KARTHIK

June 20, 2025

[2]:
```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
```

[4]:
```
# Load the dataset
df = pd.read_csv('/content/Auto Sales data.csv')
df
```

[4]:
```
      ORDERNUMBER  QUANTITYORDERED  PRICEEACH  ORDERLINENUMBER    SALES  \
0           10107               30      95.70                2  2871.00
1           10121               34      81.35                5  2765.90
2           10134               41      94.74                2  3884.34
3           10145               45      83.26                6  3746.70
4           10168               36      96.66                1  3479.76
…             …                …         …                  …       …
2742        10350               20     112.22               15  2244.40
2743        10373               29     137.19                1  3978.51
2744        10386               43     125.99                4  5417.57
2745        10397               34      62.24                1  2116.16
2746        10414               47      65.52                9  3079.44

       ORDERDATE  DAYS_SINCE_LASTORDER    STATUS  PRODUCTLINE  MSRP  \
0     24/02/2018                   828   Shipped  Motorcycles    95
1     07/05/2018                   757   Shipped  Motorcycles    95
2     01/07/2018                   703   Shipped  Motorcycles    95
3     25/08/2018                   649   Shipped  Motorcycles    95
4     28/10/2018                   586   Shipped  Motorcycles    95
…            …                      …        …            …     …
2742  02/12/2019                  2924   Shipped        Ships    54
2743  31/01/2020                  2865   Shipped        Ships    54
2744  01/03/2020                  2836  Resolved        Ships    54
2745  28/03/2020                  2810   Shipped        Ships    54
2746  06/05/2020                  2772   On Hold        Ships    54

     PRODUCTCODE            CUSTOMERNAME            PHONE  \
0       S10_1678        Land of Toys Inc.        2125557818
1       S10_1678       Reims Collectables        26.47.1555
2       S10_1678          Lyon Souveniers  +33 1 46 62 7555
3       S10_1678        Toys4GrownUps.com        6265557265
4       S10_1678      Technics Stores Inc.        6505556809
…           …                       …                 …
2742    S72_3212     Euro Shopping Channel    (91) 555 94 44
2743    S72_3212   Oulu Toy Supplies, Inc.        981-443655
2744    S72_3212     Euro Shopping Channel    (91) 555 94 44
2745    S72_3212             Alpha Cognac        61.77.6555
2746    S72_3212        Gifts4AllAges.com        6175559555

                       ADDRESSLINE1        CITY POSTALCODE  COUNTRY  \
0           897 Long Airport Avenue         NYC      10022      USA
1                59 rue de l'Abbaye       Reims      51100   France
2      27 rue du Colonel Pierre Avia       Paris      75508   France
3                78934 Hillside Dr.    Pasadena      90003      USA
4                  9408 Furth Circle  Burlingame      94217      USA
…                               …           …         …        …
2742             C/ Moralzarzal, 86      Madrid      28034    Spain
2743                    Torikatu 38        Oulu      90110  Finland
2744             C/ Moralzarzal, 86      Madrid      28034    Spain
2745          1 rue Alsace-Lorraine    Toulouse      31000   France
2746               8616 Spinnaker Dr.      Boston      51003      USA

     CONTACTLASTNAME CONTACTFIRSTNAME DEALSIZE
0                 Yu             Kwai    Small
1            Henriot             Paul    Small
2           Da Cunha           Daniel   Medium
3              Young            Julie   Medium
4             Hirano             Juri   Medium
…                …                …        …
2742          Freyre            Diego    Small
2743       Koskitalo           Pirkko   Medium
2744          Freyre            Diego   Medium
2745          Roulet          Annette    Small
2746         Yoshido             Juri   Medium

[2747 rows x 20 columns]
```

[5]:
```
# Data Cleaning
# Convert ORDERDATE to datetime
df['ORDERDATE'] = pd.to_datetime(df['ORDERDATE'], format='%d/%m/%Y')
```

[6]:
```
# Check for missing values
print("Missing Values:\n", df.isnull().sum())
```

```
Missing Values:
 ORDERNUMBER            0
QUANTITYORDERED        0
PRICEEACH              0
ORDERLINENUMBER        0
SALES                  0
ORDERDATE              0
DAYS_SINCE_LASTORDER   0
STATUS                 0
PRODUCTLINE            0
MSRP                   0
PRODUCTCODE            0
CUSTOMERNAME           0
PHONE                  0
ADDRESSLINE1           0
CITY                   0
POSTALCODE             0
COUNTRY                0
CONTACTLASTNAME        0
CONTACTFIRSTNAME       0
DEALSIZE               0
dtype: int64
```

