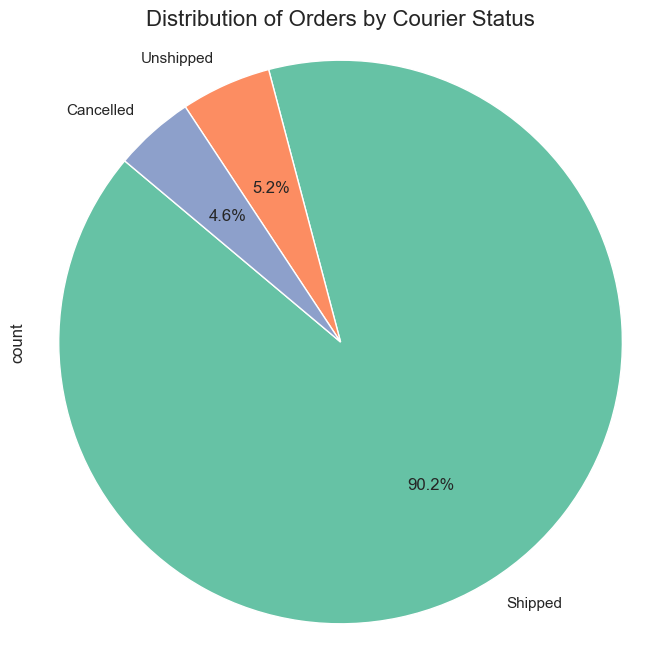
import pandas as pd

data=pd.read\_csv('Amazon Sale Report.csv.zip')  
data['ship-city'].fillna(data['ship-city'].mode()[0], inplace=True)  
data['ship-state'].fillna(data['ship-state'].mode()[0], inplace=True)  
data['ship-country'].fillna(data['ship-country'].mode()[0], inplace=True)  
data['ship-postal-code'].fillna(data['ship-postal-code'].mode()[0], inplace=True)  
data['Amount'].fillna(data['Amount'].mean(), inplace=True)  
data['currency'].fillna('USD', inplace=True)  
data['promotion-ids'].fillna(data['promotion-ids'].mode()[0], inplace=True)  
data['fulfilled-by'].fillna(data['fulfilled-by'].mode()[0], inplace=True)  
data['Unnamed: 22'].fillna(data['Unnamed: 22'].mode()[0], inplace=True)  
data['Courier Status'].fillna(data['Courier Status'].mode()[0], inplace=True)  
print(data.isnull().sum())  
print(data.dtypes)  
print(data.columns)  
duplicates = data.duplicated()  
print(data.head())

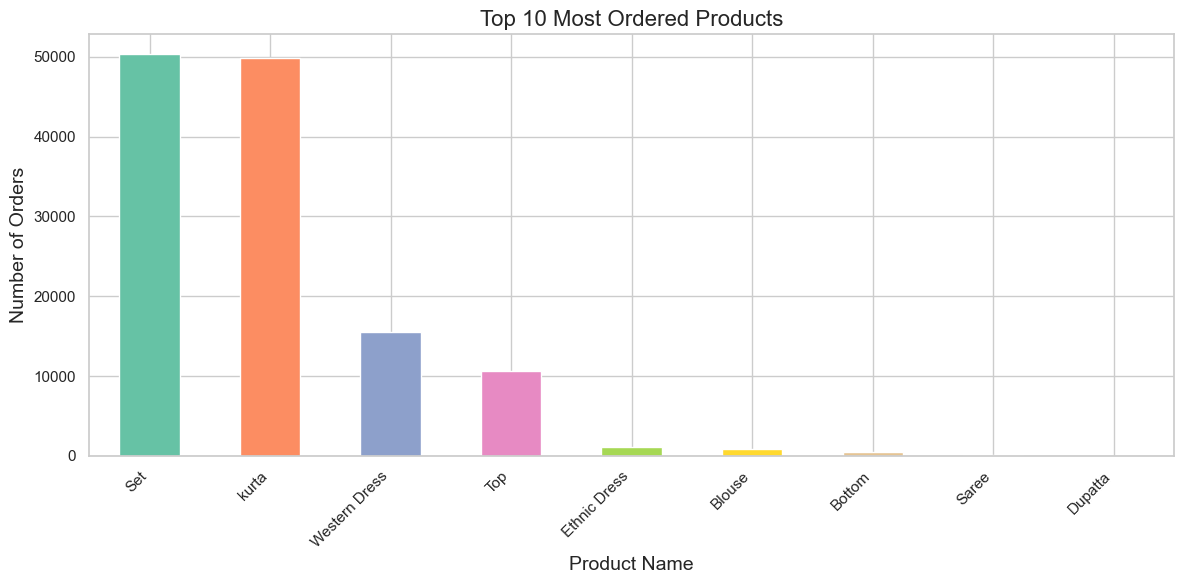
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:1: DtypeWarning: Columns (23) have mixed types. Specify dtype option on import or set low\_memory=False.  
 data=pd.read\_csv('Amazon Sale Report.csv.zip')  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['ship-city'].fillna(data['ship-city'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['ship-state'].fillna(data['ship-state'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:4: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['ship-country'].fillna(data['ship-country'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:5: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['ship-postal-code'].fillna(data['ship-postal-code'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:6: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['Amount'].fillna(data['Amount'].mean(), inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:7: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['currency'].fillna('USD', inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:8: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['promotion-ids'].fillna(data['promotion-ids'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:9: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['fulfilled-by'].fillna(data['fulfilled-by'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:10: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['Unnamed: 22'].fillna(data['Unnamed: 22'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:10: FutureWarning: Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and will change in a future version. Call result.infer\_objects(copy=False) instead. To opt-in to the future behavior, set `pd.set\_option('future.no\_silent\_downcasting', True)`  
 data['Unnamed: 22'].fillna(data['Unnamed: 22'].mode()[0], inplace=True)  
C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\3032738106.py:11: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.  
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.  
  
