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

```
dataset = pd.read_excel("/content/Superstore_USA.xlsx")
```

```
dataset.head(5)
```

	Row ID	Order Priority	Discount	Unit Price	Shipping Cost	Customer ID	Customer Name	Ship Mode	Cus Se
0	18606	Not Specified	0.01	2.88	0.50	2	Janice Fletcher	Regular Air	Co
1	20847	High	0.01	2.84	0.93	3	Bonnie Potter	Express Air	Co
2	23086	Not Specified	0.03	6.68	6.15	3	Bonnie Potter	Express Air	Co
3	23087	Not Specified	0.01	5.68	3.60	3	Bonnie Potter	Regular Air	Co
4	23088	Not Specified	0.00	205.99	2.50	3	Bonnie Potter	Express Air	Co

5 rows × 24 columns

```
dataset.shape  # WILL DISPLAY ROW,COLUMN
```

```
(9426, 24)
```

```
# 1  MISSING VALUE ANALYSIS
dataset.isnull().sum()  # it will identify all missiing values in our data
```

```
                          0
Row ID                    0
Order Priority            0
Discount                  0
Unit Price                0
Shipping Cost             0
Customer ID               0
Customer Name             0
Ship Mode                 0
Customer Segment          0
Product Category          0
Product Sub-Category      0
Product Container         0
Product Name              0
Product Base Margin      72
Region                    0
State or Province         0
City                      0
Postal Code               0
Order Date                0
Ship Date                 0
Profit                    0
Quantity ordered new      0
Sales                     0
Order ID                  0

dtype: int64
```

```
# FILLING OF MISSING VALLUE IN PRODUCT BASE MARGIN
dataset['Product Base Margin'].fillna(dataset['Product Base Margin'].mean(),inp
# 1  MISSING VALUE ANALYSIS
dataset.isnull().sum()  # it will identify all missiing values in our data
```

	0
Row ID	0
Order Priority	0
Discount	0
Unit Price	0
Shipping Cost	0
Customer ID	0
Customer Name	0
Ship Mode	0
Customer Segment	0
Product Category	0
Product Sub-Category	0
Product Container	0
Product Name	0
Product Base Margin	0
Region	0
State or Province	0
City	0
Postal Code	0
Order Date	0
Ship Date	0
Profit	0
Quantity ordered new	0
Sales	0
Order ID	0

dtype: int64

```
#.  ORDER PRIORITY ANALYSIS
```

```
# Now analysis part
# we cannot analyse using row id so moving to order priority
dataset['Order Priority'].value_counts()
```

```
                  count
Order Priority
          High     1970
           Low     1926
 Not Specified     1881
        Medium     1844
      Critical     1804
      Critical        1

dtype: int64
```

```
dataset['Order Priority'].unique()
```

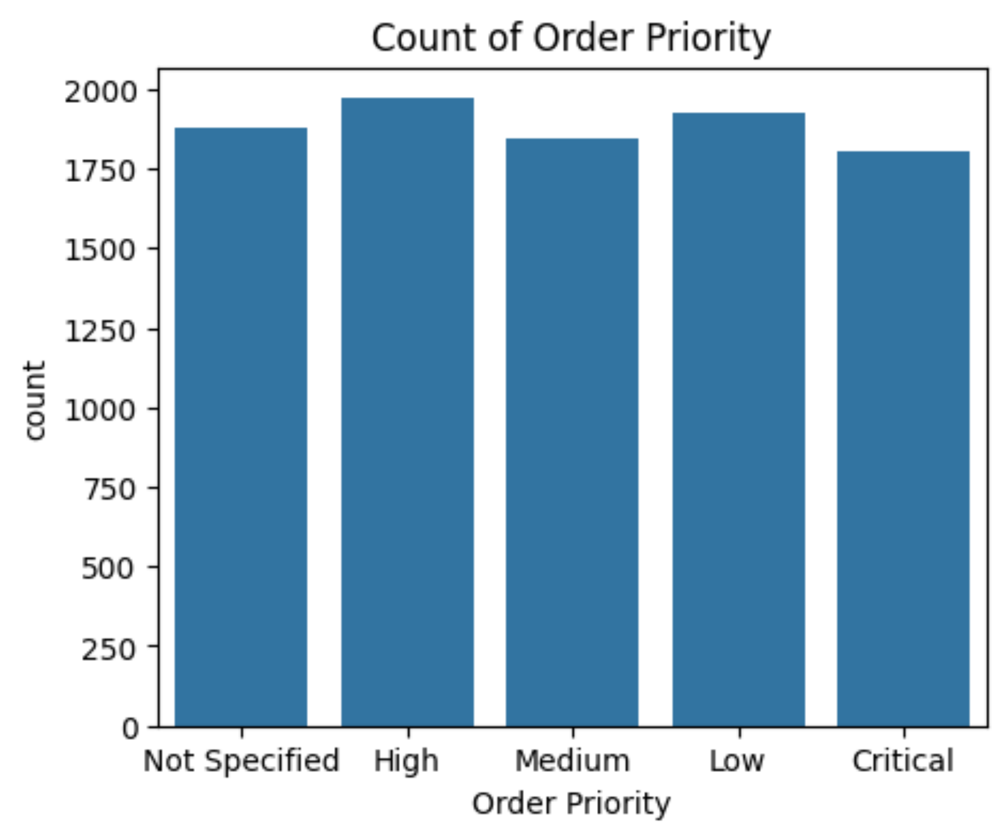
```
array(['Not Specified', 'High', 'Medium', 'Low', 'Critical', 'Critical '],
      dtype=object)
```

```
dataset["Order Priority"] = dataset["Order Priority"].replace("Critical ","Crit
dataset['Order Priority'].value_counts()
```

```
                    count
Order Priority
         High        1970
          Low        1926
Not Specified        1881
       Medium        1844
     Critical        1805

dtype: int64
```

```
plt.figure(figsize=(5,4))
plt.title("Count of Order Priority")
sns.countplot(x="Order Priority",data= dataset)
plt.show()
plt.savefig("Count of Order Priority.jpeg")
```



```
<Figure size 640x480 with 0 Axes>
```

