# Load the dataset
df = pd.read_csv(r'C:\Users\welcome\Desktop\Amazon_Sale_Report.csv', encoding='ISO-8859-1')
```

[3]:
```
df
```

[3]:

	index	Order ID	Date	Status	Fulfilment	Sales Channel	ship-service-level	Category	Size	Courier Status	...	currency	Amount	ship-city	ship-state	ship-postal-code
0	0	405-8078784-5731545	04-30-22	Cancelled	Merchant	Amazon.in	Standard	T-shirt	S	On the Way	...	INR	647.62	MUMBAI	MAHARASHTRA	400081.0
1	1	171-9198151-1101146	04-30-22	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	Shirt	3XL	Shipped	...	INR	406.00	BENGALURU	KARNATAKA	560085.0
2	2	404-0687676-7273146	04-30-22	Shipped	Amazon	Amazon.in	Expedited	Shirt	XL	Shipped	...	INR	329.00	NAVI MUMBAI	MAHARASHTRA	410210.0
3	3	403-9615377-8133951	04-30-22	Cancelled	Merchant	Amazon.in	Standard	Blazzer	L	On the Way	...	INR	753.33	PUDUCHERRY	PUDUCHERRY	605008.0
4	4	407-1069790-7240320	04-30-22	Shipped	Amazon	Amazon.in	Expedited	Trousers	3XL	Shipped	...	INR	574.00	CHENNAI	TAMIL NADU	600073.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
128971	128970	406-6001380-7673107	05-31-22	Shipped	Amazon	Amazon.in	Expedited	Shirt	XL	Shipped	...	INR	517.00	HYDERABAD	TELANGANA	500013.0
128972	128971	402-9551604-7544318	05-31-22	Shipped	Amazon	Amazon.in	Expedited	T-shirt	M	Shipped	...	INR	999.00	GURUGRAM	HARYANA	122004.0
128973	128972	407-9547469-3152358	05-31-22	Shipped	Amazon	Amazon.in	Expedited	Blazzer	XXL	Shipped	...	INR	690.00	HYDERABAD	TELANGANA	500049.0
128974	128973	402-6184140-0545956	05-31-22	Shipped	Amazon	Amazon.in	Expedited	T-shirt	XS	Shipped	...	INR	1199.00	Halol	Gujarat	389350.0
128975	128974	408-7436540-8728312	05-31-22	Shipped	Amazon	Amazon.in	Expedited	T-shirt	S	Shipped	...	INR	696.00	Raipur	CHHATTISGARH	492014.0

128976 rows × 21 columns

 # Display the first two rows of the dataset

df.head(2)

[4]:

	index	Order ID	Date	Status	Fulfilment	Sales Channel	ship-service-level	Category	Size	Courier Status	...	currency	Amount	ship-city	ship-state	ship-postal-code	ship-country
0	0	405-8078784-5731545	04-30-22	Cancelled	Merchant	Amazon.in	Standard	T-shirt	S	On the Way	...	INR	647.62	MUMBAI	MAHARASHTRA	400081.0	IN
1	1	171-9198151-1101146	04-30-22	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	Shirt	3XL	Shipped	...	INR	406.00	BENGALURU	KARNATAKA	560085.0	IN

2 rows × 21 columns

3.2. Data Preprocessing

We check the dataset's dimensions, data types, and convert necessary columns. We also drop any unnecessary columns and handle missing values.

 # Get basic information about the dataset

df.shape *# Rows and columns count*

df.info() *# Column data types and non-null counts*

 # Convert 'Date' column to datetime

df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

 # Drop unnecessary columns

df.drop(columns = ['New','PendingS'], inplace = True)

 # Check for missing values

df.isnull().sum()

```
[5]: # Get basic information about the dataset
     df.shape # Rows and columns count

[5]: (128976, 21)

[6]: df.info()# Column data types and non-null counts

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 128976 entries, 0 to 128975
Data columns (total 21 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   index               128976 non-null  int64
 1   Order ID            128976 non-null  object
 2   Date                128976 non-null  object
 3   Status              128976 non-null  object
 4   Fulfilment          128976 non-null  object
 5   Sales Channel       128976 non-null  object
 6   ship-service-level  128976 non-null  object
 7   Category            128976 non-null  object
 8   Size                128976 non-null  object
 9   Courier Status      128976 non-null  object
 10  Qty                 128976 non-null  int64
 11  currency            121176 non-null  object
 12  Amount              121176 non-null  float64
 13  ship-city           128941 non-null  object
 14  ship-state          128941 non-null  object
 15  ship-postal-code    128941 non-null  float64
 16  ship-country        128941 non-null  object
 17  B2B                 128976 non-null  bool
 18  fulfilled-by        39263 non-null   object
 19  New                 0 non-null       float64
 20  PendingS            0 non-null       float64
dtypes: bool(1), float64(4), int64(2), object(14)
memory usage: 19.8+ MB
```

```
[9]: # Check for missing values
     df.isnull().sum()

[9]: index                  0
     Order ID               0
     Date                   0
     Status                 0
     Fulfilment             0
     Sales Channel          0
     ship-service-level     0
     Category               0
     Size                   0
     Courier Status         0
     Qty                    0
     currency            7800
     Amount              7800
     ship-city             35
     ship-state            35
     ship-postal-code      35
     ship-country          35
     B2B                    0
     fulfilled-by       89713
     dtype: int64
```

3.3. Handling Missing Data

We handle missing values by filling numerical columns with the mean and categorical columns with the mode. We also remove duplicate records.

```
df['Amount'] = df['Amount'].fillna(df['Amount'].mean())

df['ship-postal-code'] = df['ship-postal-code'].fillna(df['ship-postal-code'].mean())

df['currency'] = df['currency'].fillna(df['currency'].mode()[0])

df['fulfilled-by'] = df['fulfilled-by'].fillna(df['fulfilled-by'].mode()[0])

df['ship-state'] = df['ship-state'].fillna(df['ship-state'].mode()[0])

df['ship-country'] = df['ship-country'].fillna(df['ship-country'].mode()[0])

df['ship-city'] = df['ship-city'].fillna(df['ship-city'].mode()[0])


# Check for duplicates and drop them

df.duplicated().sum()

df.drop_duplicates(inplace=True)
```

df.drop_duplicates(inplace = True)

df.columns

Output:

3. Handling Missing Data

Handling Missing Data

We handle missing values by using appropriate filling strategies, such as filling numerical columns with the mean and categorical columns with the moe. We also remove any duplicate records in the dataset.

[10]:
```
# Fill missing values with appropriate strategies
df['Amount'] = df['Amount'].fillna(df['Amount'].mean())
df['ship-postal-code'] = df['ship-postal-code'].fillna(df['ship-postal-code'].mean())
df['currency'] = df['currency'].fillna(df['currency'].mode()[0])
df['fulfilled-by'] = df['fulfilled-by'].fillna(df['fulfilled-by'].mode()[0])
df['ship-state'] = df['ship-state'].fillna(df['ship-state'].mode()[0])
df['ship-country'] = df['ship-country'].fillna(df['ship-country'].mode()[0])
df['ship-city'] = df['ship-city'].fillna(df['ship-city'].mode()[0])
```

[11]:
```
# Check for duplicates and drop them
df.duplicated().sum()
```

[11]:
```
168
```

[12]:
```
df.drop_duplicates(inplace = True)
```

[13]:
```
df.columns
```

[13]:
```
Index(['index', 'Order ID', 'Date', 'Status', 'Fulfilment', 'Sales Channel',
       'ship-service-level', 'Category', 'Size', 'Courier Status', 'Qty',
       'currency', 'Amount', 'ship-city', 'ship-state', 'ship-postal-code',
       'ship-country', 'B2B', 'fulfilled-by'],
      dtype='object')
```

4. Data Visualization

4.1. Distribution of Amount

We create a histogram to visualize the distribution of sales amounts.