[ ]:
```
# Basic Data Overview
print("\nDataset Info:")
print(df.info())
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2747 entries, 0 to 2746
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   ORDERNUMBER           2747 non-null   int64
 1   QUANTITYORDERED       2747 non-null   int64
 2   PRICEEACH             2747 non-null   float64
 3   ORDERLINENUMBER       2747 non-null   int64
 4   SALES                 2747 non-null   float64
 5   ORDERDATE             2747 non-null   datetime64[ns]
 6   DAYS_SINCE_LASTORDER  2747 non-null   int64
 7   STATUS                2747 non-null   object
 8   PRODUCTLINE           2747 non-null   object
 9   MSRP                  2747 non-null   int64
 10  PRODUCTCODE           2747 non-null   object
 11  CUSTOMERNAME          2747 non-null   object
 12  PHONE                 2747 non-null   object
 13  ADDRESSLINE1          2747 non-null   object
 14  CITY                  2747 non-null   object
 15  POSTALCODE          2747 non-null   object
 16  COUNTRY             2747 non-null   object
 17  CONTACTLASTNAME     2747 non-null   object
 18  CONTACTFIRSTNAME    2747 non-null   object
 19  DEALSIZE            2747 non-null   object
dtypes: datetime64[ns](1), float64(2), int64(5), object(12)
memory usage: 429.3+ KB
None
```

[ ]:
```
#Total number of orders
total_orders = len(df)
print(f"\n1. Total number of orders: {total_orders}")
```

```

1. Total number of orders: 2747
```

[ ]:
```
#Total sales value
total_sales = df['SALES'].sum()
print(f"2. Total sales value: ${total_sales:,.2f}")
```

```
2. Total sales value: $9,760,221.71
```

[ ]:
```
#Average order value
avg_order_value = df['SALES'].mean()
print(f"3. Average order value: ${avg_order_value:,.2f}")
```

```
3. Average order value: $3,553.05
```

[ ]:
```
#Number of unique customers
unique_customers = df['CUSTOMERNAME'].nunique()
print(f"4. Number of unique customers: {unique_customers}")
```

```
4. Number of unique customers: 89
```

[ ]:
```
#Number of unique products
unique_products = df['PRODUCTCODE'].nunique()
print(f"5. Number of unique products: {unique_products}")
```

```
5. Number of unique products: 109
```

