In [9]:
```
import pandas as pd

# Load the dataset
file_path = r'C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv'
df = pd.read_csv(file_path, encoding='ISO-8859-1')



# Display the first few rows of the dataset
print(df.head())
```

```
   index            Order ID      Date                       Status  \
0      0  405-8078784-5731545  04-30-22                    Cancelled
1      1  171-9198151-1101146  04-30-22  Shipped - Delivered to Buyer
2      2  404-0687676-7273146  04-30-22                      Shipped
3      3  403-9615377-8133951  04-30-22                    Cancelled
4      4  407-1069790-7240320  04-30-22                      Shipped

  Fulfilment Sales Channel ship-service-level  Category Size Courier Status  \
0   Merchant     Amazon.in           Standard   T-shirt    S     On the Way
1   Merchant     Amazon.in           Standard     Shirt  3XL        Shipped
2     Amazon     Amazon.in          Expedited     Shirt   XL        Shipped
3   Merchant     Amazon.in           Standard   Blazzer    L     On the Way
4     Amazon     Amazon.in          Expedited  Trousers  3XL        Shipped

   ...  currency  Amount    ship-city   ship-state ship-postal-code  \
0  ...       INR  647.62       MUMBAI  MAHARASHTRA         400081.0
1  ...       INR  406.00    BENGALURU    KARNATAKA         560085.0
2  ...       INR  329.00  NAVI MUMBAI  MAHARASHTRA         410210.0
3  ...       INR  753.33   PUDUCHERRY   PUDUCHERRY         605008.0
4  ...       INR  574.00      CHENNAI   TAMIL NADU         600073.0

   ship-country    B2B  fulfilled-by New  PendingS
0            IN  False     Easy Ship NaN       NaN
1            IN  False     Easy Ship NaN       NaN
2            IN   True           NaN NaN       NaN
3            IN  False     Easy Ship NaN       NaN
4            IN  False           NaN NaN       NaN

[5 rows x 21 columns]
```

In [10]:
```
# Check for missing values
print(df.isnull().sum())
```

```
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
New                 128976
PendingS            128976
dtype: int64
```

In [11]:
```
# Check for duplicates
print(df.duplicated().sum())
```

```
168
```

In [14]:
```
# Fill missing values (example: filling with median or mode)
df.fillna(df.median(), inplace=True)
```

```
C:\Users\Anurag Kumar\AppData\Local\Temp\ipykernel_24484\2621753063.py:2: FutureWarning: The default value of numeric_only in DataFrame.median is deprecated. In a future version,
it will default to False. In addition, specifying 'numeric_only=None' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.fillna(df.median(), inplace=True)
```

In [15]:
```
# Remove duplicates
df.drop_duplicates(inplace=True)
```

In [16]:
```
# Display the first few rows of the dataset
print(df.head())

# Check for missing values
print(df.isnull().sum())

# Check for duplicates
print(df.duplicated().sum())
```

```
   index            Order ID      Date                       Status  \
0      0  405-8078784-5731545  04-30-22                    Cancelled
1      1  171-9198151-1101146  04-30-22  Shipped - Delivered to Buyer
2      2  404-0687676-7273146  04-30-22                      Shipped
3      3  403-9615377-8133951  04-30-22                    Cancelled
4      4  407-1069790-7240320  04-30-22                      Shipped

  Fulfilment Sales Channel ship-service-level  Category Size Courier Status  \
0   Merchant     Amazon.in           Standard   T-shirt    S     On the Way
1   Merchant     Amazon.in           Standard     Shirt  3XL        Shipped
2     Amazon     Amazon.in          Expedited     Shirt   XL        Shipped
3   Merchant     Amazon.in           Standard   Blazzer    L     On the Way
4     Amazon     Amazon.in          Expedited  Trousers  3XL        Shipped

   ...  currency  Amount    ship-city   ship-state ship-postal-code  \
0  ...       INR  647.62       MUMBAI  MAHARASHTRA         400081.0
1  ...       INR  406.00    BENGALURU    KARNATAKA         560085.0
2  ...       INR  329.00  NAVI MUMBAI  MAHARASHTRA         410210.0
3  ...       INR  753.33   PUDUCHERRY   PUDUCHERRY         605008.0
4  ...       INR  574.00      CHENNAI   TAMIL NADU         600073.0

   ship-country    B2B  fulfilled-by New  PendingS
0            IN  False     Easy Ship NaN       NaN
1            IN  False     Easy Ship NaN       NaN
2            IN   True           NaN NaN       NaN
3            IN  False     Easy Ship NaN       NaN
4            IN  False           NaN NaN       NaN

[5 rows x 21 columns]
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               7789
Amount                    0
ship-city                33
ship-state               33
ship-postal-code          0
ship-country             33
B2B                       0
fulfilled-by          89595
New                  128808
PendingS             128808
dtype: int64
0
```

In [17]:
```
# Fill missing values with a specific value, e.g., 0 or the mean
df.fillna(value=0, inplace=True)
# Or use the mean value for numerical columns
df.fillna(df.mean(), inplace=True)
```

```
C:\Users\Anurag Kumar\AppData\Local\Temp\ipykernel_24484\920851928.py:4: FutureWarning: The default value of numeric_only in DataFrame.mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=None' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.fillna(df.mean(), inplace=True)
```

In [18]:
```
# Drop rows with missing values
df.dropna(inplace=True)
# Or drop columns with missing values
df.dropna(axis=1, inplace=True)
```

In [19]:
```
df.drop_duplicates(inplace=True)
```

In [ ]:
```
# Convert text data to lowercase
df['column_name'] = df['column_name'].str.lower()
```

In [ ]:

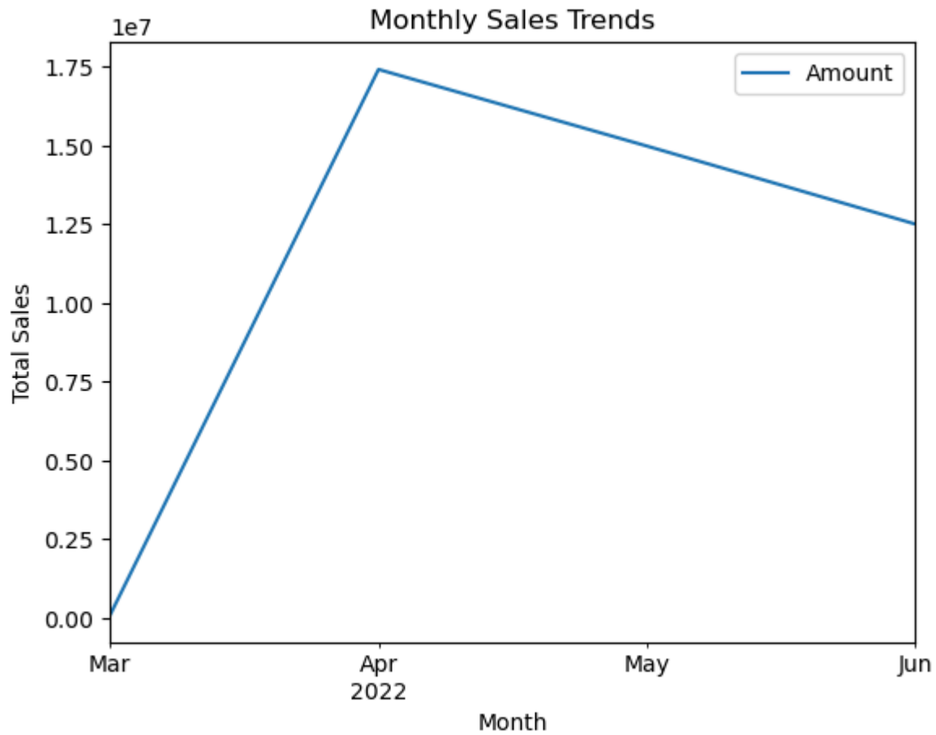
In [ ]:

In [28]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Load the Dataset with encoding specified
try:
    df = pd.read_csv r"C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv", encoding='ISO-8859-1')
except UnicodeDecodeError:
    df = pd.read_csv r"C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv", encoding='utf-16')

# Inspect column names and first few rows
print("Columns in the dataset:", df.columns)
print(df.head())

# Remove leading/trailing spaces from column names
df.columns = df.columns.str.strip()

# Step 2: Data Cleaning
# Checking for missing values and duplicates
print("\nMissing values:\n", df.isnull().sum())
print("\nDuplicates:", df.duplicated().sum())

# Handle missing values (example: fill with 0 for numerical values)
df.fillna({'Amount': 0}, inplace=True)  # Fill missing values in 'Amount' with 0

# Remove duplicates
df.drop_duplicates(inplace=True)

# Step 3: Sales Overview
# Convert 'Date' column to datetime format, if exists
if 'Date' in df.columns:
    df['Date'] = pd.to_datetime(df['Date'], format='%m-%d-%y', errors='coerce')
else:
    print("Column 'Date' is missing from the dataset.")

# Calculate total sales
if 'Amount' in df.columns:
    total_sales = df['Amount'].sum()
    print(f"\nTotal Sales: ${total_sales:.2f}")
else:
    print("Column 'Amount' is missing from the dataset.")

# Sales trends over time (monthly)
if 'Date' in df.columns:
    df['Month'] = df['Date'].dt.to_period('M')
    sales_trends = df.groupby('Month').agg({'Amount': 'sum'})
    sales_trends.plot(kind='line', title='Monthly Sales Trends')
    plt.xlabel('Month')
    plt.ylabel('Total Sales')
    plt.show()

# Step 4: Product Analysis
# Analyze distribution of product categories
if 'Category' in df.columns and 'Qty' in df.columns:
    product_distribution = df.groupby('Category').agg({'Qty': 'sum', 'Amount': 'sum'})
    print("\nProduct Distribution:\n", product_distribution)

    # Identify popular products
    top_products = df.groupby('Category').agg({'Qty': 'sum'}).sort_values(by='Qty', ascending=False)
    print("\nTop Selling Products:\n", top_products)

    # Visualize product distribution
```

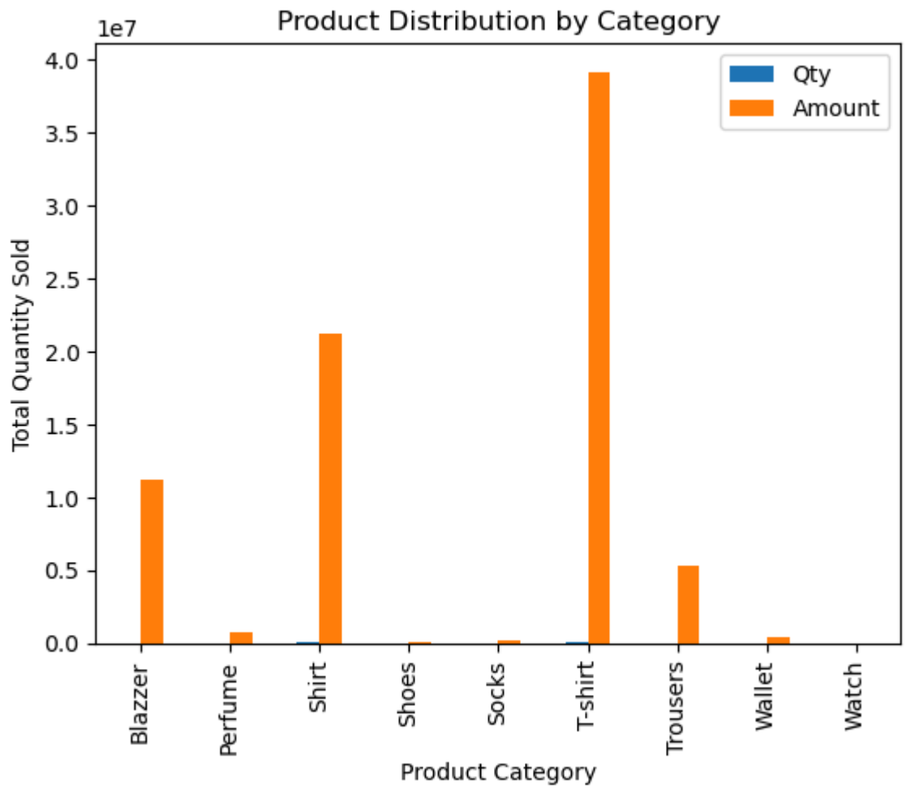
```
    product_distribution.plot(kind='bar', title='Product Distribution by Category')
    plt.xlabel('Product Category')
    plt.ylabel('Total Quantity Sold')
    plt.show()