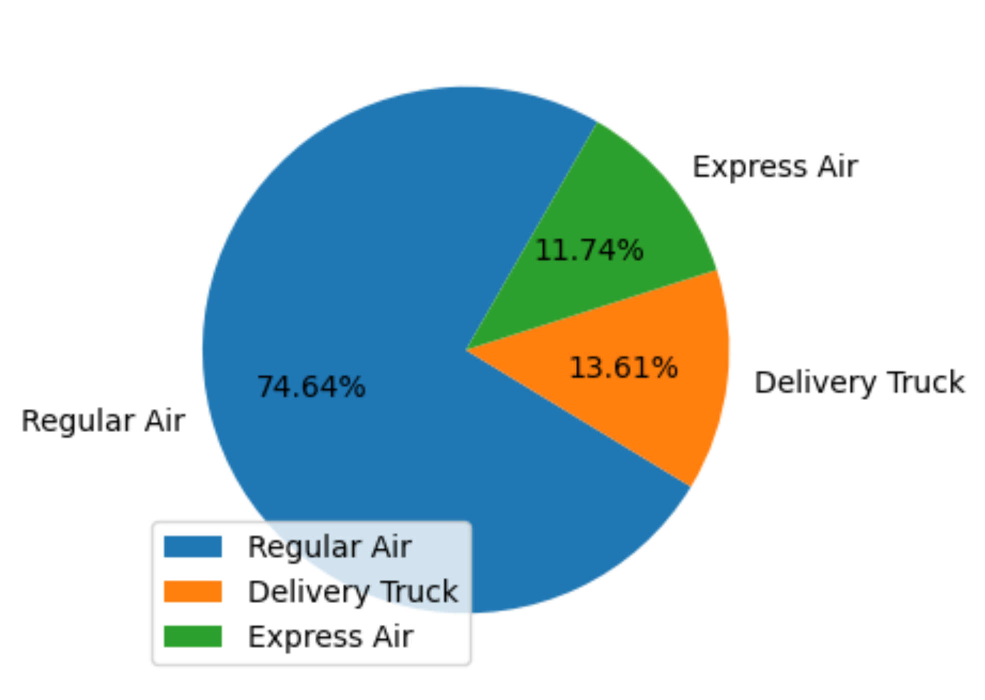
```
##. SHIPPING MODE
```

```
dataset['Ship Mode'].value_counts()
```

