Perform the EDA on Super Market Sales.

In my EDA of the Superstore_USA dataset, I cleaned the data and analyzed profit,
customer segments, and sales trends. I identified opportunities to boost profit
through better pricing strategies, cost management, and targeted customer segments.
Insights from sales patterns and customer behavior will help refine strategies to increase profitability.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

In [2]:
```
df=pd.read_excel('Superstore_USA.xlsx')
df
```

3	23087	Not Specified	0.01	5.68	3.60	3	Bonnie Potter	Regular Air	Corporate	
4	23088	Not Specified	0.00	205.99	2.50	3	Bonnie Potter	Express Air	Corporate	Te
...	...	...	...	...	...	...	...	...	...	...
9421	20275	Critical	0.06	35.89	14.72	3402	Frederick Cole	Regular Air	Consumer	
9422	20276	Critical	0.00	3.34	7.49	3402	Frederick Cole	Regular Air	Consumer	
9423	24491	Not Specified	0.08	550.98	45.70	3402	Frederick Cole	Delivery Truck	Consumer	
9424	25914	High	0.10	105.98	13.99	3403	Tammy Buckley	Express Air	Consumer	
9425	24492	Not Specified	0.09	7.78	2.50	3403	Tammy Buckley	Express Air	Consumer	

9426 rows × 24 columns

In [3]:
```
df.head()
```

Out[3]:

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

5 rows × 24 columns

Do some Basic Operation of dataset.

In [4]:
```
type(df)
```

Out[4]:
```
pandas.core.frame.DataFrame
```

In [5]:
```
df.count()
```

Out[5]:
```
Row ID                  9426
Order Priority          9426
Discount                9426
Unit Price              9426
Shipping Cost           9426
Customer ID             9426
Customer Name           9426
Ship Mode               9426
Customer Segment        9426
Product Category        9426
Product Sub-Category    9426
Product Container       9426
Product Name            9426
Product Base Margin     9354
Region                  9426
State or Province       9426
City                    9426
Postal Code             9426
Order Date              9426
Ship Date               9426
Profit                  9426
Quantity ordered new    9426
Sales                   9426
Order ID                9426
dtype: int64
```

In [6]:
```
df.shape
```

Out[6]:
```
(9426, 24)
```

Check the missing value firstly.

In [7]:
```
df.isnull().sum()
```

Out[7]:
```
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
```

In [8]:
```
df['Product Base Margin']
```

Out[8]:
```
0       0.36
1       0.54
2       0.37
3       0.56
4       0.59
        ...
9421    0.40
9422    0.54
9423    0.71
9424    0.65
9425    0.38
Name: Product Base Margin, Length: 9426, dtype: float64
```

In [9]:
```
# 1. Here we can see that the column of "Product Base Margin" have 72 missing
#  so that we have to fill this by mean of the value of "Product Base Margin"

df['Product Base Margin'].mean()
```

Out[9]:
```
0.5121894376737225
```

In [10]:
```
df['Product Base Margin'].fillna(df['Product Base Margin'].mean(),inplace=True
```