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.  
  
  
 data['Courier Status'].fillna(data['Courier Status'].mode()[0], inplace=True)

index 0  
Order ID 0  
Date 0  
Status 0  
Fulfilment 0  
Sales Channel 0  
ship-service-level 0  
Style 0  
SKU 0  
Category 0  
Size 0  
ASIN 0  
Courier Status 0  
Qty 0  
currency 0  
Amount 0  
ship-city 0  
ship-state 0  
ship-postal-code 0  
ship-country 0  
promotion-ids 0  
B2B 0  
fulfilled-by 0  
Unnamed: 22 0  
dtype: int64  
index int64  
Order ID object  
Date object  
Status object  
Fulfilment object  
Sales Channel object  
ship-service-level object  
Style object  
SKU object  
Category object  
Size object  
ASIN object  
Courier Status object  
Qty int64  
currency object  
Amount float64  
ship-city object  
ship-state object  
ship-postal-code float64  
ship-country object  
promotion-ids object  
B2B bool  
fulfilled-by object  
Unnamed: 22 bool  
dtype: object  
Index(['index', 'Order ID', 'Date', 'Status', 'Fulfilment', 'Sales Channel ',  
 'ship-service-level', 'Style', 'SKU', 'Category', 'Size', 'ASIN',  
 'Courier Status', 'Qty', 'currency', 'Amount', 'ship-city',  
 'ship-state', 'ship-postal-code', 'ship-country', 'promotion-ids',  
 'B2B', 'fulfilled-by', 'Unnamed: 22'],  
 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 Style SKU \  
0 Merchant Amazon.in Standard SET389 SET389-KR-NP-S   
1 Merchant Amazon.in Standard JNE3781 JNE3781-KR-XXXL   
2 Amazon Amazon.in Expedited JNE3371 JNE3371-KR-XL   
3 Merchant Amazon.in Standard J0341 J0341-DR-L   
4 Amazon Amazon.in Expedited JNE3671 JNE3671-TU-XXXL   
  
 Category ... currency Amount ship-city ship-state \  
0 Set ... INR 647.62 MUMBAI MAHARASHTRA   
1 kurta ... INR 406.00 BENGALURU KARNATAKA   
2 kurta ... INR 329.00 NAVI MUMBAI MAHARASHTRA   
3 Western Dress ... INR 753.33 PUDUCHERRY PUDUCHERRY   
4 Top ... INR 574.00 CHENNAI TAMIL NADU   
  
 ship-postal-code ship-country \  
0 400081.0 IN   
1 560085.0 IN   
2 410210.0 IN   
3 605008.0 IN   
4 600073.0 IN   
  
 promotion-ids B2B fulfilled-by \  
0 IN Core Free Shipping 2015/04/08 23-48-5-108 False Easy Ship   
1 Amazon PLCC Free-Financing Universal Merchant ... False Easy Ship   
2 IN Core Free Shipping 2015/04/08 23-48-5-108 True Easy Ship   
3 IN Core Free Shipping 2015/04/08 23-48-5-108 False Easy Ship   
4 IN Core Free Shipping 2015/04/08 23-48-5-108 False Easy Ship   
  