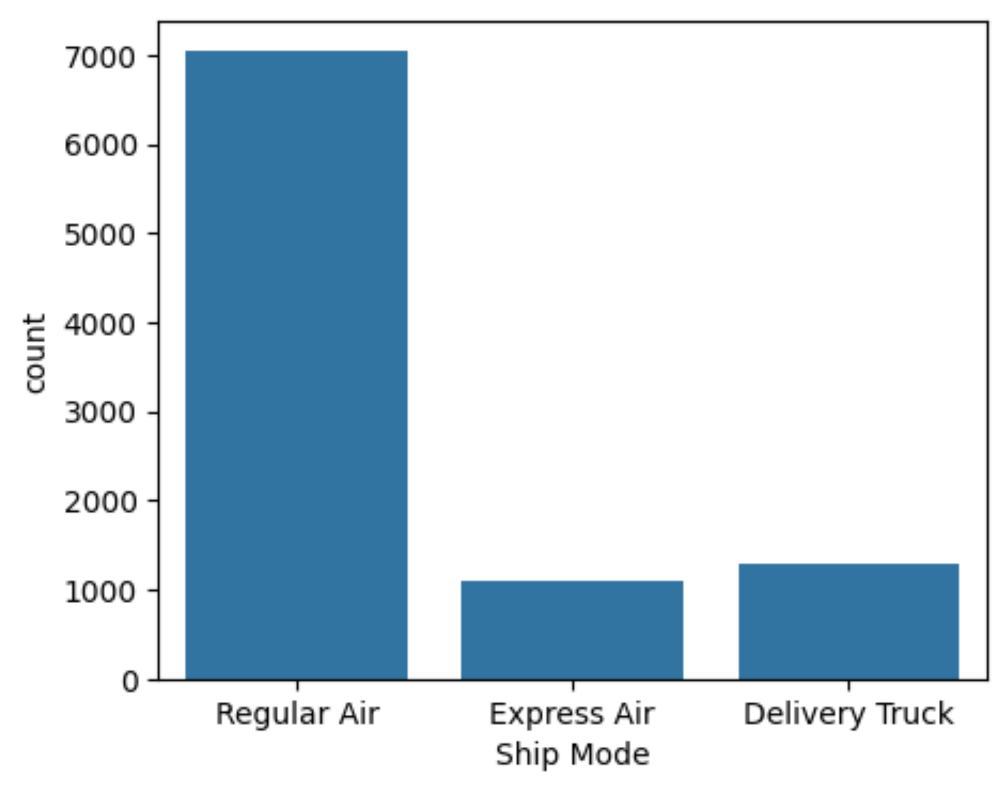
```
                 count
Ship Mode
Regular Air       7036
Delivery Truck    1283
Express Air       1107

dtype: int64
```

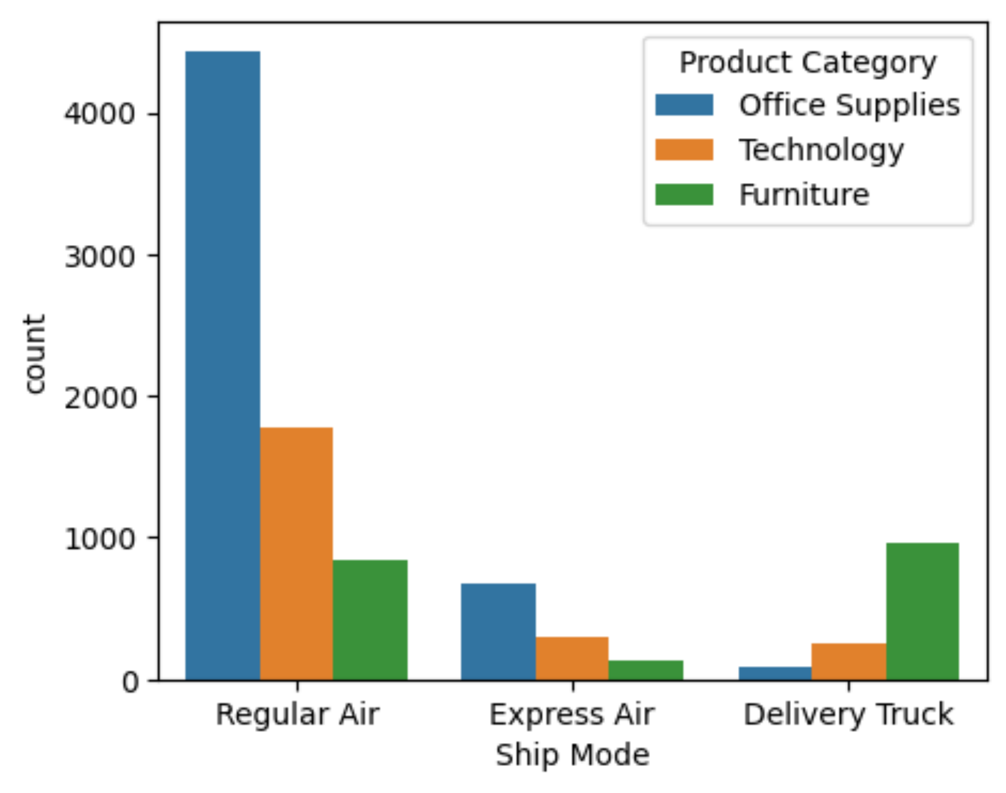
```
plt.figure(figsize=(5,4))
x= dataset['Ship Mode'].value_counts().index
y= dataset['Ship Mode'].value_counts().values
plt.pie(y,labels=x,startangle=60,autopct='%0.2f%%')
plt.legend(loc=3)
plt.show()
```



```
plt.figure(figsize=(5,4))
sns.countplot( x= "Ship Mode",data=dataset)
plt.show()
```