[ ]:
```
#Sales by product line
sales_by_productline = df.groupby('PRODUCTLINE')['SALES'].sum().
  ↪sort_values(ascending=False)
print("\n6. Sales by Product Line:\n", sales_by_productline)
```

```

6. Sales by Product Line:
 PRODUCTLINE
Classic Cars        3842868.54
Vintage Cars        1806675.68
Trucks and Buses    1111559.19
```

```
Motorcycles         1103512.19
Planes               969323.42
Ships                700039.22
Trains               226243.47
Name: SALES, dtype: float64
```

```
[ ]: #Top 5 customers by total sales
     top_customers = df.groupby('CUSTOMERNAME')['SALES'].sum().
      ↪sort_values(ascending=False).head(5)
     print("\n7. Top 5 customers by total sales:\n", top_customers)
```

```
7. Top 5 customers by total sales:
 CUSTOMERNAME
Euro Shopping Channel           912294.11
Mini Gifts Distributors Ltd.    654858.06
Australian Collectors, Co.      200995.41
Muscle Machine Inc              197736.94
La Rochelle Gifts               180124.90
Name: SALES, dtype: float64
```

```
[ ]: #Orders by status
     orders_by_status = df['STATUS'].value_counts()
     print("\n8. Orders by Status:\n", orders_by_status)
```

```
8. Orders by Status:
 STATUS
Shipped       2541
Cancelled       60
Resolved        47
On Hold         44
In Process      41
Disputed        14
Name: count, dtype: int64
```

```
[ ]: #Sales by country
     sales_by_country = df.groupby('COUNTRY')['SALES'].sum().
      ↪sort_values(ascending=False)
     print("\n9. Sales by Country:\n", sales_by_country)
```

```
9. Sales by Country:
 COUNTRY
USA          3355575.69
Spain        1215686.92
France       1110916.52
Australia     630623.10
UK            478880.46
```

```
Italy          374674.31
Finland        329581.91
Norway         307463.70
Singapore      288488.41
Denmark        245637.15
Canada         224078.56
Germany        220472.09
Sweden         210014.21
Austria        202062.53
Japan          188167.81
Switzerland    117713.56
Belgium        108412.62
Philippines     94015.73
Ireland         57756.43
Name: SALES, dtype: float64
```

```
[ ]: #Average quantity ordered
     avg_quantity = df['QUANTITYORDERED'].mean()
     print(f"\n10. Average quantity ordered: {avg_quantity:.2f}")
```

```
10. Average quantity ordered: 35.10
```

```
[ ]: #Total sales by year
     df['YEAR'] = df['ORDERDATE'].dt.year
     sales_by_year = df.groupby('YEAR')['SALES'].sum()
     print("\n11. Total sales by year:\n", sales_by_year)
```

```
11. Total sales by year:
 YEAR
2018    3353014.06
2019    4669924.56
2020    1737283.09
Name: SALES, dtype: float64
```

```
[ ]: #Number of orders by year
     orders_by_year = df.groupby('YEAR')['ORDERNUMBER'].nunique()
     print("\n12. Number of orders by year:\n", orders_by_year)
```

```
12. Number of orders by year:
 YEAR
2018     99
2019    142
2020     57
Name: ORDERNUMBER, dtype: int64
```

```
[ ]: #Most popular product line by order count
     popular_productline = df.groupby('PRODUCTLINE')['ORDERNUMBER'].count().
       ↪sort_values(ascending=False)
     print("\n13. Most popular product line by order count:\n", popular_productline)
```

```
13. Most popular product line by order count:
 PRODUCTLINE
Classic Cars        949
Vintage Cars        579
Motorcycles         313
Planes              304
Trucks and Buses    295
Ships               230
Trains               77
Name: ORDERNUMBER, dtype: int64
```

```
[ ]: #Average price per product line
     avg_price_productline = df.groupby('PRODUCTLINE')['PRICEEACH'].mean()
     print("\n14. Average price per product line:\n", avg_price_productline)
```

```
14. Average price per product line:
 PRODUCTLINE
Classic Cars        115.195680
Motorcycles          99.767125
Planes               90.517829
Ships                88.169261
Trains               84.108701
Trucks and Buses    104.344983
Vintage Cars         90.011261
Name: PRICEEACH, dtype: float64
```

```
[ ]: #Top 5 products by sales
     top_products = df.groupby('PRODUCTCODE')['SALES'].sum().
       ↪sort_values(ascending=False).head(5)
     print("\n16. Top 5 products by sales:\n", top_products)
```

```
16. Top 5 products by sales:
 PRODUCTCODE
S18_3232    284249.02
S10_1949    179815.23
S12_1108    168585.32
S10_4698    158202.48
S18_2238    154623.95
Name: SALES, dtype: float64
```

```
[ ]: #Average days since last order
     avg_days_since_last = df['DAYS_SINCE_LASTORDER'].mean()
     print(f"\n17. Average days since last order: {avg_days_since_last:.2f}")
```

```
17. Average days since last order: 1757.09
```

```
[ ]: #Sales by city
     sales_by_city = df.groupby('CITY')['SALES'].sum().sort_values(ascending=False).
      ↪head(5)
     print("\n18. Top 5 cities by sales:\n", sales_by_city)
```

```
18. Top 5 cities by sales:
 CITY
Madrid         1082551.44
San Rafael      654858.06
NYC             560787.77
Singapore       288488.41
Paris           268944.68
Name: SALES, dtype: float64
```

```
[ ]: #Orders by month
     df['MONTH'] = df['ORDERDATE'].dt.month
     orders_by_month = df.groupby('MONTH')['ORDERNUMBER'].count()
     print("\n19. Orders by month:\n", orders_by_month)
```

```
19. Orders by month:
 MONTH
1     221
2     211
3     206
4     178
5     252
6     131
7     141
8     191
9     171
10    283
11    589
12    173
Name: ORDERNUMBER, dtype: int64
```

```
[ ]: #Sales by month
     sales_by_month = df.groupby('MONTH')['SALES'].sum()
     print("\n20. Sales by month:\n", sales_by_month)
```

```
20. Sales by month:
 MONTH
1      761985.12
2      756238.28
3      735805.81
4      669390.96
5      923972.56
6      454756.78
7      514875.97
8      659310.57
9      584724.27
10    1001377.20
11    2088536.95
12     609247.24
Name: SALES, dtype: float64
```

```
[ ]: #Most frequent contact person
     frequent_contact = df['CONTACTLASTNAME'].value_counts().head(5)
     print("\n21. Most frequent contact persons:\n", frequent_contact)
```

```
21. Most frequent contact persons:
 CONTACTLASTNAME
Freyre    259
Nelson    204
Young     115
Frick      91
Yu         80
Name: count, dtype: int64
```

```
[ ]: #Sales by MSRP
     sales_by_msrp = df.groupby('MSRP')['SALES'].sum().sort_values(ascending=False).
      ↪head(5)
     print("\n22. Top 5 MSRP values by sales:\n", sales_by_msrp)
```

```
22. Top 5 MSRP values by sales:
 MSRP
118    415755.91
99     348067.44
136    339888.27
169    284249.02
141    258075.54
Name: SALES, dtype: float64
```

```
[ ]: #Orders by deal size
     orders_by_dealsize = df['DEALSIZE'].value_counts()
     print("\n23. Orders by deal size:\n", orders_by_dealsize)
```

```
23. Orders by deal size:
 DEALSIZE
Medium    1349
Small     1246
Large      152
Name: count, dtype: int64
```