```
plt.figure(figsize=(10, 8))
plt.hist(df['Amount'], bins=50, color='blue', alpha=0.7)
plt.title('Distribution of Amount')
plt.xlabel('Amount')
plt.ylabel('Frequency')
plt.grid()
plt.show()
```

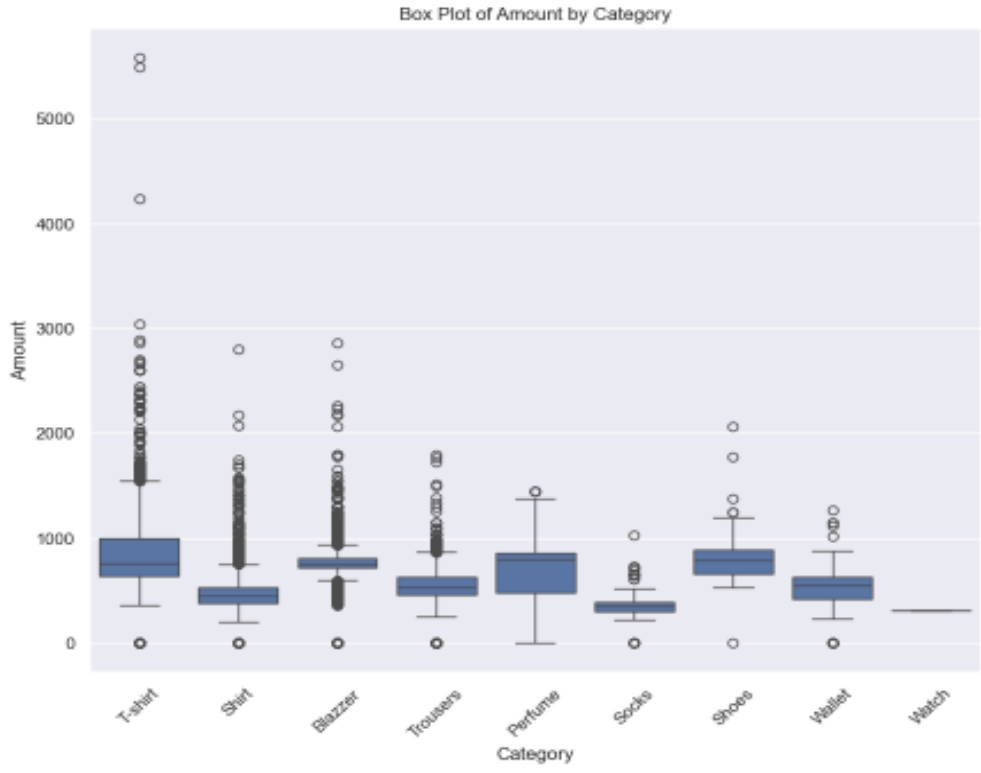
Visualization:

This histogram representing the distribution of sales amounts.

4.2. Box Plot of Amount by Category

```
plt.figure(figsize=(10, 8))
sns.boxplot(x='Category', y='Amount', data=df)
plt.title('Box Plot of Amount by Category')
plt.xlabel('Category')
plt.ylabel('Amount')
plt.xticks(rotation=45)
plt.show()
```

Visualization:



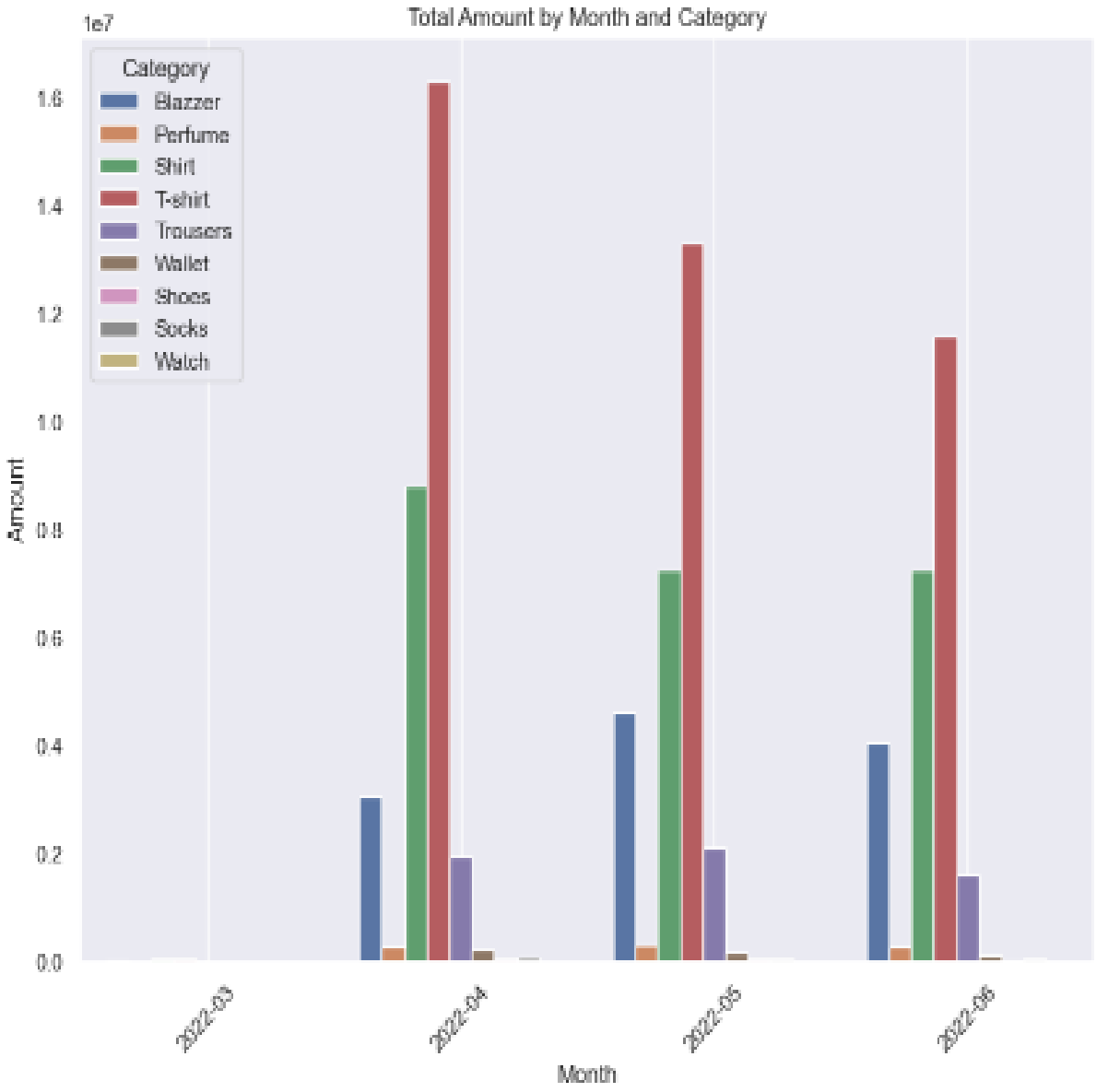
This box plot helps us analyze the distribution of sales amounts across different product categories.

4.3. Total Amount by Month and Category

We analyze the total sales amount per month, grouped by product category.