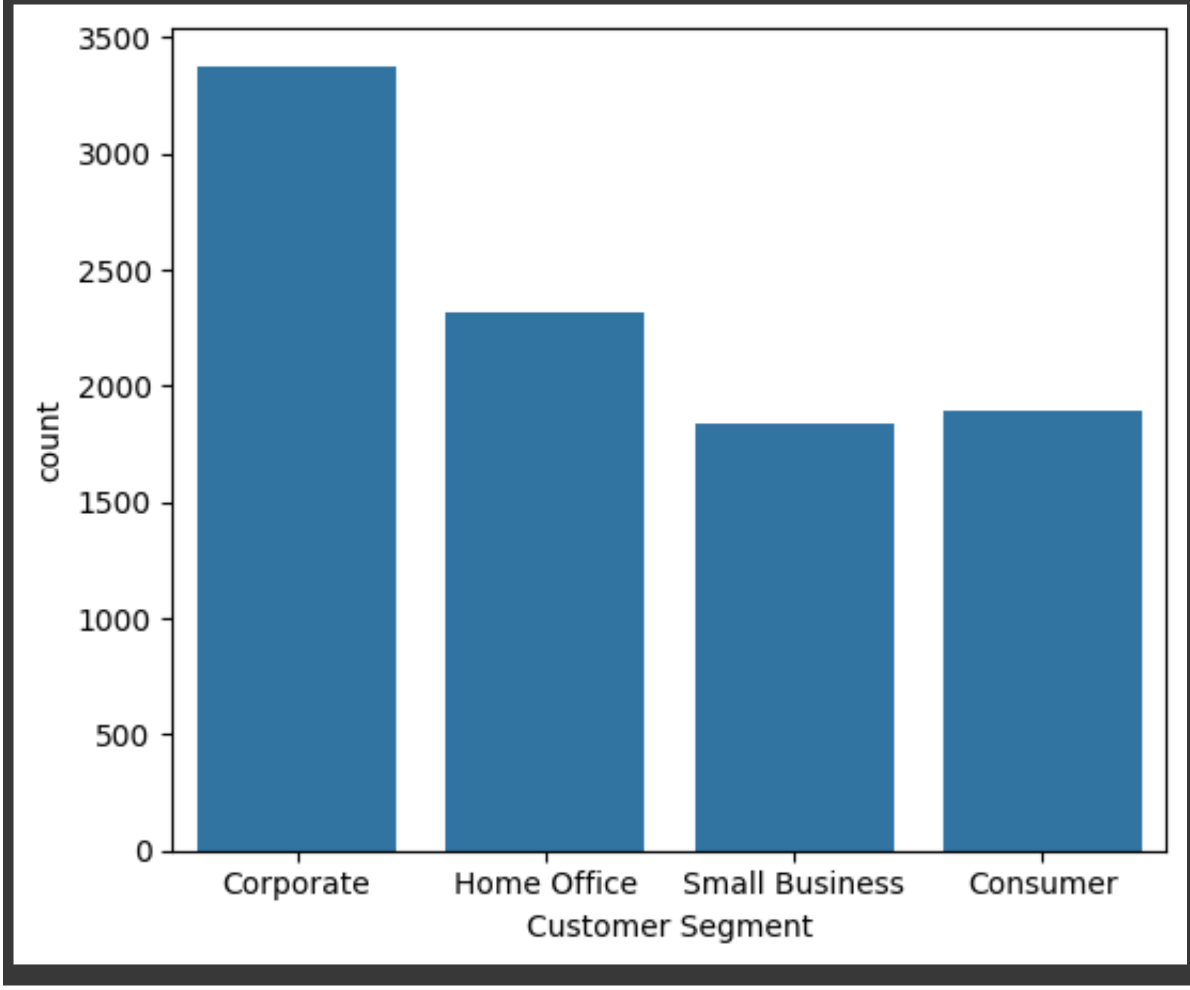


```
  ## hue - differentiates
plt.figure(figsize=(5,4))
sns.countplot( x= "Ship Mode",data=dataset,hue="Product Category")
plt.show()
```