# Step 5: Fulfillment Analysis
# Analyze fulfillment methods
if 'Fulfilment' in df.columns:
    fulfillment_analysis = df.groupby('Fulfilment').agg({'Amount': 'sum', 'Qty': 'sum'})
    print("\nFulfillment Analysis:\n", fulfillment_analysis)

    # Visualize fulfillment effectiveness
    fulfillment_analysis.plot(kind='bar', title='Fulfillment Method Analysis')
    plt.xlabel('Fulfillment Method')
    plt.ylabel('Total Sales and Quantity')
    plt.show()

# Step 6: Customer Segmentation
# Segment customers and analyze
if 'Order ID' in df.columns:
    customer_segments = df.groupby('Order ID').agg({'Amount': 'sum', 'Qty': 'sum'})
    print("\nCustomer Segments:\n", customer_segments)

    # Visualize customer segments
    customer_segments['Amount'].plot(kind='hist', title='Customer Purchase Distribution')
    plt.xlabel('Total Purchase Amount')
    plt.ylabel('Number of Orders')
    plt.show()

# Step 7: Geographical Analysis
# Analyze sales by location
if 'ship-city' in df.columns and 'Amount' in df.columns:
    geographical_sales = df.groupby('ship-city').agg({'Amount': 'sum'})
    print("\nGeographical Sales Distribution:\n", geographical_sales)

    # Visualize geographical distribution
    geographical_sales.plot(kind='bar', title='Sales Distribution by City')
    plt.xlabel('City')
    plt.ylabel('Total Sales')
    plt.show()
```

```
Columns in the dataset: 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', 'New', 'PendingS'],
      dtype='object')
   index            Order ID      Date                       Status  \
0      0  405-8078784-5731545  04-30-22                    Cancelled
1      1  171-9198151-1101146  04-30-22  Shipped - Delivered to Buyer
2      2  404-0687676-7273146  04-30-22                      Shipped
3      3  403-9615377-8133951  04-30-22                    Cancelled
4      4  407-1069790-7240320  04-30-22                      Shipped

  Fulfilment Sales Channel ship-service-level  Category Size Courier Status  \
0   Merchant     Amazon.in           Standard   T-shirt    S     On the Way
1   Merchant     Amazon.in           Standard     Shirt  3XL        Shipped
2     Amazon     Amazon.in          Expedited     Shirt   XL        Shipped
3   Merchant     Amazon.in           Standard   Blazzer    L     On the Way
4     Amazon     Amazon.in          Expedited  Trousers  3XL        Shipped

   ...  currency  Amount    ship-city   ship-state ship-postal-code  \
0  ...       INR  647.62       MUMBAI  MAHARASHTRA         400081.0
1  ...       INR  406.00    BENGALURU    KARNATAKA         560085.0
2  ...       INR  329.00  NAVI MUMBAI  MAHARASHTRA         410210.0
3  ...       INR  753.33   PUDUCHERRY   PUDUCHERRY         605008.0
4  ...       INR  574.00      CHENNAI   TAMIL NADU         600073.0

   ship-country    B2B  fulfilled-by New  PendingS
0            IN  False     Easy Ship NaN       NaN
1            IN  False     Easy Ship NaN       NaN
2            IN   True           NaN NaN       NaN
3            IN  False     Easy Ship NaN       NaN
4            IN  False           NaN NaN       NaN

[5 rows x 21 columns]

Missing values:
 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
New                 128976
PendingS            128976
dtype: int64

Duplicates: 168
```

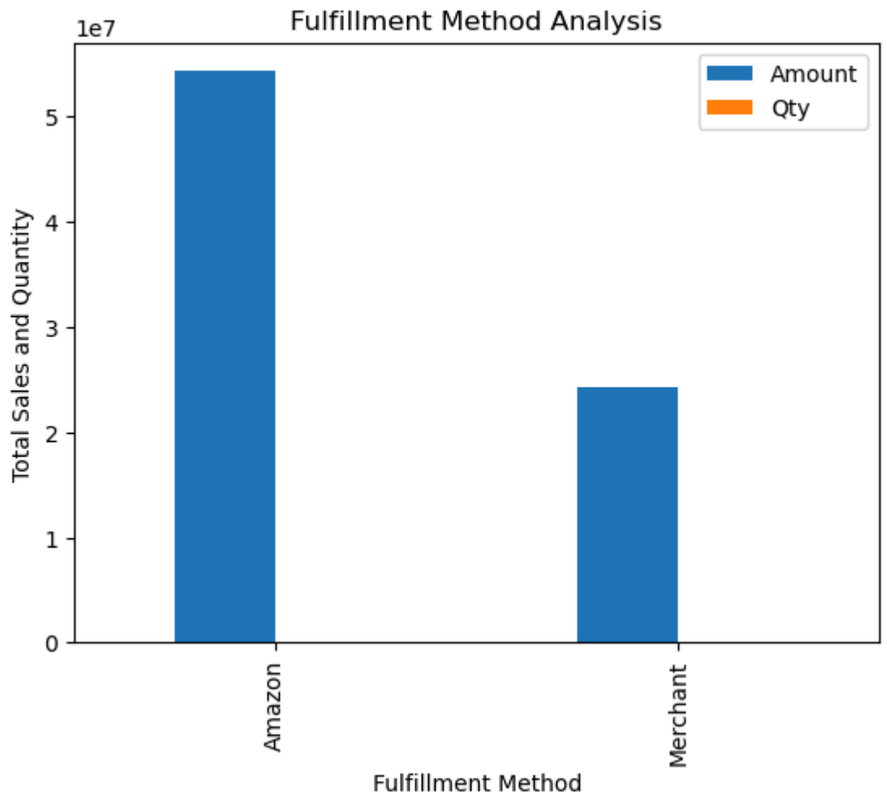
```
Total Sales: $78496786.39
```



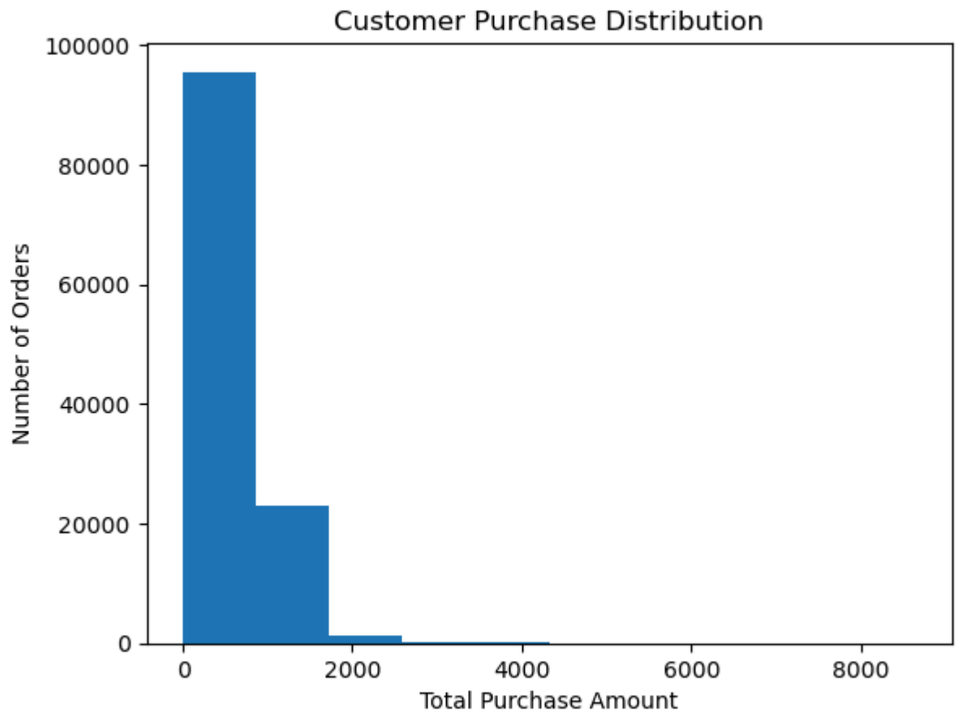

```
Product Distribution:
             Qty       Amount
Category
Blazzer   13934  11208506.12
Perfume    1051    789419.66
Shirt     44978  21269768.70
Shoes       152    123933.76
Socks       398    150397.50
T-shirt   45228  39154132.17
Trousers   9889   5341305.30
Wallet      863    458408.18
Watch         3       915.00

Top Selling Products:
             Qty
Category
T-shirt   45228
Shirt     44978
Blazzer   13934
Trousers   9889
Perfume    1051
Wallet      863
Socks       398
Shoes       152
Watch         3
```




```
Fulfillment Analysis:
                Amount    Qty
Fulfilment
Amazon      54262165.00  83990
Merchant    24234621.39  32506
```



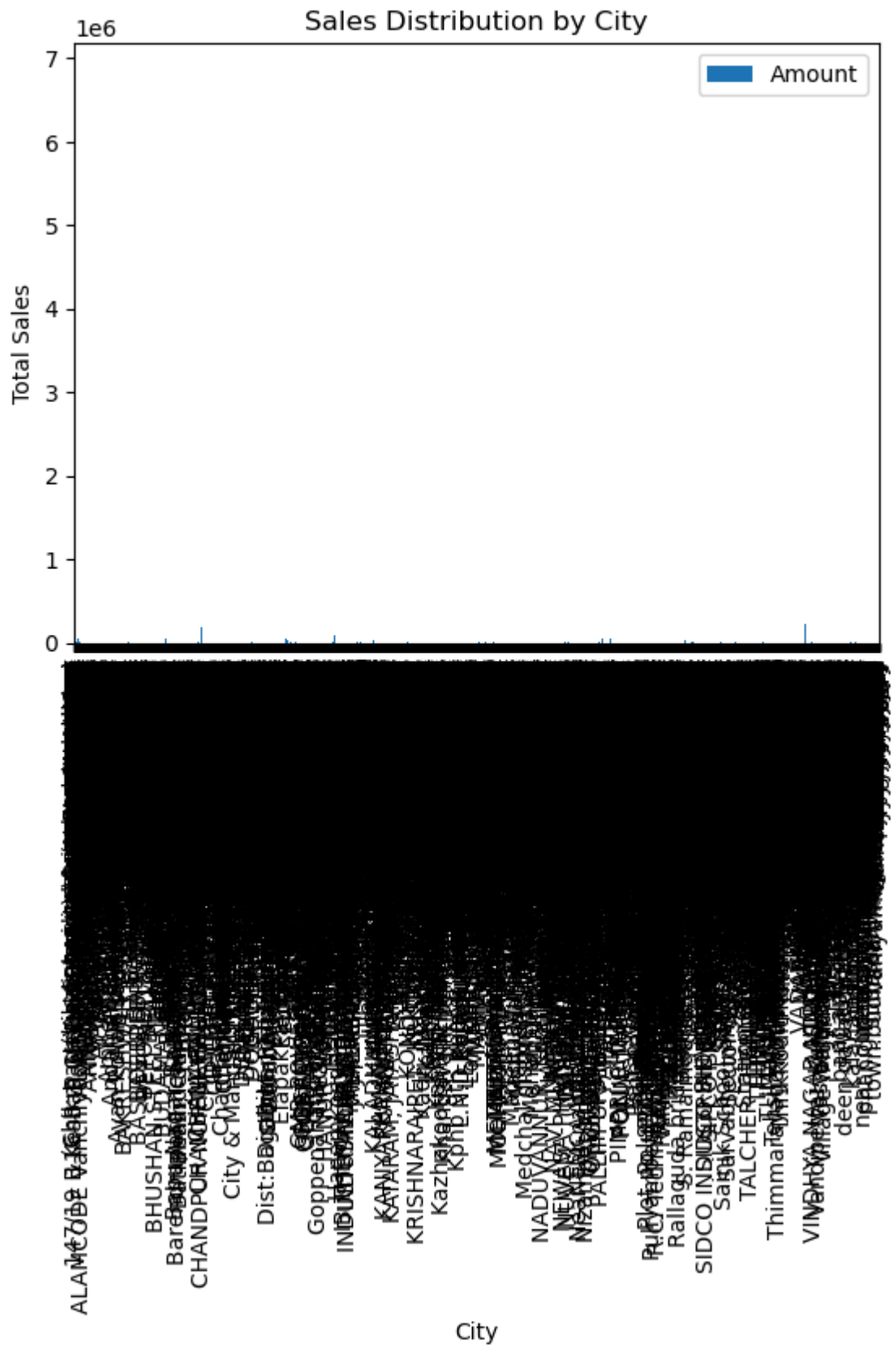