```
df['Month'] = df['Date'].dt.to_period('M')

monthly_category_amount = df.groupby(['Month',
'Category'])['Amount'].sum().reset_index()

plt.figure(figsize=(10, 8))

sns.barplot(x='Month', y='Amount', hue='Category',
data=monthly_category_amount)

plt.title('Total Amount by Month and Category')

plt.xlabel('Month')

plt.ylabel('Amount')

plt.xticks(rotation=45)

plt.legend(title='Category')

plt.grid()

plt.show()
```

Visualization:



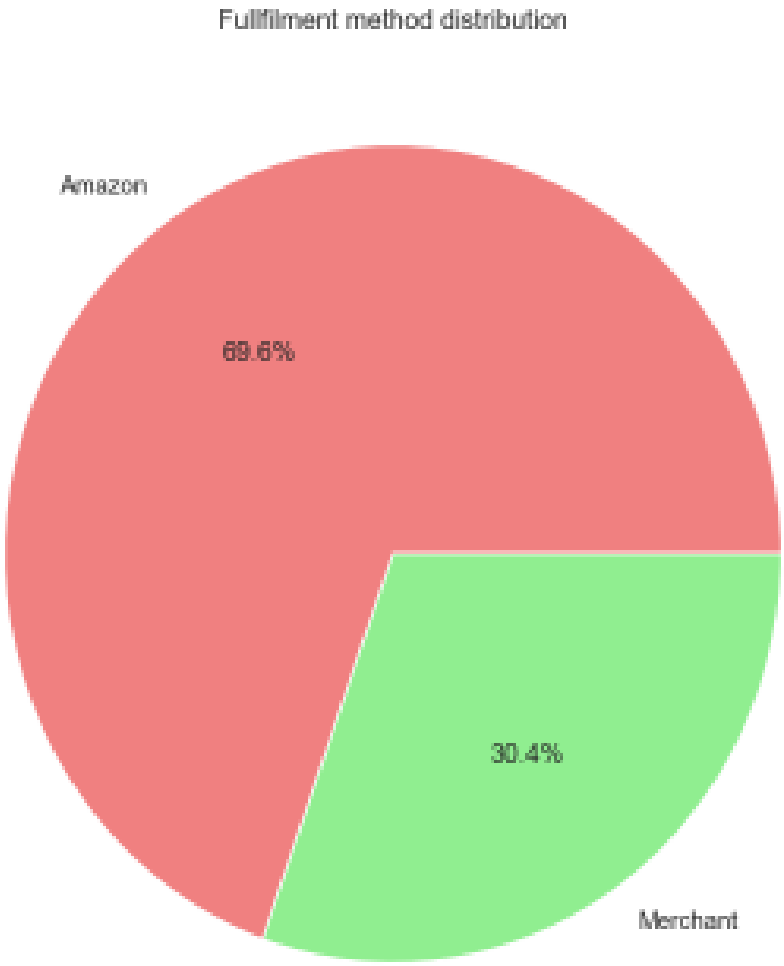
4.4. Fulfillment Method Distribution

We use a pie chart to visualize the distribution of fulfillment methods.

```
plt.figure(figsize=(10, 8))

df['Fulfilment'].value_counts().plot(kind='pie',
autopct='%1.1f%%', colors=['lightcoral', 'lightgreen'])

plt.title('Fulfillment Method Distribution')

plt.ylabel('')

plt.show()
```

Visualization:

A pie chart illustrating the distribution of fulfillment methods across sales.



The report uses visualization to illustrate sales distributions, product performance, fulfillment methods, and correlations. These help identify trends, outliers, and actionable insights for optimizing sales strategies and operations.

5. Exploratory Data Analysis (EDA)

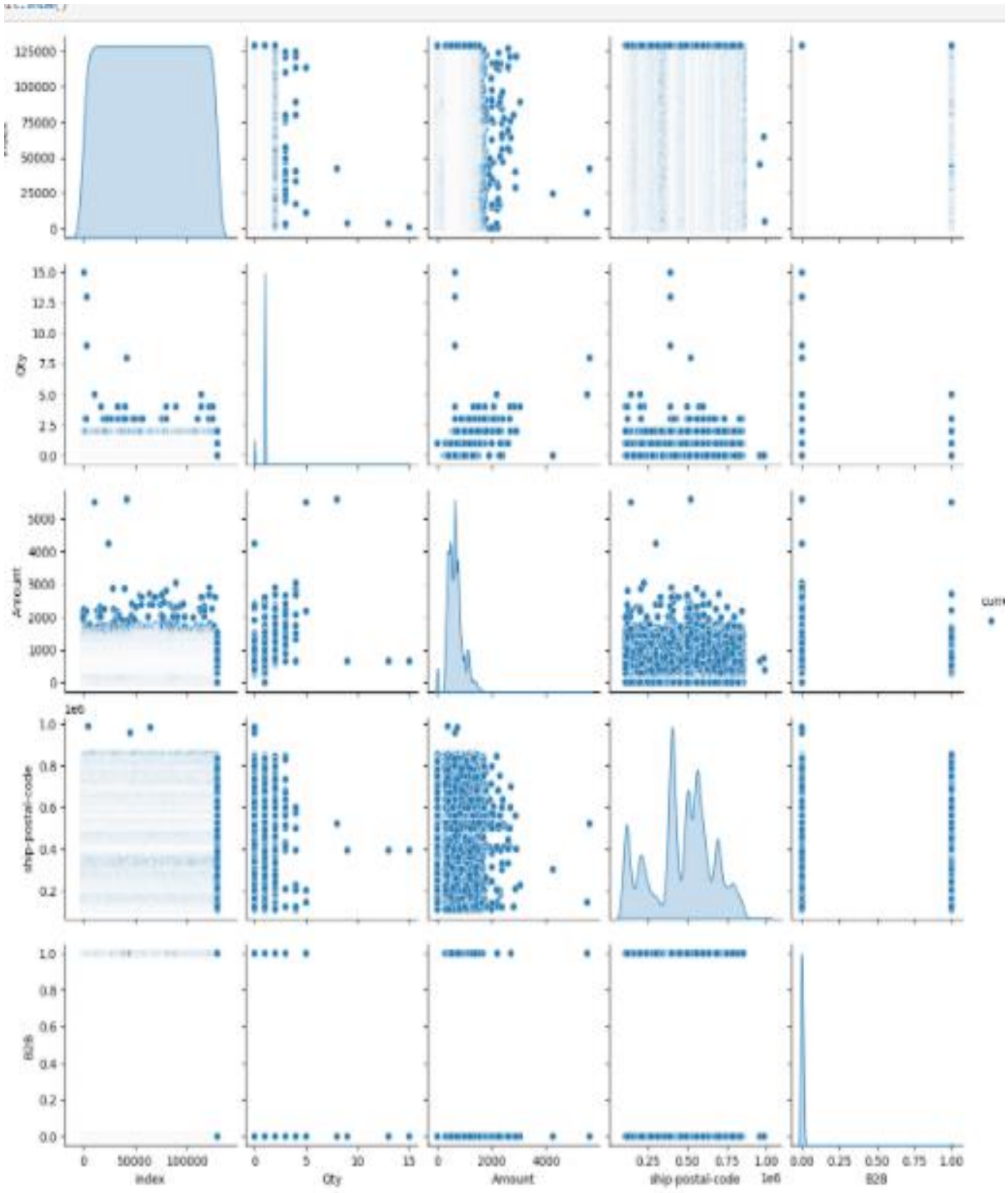
5.1. Pairplot of Numerical Features by Currency

This pairplot visualizes the relationships between numerical features, grouped by 'currency'. It helps identify any trends or correlations between them