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
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
```

Analysis on Order Priority

In [12]:
```
df["Order Priority"].value_counts()
```

Out[12]:
```
Order Priority
High             1970
Low              1926
Not Specified    1881
Medium           1844
Critical         1804
Critical            1
Name: count, dtype: int64
```

In [ ]:
```

```

In [13]:
```
df['Order Priority'].unique()
```

Out[13]:
```
array(['Not Specified', 'High', 'Medium', 'Low', 'Critical', 'Critical '],
      dtype=object)
```

There is two types of ['Critical', 'Critical '] column which is parts of data cleaning.

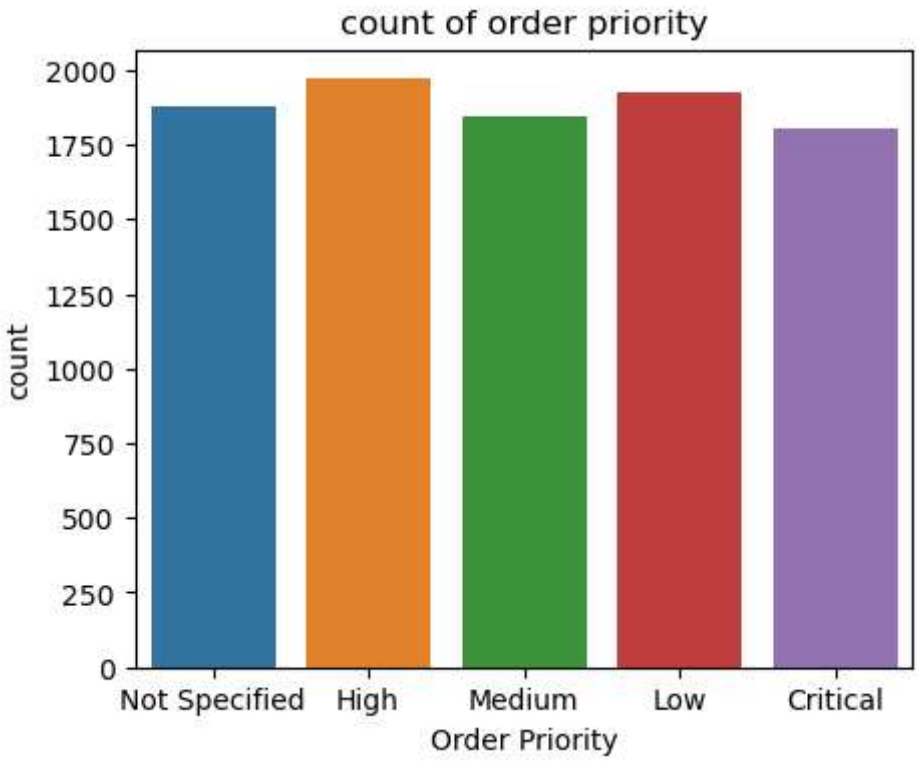
In [14]:
```
df['Order Priority']=df["Order Priority"].replace('Critical ','Critical')
```

In [15]:
```
df['Order Priority'].unique()
```

Out[15]:
```
array(['Not Specified', 'High', 'Medium', 'Low', 'Critical'], dtype=object)
```

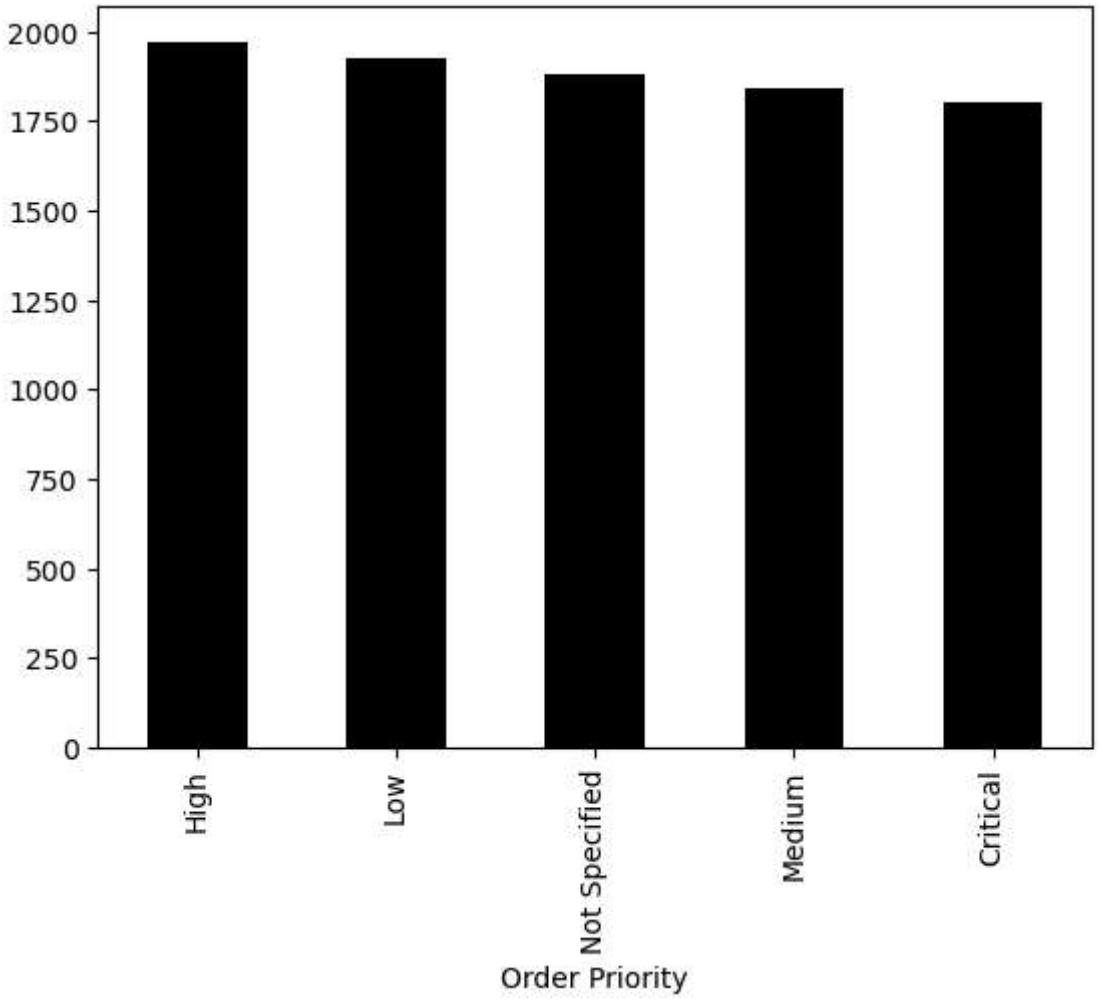
In [ ]:
```

```

In [16]:
```
plt.figure(figsize=(5,4))
