In [ ]:

```
Name: Vrushali Khade
Div: A
Roll No: 44
```

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
data = pd.read_csv("C:/Users/Vrushali Khade/Downloads/sales_data_sample.csv", enco
data.head()

# While utf-8 supports all languages according to pandas' documentation, utf-8 has
```

Out[2]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORI
0	10107	30	95.70	2	2871.00	2
1	10121	34	81.35	5	2765.90	
2	10134	41	94.74	2	3884.34	
3	10145	45	83.26	6	3746.70	8
4	10159	49	100.00	14	5205.27	10

5 rows × 25 columns

In [3]:

```
data.shape
```

Out[3]:

```
(2823, 25)
```

In [4]:

```
# Number of NAN values per column in the dataset
data.isnull().sum
```

Out[4]:

```
<bound method DataFrame.sum of       ORDERNUMBER  QUANTITYORDERED  PRICEEACH  ORD
ERLINENUMBER  SALES  \
0           False            False      False            False  False
1           False            False      False            False  False
2           False            False      False            False  False
3           False            False      False            False  False
4           False            False      False            False  False
...           ...              ...        ...              ...    ...
2818        False            False      False            False  False
2819        False            False      False            False  False
2820        False            False      False            False  False
2821        False            False      False            False  False
2822        False            False      False            False  False

      ORDERDATE  STATUS  QTR_ID  MONTH_ID  YEAR_ID  ...  ADDRESSLINE1  \
0         False   False   False     False    False  ...         False
1         False   False   False     False    False  ...         False
2         False   False   False     False    False  ...         False
3         False   False   False     False    False  ...         False
4         False   False   False     False    False  ...         False
...         ...     ...     ...       ...      ...  ...           ...
2818      False   False   False     False    False  ...         False
2819      False   False   False     False    False  ...         False
2820      False   False   False     False    False  ...         False
2821      False   False   False     False    False  ...         False
2822      False   False   False     False    False  ...         False

      ADDRESSLINE2   CITY  STATE  POSTALCODE  COUNTRY  TERRITORY  \
0             True  False  False       False    False       True
1             True  False   True       False    False      False
2             True  False   True       False    False      False
3             True  False  False       False    False       True
4             True  False  False        True    False       True
...            ...    ...    ...         ...      ...        ...
2818          True  False   True       False    False      False
2819          True  False   True       False    False      False
2820          True  False   True       False    False      False
2821          True  False   True       False    False      False
2822          True  False  False       False    False       True

      CONTACTLASTNAME  CONTACTFIRSTNAME  DEALSIZE
0               False             False     False
1               False             False     False
2               False             False     False
3               False             False     False
4               False             False     False
...               ...               ...       ...
2818            False             False     False
2819            False             False     False
2820            False             False     False
2821            False             False     False
2822            False             False     False

[2823 rows x 25 columns]>
```

In [5]:

```
data.drop(["ORDERNUMBER", "PRICEEACH", "ORDERDATE", "PHONE", "ADDRESSLINE1", "ADDR
```

In [6]:

```
data.head()
```

Out[6]:

	QUANTITYORDERED	ORDERLINENUMBER	SALES	STATUS	QTR_ID	MONTH_ID	YE
0	30	2	2871.00	Shipped	1	2	
1	34	5	2765.90	Shipped	2	5	
2	41	2	3884.34	Shipped	3	7	
3	45	6	3746.70	Shipped	3	8	
4	49	14	5205.27	Shipped	4	10	

In [7]:

```
data.isnull().sum()
```

Out[7]:

```
QUANTITYORDERED    0
ORDERLINENUMBER    0
SALES              0
STATUS             0
QTR_ID             0
MONTH_ID           0
YEAR_ID            0
PRODUCTLINE        0
MSRP               0
PRODUCTCODE        0
CUSTOMERNAME       0
COUNTRY            0
DEALSIZE           0
dtype: int64
```

In [8]:

```
data.describe()
```

Out[8]:

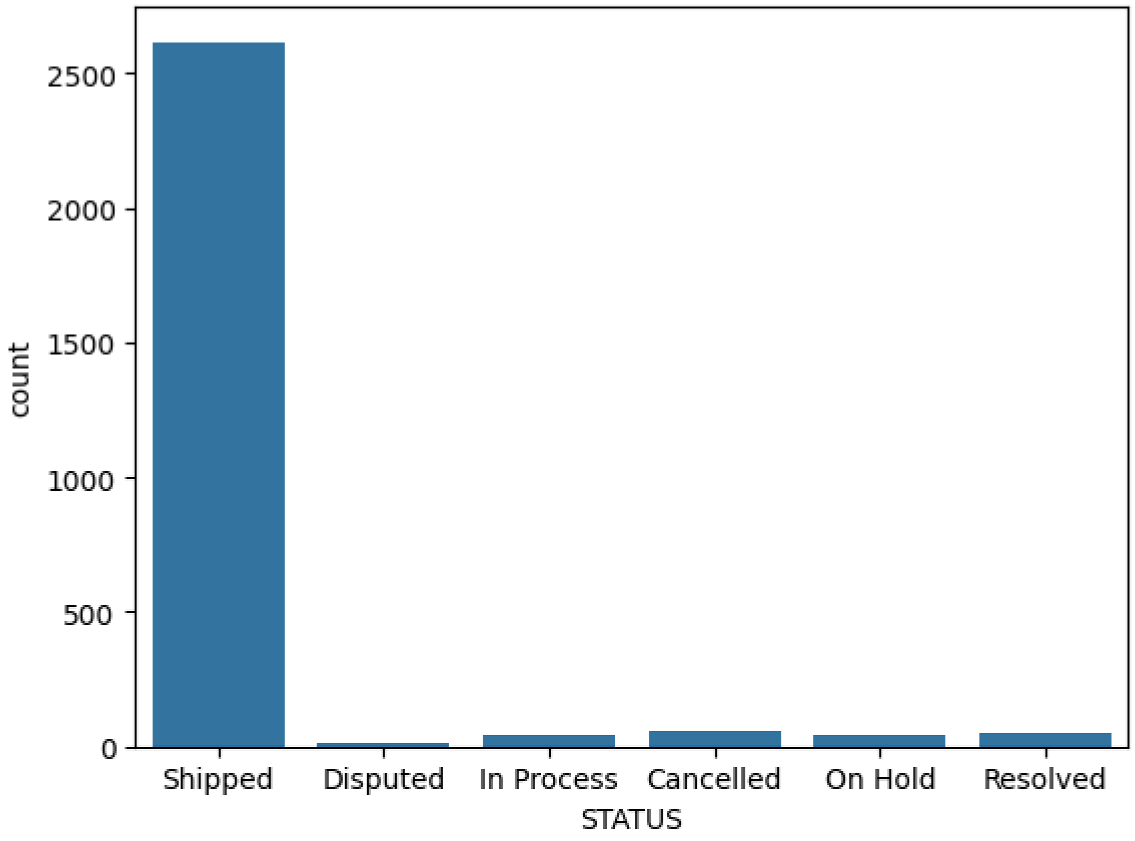
	QUANTITYORDERED	ORDERLINENUMBER	SALES	QTR_ID	MONTH_II
count	2823.000000	2823.000000	2823.000000	2823.000000	2823.00000
mean	35.092809	6.466171	3553.889072	2.717676	7.09245
std	9.741443	4.225841	1841.865106	1.203878	3.65663
min	6.000000	1.000000	482.130000	1.000000	1.00000
25%	27.000000	3.000000	2203.430000	2.000000	4.00000
50%	35.000000	6.000000	3184.800000	3.000000	8.00000
75%	43.000000	9.000000	4508.000000	4.000000	11.00000
max	97.000000	18.000000	14082.800000	4.000000	12.00000

In [9]:

```
sns.countplot(data = data , x = 'STATUS')
```

Out[9]:

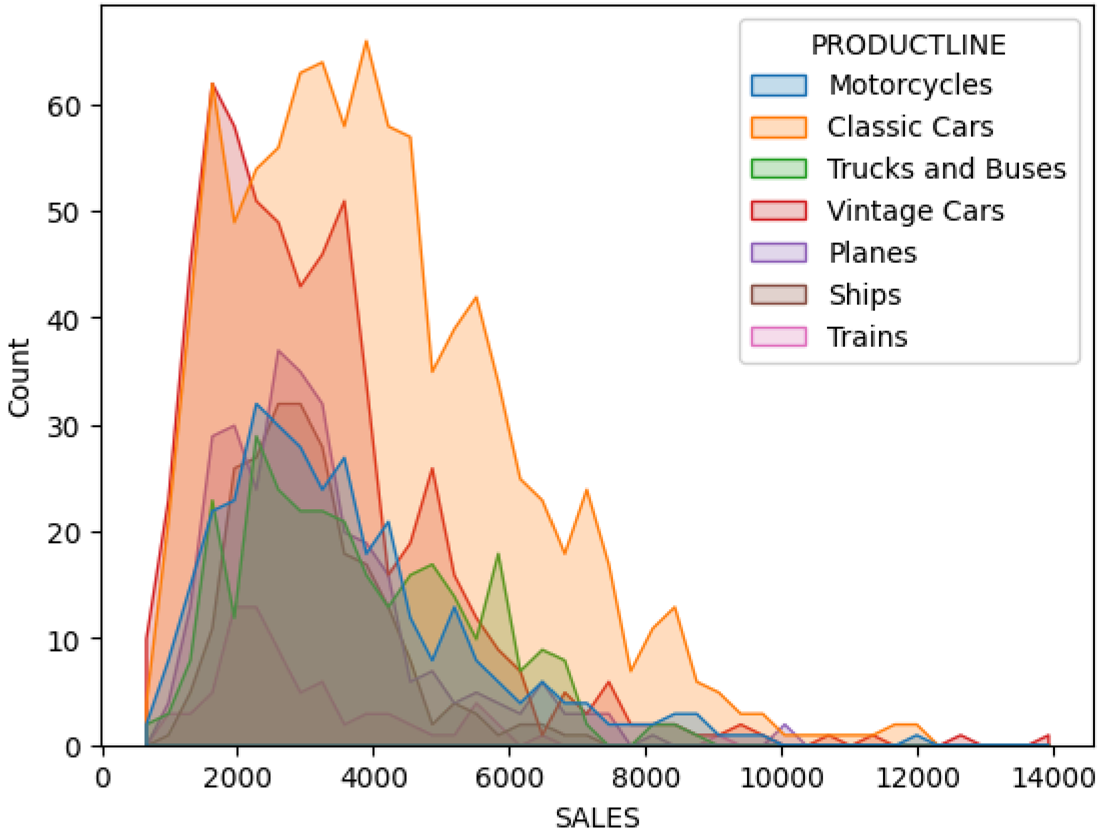
```
<Axes: xlabel='STATUS', ylabel='count'>
```