```
Customer Segments:
                      Amount  Qty
Order ID
171-0000547-8192359   301.0    1
171-0000902-4490745   544.0    1
171-0001409-6228339   422.0    1
171-0003082-5110755   563.0    1
171-0003738-2052324   379.0    1
...                     ...  ...
S02-9578181-3610412     0.0    1
S02-9599483-2736812     0.0    1
S02-9649067-3246849     0.0    1
S02-9736323-0094708     0.0    1
S02-9878098-5959538     0.0    1

[120229 rows x 2 columns]
```



Geographical Sales Distribution:

ship-city	Amount
(Chikmagalur disterict). (N.R pur thaluku)	389.0
(Via Cuncolim)Quepem,South Goa	1163.0
,HYDERABAD	563.0
,raibarely road faizabad (Ayodhya)	1122.0
..katra	641.0
...	...
yavatmal	735.0
yazali	487.0
yellapur	824.0
zirakpur	852.0
ýýýýýýýýýýý	2003.0

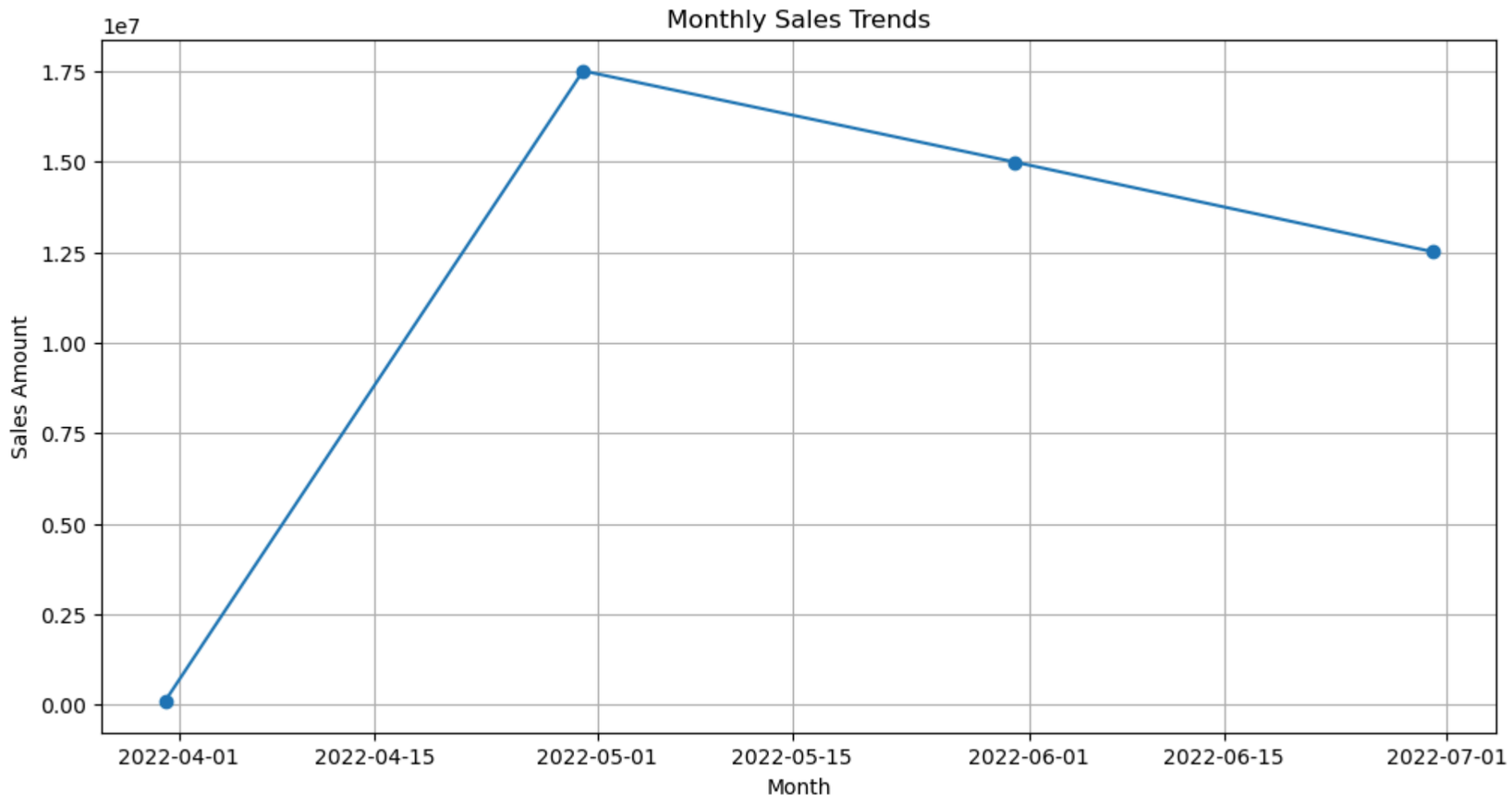
[8948 rows x 1 columns]

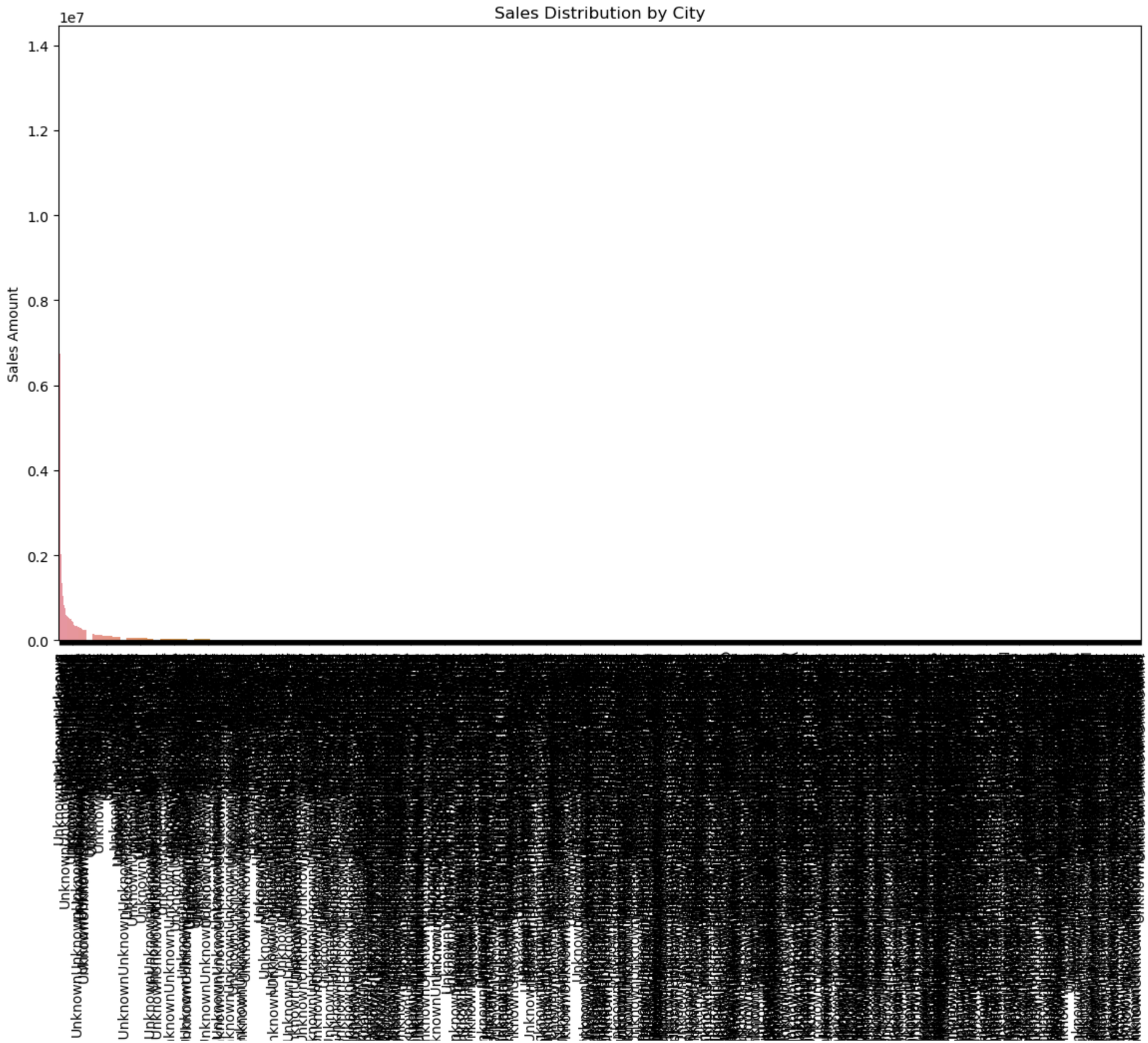

Sales Distribution by City
1e6
Amount
Total Sales
0
1
2
3
4
5
6
7
City

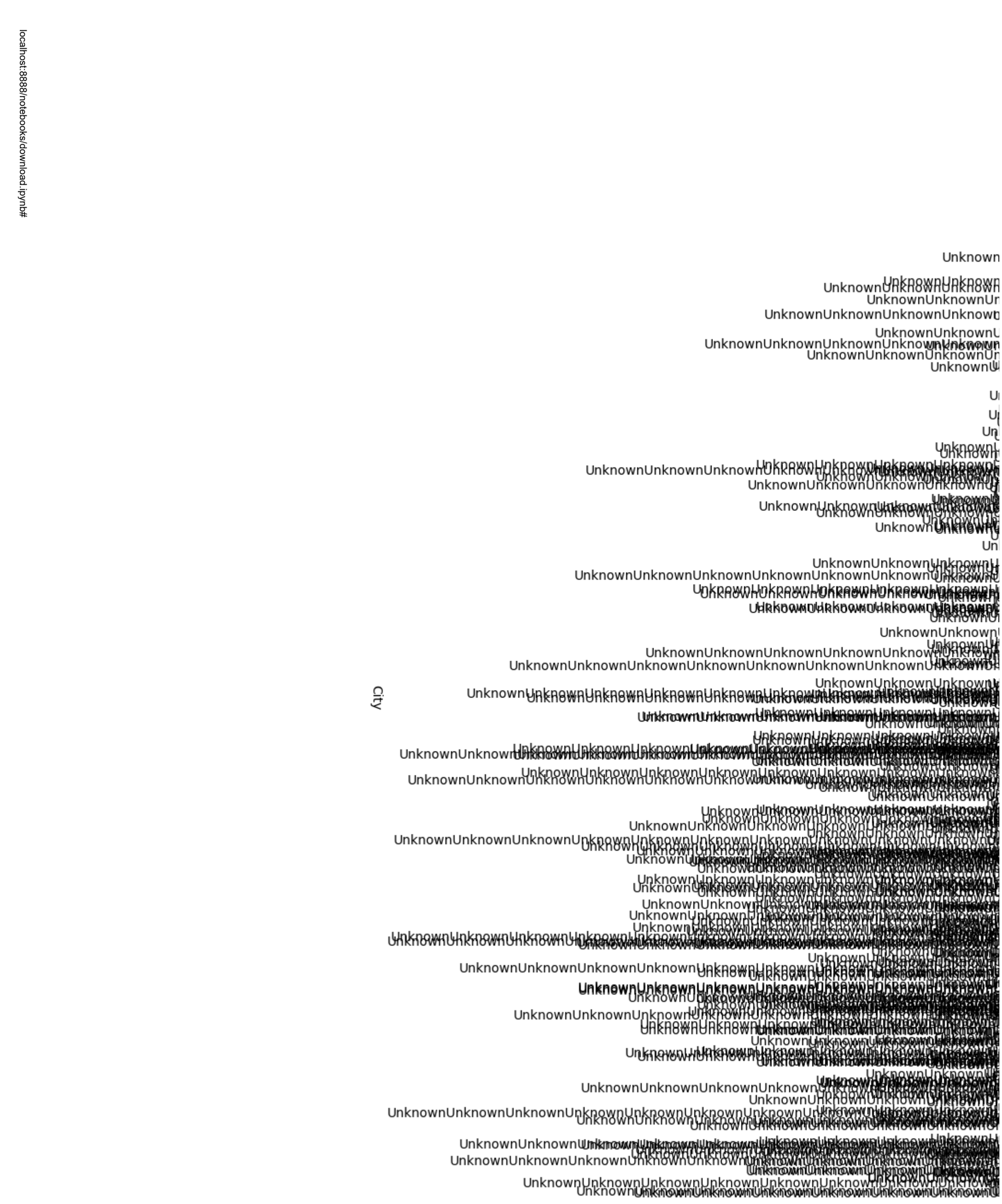
In [30]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

# Load the Dataset
df = pd.read_csv r"C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv", encoding='ISO-8859-1')

# Data Cleaning
df['ship-city'] = df['ship-city'].str.strip().replace({'(Chikmagalur disterict).': 'Chikmagalur', '...': 'Unknown'}, regex=True)
df['currency'].fillna('Unknown', inplace=True)
df['Amount'].fillna(0, inplace=True)
df.dropna(subset=['ship-city', 'ship-state', 'ship-postal-code', 'ship-country'], inplace=True)

# Inspect Date Format
print(df['Date'].head())

# Convert 'Date' to datetime format with different formats if necessary
try:
    df['Date'] = pd.to_datetime(df['Date'], format='%m-%d-%y', errors='coerce')
except ValueError:
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Check if conversion was successful
print(df['Date'].head())

# Drop rows with NaT in 'Date' column if any
df.dropna(subset=['Date'], inplace=True)
df.set_index('Date', inplace=True)

# Monthly Sales Trends
monthly_sales = df['Amount'].resample('M').sum()
plt.figure(figsize=(12, 6))
plt.plot(monthly_sales.index, monthly_sales.values, marker='o')
plt.title('Monthly Sales Trends')
plt.xlabel('Month')
plt.ylabel('Sales Amount')
plt.grid(True)
plt.show()

# Geographical Sales Distribution
city_sales = df.groupby('ship-city')['Amount'].sum().sort_values(ascending=False)
plt.figure(figsize=(14, 8))
sns.barplot(x=city_sales.index, y=city_sales.values)
plt.xticks(rotation=90)
plt.title('Sales Distribution by City')
plt.xlabel('City')
plt.ylabel('Sales Amount')
plt.show()

# Top Selling Products
top_products = df.groupby('Category')['Qty'].sum().sort_values(ascending=False)
plt.figure(figsize=(12, 6))
top_products.plot(kind='bar')
plt.title('Top Selling Products by Quantity')
plt.xlabel('Product Category')
plt.ylabel('Quantity Sold')
plt.grid(True)
plt.show()
```