 Unnamed: 22   
0 False   
1 False   
2 False   
3 False   
4 False   
  
[5 rows x 24 columns]

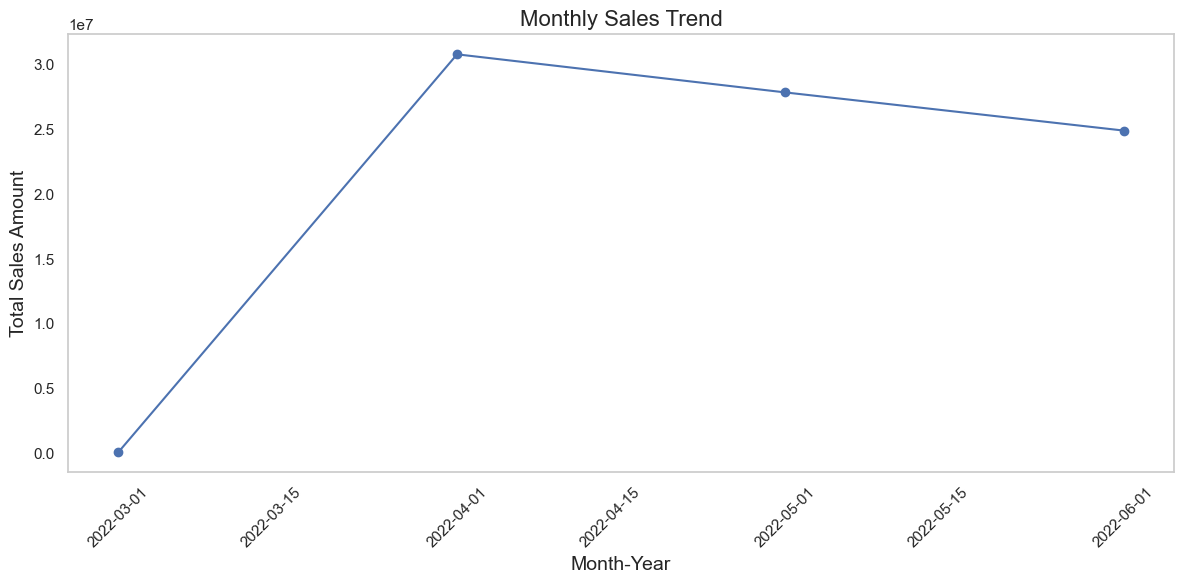
#pie chart to visualize the no. of orders that got accepted, rejected and cancelled  
import matplotlib.pyplot as plt  
import seaborn as sns  
# Create a pie chart   
plt.figure(figsize=(8, 8))  
data['Courier Status'].value\_counts().plot.pie(autopct='%1.1f%%', startangle=140, colors=sns.color\_palette('Set2'))  
  
# Set the title  
plt.title('Distribution of Orders by Courier Status', fontsize=16)  
# Show the plot  
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.  
plt.show()



#Bar chart to visualize which is the most ordered item  
import matplotlib.pyplot as plt  
import seaborn as sns  
# Create a bar chart  
plt.figure(figsize=(12, 6))  
data['Category'].value\_counts().head(10).plot.bar(color=sns.color\_palette('Set2'))  
# Set the title and labels  
plt.title('Top 10 Most Ordered Products', fontsize=16)  
plt.xlabel('Product Name', fontsize=14)  
plt.ylabel('Number of Orders', fontsize=14)  
  
# Show the plot  
plt.xticks(rotation=45, ha='right')   
plt.tight\_layout() # Adjust layout to prevent clipping of tick-labels  
plt.show()