[ ]:
```
#Average sales per order by country
avg_sales_country = df.groupby('COUNTRY')['SALES'].mean().
 ↪sort_values(ascending=False)
print("\n24. Average sales per order by country:\n", avg_sales_country)
```

```
24. Average sales per order by country:
 COUNTRY
Denmark        3899.002381
Switzerland    3797.211613
Sweden         3684.459825
Austria        3673.864182
Singapore      3651.752025
Japan          3618.611731
Norway         3617.220000
Philippines    3615.989615
USA            3615.922080
Ireland        3609.776875
Finland        3582.412065
Germany        3556.001452
Spain          3554.640117
France         3537.950701
Australia      3408.773514
UK             3325.558750
Italy          3315.701858
Belgium        3285.230909
Canada         3201.122286
Name: SALES, dtype: float64
```

[ ]:
```
#Total quantity ordered by product line
quantity_by_productline = df.groupby('PRODUCTLINE')['QUANTITYORDERED'].sum()
print("\n25. Total quantity ordered by product line:\n",␣
 ↪quantity_by_productline)
```

```
25. Total quantity ordered by product line:
 PRODUCTLINE
Classic Cars        33373
Motorcycles         11080
Planes              10636
Ships                7989
```

```
Trains                 2712
Trucks and Buses     10579
Vintage Cars         20059
Name: QUANTITYORDERED, dtype: int64
```

```
[ ]: #Sales growth year-over-year
     sales_yoy = sales_by_year.pct_change() * 100
     print("\n26. Sales growth year-over-year (%):\n", sales_yoy)
```

```
26. Sales growth year-over-year (%):
 YEAR
2018          NaN
2019    39.275424
2020   -62.798476
Name: SALES, dtype: float64
```

```
[ ]: #Top 5 customers by order count
     top_customers_orders = df.groupby('CUSTOMERNAME')['ORDERNUMBER'].nunique().
      ↪sort_values(ascending=False).head(5)
     print("\n27. Top 5 customers by order count:\n", top_customers_orders)
```

```
27. Top 5 customers by order count:
 CUSTOMERNAME
Euro Shopping Channel           26
Mini Gifts Distributors Ltd.    17
Dragon Souveniers, Ltd.          5
Danish Wholesale Imports         5
Reims Collectables               5
Name: ORDERNUMBER, dtype: int64
```

```
[ ]: #Sales variance by product line
     sales_variance = df.groupby('PRODUCTLINE')['SALES'].var()
     print("\n28. Sales variance by product line:\n", sales_variance)
```

```
28. Sales variance by product line:
 PRODUCTLINE
Classic Cars        4.209902e+06
Motorcycles         3.372306e+06
Planes              2.320256e+06
Ships               1.120958e+06
Trains              2.121672e+06
Trucks and Buses    2.802463e+06
Vintage Cars        3.136670e+06
Name: SALES, dtype: float64
```

```
[ ]: #Orders with price above MSRP
     above_msrp = df[df['PRICEEACH'] > df['MSRP']].shape[0]
     print(f"\n29. Number of orders with price above MSRP: {above_msrp}")
```

```
29. Number of orders with price above MSRP: 1398
```

```
[ ]: #Average discount (PRICEEACH vs MSRP)
     df['DISCOUNT'] = df['MSRP'] - df['PRICEEACH']
     avg_discount = df['DISCOUNT'].mean()
     print(f"\n30. Average discount (MSRP - PRICEEACH): ${avg_discount:.2f}")
```

```
30. Average discount (MSRP - PRICEEACH): $-0.41
```

```
[ ]: #Sales by contact first name
     sales_by_contact = df.groupby('CONTACTFIRSTNAME')['SALES'].sum().
      ↪sort_values(ascending=False).head(5)
     print("\n31. Top 5 contact first names by sales:\n", sales_by_contact)
```

```
31. Top 5 contact first names by sales:
 CONTACTFIRSTNAME
Valarie    944892.37
Diego      912294.11
Sue        292979.86
Michael    240748.74
Maria      212054.53
Name: SALES, dtype: float64
```

```
[ ]: #Orders by postal code
     orders_by_postal = df['POSTALCODE'].value_counts().head(5)
     print("\n32. Top 5 postal codes by order count:\n", orders_by_postal)
```

```
32. Top 5 postal codes by order count:
 POSTALCODE
28034    259
97562    205
10022    152
94217     89
50553     61
Name: count, dtype: int64
```

```
[ ]: #Sales by order line number
     sales_by_orderline = df.groupby('ORDERLINENUMBER')['SALES'].sum()
     print("\n33. Sales by order line number:\n", sales_by_orderline)
```

```
33. Sales by order line number:
 ORDERLINENUMBER
1     1083272.58
2     1048766.55
3      983467.54
4      892171.95
5      829040.91
6      743256.38
7      607391.74
8      683777.11
9      578056.32
10     475612.54
11     456054.44
12     347920.48
13     317780.58
14     280908.97
15     172009.40
16     147526.35
17      91214.41
18      21993.46
Name: SALES, dtype: float64
```