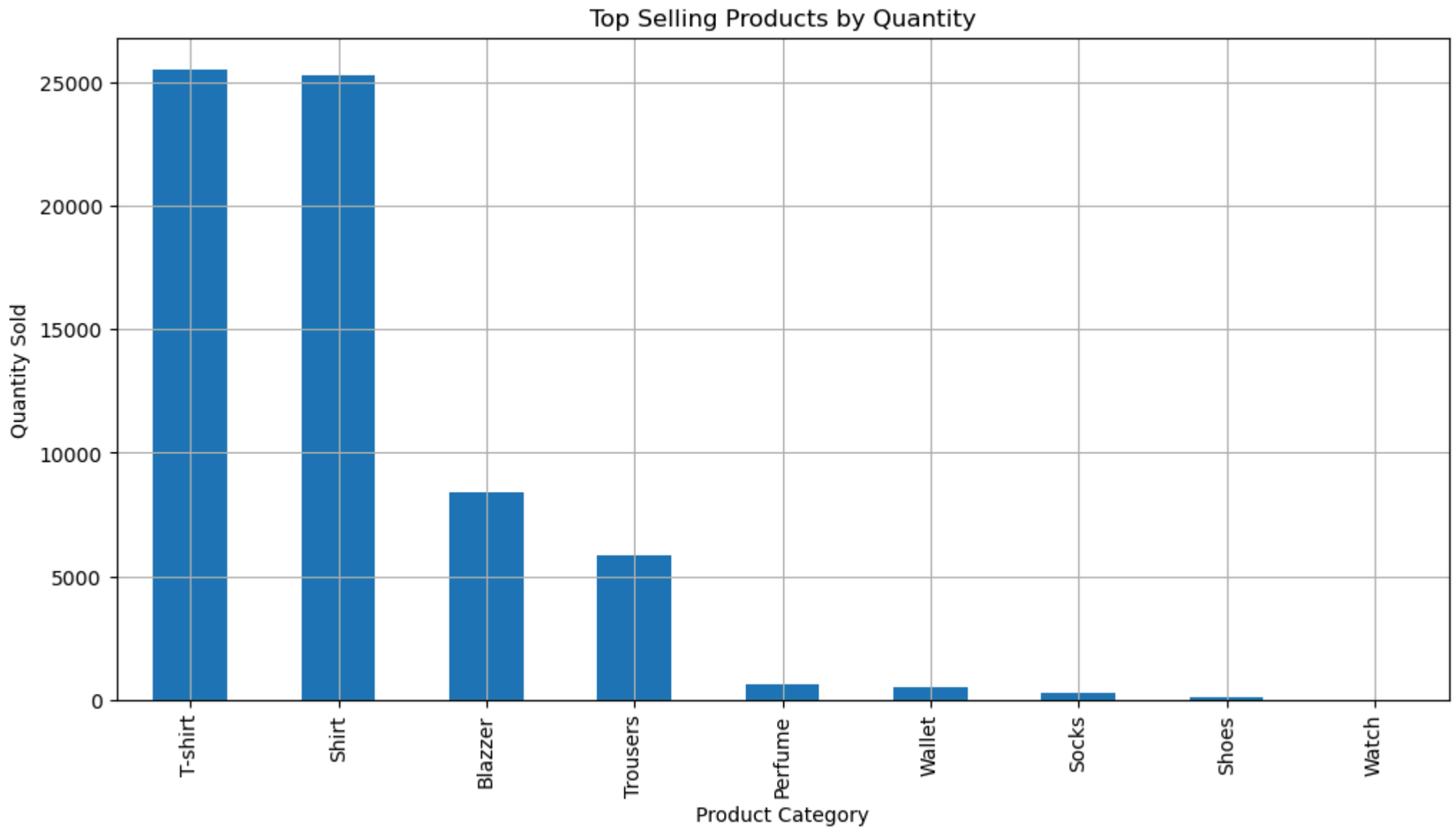
```
# Customer Segmentation
customer_data = df.groupby('Order ID').agg({'Amount': 'sum', 'Qty': 'sum'}).reset_index()
scaler = StandardScaler()
customer_data_scaled = scaler.fit_transform(customer_data[['Amount', 'Qty']])
kmeans = KMeans(n_clusters=3, random_state=0).fit(customer_data_scaled)
customer_data['Cluster'] = kmeans.labels_
print(customer_data.head())

# Fulfillment Analysis
fulfillment_analysis = df.groupby('Fulfilment').agg({'Amount': 'sum', 'Qty': 'sum'})
fulfillment_analysis.plot(kind='bar', figsize=(10, 6))
plt.title('Fulfillment Analysis')
plt.xlabel('Fulfillment Method')
plt.ylabel('Amount and Quantity')
plt.grid(True)
plt.show()

# Sales Distribution by Product Size
size_sales = df.groupby('Size')['Qty'].sum().sort_values(ascending=False)
plt.figure(figsize=(12, 6))
size_sales.plot(kind='bar')
plt.title('Sales Distribution by Product Size')
plt.xlabel('Size')
plt.ylabel('Quantity Sold')
plt.grid(True)
plt.show()
```

```
0    04-30-22
1    04-30-22
2    04-30-22
3    04-30-22
4    04-30-22
Name: Date, dtype: object
0   2022-04-30
1   2022-04-30
2   2022-04-30
3   2022-04-30
4   2022-04-30
Name: Date, dtype: datetime64[ns]
```


Monthly Sales Trends
1e7
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Sales Amount
2022-04-01
2022-04-15
2022-05-01
2022-05-15
2022-06-01
2022-06-15
2022-07-01
Month