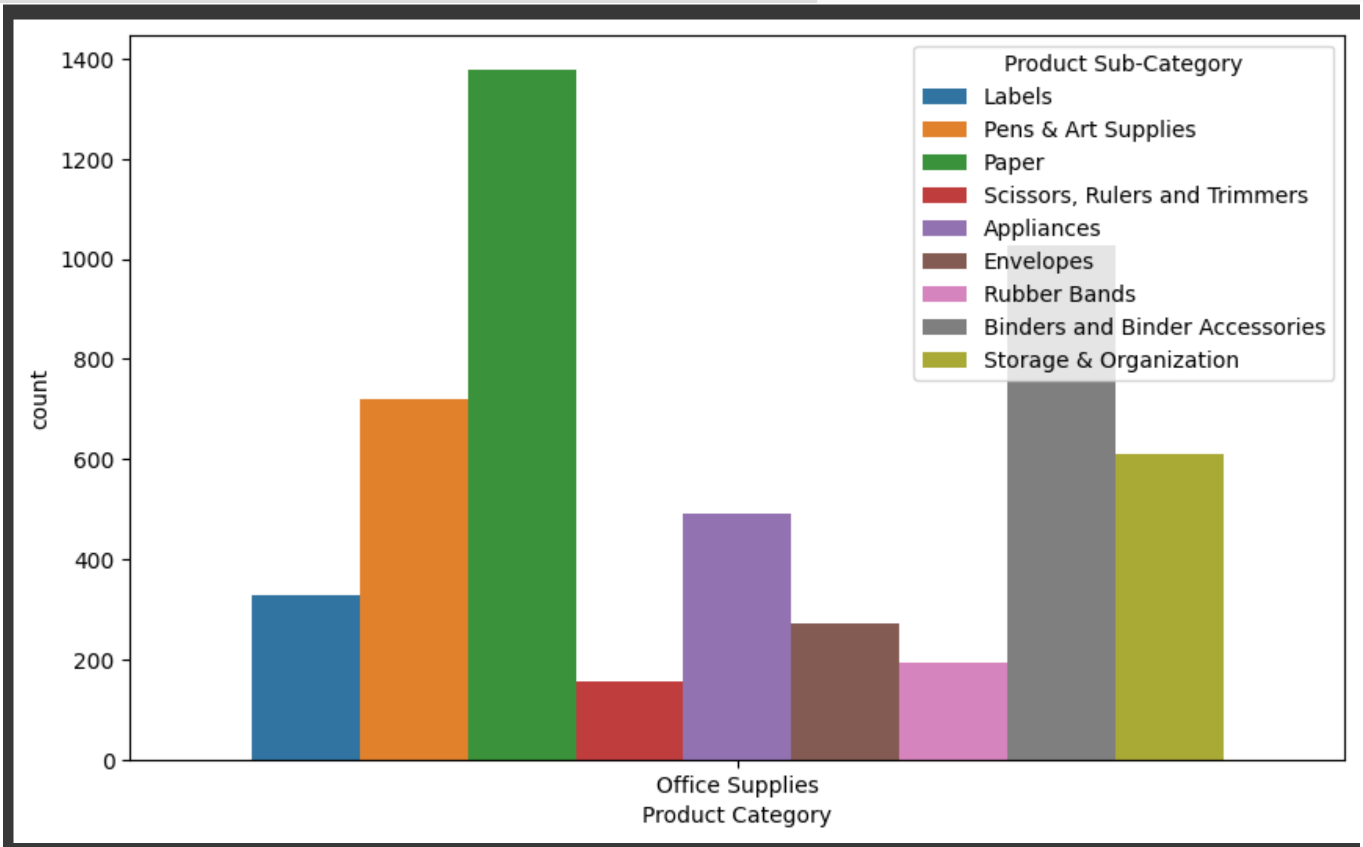


```
## CUSTOMER SEGMENT
```

```
    ## hue - differentiates
plt.figure(figsize=(6,5))
sns.countplot(x="Customer Segment",data=dataset)
plt.show()
```



```
 #. PRODUCT CATEGORY
plt.figure(figsize=(10,6))
sns.countplot(x="Product Category",data = dataset[dataset["Product Category"]==
plt.show()
```



```
##. TO KNOW SALES BY YEAR
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9426 entries, 0 to 9425
Data columns (total 24 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Row ID                9426 non-null   int64
 1   Order Priority        9426 non-null   object
 2   Discount              9426 non-null   float64
 3   Unit Price            9426 non-null   float64
 4   Shipping Cost         9426 non-null   float64
 5   Customer ID           9426 non-null   int64
 6   Customer Name         9426 non-null   object
 7   Ship Mode             9426 non-null   object
 8   Customer Segment      9426 non-null   object
 9   Product Category      9426 non-null   object
 10  Product Sub-Category  9426 non-null   object
 11  Product Container     9426 non-null   object
 12  Product Name          9426 non-null   object
 13  Product Base Margin   9426 non-null   float64
 14  Region                9426 non-null   object
 15  State or Province     9426 non-null   object
 16  City                  9426 non-null   object
 17  Postal Code           9426 non-null   int64
 18  Order Date            9426 non-null   datetime64[ns]
 19  Ship Date             9426 non-null   datetime64[ns]
 20  Profit                9426 non-null   float64
 21  Quantity ordered new  9426 non-null   int64
 22  Sales                 9426 non-null   float64
 23  Order ID              9426 non-null   int64
dtypes: datetime64[ns](2), float64(6), int64(5), object(11)
memory usage: 1.7+ MB
```

```
dataset["Order Year"] = dataset['Order Date'].dt.year
dataset.info()
```