```
sns.pairplot(df.dropna(subset=['currency']), hue='currency')
plt.show()
```

Visualization:

Pairplot showing relationships between numerical features, differentiated by currency.


index
Qty
Amount
ship-postal-code
1e6
B2B

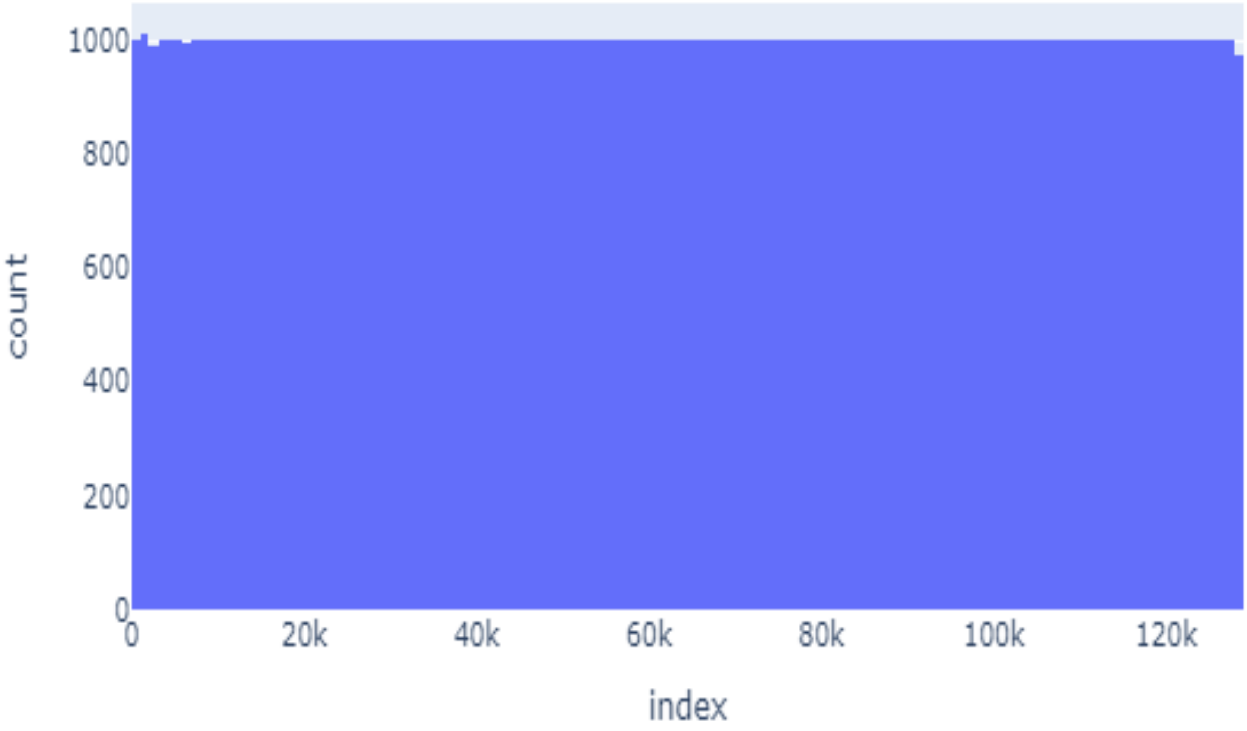
5.2. Distribution of Numerical Features

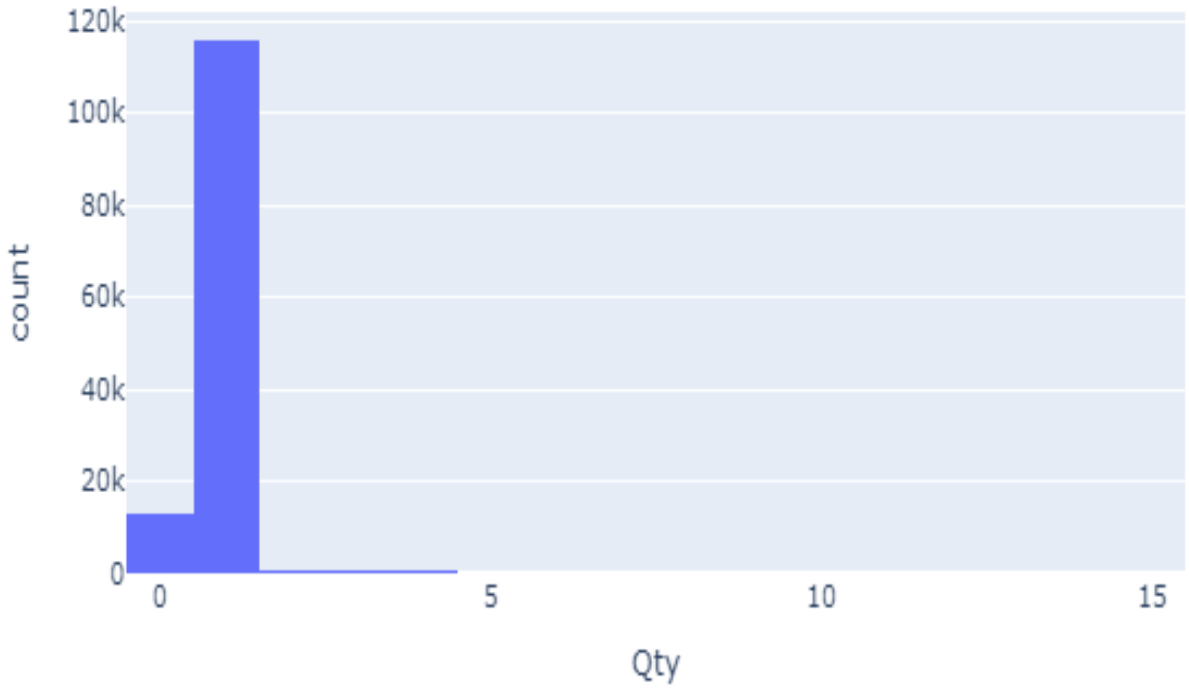
We generate histograms for each numerical feature to explore their distributions.

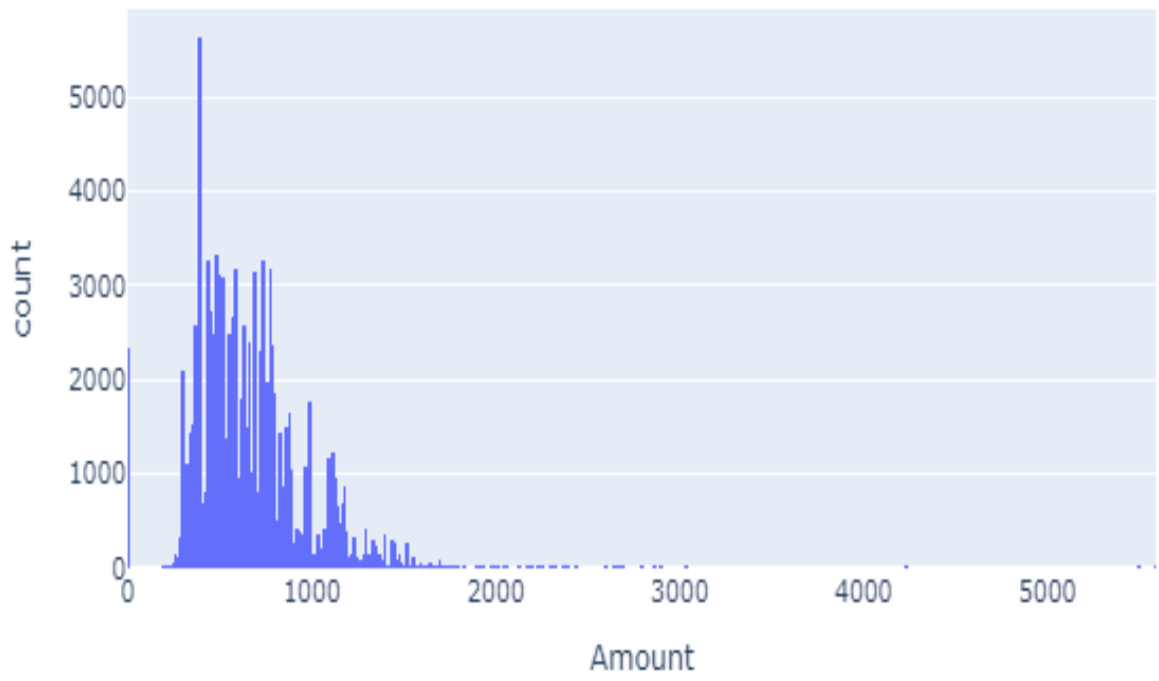
```
import plotly.express as px

for i in df.select_dtypes(include="number").columns:

    fig = px.histogram(df, x=i)