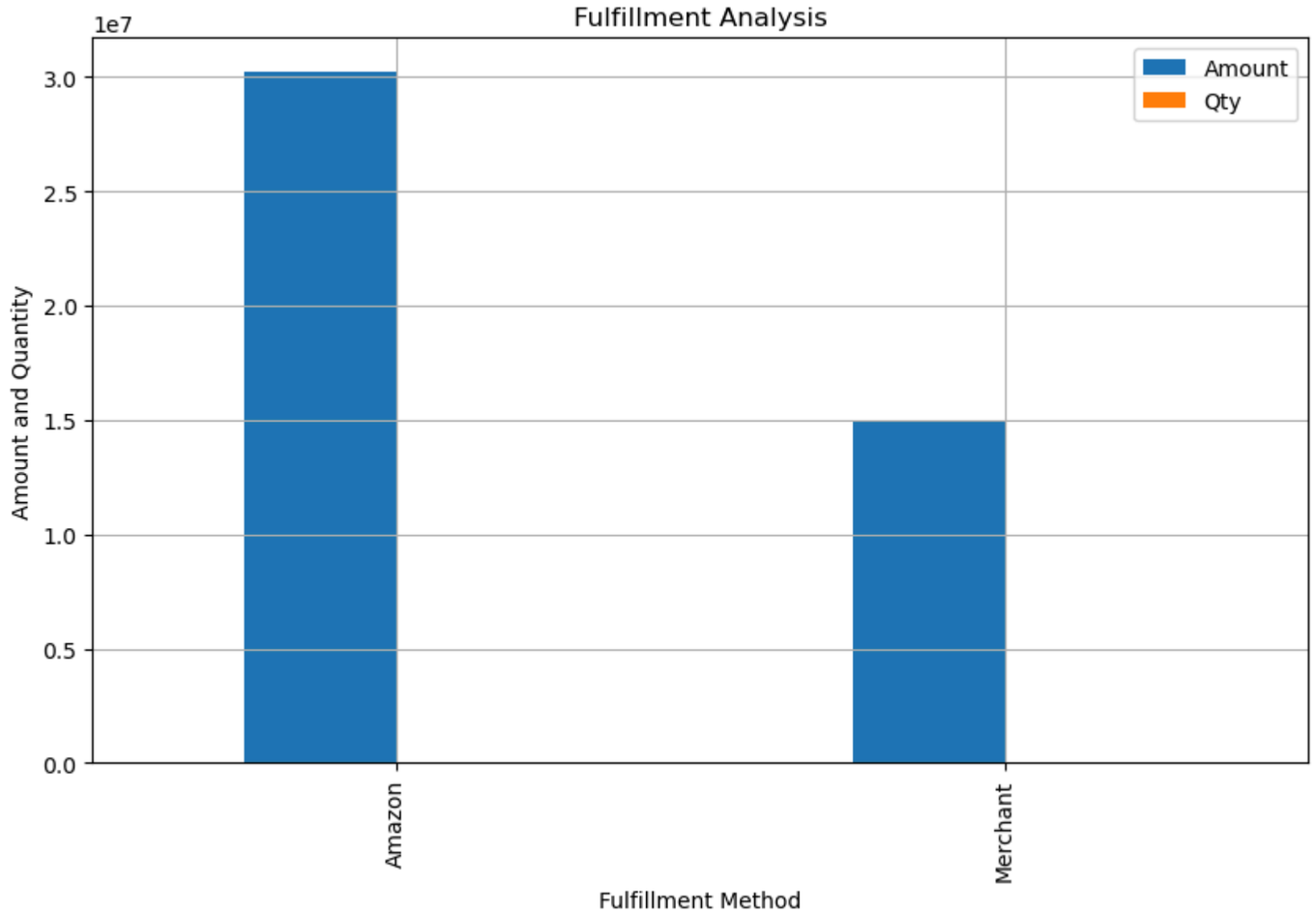

Sales Distribution by City
1e7
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Sales Amount

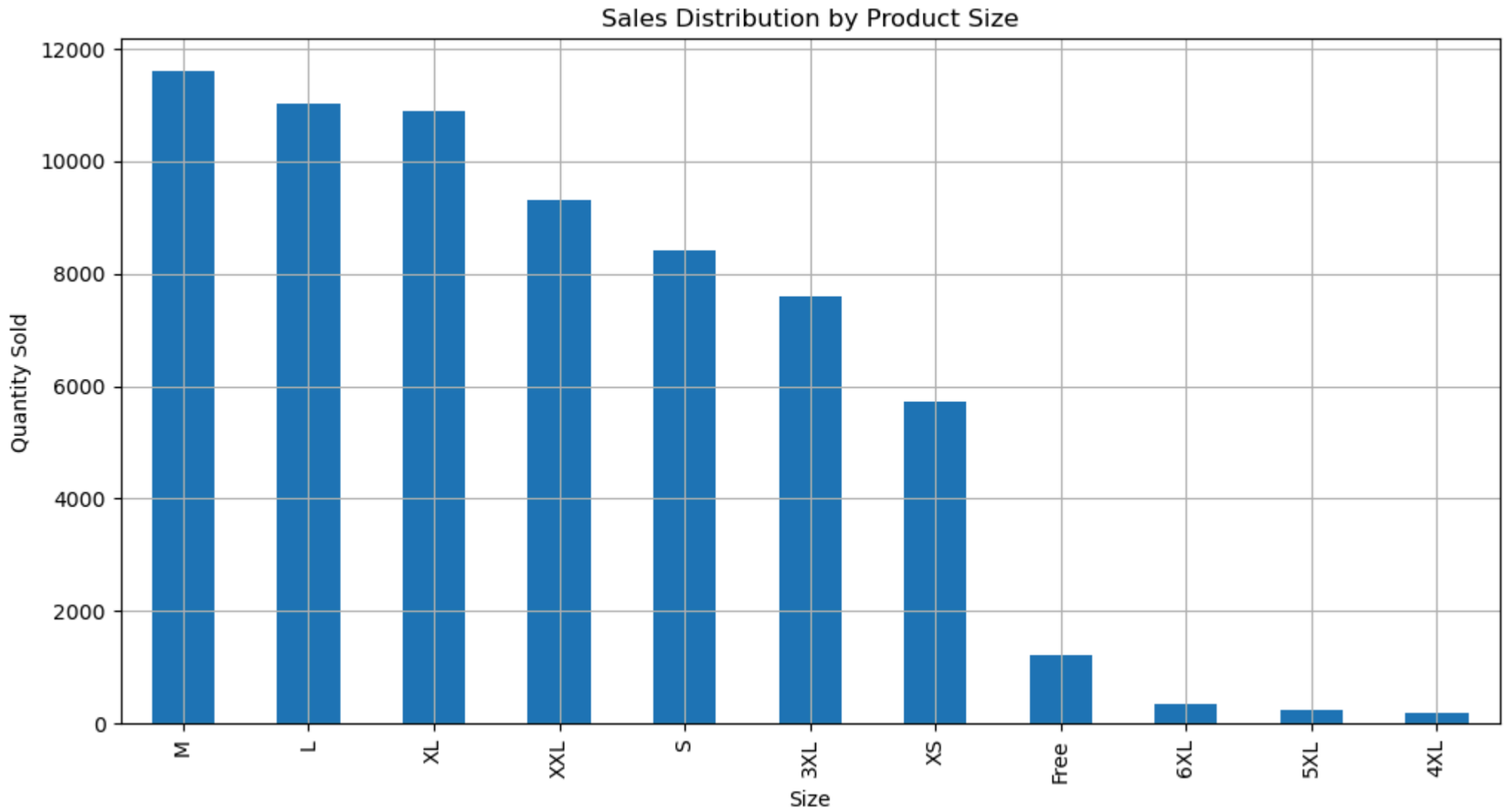

City




```
C:\Users\Anurag Kumar\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the valu
e of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

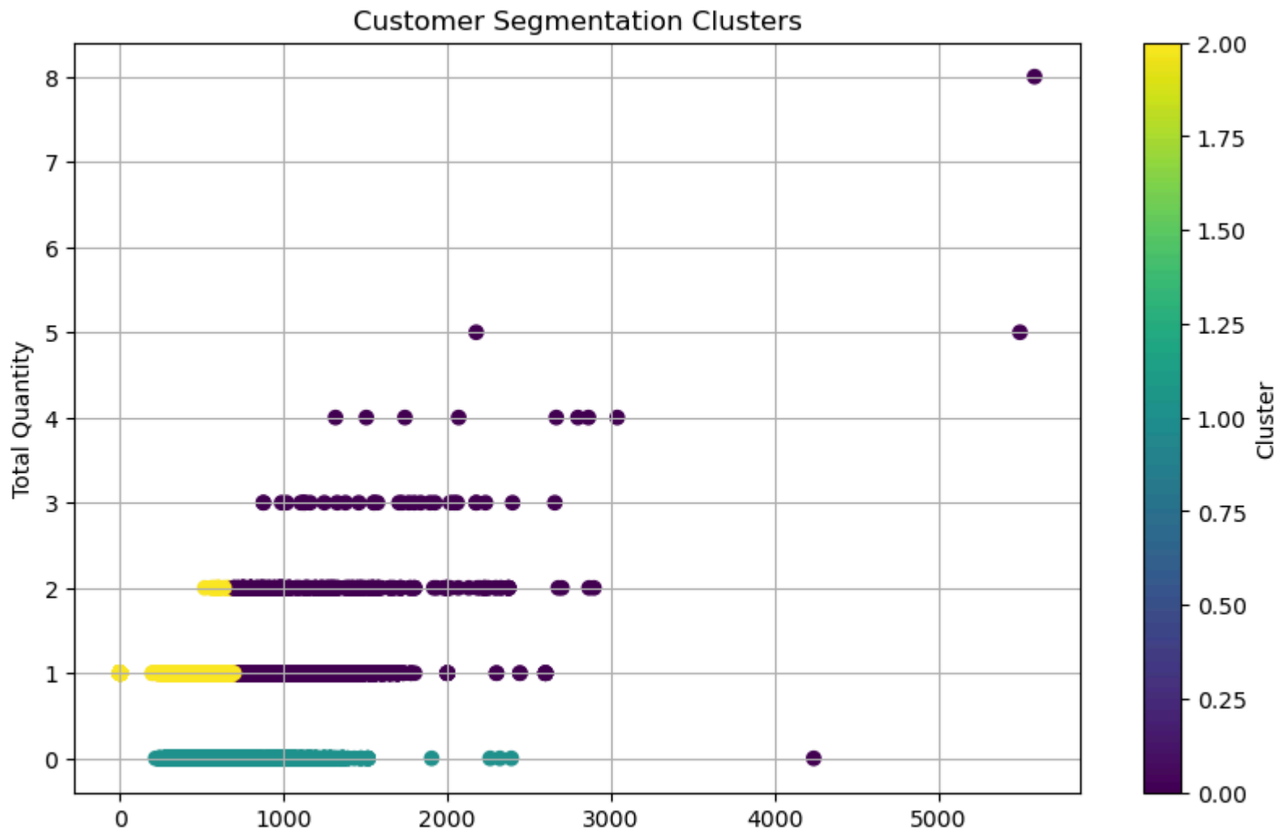
	Order ID	Amount	Qty	Cluster
0	171-0005637-8167567	579.0	1	1
1	171-0005741-2261112	558.0	1	1
2	171-0005999-3189913	1115.0	1	1
3	171-0006482-2020369	368.0	1	1
4	171-0007212-7125106	1092.0	1	1


Fulfillment Analysis
1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Amount and Quantity
Amount
Qty
Amazon
Merchant
Fulfillment Method


Sales Distribution by Product Size
Quantity Sold
0
2000
4000
6000
8000
10000
12000
M
L
XL
XXL
S
3XL
XS
Free
6XL
5XL
4XL
Size

In [32]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

# Load the Dataset
df = pd.read_csv r"C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv", encoding='ISO-8859-1')

# Drop rows with missing values in critical columns
df.dropna(subset=['ship-city', 'ship-state', 'ship-postal-code', 'ship-country'], inplace=True)

# Convert 'Date' to datetime format using infer_datetime_format
try:
    df['Date'] = pd.to_datetime(df['Date'], infer_datetime_format=True)
except Exception as e:
    print(f"Error parsing dates: {e}")

# Check if conversion was successful
print(df['Date'].head())

# Set 'Date' as index
df.set_index('Date', inplace=True)

# Select relevant columns for clustering
customer_data = df[['Amount', 'Qty']].copy()
customer_data.dropna(inplace=True)  # Drop rows with missing values in Amount or Qty

# Standardize the data
scaler = StandardScaler()
customer_data_scaled = scaler.fit_transform(customer_data)

# Apply KMeans Clustering
kmeans = KMeans(n_clusters=3, n_init=10, random_state=0).fit(customer_data_scaled)

# Adding cluster information to the dataframe
customer_data['Cluster'] = kmeans.labels_

# Plotting clusters
plt.figure(figsize=(10, 6))
scatter = plt.scatter(customer_data['Amount'], customer_data['Qty'], c=customer_data['Cluster'], cmap='viridis')
plt.colorbar(scatter, label='Cluster')
plt.title('Customer Segmentation Clusters')
plt.xlabel('Total Amount')
plt.ylabel('Total Quantity')
plt.grid(True)
plt.show()

# Analyzing clusters
cluster_summary = customer_data.groupby('Cluster').agg({'Amount': ['mean', 'sum'], 'Qty': ['mean', 'sum']})
print(cluster_summary)
```

```
0   2022-04-30
1   2022-04-30
2   2022-04-30
3   2022-04-30
4   2022-04-30
Name: Date, dtype: datetime64[ns]
```