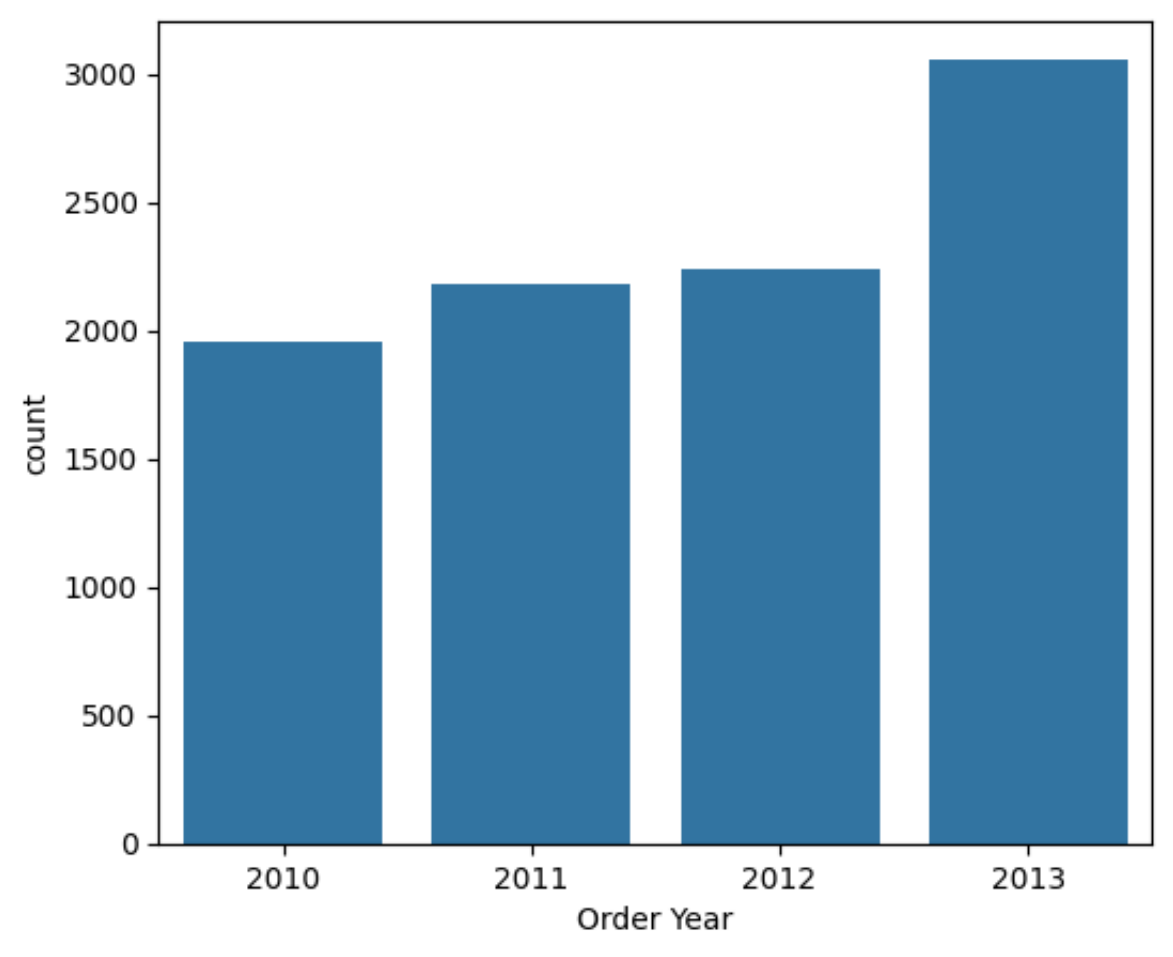
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9426 entries, 0 to 9425
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Row ID                9426 non-null   int64
 1   Order Priority        9426 non-null   object
 2   Discount              9426 non-null   float64
 3   Unit Price            9426 non-null   float64
 4   Shipping Cost         9426 non-null   float64
 5   Customer ID           9426 non-null   int64
 6   Customer Name         9426 non-null   object
 7   Ship Mode             9426 non-null   object
 8   Customer Segment      9426 non-null   object
 9   Product Category      9426 non-null   object
 10  Product Sub-Category  9426 non-null   object
 11  Product Container     9426 non-null   object
 12  Product Name          9426 non-null   object
 13  Product Base Margin   9426 non-null   float64
 14  Region                9426 non-null   object
 15  State or Province     9426 non-null   object
 16  City                  9426 non-null   object
 17  Postal Code           9426 non-null   int64
 18  Order Date            9426 non-null   datetime64[ns]
 19  Ship Date             9426 non-null   datetime64[ns]
 20  Profit                9426 non-null   float64
 21  Quantity ordered new  9426 non-null   int64
 22  Sales                 9426 non-null   float64
 23  Order ID              9426 non-null   int64
 24  Order Year            9426 non-null   int32
dtypes: datetime64[ns](2), float64(6), int32(1), int64(5), object(11)
memory usage: 1.8+ MB
```