```
In [10]: import seaborn as sns
```

```
In [11]: sns.histplot(x = 'SALES' , hue = 'PRODUCTLINE', data = data, element="poly")
```

```
Out[11]: <Axes: xlabel='SALES', ylabel='Count'>
```



```
In [12]: data['PRODUCTLINE'].unique()
```

```
Out[12]: array(['Motorcycles', 'Classic Cars', 'Trucks and Buses', 'Vintage Cars',
                'Planes', 'Ships', 'Trains'], dtype=object)
```

```
In [13]: #checking the duplicated values
         data.drop_duplicates(inplace=True)
```

```
In [14]: data.info()
```

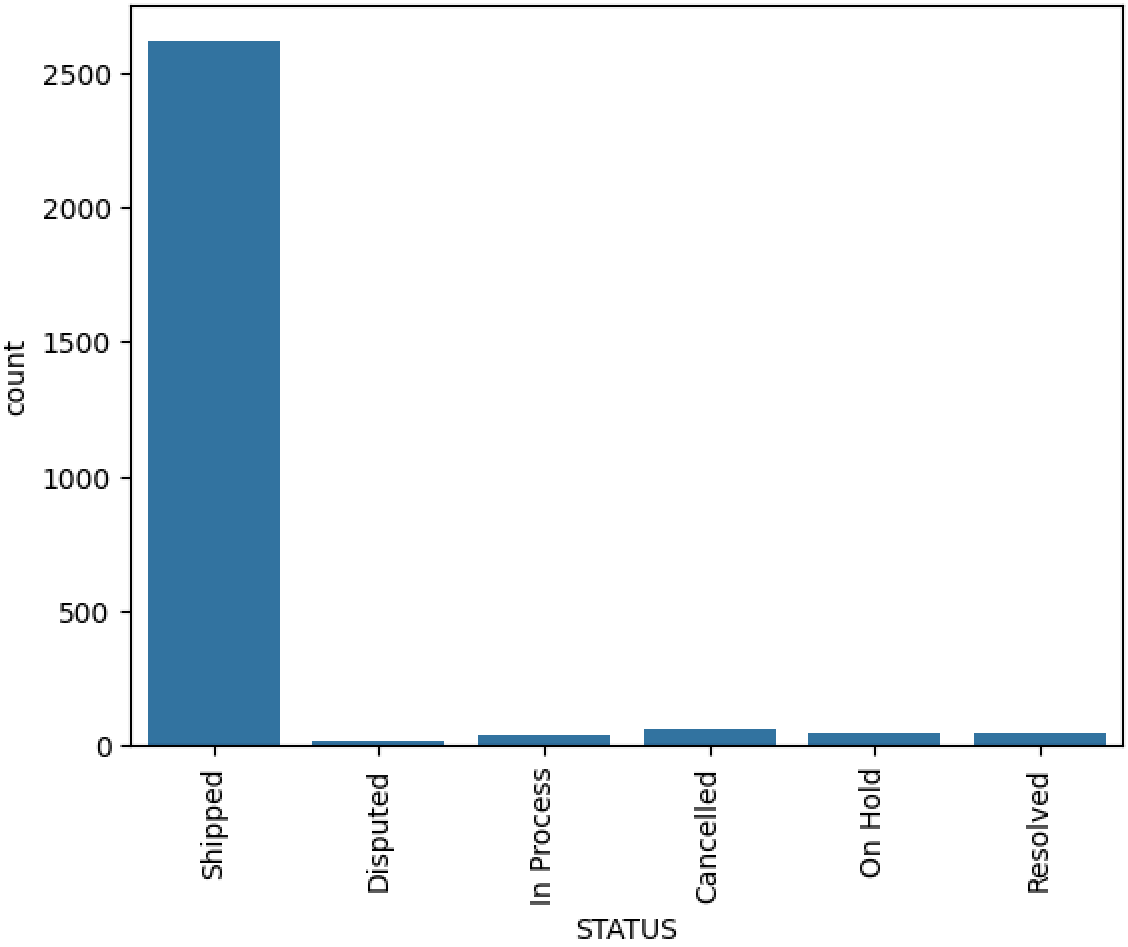
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   QUANTITYORDERED  2823 non-null   int64  
 1   ORDERLINENUMBER  2823 non-null   int64  
 2   SALES            2823 non-null   float64
 3   STATUS           2823 non-null   object 
 4   QTR_ID           2823 non-null   int64  
 5   MONTH_ID         2823 non-null   int64  
 6   YEAR_ID          2823 non-null   int64  
 7   PRODUCTLINE      2823 non-null   object 
 8   MSRP             2823 non-null   int64  
 9   PRODUCTCODE      2823 non-null   object 
 10  CUSTOMERNAME     2823 non-null   object 
 11  COUNTRY          2823 non-null   object 
 12  DEALSIZE         2823 non-null   object 
dtypes: float64(1), int64(6), object(6)
memory usage: 286.8+ KB
```