	Amount		Qty	
	mean	sum	mean	sum
Cluster				
0	937.400523	40697243.72	1.009582	43831
1	620.755854	3184477.53	0.000000	0
2	477.820698	34688827.00	1.000275	72618

In [ ]:

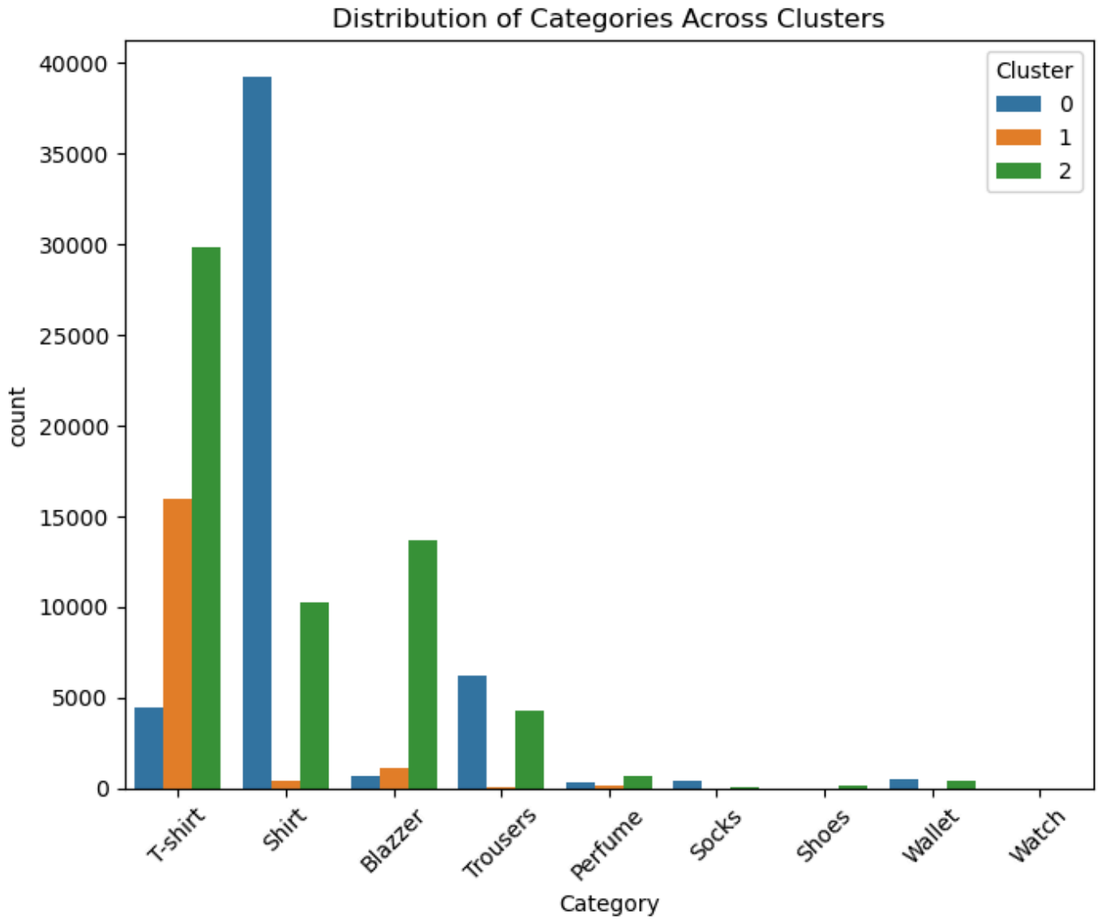
In [36]:
```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.impute import SimpleImputer
import matplotlib.pyplot as plt
import seaborn as sns

# Load your DataFrame (example)
df = pd.read_csv r"C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv", encoding='ISO-8859-1')

# Print initial data and info
print("Initial Data:")
print(df.head())
print("\nData Info:")
print(df.info())

# Check for missing values in features
print("\nMissing Values in Features:")
print(df[['Amount', 'Qty']].isna().sum())

# Impute missing values for 'Amount'
imputer = SimpleImputer(strategy='mean')
df['Amount'] = imputer.fit_transform(df[['Amount']])

# Ensure 'Qty' column does not contain missing values
df['Qty'].fillna(0, inplace=True)

# Prepare features for clustering
features = df[['Amount', 'Qty']]

# Perform KMeans clustering
kmeans = KMeans(n_clusters=3, n_init=10, random_state=42)
df['Cluster'] = kmeans.fit_predict(features)

# Print data with clusters
print("\nData with Clusters:")
print(df.head())

# Analyze characteristics of each cluster
cluster_analysis = df.groupby('Cluster').agg({
    'Amount': ['mean', 'sum'],
    'Qty': ['mean', 'sum'],
    'Category': lambda x: x.mode().iloc[0] if not x.mode().empty else None,  # Most frequent category
    'Sales Channel': lambda x: x.mode().iloc[0] if not x.mode().empty else None,  # Most frequent sales channel
    'Fulfilment': lambda x: x.mode().iloc[0] if not x.mode().empty else None  # Most frequent fulfilment method
})

print("\nCluster Analysis:")
print(cluster_analysis)

# Visualization
plt.figure(figsize=(14, 6))

# Distribution of Categories Across Clusters
plt.subplot(1, 2, 1)
sns.countplot(data=df, x='Category', hue='Cluster')
plt.title('Distribution of Categories Across Clusters')
plt.xticks(rotation=45)

# Distribution of Amount and Quantity Across Clusters
plt.subplot(1, 2, 2)
sns.boxplot(data=df, x='Cluster', y='Amount')
```

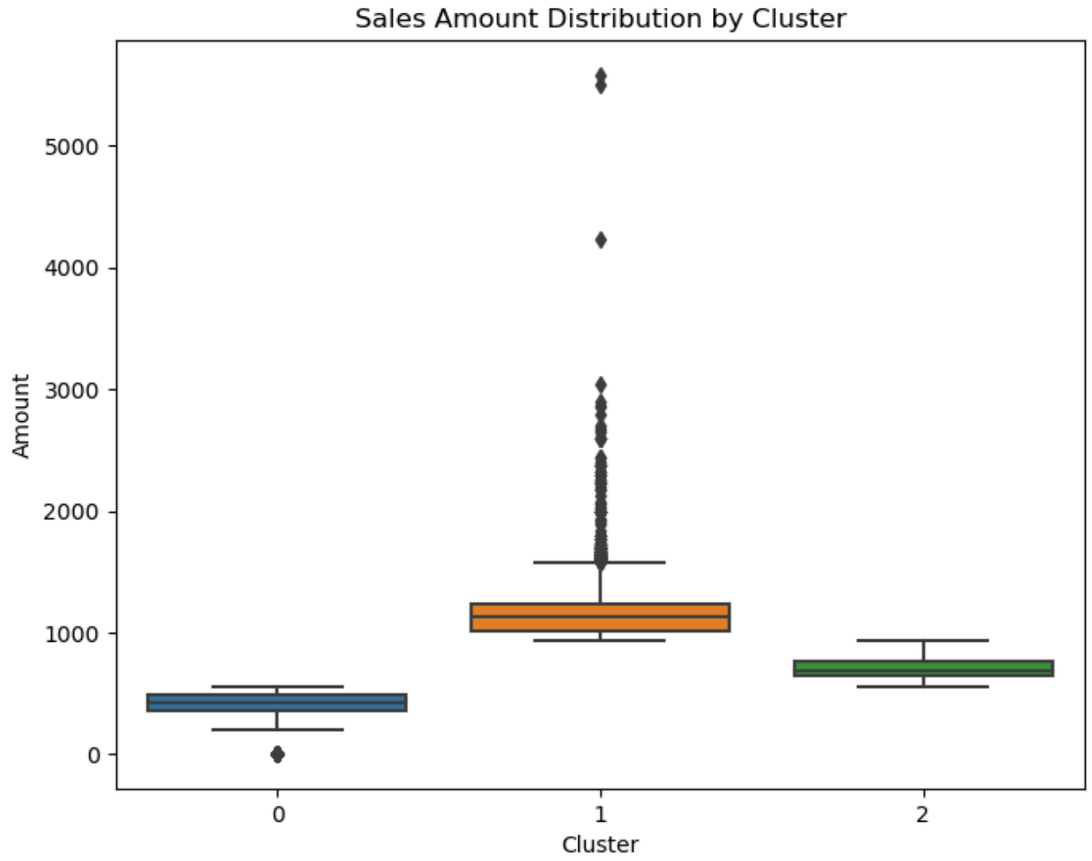
```
plt.title('Sales Amount Distribution by Cluster')

plt.tight_layout()
plt.show()

# Additional plot for Quantity distribution
plt.figure(figsize=(7, 6))
sns.boxplot(data=df, x='Cluster', y='Qty')
plt.title('Quantity Distribution by Cluster')
plt.tight_layout()
plt.show()
```

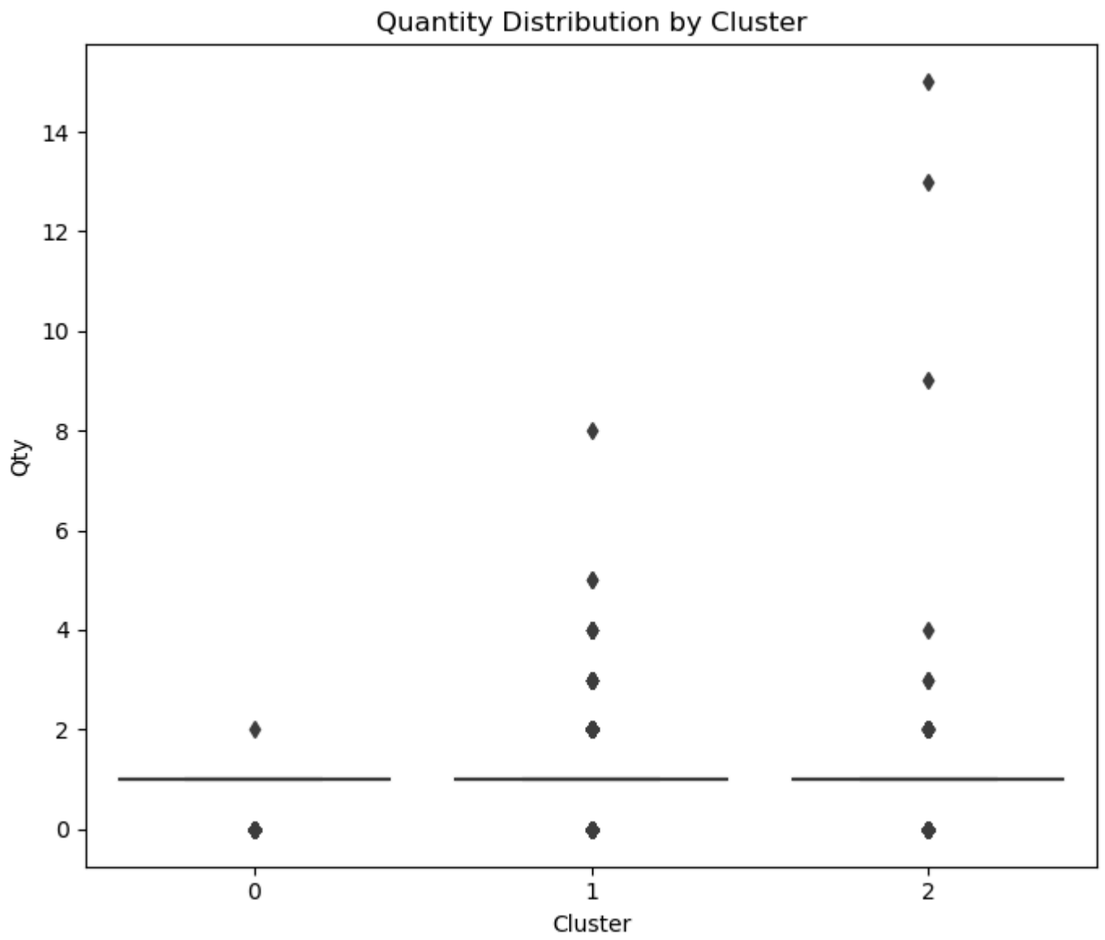
```
Initial Data:
   index            Order ID      Date                       Status  \
0      0  405-8078784-5731545  04-30-22                    Cancelled
1      1  171-9198151-1101146  04-30-22  Shipped - Delivered to Buyer
2      2  404-0687676-7273146  04-30-22                      Shipped
3      3  403-9615377-8133951  04-30-22                    Cancelled
4      4  407-1069790-7240320  04-30-22                      Shipped

  Fulfilment Sales Channel ship-service-level  Category Size Courier Status  \
0   Merchant     Amazon.in           Standard   T-shirt    S     On the Way
1   Merchant     Amazon.in           Standard     Shirt  3XL        Shipped
2     Amazon     Amazon.in          Expedited     Shirt   XL        Shipped
3   Merchant     Amazon.in           Standard   Blazzer    L     On the Way
4     Amazon     Amazon.in          Expedited  Trousers  3XL        Shipped

   ...  currency  Amount    ship-city   ship-state ship-postal-code  \
0  ...       INR  647.62       MUMBAI  MAHARASHTRA         400081.0
1  ...       INR  406.00    BENGALURU    KARNATAKA         560085.0
2  ...       INR  329.00  NAVI MUMBAI  MAHARASHTRA         410210.0
3  ...       INR  753.33   PUDUCHERRY   PUDUCHERRY         605008.0
4  ...       INR  574.00      CHENNAI   TAMIL NADU         600073.0

   ship-country    B2B  fulfilled-by New  PendingS
0            IN  False     Easy Ship NaN       NaN
1            IN  False     Easy Ship NaN       NaN
2            IN   True           NaN NaN       NaN
3            IN  False     Easy Ship NaN       NaN
4            IN  False           NaN NaN       NaN

[5 rows x 21 columns]
```

```
Data Info:
<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
None
```