```
dataset['Order Year'].value_counts()
```

```
            count
Order Year
      2013   3054
      2012   2241
      2011   2179
      2010   1952

dtype: int64
```

```
plt.figure(figsize=(6,5))
sns.countplot(x="Order Year",data=dataset)
plt.show()
```

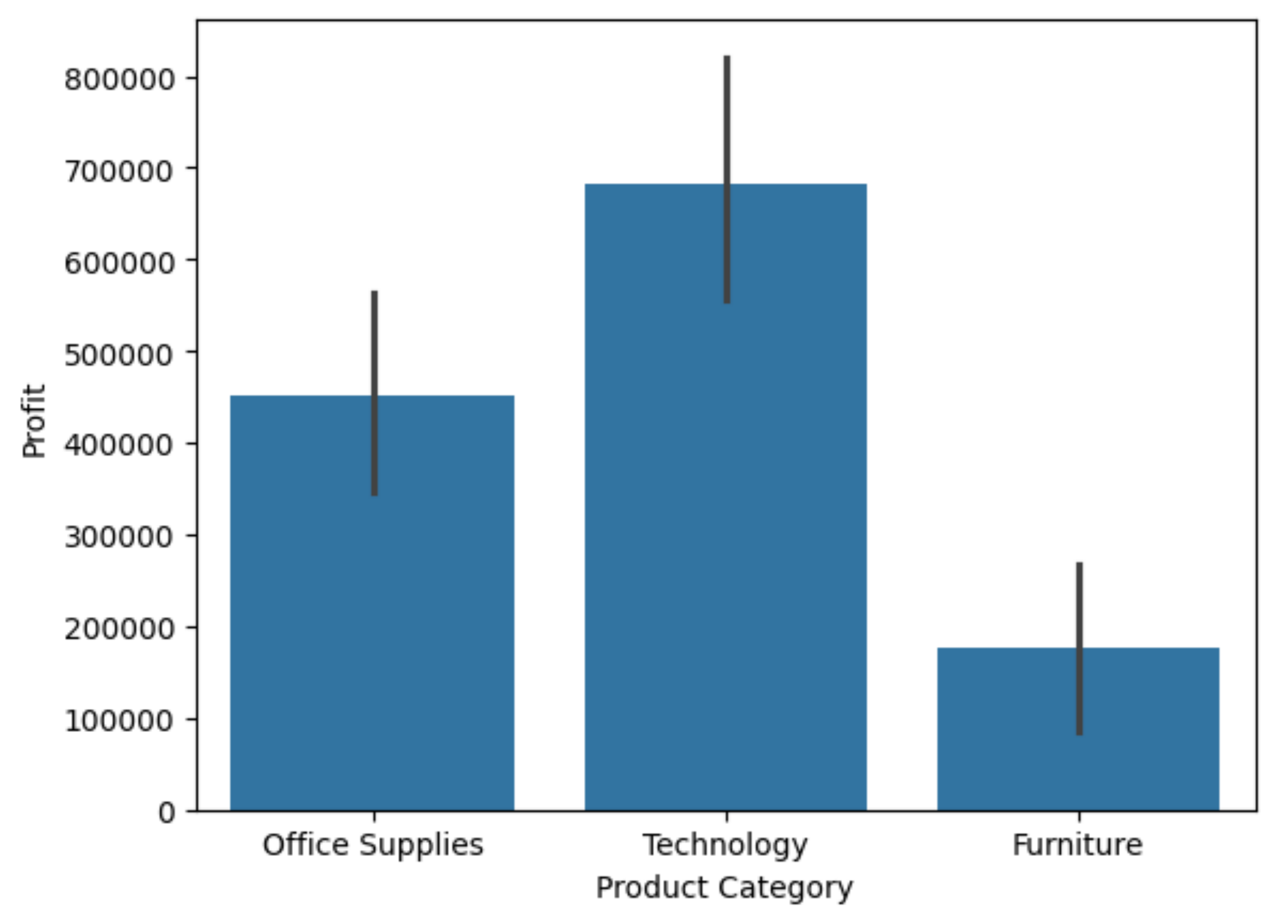


```
##. TO KNOW PROFIT BY YEAR
sns.barplot(x="Product Category",y="Profit",data=dataset,estimator='sum')
plt.show
```

```
matplotlib.pyplot.show
def show(*args, **kwargs)

Display all open figures.