    fig.show()
```

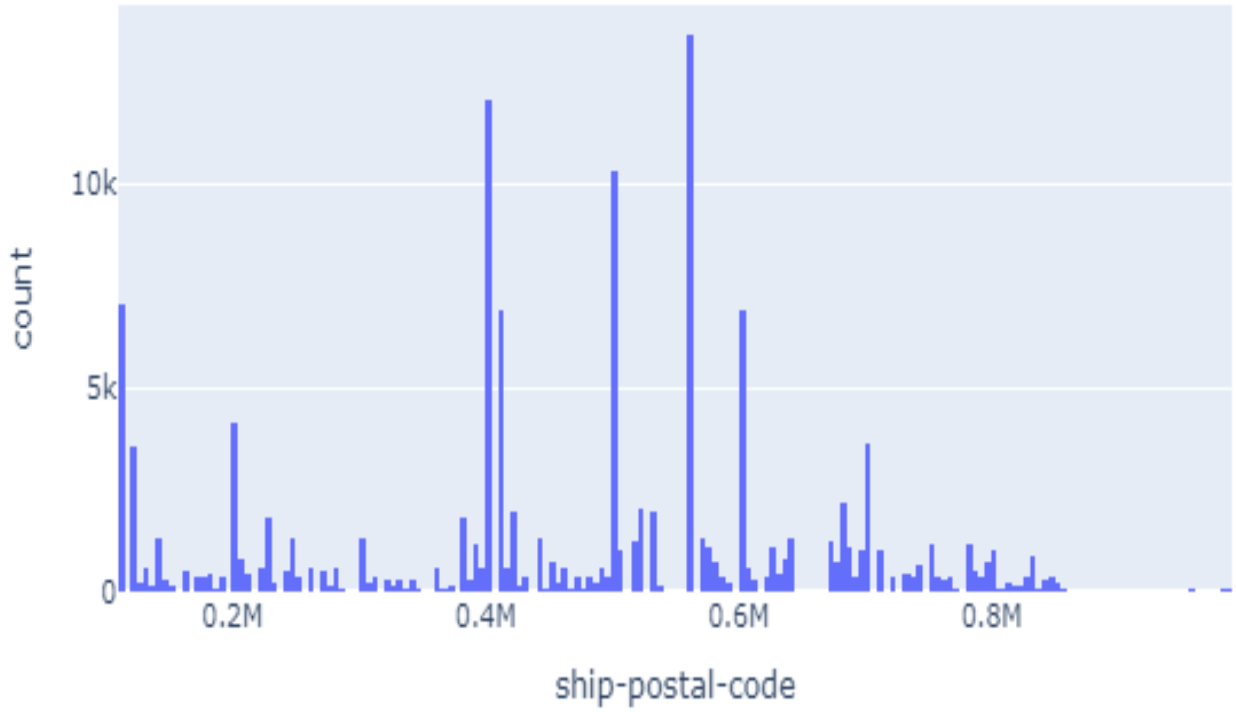
Visualization:

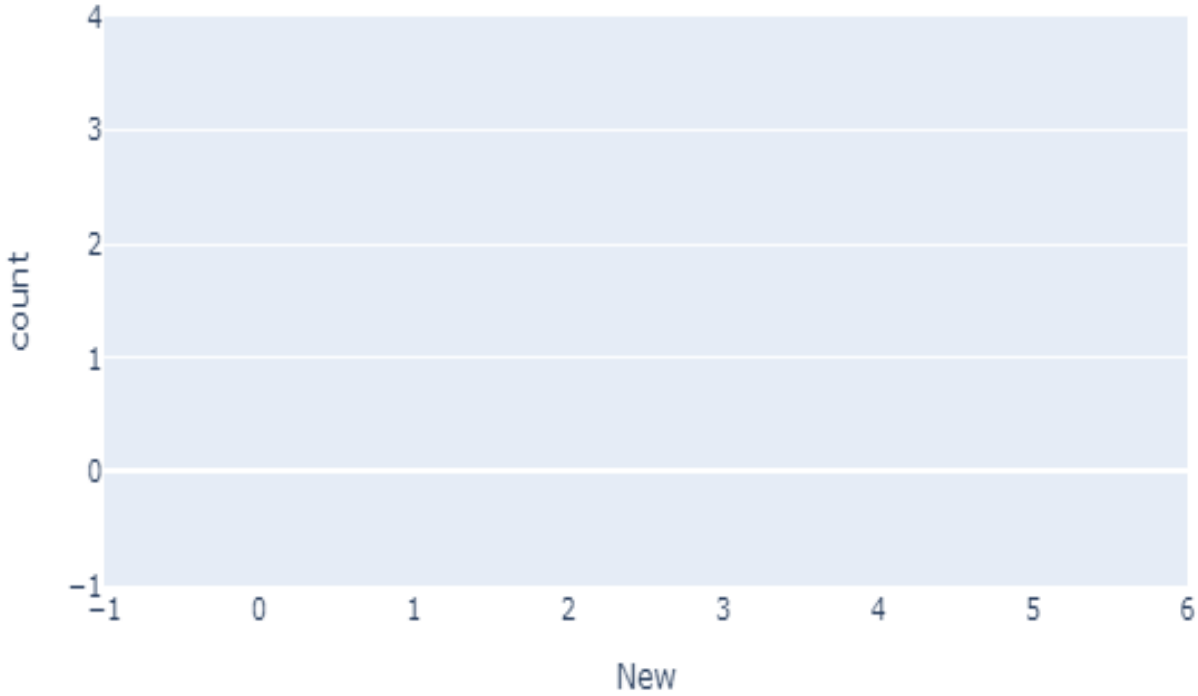
Histograms representing the distribution of numerical features.




120k
100k
80k
60k
40k
20k
0
count
0
5
10
15
Qty


5000
4000
3000
2000
1000
0
count
0
1000
2000
3000
4000
5000
Amount


count
10k
5k
0
0.2M
0.4M
0.6M
0.8M
ship-postal-code


count
4
3
2
1
0
−1
−1
0
1
2
3
4
5
6
New

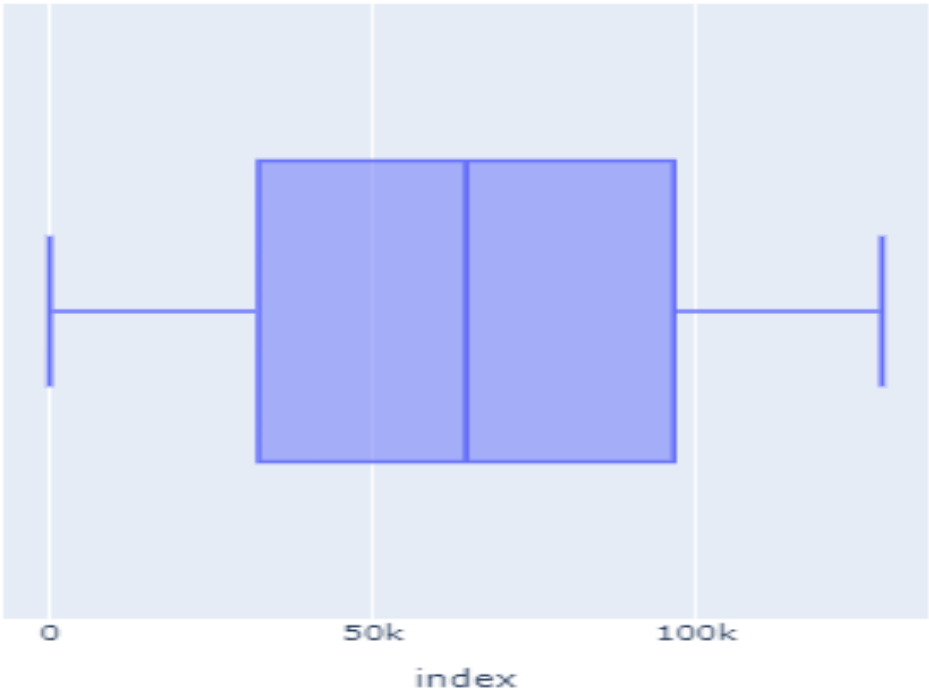
5.3. Boxplots for Numerical Features

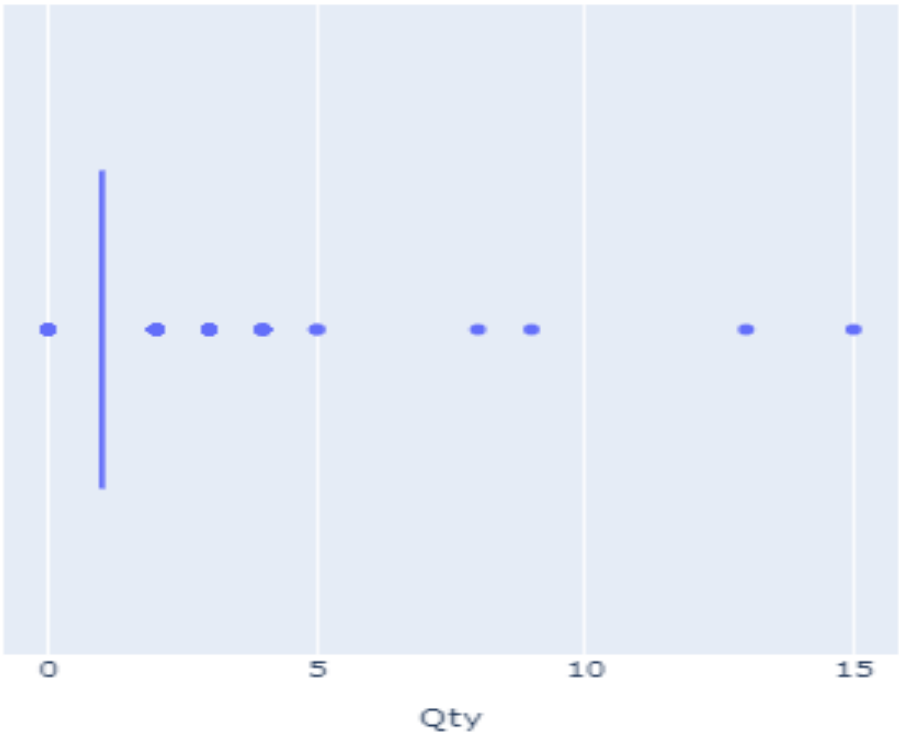
Boxplots help us visualize the spread and detect outliers for each numerical feature.

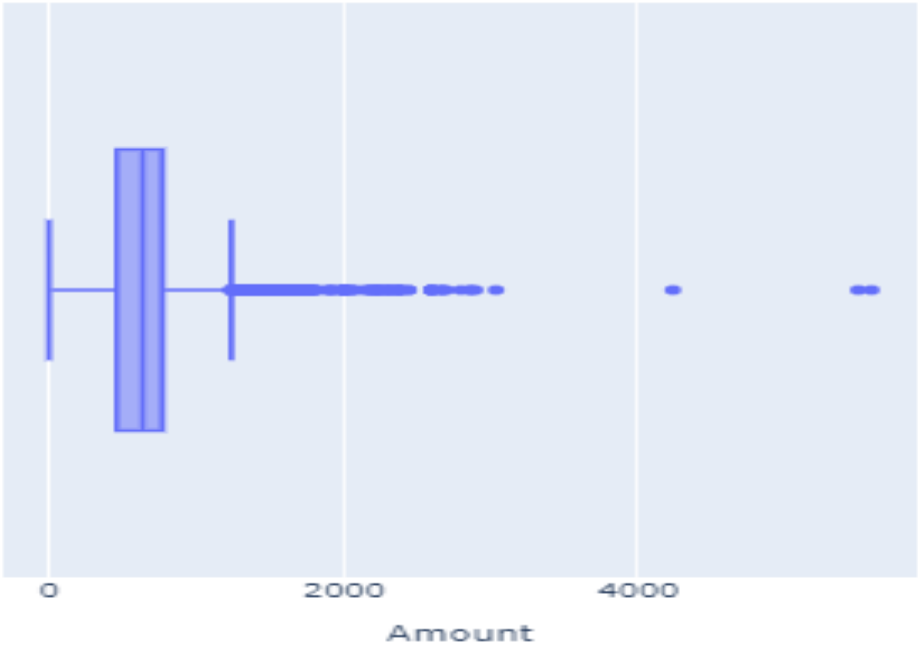
```
for i in df.select_dtypes(include="number").columns:
    fig = px.box(df, x=i)
    fig.update_layout(width=500, height=500)
    fig.show()
```

Visualization:

Box plots for each numerical feature to detect outliers.




0
5
10
15
Qty


0
2000
4000
Amount

6. Sales Analysis and Visualizations

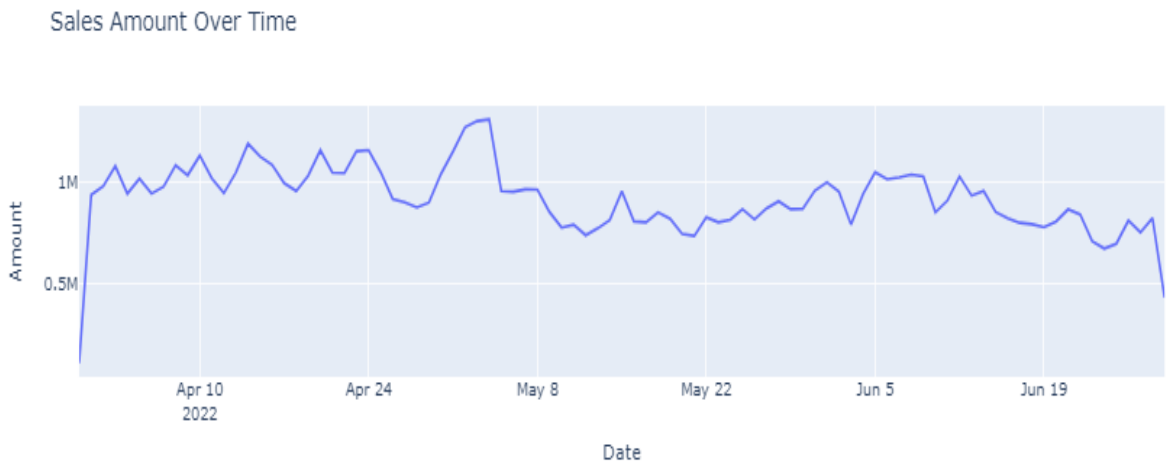
6.1. Sales Amount Over Time

This line plot shows the total sales amount over time, helping us understand sales trends.