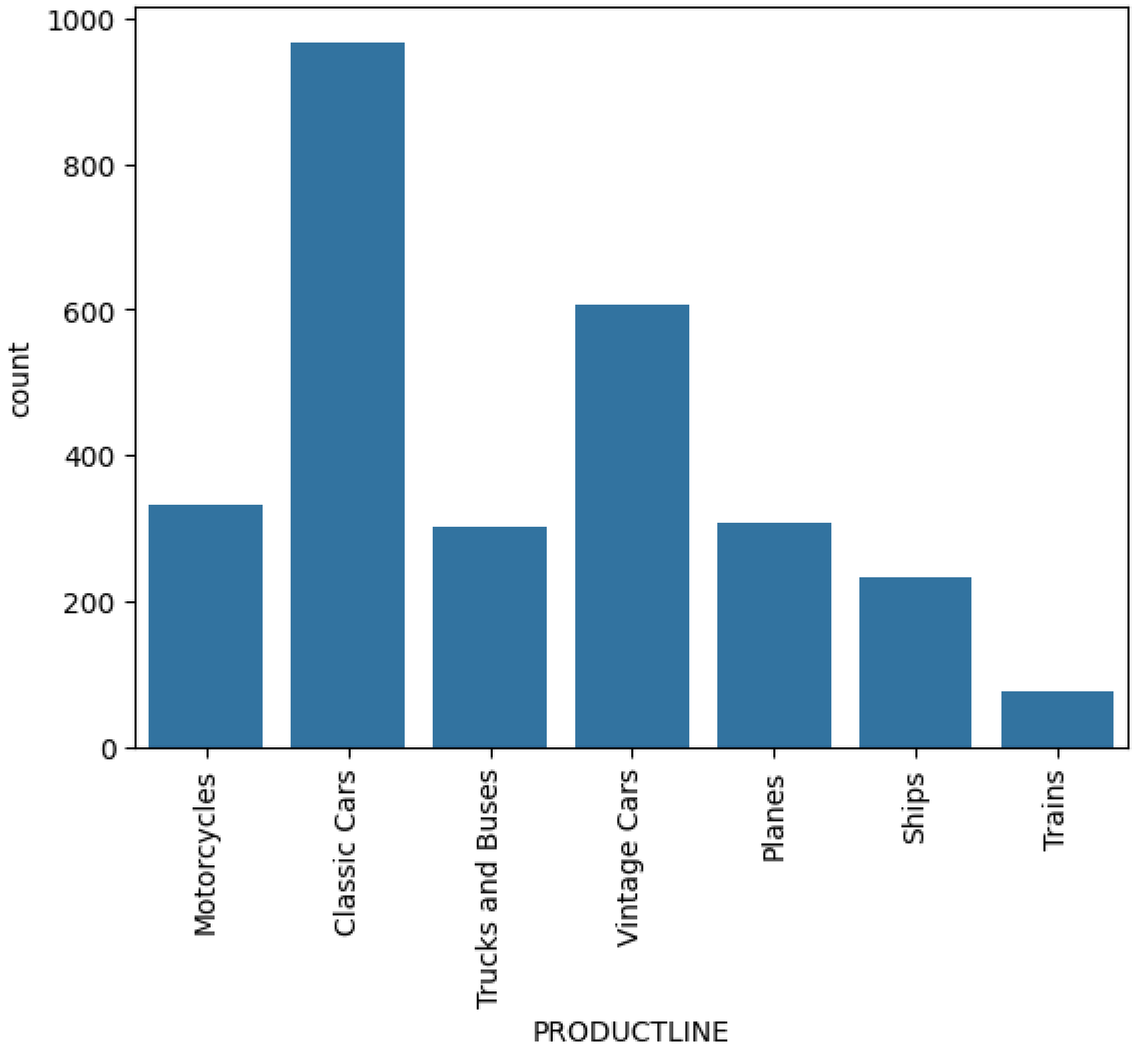
```
In [15]: list_cat = data.select_dtypes(include=['object']).columns.tolist()
```