Parameters
----------
block : bool, optional
    Whether to wait for all figures to be closed before returning.
```



```
#### orders/sales  by state
dataset['State or Province'].value_counts()
```

	count
State or Province	
California	1021
Texas	646
Illinois	584

Illinois	[illegible]
New York	574
Florida	522
Ohio	396
Washington	327
Michigan	327
Pennsylvania	271
North Carolina	251
Indiana	241
Minnesota	239
Massachusetts	222
Georgia	214
Virginia	198
Maryland	178
Colorado	177
New Jersey	177
Wisconsin	169
Oregon	168
Tennessee	166
Missouri	161
Iowa	156
Utah	146
Arizona	134
Kansas	133
Maine	128
Alabama	125
Arkansas	123
Idaho	114
South Carolina	105
Oklahoma	104

```
#### top 5 orders/sales  by state
dataset['State or Province'].value_counts()[:5]
```

```
                         count
 State or Province

      California          1021
        Texas              646
       Illinois            584
       New York            574
       Florida             522

dtype: int64

        Nevada              43
     North Dakota           34
     South Dakota           28
       Wyoming              21
     Rhode Island           20
       Delaware             15

dtype: int64
```

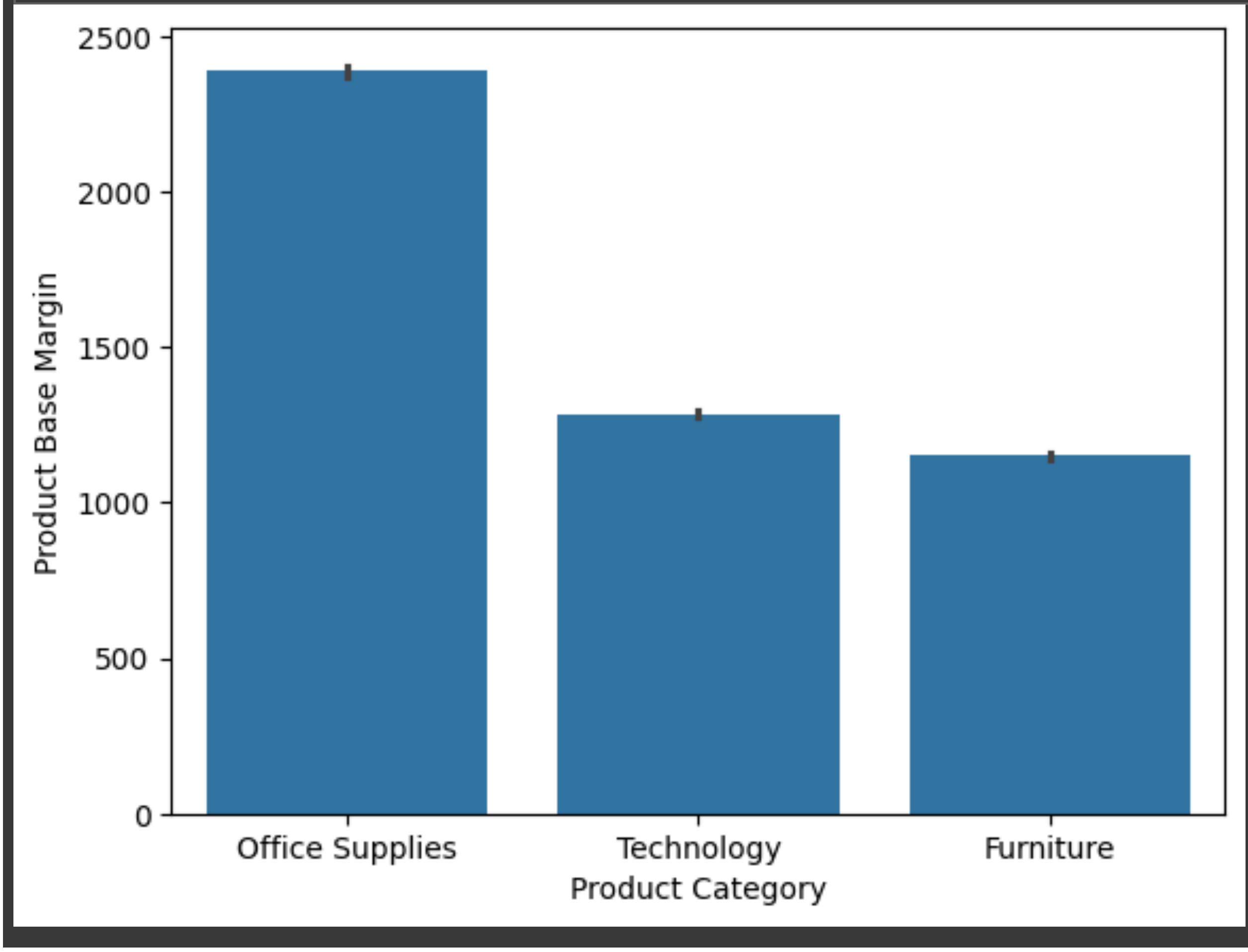
```
# to find product base margin/product

sns.barplot(x="Product Category",y="Product Base Margin",data=dataset,estimator
plt.show
```

```
matplotlib.pyplot.show
def show(*args, **kwargs)
```

```
Display all open figures.

Parameters
----------
block : bool, optional
    Whether to wait for all figures to be closed before returning.
```



```
Start coding or generate with AI.
```