[ ]:
```
#Correlation between quantity ordered and sales
correlation = df['QUANTITYORDERED'].corr(df['SALES'])
print(f"\n34. Correlation between quantity ordered and sales: {correlation:.
  ↪2f}")
```

```
34. Correlation between quantity ordered and sales: 0.55
```

[ ]:
```
#Sales distribution by deal size
sales_dist_dealsize = df.groupby('DEALSIZE')['SALES'].describe()
print("\n35. Sales distribution by deal size:\n", sales_dist_dealsize)
```

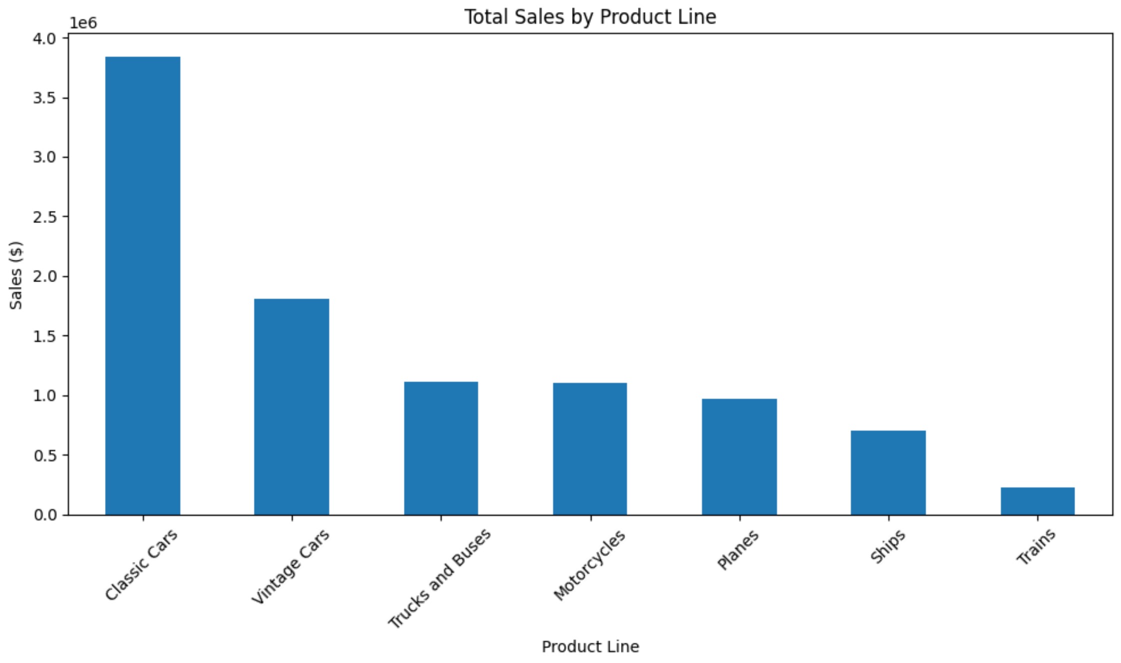
```
35. Sales distribution by deal size:
            count         mean          std      min       25%       50%  \
DEALSIZE
Large      152.0  8282.607895  1289.289903  7016.31  7324.295  7972.400
Medium    1349.0  4396.761653  1051.366317  3002.40  3510.000  4149.070
Small     1246.0  2062.627480   578.811332   482.13  1638.335  2113.975

                75%       max
DEALSIZE
Large     8777.0475  14082.80
Medium    5176.3800   6996.42
Small     2558.7450   2999.97
```