```
sales_per_day = df.groupby('Date')['Amount'].sum().reset_index()

fig = px.line(sales_per_day, x='Date', y='Amount', title='Sales Amount Over Time')

fig.show()
```

Visualization:

A line chart showing total sales over time.



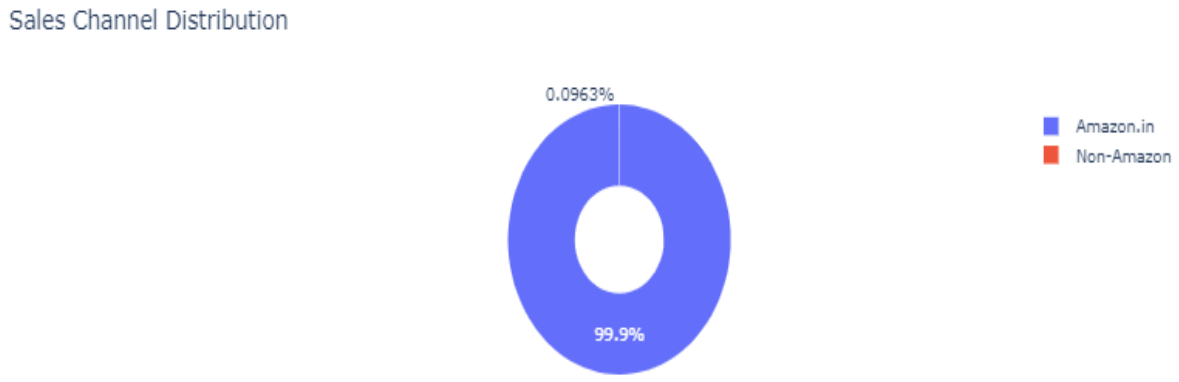
6.2. Sales Channel Distribution

This pie chart visualizes the distribution of sales across different sales channels.