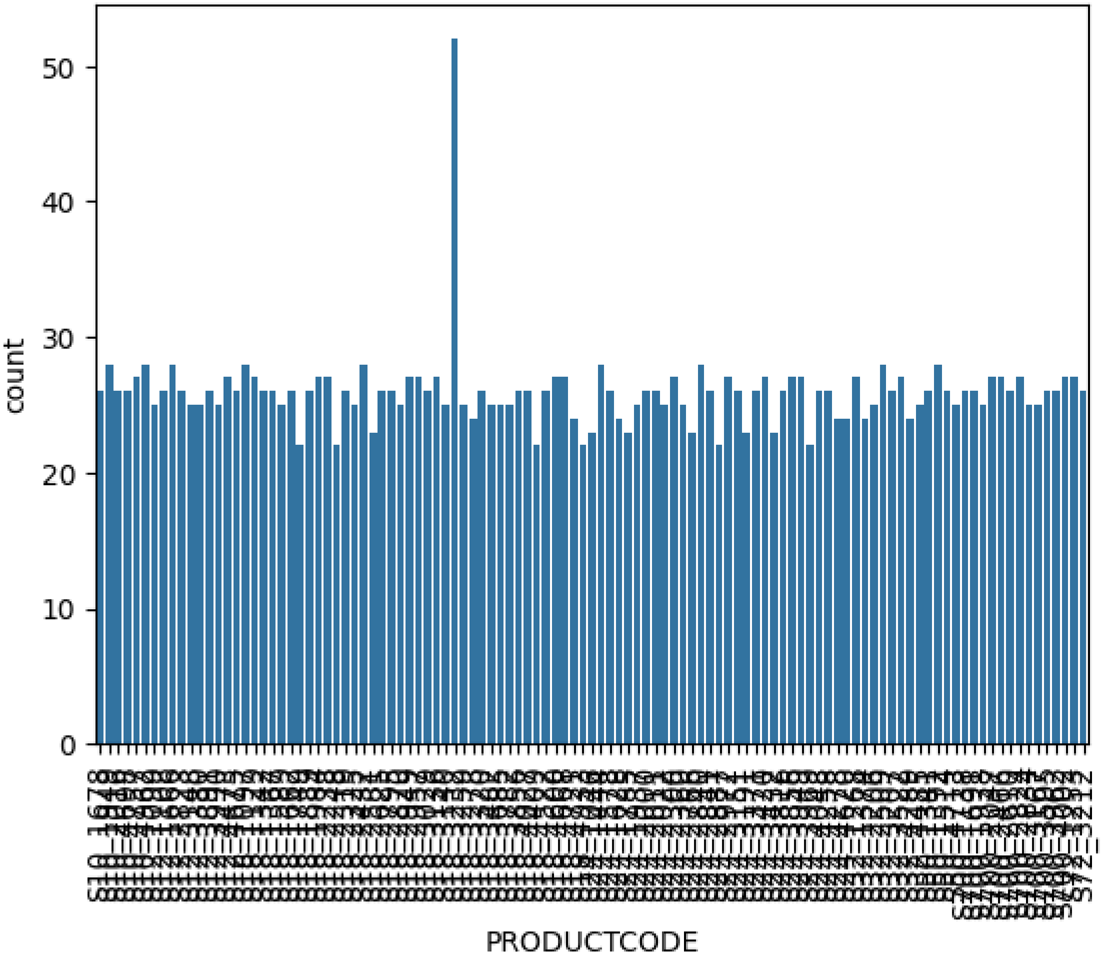
```
In [16]: list_cat
```

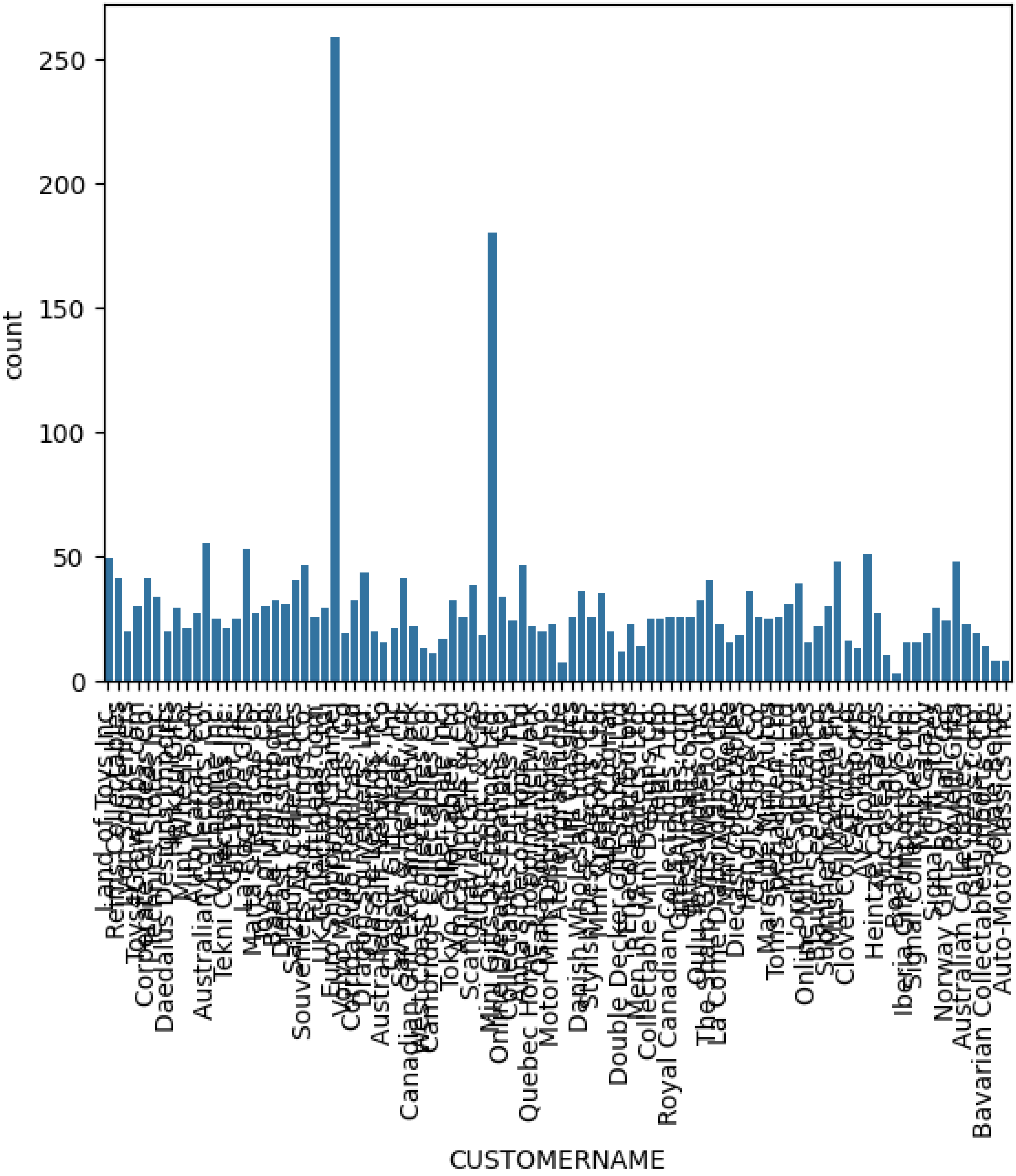
```
Out[16]: ['STATUS', 'PRODUCTLINE', 'PRODUCTCODE', 'CUSTOMERNAME', 'COUNTRY', 'DEALSIZE']
```