sns.countplot(x="Order Priority",data=df)
plt.title("count of order priority")
plt.show()
```



In [17]:
```
df["Order Priority"].value_counts().plot(kind='bar',color="black",x="mmm")
```

Out[17]:
```
<Axes: xlabel='Order Priority'>
```



Analysis of shiping mode

In [18]:
```
df["Ship Mode"]
```

Out[18]:
```
0          Regular Air
1          Express Air
2          Express Air
3          Regular Air
4          Express Air
              ...
9421       Regular Air
9422       Regular Air
9423    Delivery Truck
9424       Express Air
9425       Express Air
Name: Ship Mode, Length: 9426, dtype: object
```

In [19]:
```
df["Ship Mode"].value_counts()
```

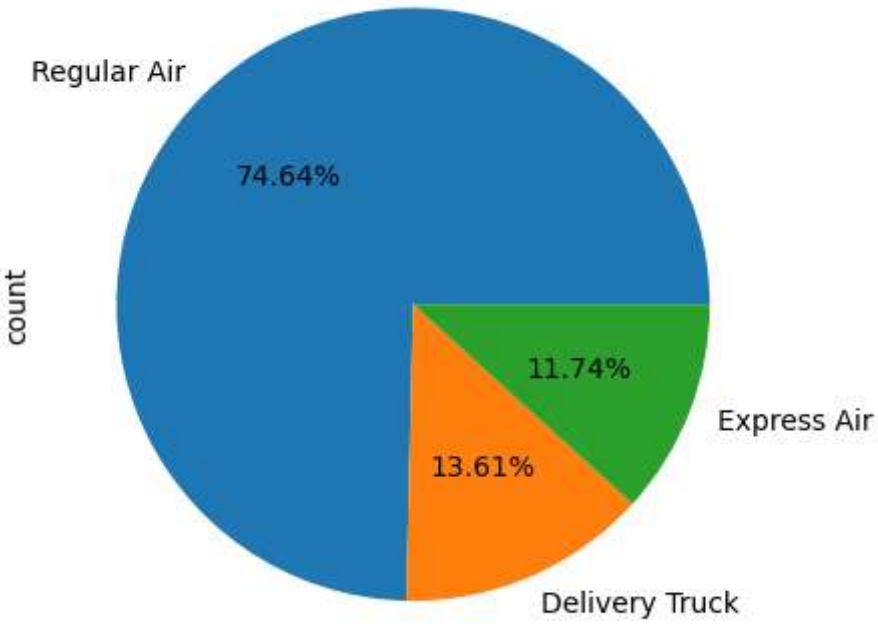
Out[19]:
```
Ship Mode
Regular Air       7036
Delivery Truck    1283
Express Air       1107
Name: count, dtype: int64
```

In [20]:
```
df["Ship Mode"].value_counts().index
```

Out[20]:
```
Index(['Regular Air', 'Delivery Truck', 'Express Air'], dtype='object', name
='Ship Mode')
```

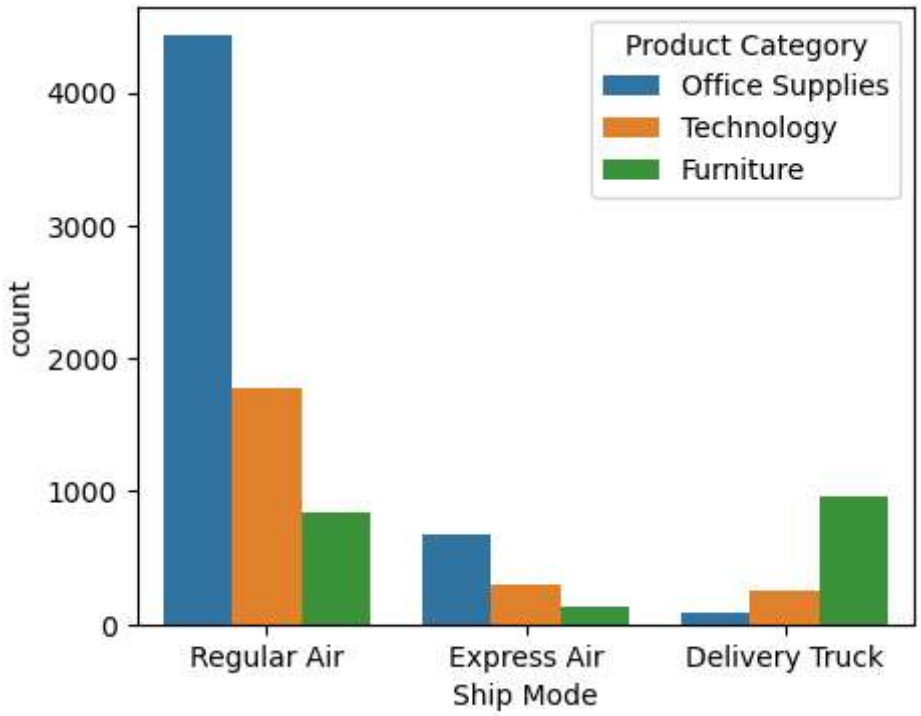
In [21]:
```