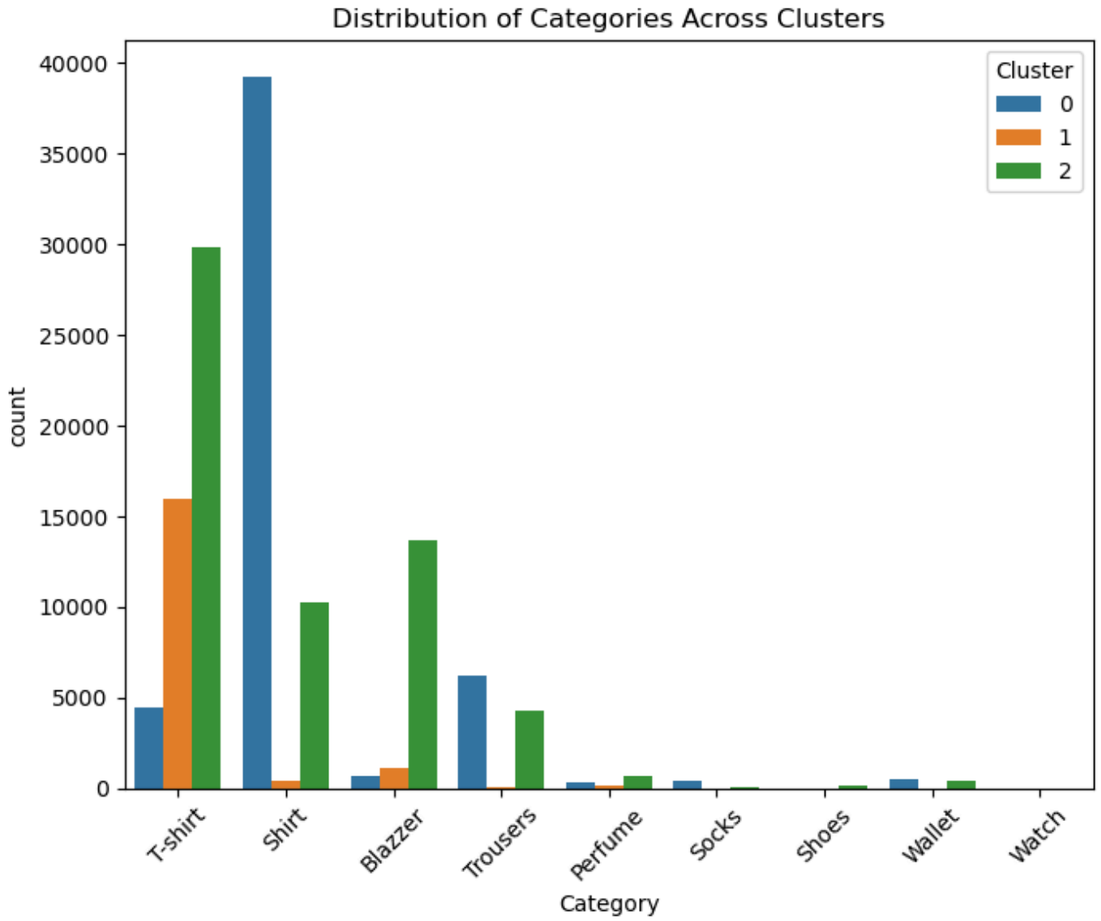
```
Missing Values in Features:
Amount    7800
Qty          0
dtype: int64

Data with Clusters:
   index             Order ID      Date                        Status  \
0      0  405-8078784-5731545  04-30-22                     Cancelled
1      1  171-9198151-1101146  04-30-22  Shipped - Delivered to Buyer
2      2  404-0687676-7273146  04-30-22                       Shipped
3      3  403-9615377-8133951  04-30-22                     Cancelled
4      4  407-1069790-7240320  04-30-22                       Shipped

  Fulfilment Sales Channel ship-service-level  Category Size Courier Status  \
0   Merchant     Amazon.in           Standard   T-shirt    S     On the Way
1   Merchant     Amazon.in           Standard     Shirt  3XL        Shipped
2     Amazon     Amazon.in          Expedited     Shirt   XL        Shipped
3   Merchant     Amazon.in           Standard   Blazzer    L     On the Way
4     Amazon     Amazon.in          Expedited  Trousers  3XL        Shipped

   ...  Amount    ship-city   ship-state ship-postal-code ship-country    B2B  \
0  ...  647.62       MUMBAI  MAHARASHTRA         400081.0           IN  False
1  ...  406.00    BENGALURU    KARNATAKA         560085.0           IN  False
2  ...  329.00  NAVI MUMBAI  MAHARASHTRA         410210.0           IN   True
3  ...  753.33   PUDUCHERRY   PUDUCHERRY         605008.0           IN  False
4  ...  574.00      CHENNAI   TAMIL NADU         600073.0           IN  False

  fulfilled-by  New PendingS  Cluster
0    Easy Ship  NaN      NaN        2
1    Easy Ship  NaN      NaN        0
2          NaN  NaN      NaN        0
3    Easy Ship  NaN      NaN        2
4          NaN  NaN      NaN        2

[5 rows x 22 columns]

Cluster Analysis:
              Amount                     Qty        Category Sales Channel  \
                mean           sum      mean    sum <lambda>      <lambda>
Cluster
0         407.867312  2.117892e+07  0.956342  49659    Shirt     Amazon.in
1        1158.361793  2.049605e+07  0.982593  17386  T-shirt     Amazon.in
2         707.156541  4.197398e+07  0.835653  49601  T-shirt     Amazon.in

        Fulfilment
          <lambda>
Cluster
0           Amazon
1           Amazon
2           Amazon
```

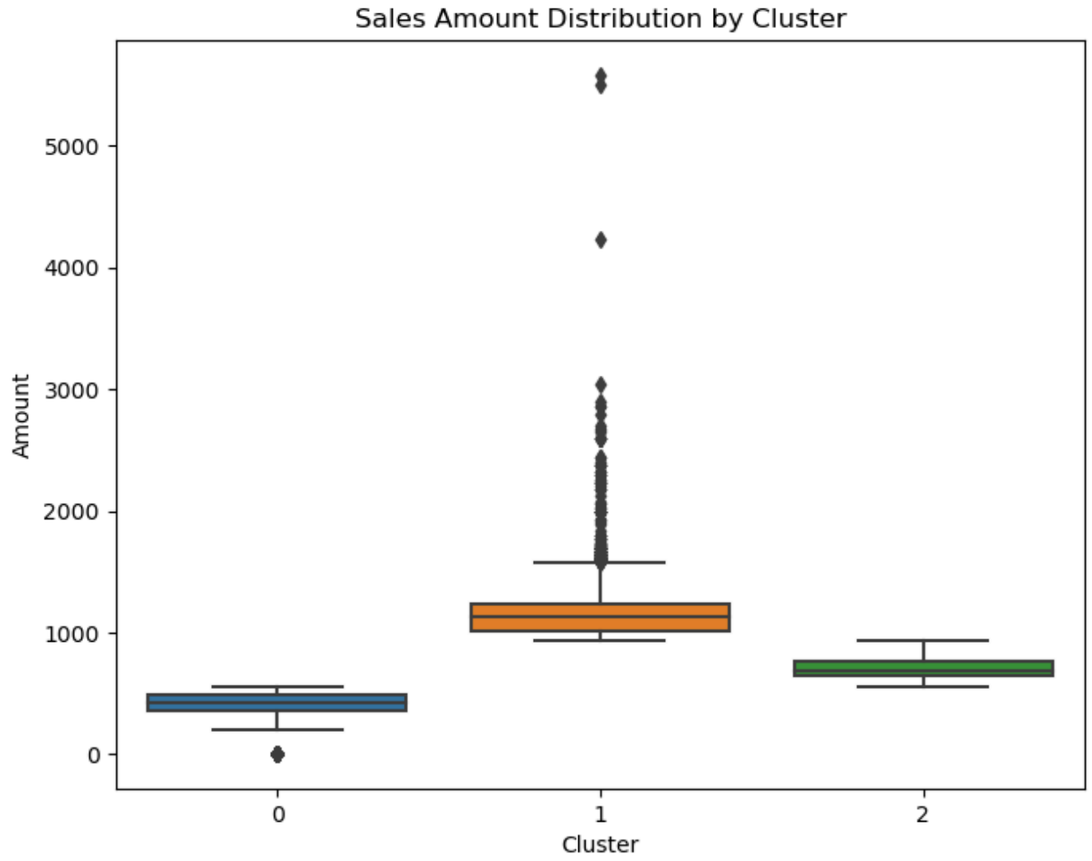

Distribution of Categories Across Clusters
Cluster
0
1
2
count
40000
35000
30000
25000
20000
15000
10000
5000
0
T-shirt
Shirt
Blazzer
Trousers
Perfume
Socks
Shoes
Wallet
Watch
Category