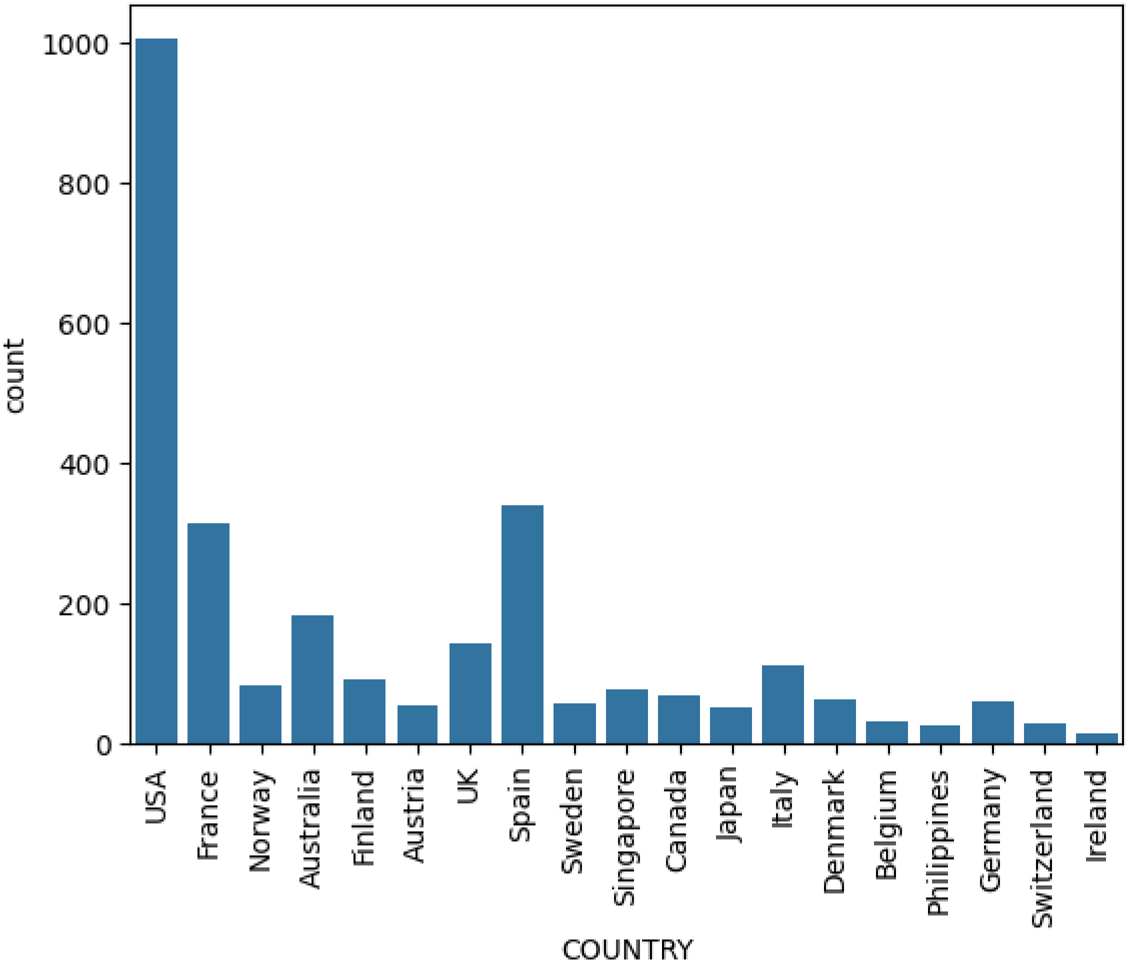
```
In [17]: for i in list_cat:
           sns.countplot(data = data ,x = i)
           plt.xticks(rotation = 90)
           plt.show()
```