df["Ship Mode"].value_counts().plot(kind='pie',autopct='%0.2f%%')
```

Out[21]:
```
<Axes: ylabel='count'>
```



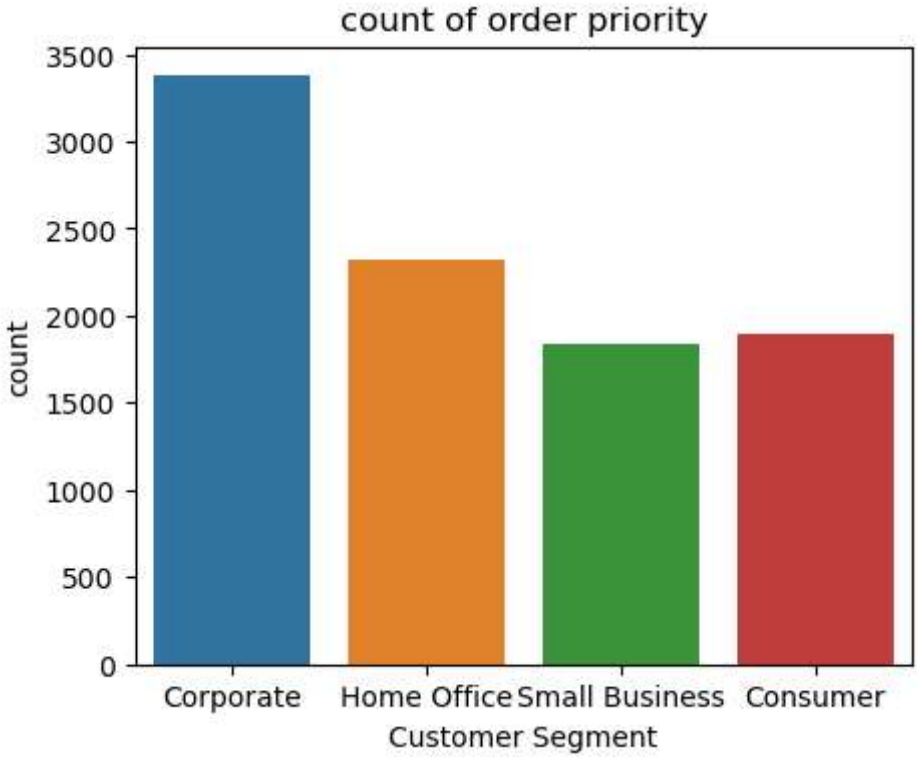
Bivarient analysis with (Product Category vs Ship Mode)

In [22]:
```
plt.figure(figsize=(5,4))
sns.countplot(x='Ship Mode', data=df, hue ="Product Category")
plt.show()
```



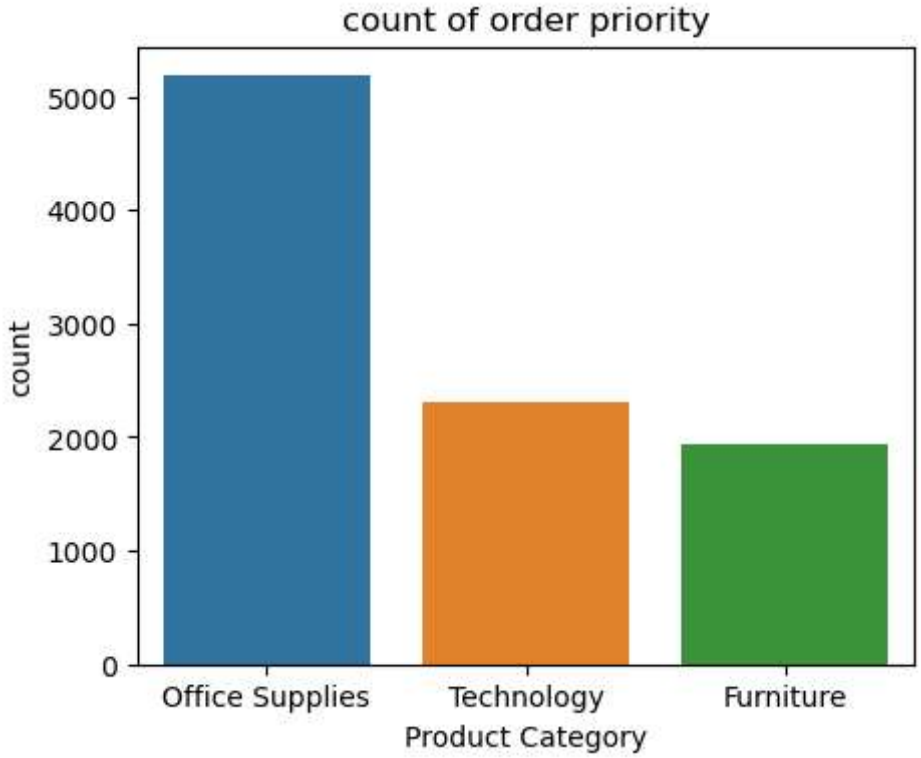
Customer segments

In [23]:
```
plt.figure(figsize=(5,4))
sns.countplot(x="Customer Segment",data=df)
plt.title("count of order priority")
plt.show()
```