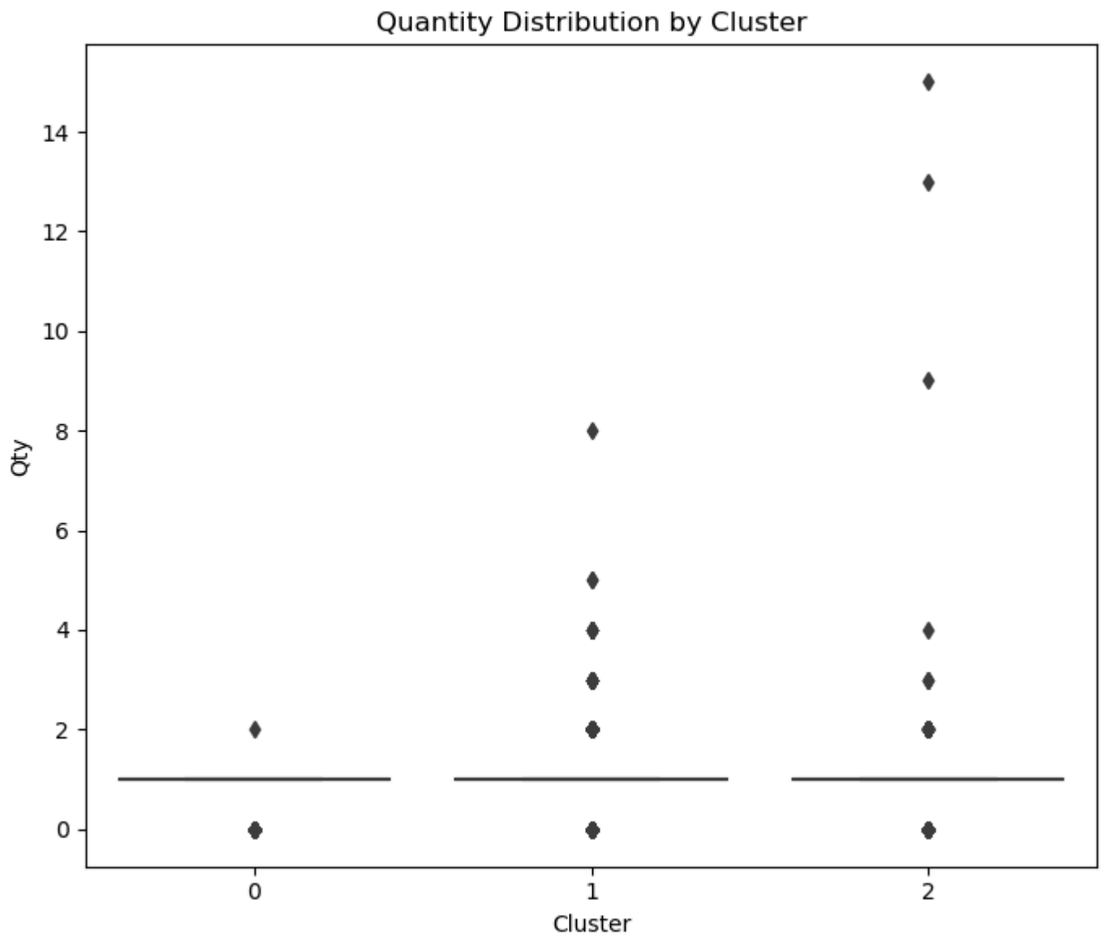

Sales Amount Distribution by Cluster
Amount
5000
4000
3000
2000
1000
0
0
1
2
Cluster


Quantity Distribution by Cluster
Qty
0
2
4
6
8
10
12
14
0
1
2
Cluster

In [37]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Visualize clusters
plt.figure(figsize=(14, 6))

# a. Distribution of Categories Across Clusters
plt.subplot(1, 2, 1)
sns.countplot(data=df, x='Category', hue='Cluster')
plt.title('Distribution of Categories Across Clusters')
plt.xticks(rotation=45)

# b. Distribution of Amount Across Clusters
plt.subplot(1, 2, 2)
sns.boxplot(data=df, x='Cluster', y='Amount')
plt.title('Sales Amount Distribution by Cluster')

plt.tight_layout()
plt.show()

# Additional plot for Quantity distribution
plt.figure(figsize=(7, 6))
sns.boxplot(data=df, x='Cluster', y='Qty')
plt.title('Quantity Distribution by Cluster')
plt.tight_layout()
plt.show()
```


Distribution of Categories Across Clusters
Cluster
0
1
2
count
40000
35000
30000
25000
20000
15000
10000
5000
0
T-shirt
Shirt
Blazzer
Trousers
Perfume
Socks
Shoes
Wallet
Watch
Category


Sales Amount Distribution by Cluster
Amount
5000
4000
3000
2000
1000
0
0
1
2
Cluster



In [ ]:

In [39]:
```
# Inspect the Date column to understand its format
print(df['Date'].head())
```

```
0    04-30-22
1    04-30-22
2    04-30-22
3    04-30-22
4    04-30-22
Name: Date, dtype: object
```

In [40]:
```
import pandas as pd

# Convert 'Date' to datetime format
df['Date'] = pd.to_datetime(df['Date'], format='%m-%d-%y', errors='coerce')

# Check for any NaT values after conversion
print(df['Date'].isna().sum())
```

```
55109
```

In [41]:
```
import pandas as pd

# Sample data
data = {
    'Date': ['04-30-22', '04-30-22', '04-30-22', '04-30-22', '04-30-22']
    # Add other columns as needed
}

# Create DataFrame
df = pd.DataFrame(data)

# Convert 'Date' to datetime format
df['Date'] = pd.to_datetime(df['Date'], format='%m-%d-%y', errors='coerce')

# Handle missing dates
df = df.dropna(subset=['Date'])  # or use fillna to fill missing dates

# Print the DataFrame to verify
print(df.head())

# Continue with your analysis
```

```
        Date
0 2022-04-30
1 2022-04-30
2 2022-04-30
3 2022-04-30
4 2022-04-30
```

In [ ]:

In [43]:
```
import pandas as pd

# Load your dataset
df = pd.read_csv r"C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv", encoding='ISO-8859-1')

# Print a few date entries to inspect
print(df['Date'].head())

# Convert 'Date' to datetime format
df['Date'] = pd.to_datetime(df['Date'], format='%m-%d-%y', errors='coerce')

# Check if conversion was successful
print(df['Date'].head())
print(df['Date'].isnull().sum())  # Check for any null values after conversion
```

```
0    04-30-22
1    04-30-22
2    04-30-22
3    04-30-22
4    04-30-22
Name: Date, dtype: object
0   2022-04-30
1   2022-04-30
2   2022-04-30
3   2022-04-30
4   2022-04-30
Name: Date, dtype: datetime64[ns]
55109
```

In [46]:
```
import pandas as pd

# Load the dataset
df = pd.read_csv r"C:\Users\Anurag Kumar\Downloads\Amazon Sale Report.csv", encoding='ISO-8859-1')

# Try automatic date conversion
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Check if there are any missing values in 'Date' column after conversion
missing_dates = df['Date'].isna().sum()
print(f"Missing Dates after Conversion: {missing_dates}")

# If there are missing dates, inspect a sample to understand the issue
if missing_dates > 0:
    print("Sample of rows with conversion issues:")
    print(df[df['Date'].isna()].head())

# Proceed with your analysis if the conversion is successful
if missing_dates == 0:
    # 1. Sales Overview
    sales_overview = df.groupby(df['Date'].dt.to_period('M')).agg({
        'Amount': ['sum', 'mean'],
        'Qty': 'sum'
    }).reset_index()
    sales_overview.columns = ['Date', 'Total Amount', 'Average Amount', 'Total Qty']
    print("\nSales Overview:")
    print(sales_overview.head())

    # 2. Product Analysis
    product_analysis = df.groupby('Category').agg({
        'Amount': 'sum',
        'Qty': 'sum'
    }).reset_index()
    product_analysis.columns = ['Category', 'Total Revenue', 'Total Qty']
    print("\nProduct Analysis:")
    print(product_analysis.head())

    size_analysis = df.groupby('Size').agg({
        'Amount': 'sum',
        'Qty': 'sum'
    }).reset_index()
    size_analysis.columns = ['Size', 'Total Revenue', 'Total Qty']
    print("\nSize Analysis:")
    print(size_analysis.head())

    # 3. Fulfillment Analysis
    fulfillment_analysis = df.groupby('Fulfilment').agg({
        'Amount': 'sum',
        'Qty': 'sum'
    }).reset_index()
    fulfillment_analysis.columns = ['Fulfilment Method', 'Total Revenue', 'Total Qty']
    print("\nFulfillment Analysis:")
    print(fulfillment_analysis.head())

    # 4. Customer Segmentation
    customer_segmentation = df.groupby('Order ID').agg({
        'Amount': 'sum'
    }).reset_index()
    customer_segmentation.columns = ['Order ID', 'Total Spending']
    print("\nCustomer Segmentation:")
    print(customer_segmentation.head())
```

```
    # 5. Geographical Analysis
    state_analysis = df.groupby('ship-state').agg({
        'Amount': 'sum',
        'Qty': 'sum'
    }).reset_index()
    state_analysis.columns = ['State', 'Total Revenue', 'Total Qty']
    print("\nState Analysis:")
    print(state_analysis.head())

    city_analysis = df.groupby('ship-city').agg({
        'Amount': 'sum',
        'Qty': 'sum'
    }).reset_index()
    city_analysis.columns = ['City', 'Total Revenue', 'Total Qty']
    print("\nCity Analysis:")
    print(city_analysis.head())

    # 6. Business Insights
    insights = {
        "Sales Trends": "Analyze the trends from the sales_overview DataFrame.",
        "Popular Products": "Review the product_analysis and size_analysis DataFrames.",
        "Fulfillment Efficiency": "Evaluate the fulfillment_analysis DataFrame for efficiency insights.",
        "Customer Segments": "Examine the customer_segmentation DataFrame for spending patterns.",
        "Geographic Focus": "Look at state_analysis and city_analysis DataFrames to identify key regions."
    }

    print("\nBusiness Insights:")
    for key, value in insights.items():
        print(f"{key}: {value}")
else:
    print("Please check the date format or data for issues.")
```

```
Missing Dates after Conversion: 0

Sales Overview:
      Date  Total Amount  Average Amount  Total Qty
0  2022-03     101683.85      627.678086        156
1  2022-04   28836200.27      626.002958      44203
2  2022-05   26226476.75      663.356858      38011
3  2022-06   23425809.38      661.484424      34276

Product Analysis:
  Category  Total Revenue  Total Qty
0  Blazzer    11215104.12      13943
1  Perfume      789419.66       1051
2    Shirt    21297770.08      45044
3    Shoes      124752.76        153
4    Socks      150757.50        399

Size Analysis:
   Size  Total Revenue  Total Qty
0   3XL     9034156.30      13360
1   4XL      334451.64        398
2   5XL      425156.63        513
3   6XL      576249.33        688
4  Free     1373495.60       2070

Fulfillment Analysis:
  Fulfilment Method  Total Revenue  Total Qty
0            Amazon    54327540.00      84097
1          Merchant    24262630.25      32549

Customer Segmentation:
              Order ID  Total Spending
0  171-0000547-8192359           301.0
1  171-0000902-4490745           544.0
2  171-0001409-6228339           422.0
3  171-0003082-5110755           563.0
4  171-0003738-2052324           379.0

State Analysis:
               State  Total Revenue  Total Qty
0  ANDAMAN & NICOBAR      157424.62        225
1     ANDHRA PRADESH     3217859.86       4816
2                APO           0.00          0
3                 AR         493.00          1
4  ARUNACHAL PRADESH       95235.00        130

City Analysis:
                                        City  Total Revenue  Total Qty
0  (Chikmagalur disterict).     (N.R pur thaluku)          389.0          1
1                 (Via Cuncolim)Quepem,South Goa         1163.0          1
2                                    ,HYDERABAD          563.0          1
3             ,raibarely road faizabad (Ayodhya)         1122.0          1
4                                     ..katra           641.0          1

Business Insights:
Sales Trends: Analyze the trends from the sales_overview DataFrame.
Popular Products: Review the product_analysis and size_analysis DataFrames.
Fulfillment Efficiency: Evaluate the fulfillment_analysis DataFrame for efficiency insights.
Customer Segments: Examine the customer_segmentation DataFrame for spending patterns.
Geographic Focus: Look at state_analysis and city_analysis DataFrames to identify key regions.
```