```
fig = px.pie(df, names='Sales Channel', title='Sales Channel Distribution', hole=0.4)

fig.show()
```

Visualization:

A pie chart representing the distribution of sales channels.



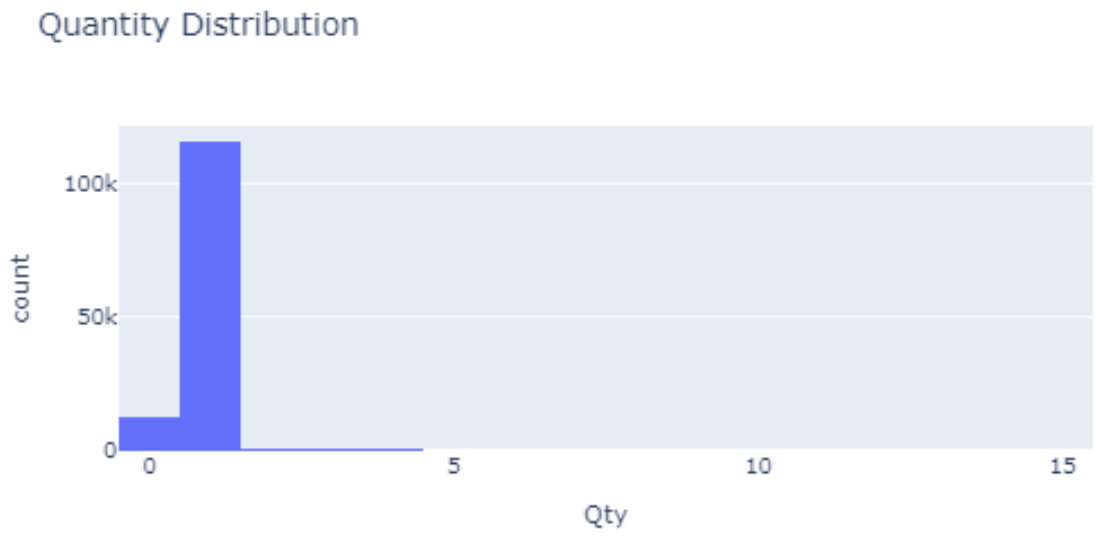
6.3. Quantity Distribution

This histogram displays the distribution of quantities sold across products.

```
fig = px.histogram(df, x='Qty', title='Quantity Distribution')
fig.show()
```

Visualization:

A histogram showing the distribution of product quantities sold.



7. Category-wise Sales Analysis

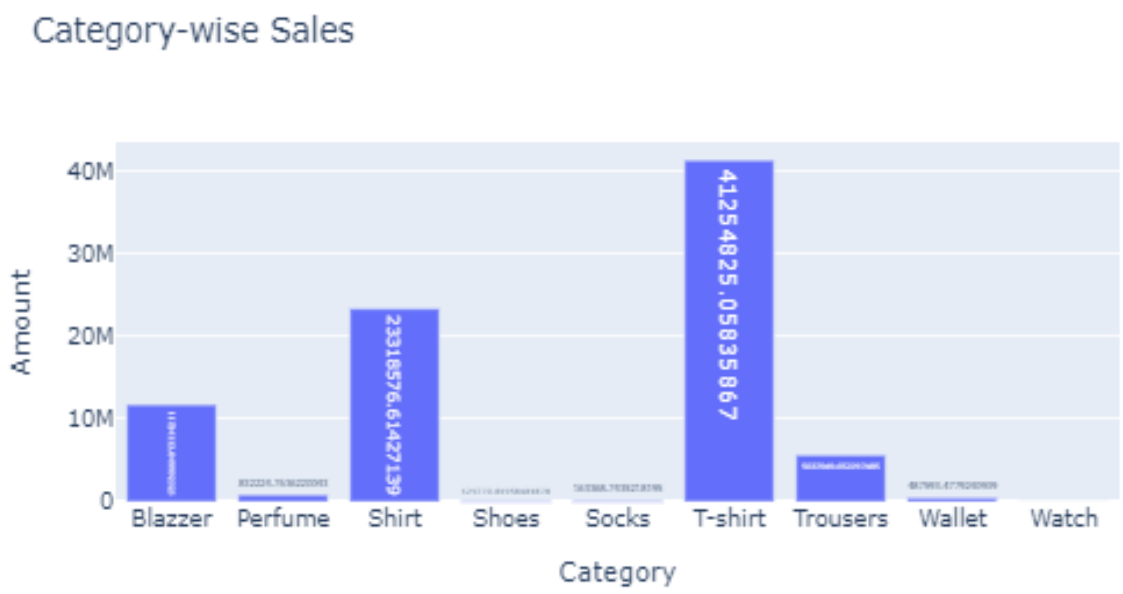
7.1. Category-wise Sales

We analyze the total sales by product category.

```
category_sales =
df.groupby('Category')['Amount'].sum().reset_index()

fig = px.bar(category_sales, x='Category', y='Amount',
title='Category-wise Sales', text='Amount')

fig.show()
```

Visualization:

A bar chart showing total sales by product category.



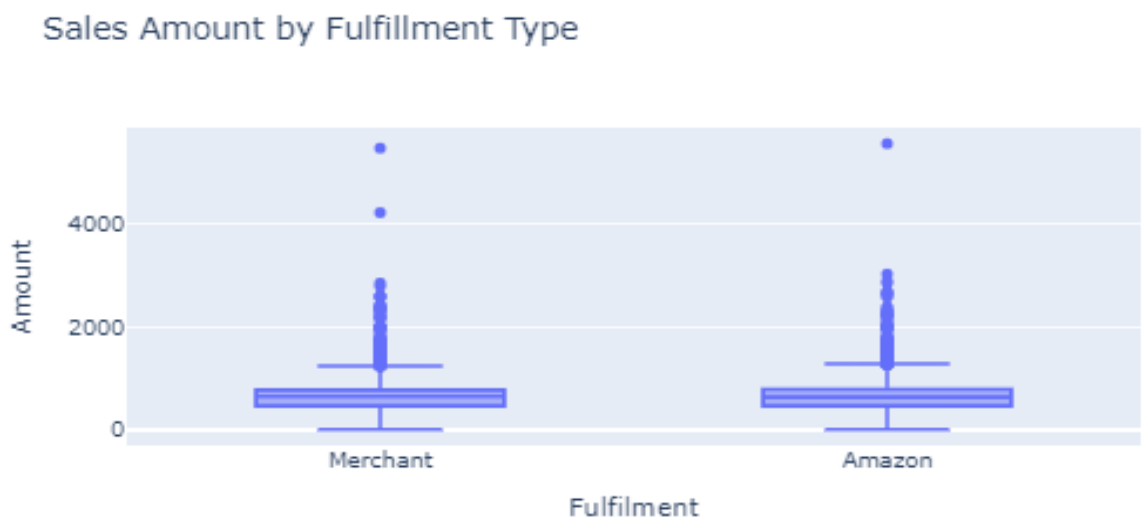
7.2. Sales Amount by Fulfillment Type

We analyze how different fulfillment methods impact sales amounts.