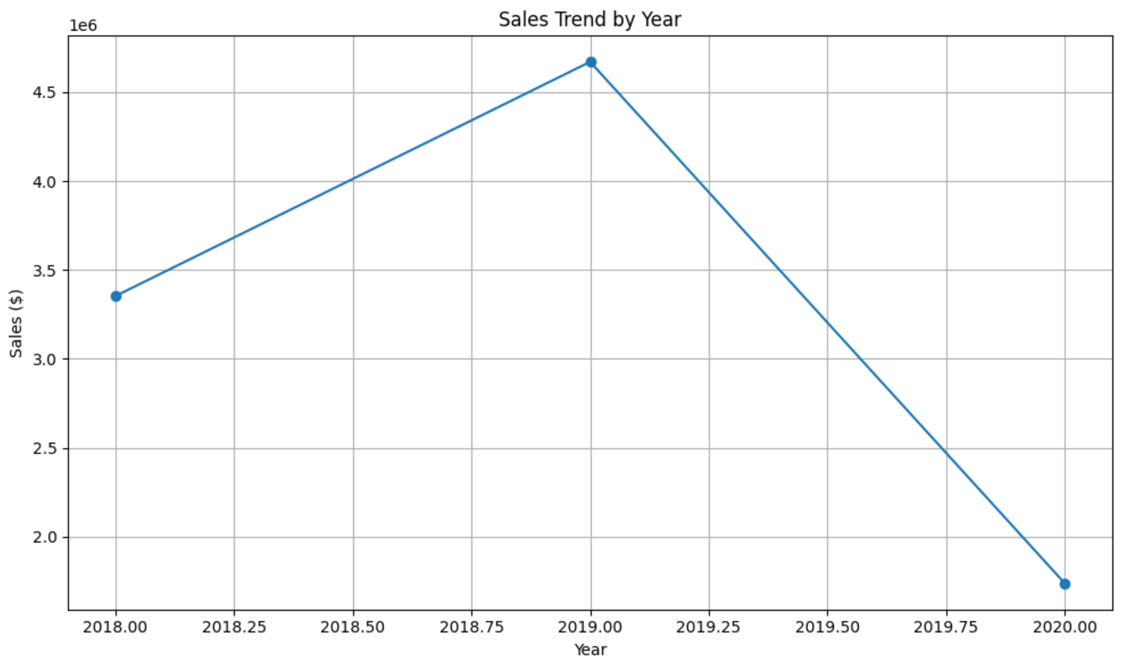
```
[ ]: #Average sales per order by product line
     avg_sales_productline = df.groupby('PRODUCTLINE')['SALES'].mean()
     print("\n36. Average sales per order by product line:\n", avg_sales_productline)
```

```
36. Average sales per order by product line:
 PRODUCTLINE
Classic Cars        4049.387292
Motorcycles         3525.598051
Planes              3188.563882
Ships               3043.648783
Trains              2938.226883
Trucks and Buses    3767.997254
Vintage Cars        3120.337962
Name: SALES, dtype: float64
```