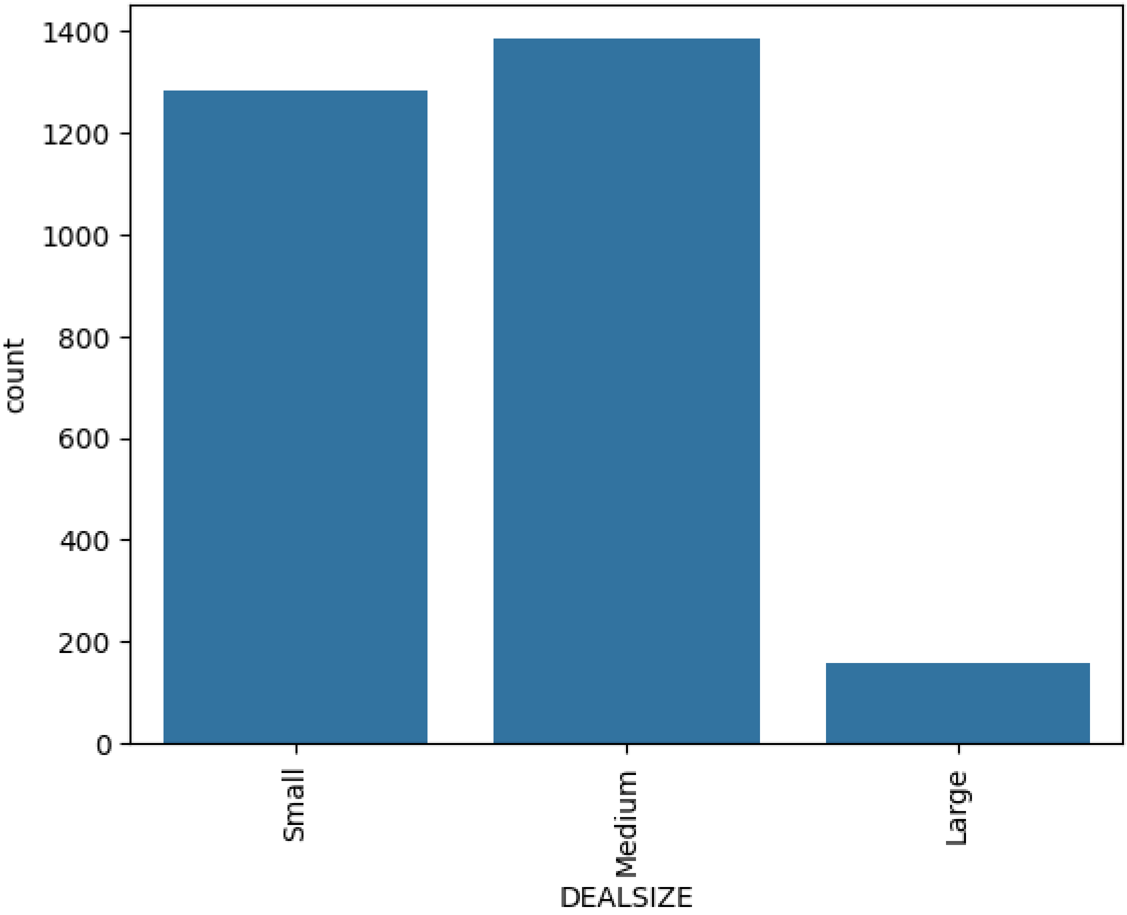

2500
2000
1500
1000
500
0
count
Shipped
Disputed
In Process
Cancelled
On Hold
Resolved
STATUS


1000
800
600
400
200
0
count
Motorcycles
Classic Cars
Trucks and Buses
Vintage Cars
Planes
Ships
Trains
PRODUCTLINE


50
40
30
20
10
0
count
S10_1678
PRODUCTCODE


count
250
200
150
100
50
0
Land of Toys Inc.
Reims Collectables
Daedalus Designs Imports
Australian Collectors, Co.
Tekni Collectables Inc.
Souveniers And Things Co.
Euro Shopping Channel
Canadian Gift Exchange Network
Cambridge Collectables Co.
Online Diecast Creations Co.
Quebec Home Shopping Network
Danish Wholesale Imports
Double Decker Gift Stores, Ltd
Collectable Mini Designs Co.
Royal Canadian Collectables, Ltd.
La Corne D'abondance, Co.
Toms Spezialitäten, Ltd
Clover Collections, Co.
Iberia Gift Imports, Corp.
Signal Collectibles Ltd.
Norway Gifts By Mail, Co.
Australian Collectables, Ltd
Bavarian Collectables Imports, Co.
Auto-Moto Classics Inc.
CUSTOMERNAME


count
0
200
400
600
800
1000
USA
France
Norway
Australia
Finland
Austria
UK
Spain
Sweden
Singapore
Canada
Japan
Italy
Denmark
Belgium
Philippines
Germany
Switzerland
Ireland
COUNTRY


count
0
200
400
600
800
1000
1200
1400
Small
Medium
Large
DEALSIZE

In [18]:
```
#dealing with the catagorical features
from sklearn import preprocessing
le = preprocessing.LabelEncoder()

# Encode labels in column 'species'.
for i in list_cat:
  data[i]= le.fit_transform(data[i])
```

In [19]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   QUANTITYORDERED  2823 non-null   int64  
 1   ORDERLINENUMBER  2823 non-null   int64  
 2   SALES            2823 non-null   float64
 3   STATUS           2823 non-null   int32  
 4   QTR_ID           2823 non-null   int64  
 5   MONTH_ID         2823 non-null   int64  
 6   YEAR_ID          2823 non-null   int64  
 7   PRODUCTLINE      2823 non-null   int32  
 8   MSRP             2823 non-null   int64  
 9   PRODUCTCODE      2823 non-null   int32  
 10  CUSTOMERNAME     2823 non-null   int32  
 11  COUNTRY          2823 non-null   int32  
 12  DEALSIZE         2823 non-null   int32  
dtypes: float64(1), int32(6), int64(6)
memory usage: 220.7 KB
```

In [20]:
```
data['SALES'] = data['SALES'].astype(int)
```

In [21]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   QUANTITYORDERED  2823 non-null   int64
 1   ORDERLINENUMBER  2823 non-null   int64
 2   SALES            2823 non-null   int32
 3   STATUS           2823 non-null   int32
 4   QTR_ID           2823 non-null   int64
 5   MONTH_ID         2823 non-null   int64
 6   YEAR_ID          2823 non-null   int64
 7   PRODUCTLINE      2823 non-null   int32
 8   MSRP             2823 non-null   int64
 9   PRODUCTCODE      2823 non-null   int32
 10  CUSTOMERNAME     2823 non-null   int32
 11  COUNTRY          2823 non-null   int32
 12  DEALSIZE         2823 non-null   int32
dtypes: int32(7), int64(6)
memory usage: 209.6 KB
```