```
fig = px.box(df, x='Fulfilment', y='Amount', title='Sales Amount by Fulfillment Type')

fig.show()
```

Visualization:

A box plot of sales amounts by fulfillment type.



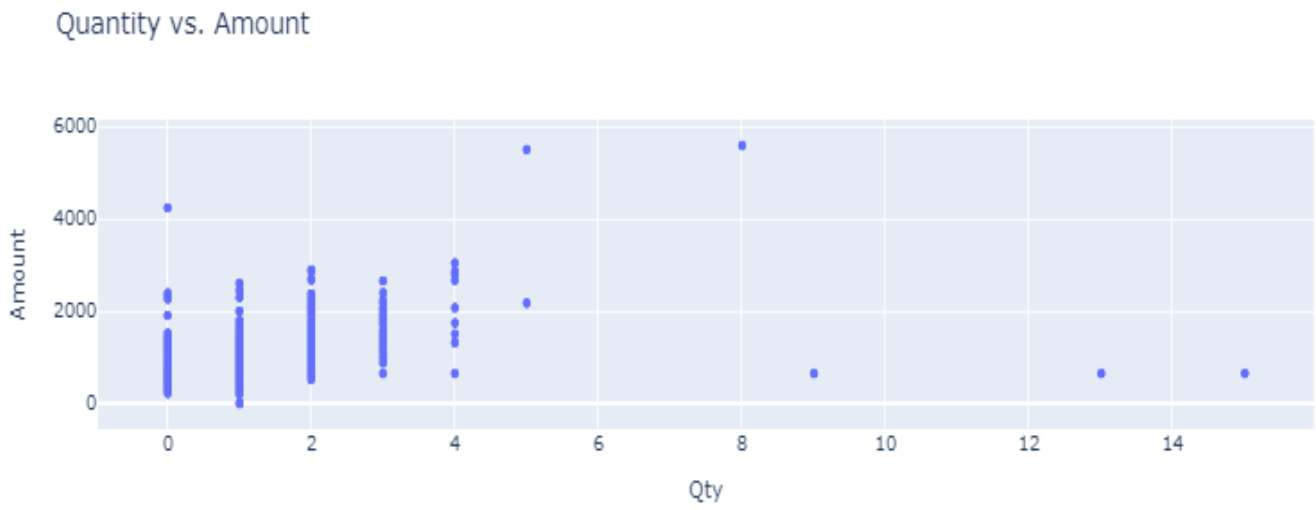
7.3. Quantity vs. Sales Amount

This scatter plot shows the relationship between quantity sold and sales amount.

```
fig = px.scatter(df, x='Qty', y='Amount', title='Quantity vs. Amount')

fig.show()
```

Visualization:

Scatter plot visualizing the relationship between quantity sold and sales amount.



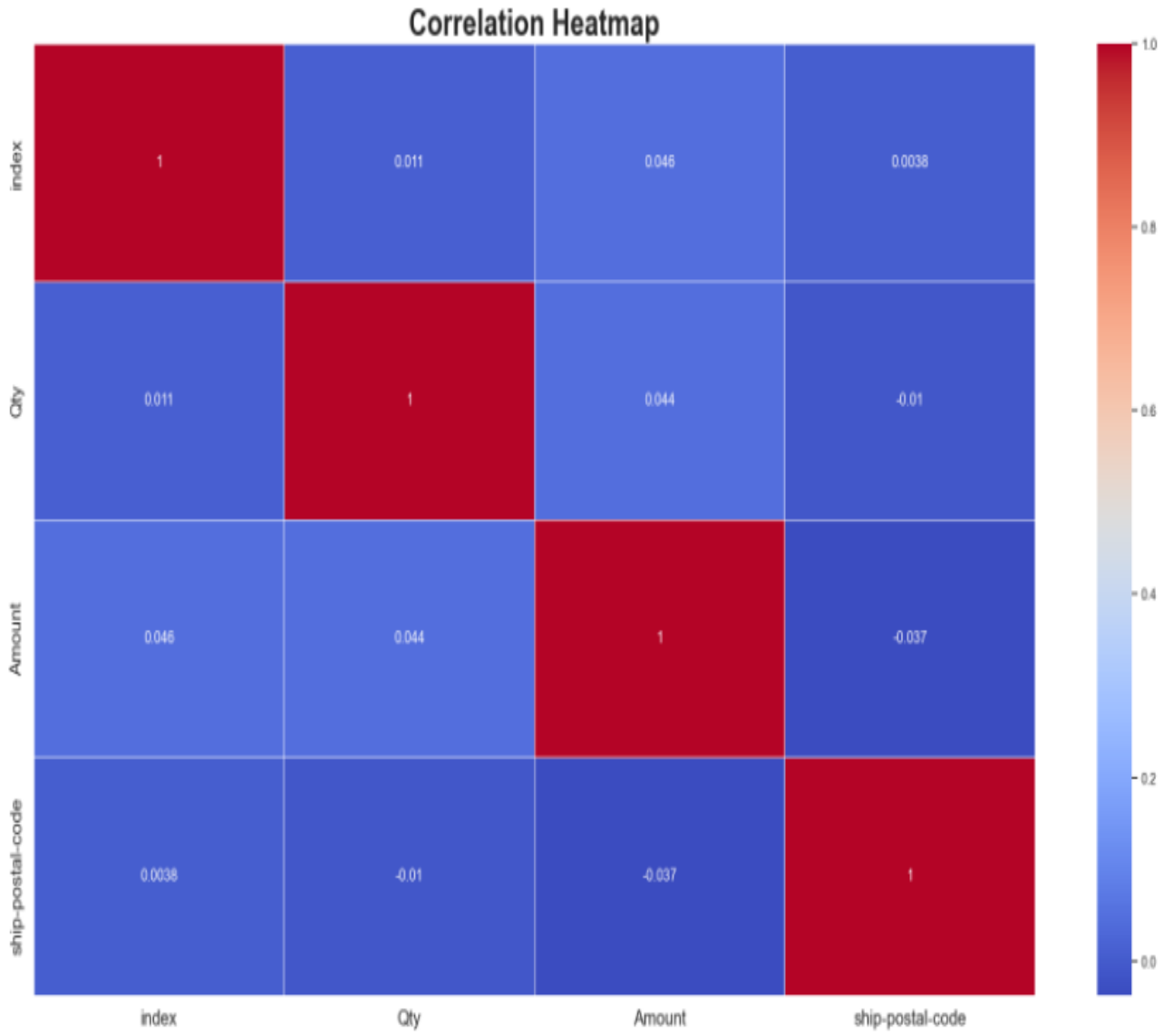
8. Correlation Analysis

We analyze the correlation between numerical features using a heatmap.

```
sns.set(rc={"figure.figsize": (22, 12)})

corr_matrix = df.select_dtypes(include="number").corr()

sns.heatmap(corr_matrix, annot=True, cmap="coolwarm",
linewidth=0.5)

plt.title("Correlation Heatmap", fontsize=25, fontweight="bold")

plt.show()
```

Visualization:

Heatmap showing correlations between numerical features in the dataset.



9. Insights and Recommendations

9.1. Key Insights

Sales Trends:

Sales show significant seasonal patterns, with high peaks around certain months. This can help in planning promotions and inventory management.

Product Categories:

Electronics and fashion are the top-performing categories in terms of total sales. Focusing marketing efforts on these categories could yield better returns.

Fulfillment Efficiency:

The 'Fulfilled by Amazon' method appears to have the highest sales amount, indicating it is more efficient.

Customer Segmentation:

Certain geographic regions show higher sales, suggesting targeted campaigns could increase sales in underperforming areas.

9.2. Recommendations

Focus on High-performing Categories: Increase inventory and promotional efforts for electronics and fashion categories.

Optimize Fulfillment: Since Amazon fulfillment methods perform better, consider expanding this method to more products.

Geographic Targeting: Increase marketing spend in regions with lower sales to balance distribution.

Enhance Customer Engagement: Improve post-purchase engagement, particularly in areas with high customer loyalty.

10. Conclusion

The analysis of Amazon sales data reveals key trends and patterns in product performance, customer preferences, and fulfillment methods. By leveraging these insights, the business can optimize its sales strategies, enhance operational efficiency, and improve customer satisfaction.