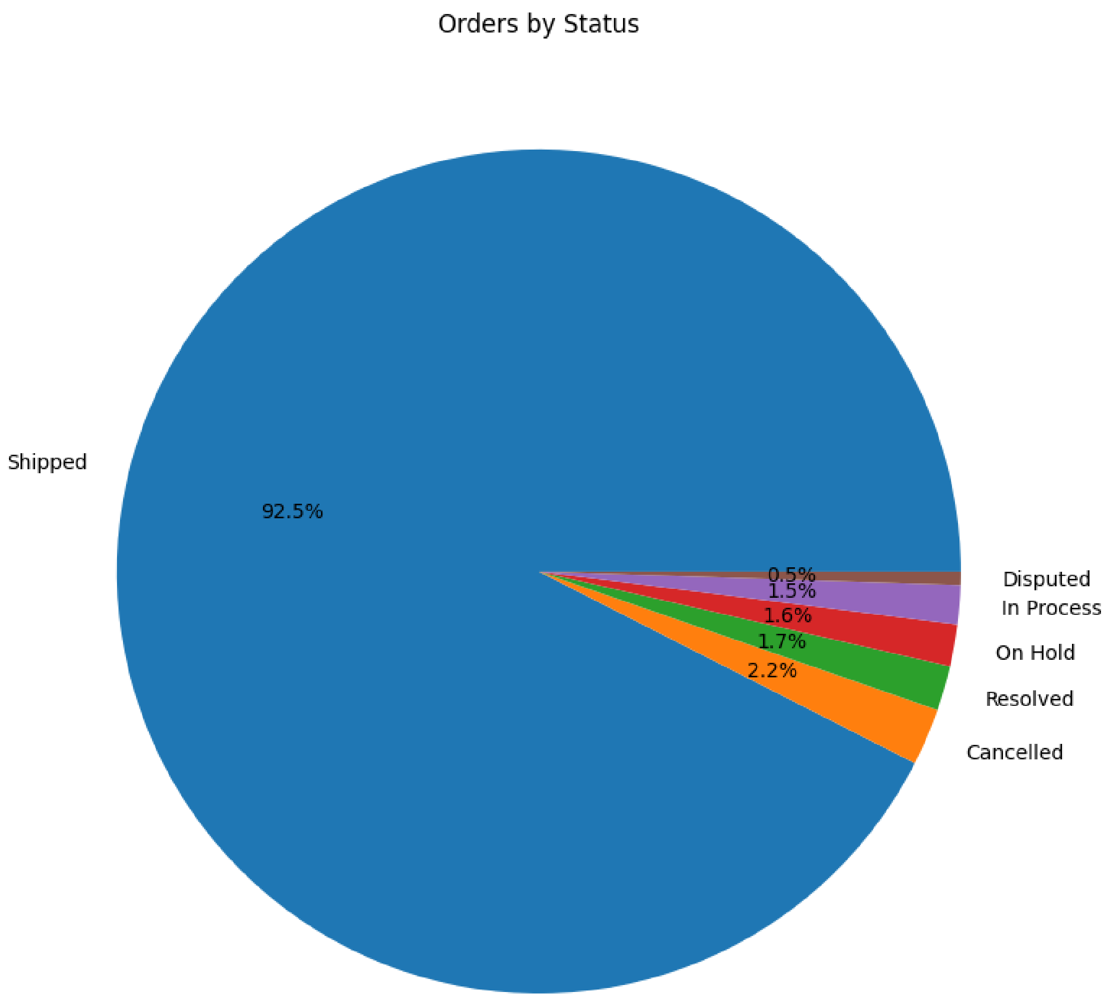
```
[ ]: # Visualizations
     # Sales by Product Line
     plt.figure(figsize=(10, 6))
     sales_by_productline.plot(kind='bar')
     plt.title('Total Sales by Product Line')
     plt.xlabel('Product Line')
     plt.ylabel('Sales ($)')
     plt.xticks(rotation=45)
     plt.tight_layout()
     plt.show()
```



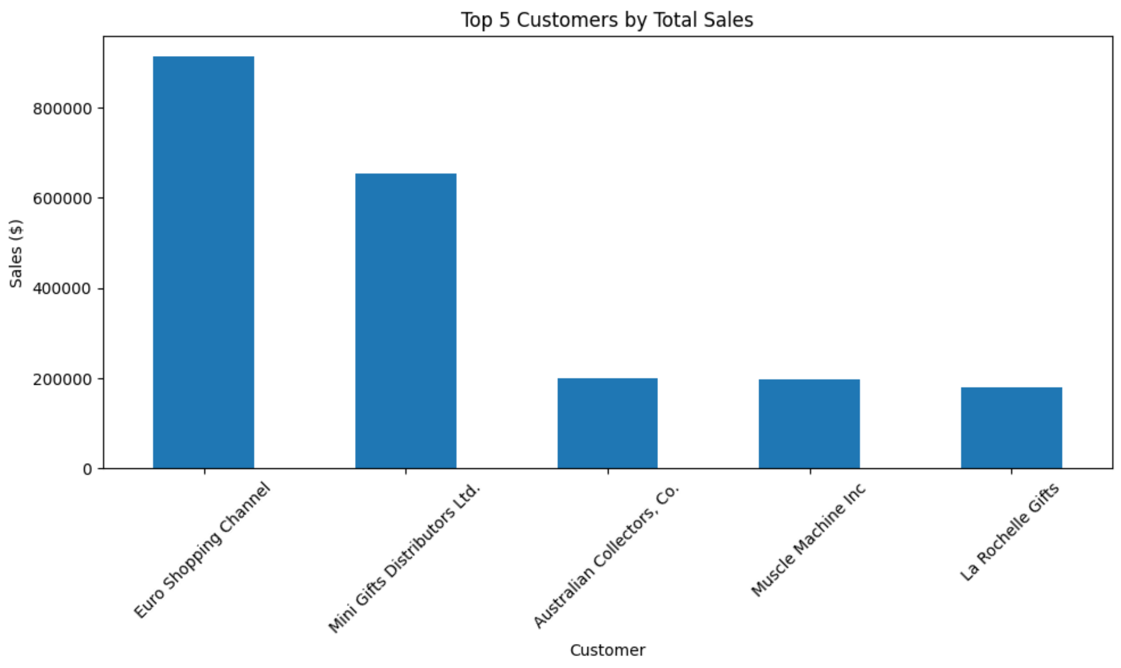
[ ]:
```python
# Sales Trend by Year
plt.figure(figsize=(10, 6))
sales_by_year.plot(kind='line', marker='o')
plt.title('Sales Trend by Year')
plt.xlabel('Year')
plt.ylabel('Sales ($)')
plt.grid(True)
plt.tight_layout()
plt.show()
```