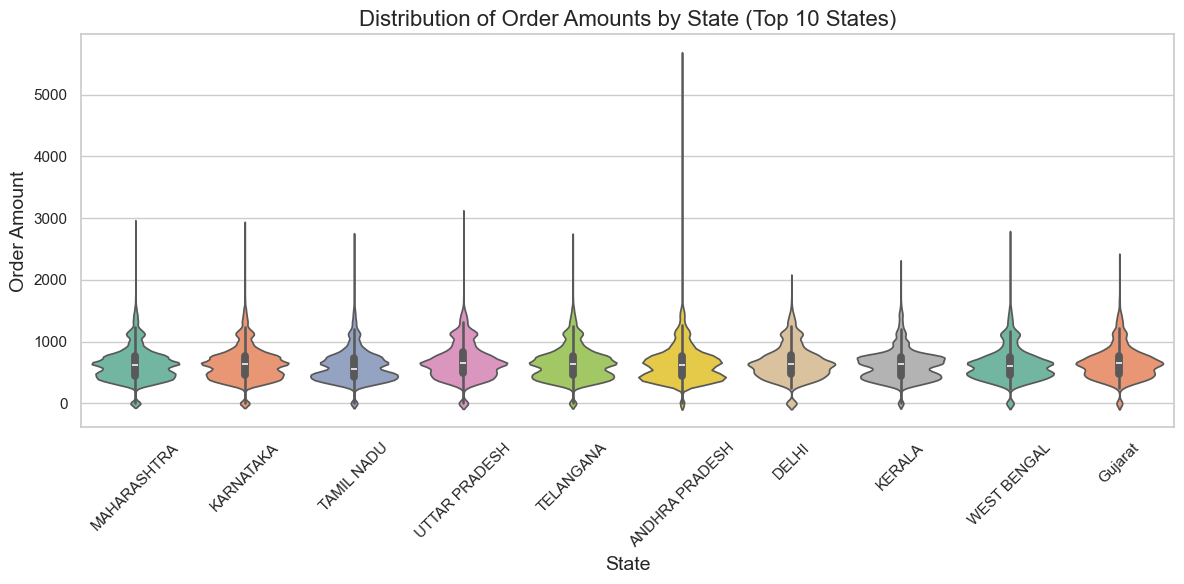


#I want to visualize the sales trend over the months  
import matplotlib.pyplot as plt  
import pandas as pd  
data['Date'] = pd.to\_datetime(data['Date'])  
data['month\_year'] = data['Date'].dt.to\_period('M')  
monthly\_sales = data.groupby('month\_year')['Amount'].sum().reset\_index()  
monthly\_sales['month\_year'] = monthly\_sales['month\_year'].dt.to\_timestamp()   
plt.figure(figsize=(12, 6))  
plt.plot(monthly\_sales['month\_year'], monthly\_sales['Amount'], marker='o', color='b')  
plt.title('Monthly Sales Trend', fontsize=16)  
plt.xlabel('Month-Year', fontsize=14)  
plt.ylabel('Total Sales Amount', fontsize=14)  
plt.xticks(rotation=45)   
plt.grid()  
plt.tight\_layout()   
plt.show()



#visualize which region ordered most items using violin plot only for top 10 states  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
top\_states = data['ship-state'].value\_counts().head(10).index  
data\_top\_states = data[data['ship-state'].isin(top\_states)]  
# Create a violin plot  
plt.figure(figsize=(12, 6))  
sns.violinplot(x='ship-state', y='Amount', data=data\_top\_states, palette='Set2')  
plt.title('Distribution of Order Amounts by State (Top 10 States)', fontsize=16)  
plt.xlabel('State', fontsize=14)  
plt.ylabel('Order Amount', fontsize=14)  
# Show the plot  
plt.xticks(rotation=45)  
plt.tight\_layout()   
plt.show()

C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\69157183.py:9: FutureWarning:   
  
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.  
  
 sns.violinplot(x='ship-state', y='Amount', data=data\_top\_states, palette='Set2')



#Visualization on the basis of ship-service-level using box plot  
import matplotlib.pyplot as plt  
import pandas as pd  
import seaborn as sns  
# Create a box plot  
plt.figure(figsize=(12, 6))  
sns.boxplot(x='ship-service-level', y='Amount', data=data, palette='Set2')  
plt.title('Distribution of Order Amounts by Shipping Service Level', fontsize=16)  
plt.xlabel('Shipping Service Level', fontsize=14)  
plt.ylabel('Order Amount', fontsize=14)  
# Show the plot  
plt.xticks(rotation=45)  
plt.grid()  
plt.tight\_layout()  
plt.show()

C:\Users\HP\AppData\Local\Temp\ipykernel\_10748\2377524825.py:7: FutureWarning:   
  
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.  
  
 sns.boxplot(x='ship-service-level', y='Amount', data=data, palette='Set2')