In [22]:
```
data.describe()
```

Out[22]:

	QUANTITYORDERED	ORDERLINENUMBER	SALES	STATUS	QTR_I
count	2823.000000	2823.000000	2823.000000	2823.000000	2823.00000
mean	35.092809	6.466171	3553.421537	4.782501	2.71767
std	9.741443	4.225841	1841.865754	0.879416	1.20387
min	6.000000	1.000000	482.000000	0.000000	1.00000
25%	27.000000	3.000000	2203.000000	5.000000	2.00000
50%	35.000000	6.000000	3184.000000	5.000000	3.00000
75%	43.000000	9.000000	4508.000000	5.000000	4.00000
max	97.000000	18.000000	14082.000000	5.000000	4.00000

In [23]:

```
## taget feature are Sales and productline
X = data[['SALES','PRODUCTCODE']]
```

In [24]:

```
data.columns
```

Out[24]:

```
Index(['QUANTITYORDERED', 'ORDERLINENUMBER', 'SALES', 'STATUS', 'QTR_ID',
       'MONTH_ID', 'YEAR_ID', 'PRODUCTLINE', 'MSRP', 'PRODUCTCODE',
       'CUSTOMERNAME', 'COUNTRY', 'DEALSIZE'],
      dtype='object')
```

In [25]:

```
## K Means implementation
```

In [26]:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4, init='k-means++', random_state=0).fit(X)
```

In [27]:

```
kmeans.labels_
```

Out[27]:

```
array([2, 2, 2, ..., 3, 0, 2])
```

In [28]:

```
kmeans.inertia_
```

Out[28]:

```
1043164092.8545704
```

In [29]:

```
kmeans.n_iter_
```

Out[29]:

```
15
```

In [30]:

```
kmeans.cluster_centers_
```

Out[30]:

```
array([[1913.93425926,   63.19907407],
       [8023.78238342,   28.35751295],
       [3489.45517241,   55.50640394],
       [5371.72523364,   40.62616822]])
```

In [31]:

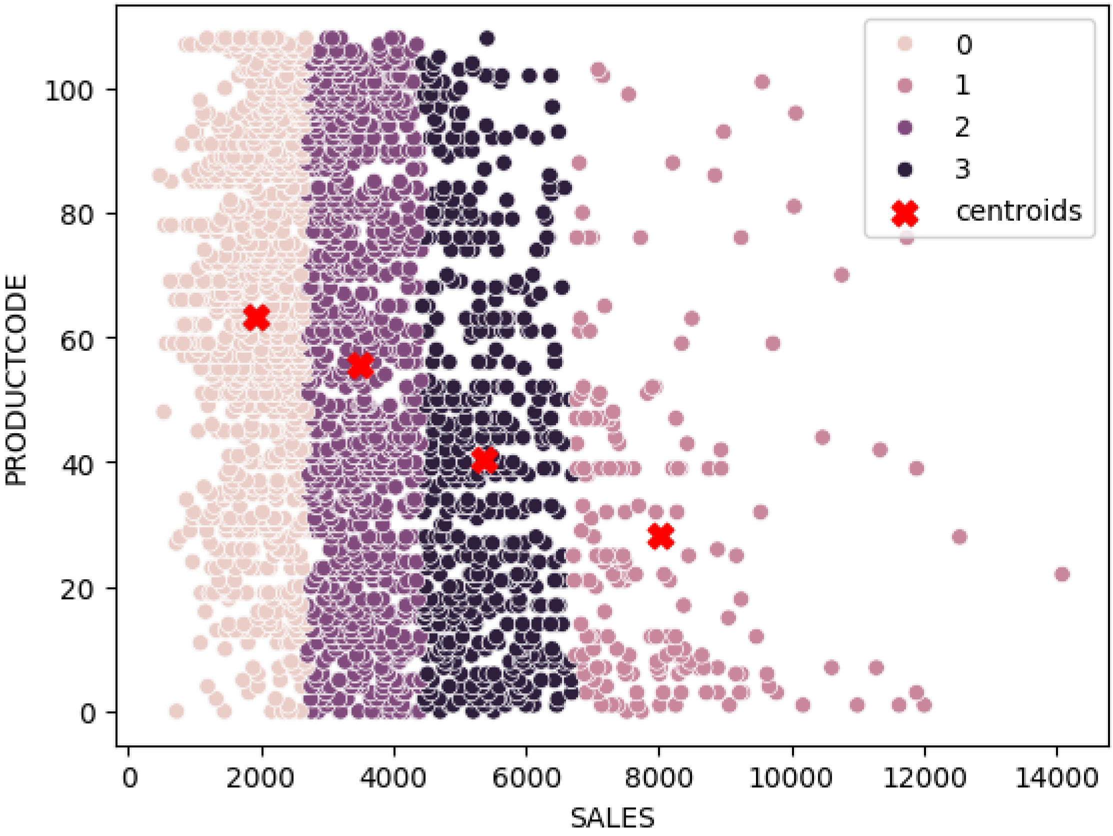
```
#getting the size of the clusters
from collections import Counter
Counter(kmeans.labels_)
```

Out[31]:

```
Counter({0: 1078, 2: 1015, 3: 537, 1: 193})
```

In [32]:

```
sns.scatterplot(data=X, x="SALES", y="PRODUCTCODE", hue=kmeans.labels_)
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1],
            marker="X", c="r", s=80, label="centroids")
plt.legend()
plt.show()
```


PRODUCTCODE
SALES
centroids