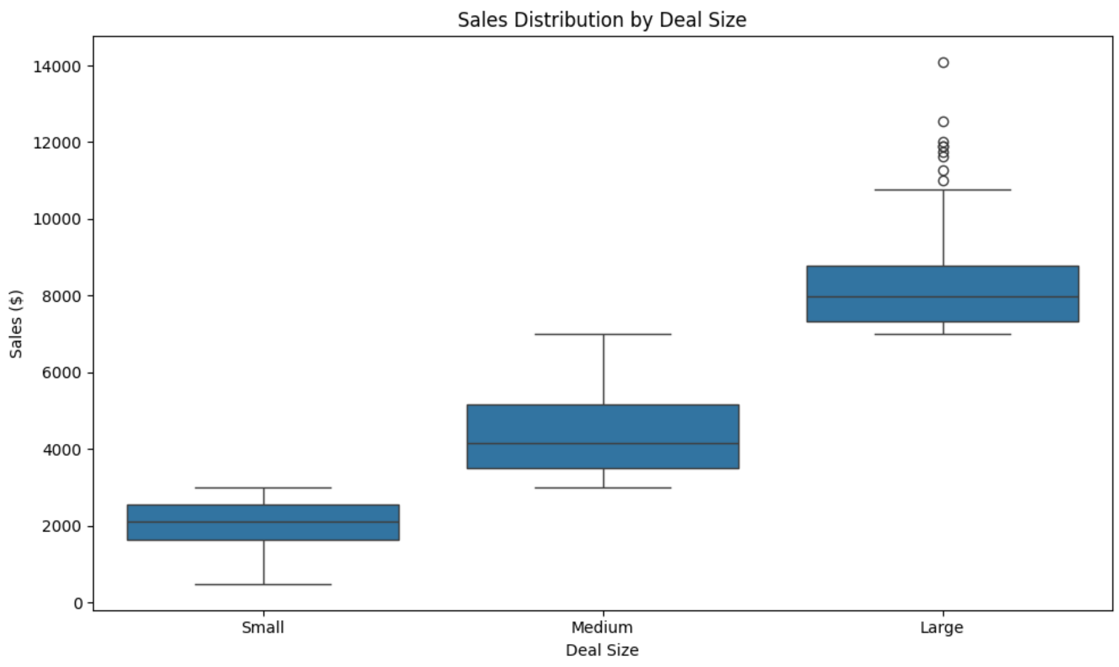
[ ]:
```python
# Orders by Status
plt.figure(figsize=(8, 8))
orders_by_status.plot(kind='pie', autopct='%1.1f%%')
plt.title('Orders by Status')
plt.ylabel('')
plt.tight_layout()
plt.show()
```



```
[ ]: # Top 5 Customers by Sales
     plt.figure(figsize=(10, 6))
     top_customers.plot(kind='bar')
     plt.title('Top 5 Customers by Total Sales')
     plt.xlabel('Customer')
     plt.ylabel('Sales ($)')
     plt.xticks(rotation=45)
     plt.tight_layout()
     plt.show()
```



[ ]:
```
# Sales Distribution by Deal Size
plt.figure(figsize=(10, 6))
sns.boxplot(x='DEALSIZE', y='SALES', data=df)
plt.title('Sales Distribution by Deal Size')
plt.xlabel('Deal Size')
plt.ylabel('Sales ($)')
plt.tight_layout()
plt.show()
```



```
[ ]: # Interesting Insight
     print("\nInteresting Insight: Classic Cars are the top-selling product line,␣
      ↪indicating strong demand among collectors or enthusiasts.")
```

```
Interesting Insight: Classic Cars are the top-selling product line, indicating
strong demand among collectors or enthusiasts.
```