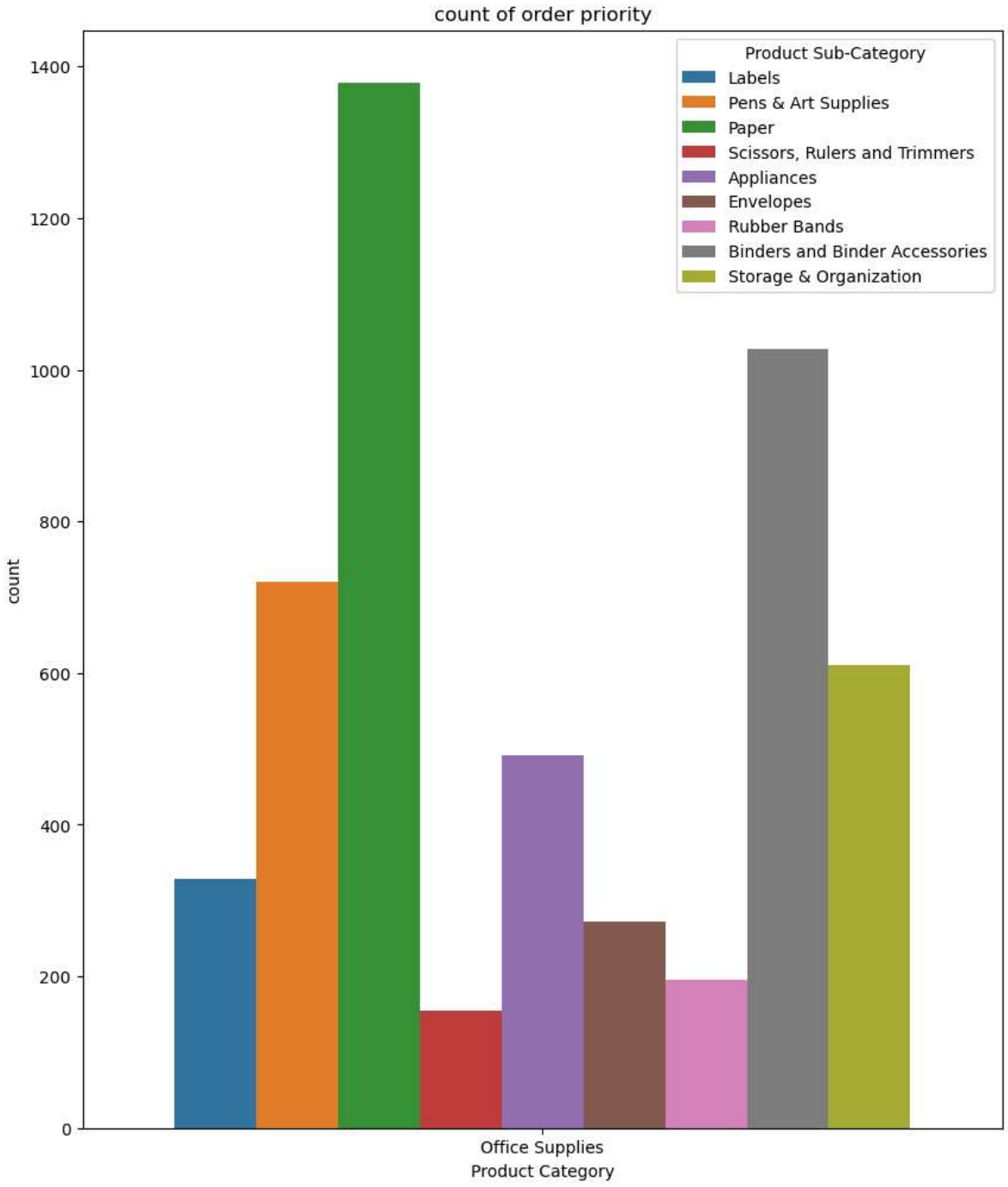


Product category

In [24]:
```
plt.figure(figsize=(5,4))
sns.countplot(x="Product Category",data=df)
plt.title("count of order priority")
plt.show()
```



In [25]:
```
plt.figure(figsize=(10,12))
sns.countplot(x="Product Category",data=df[df['Product Category']=="Office Sup
plt.title("count of order priority")
plt.show()
```



Sales per Year

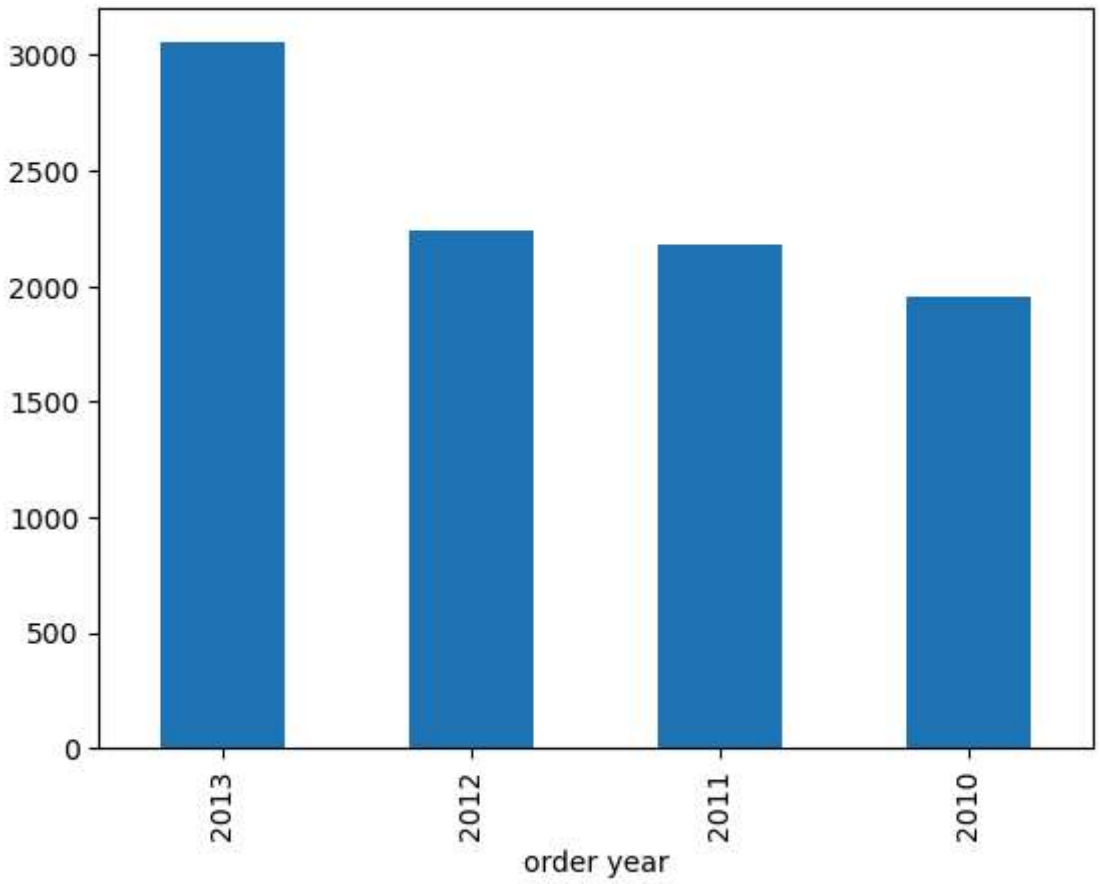
In [26]:
```
df.info()
```

```
<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
```

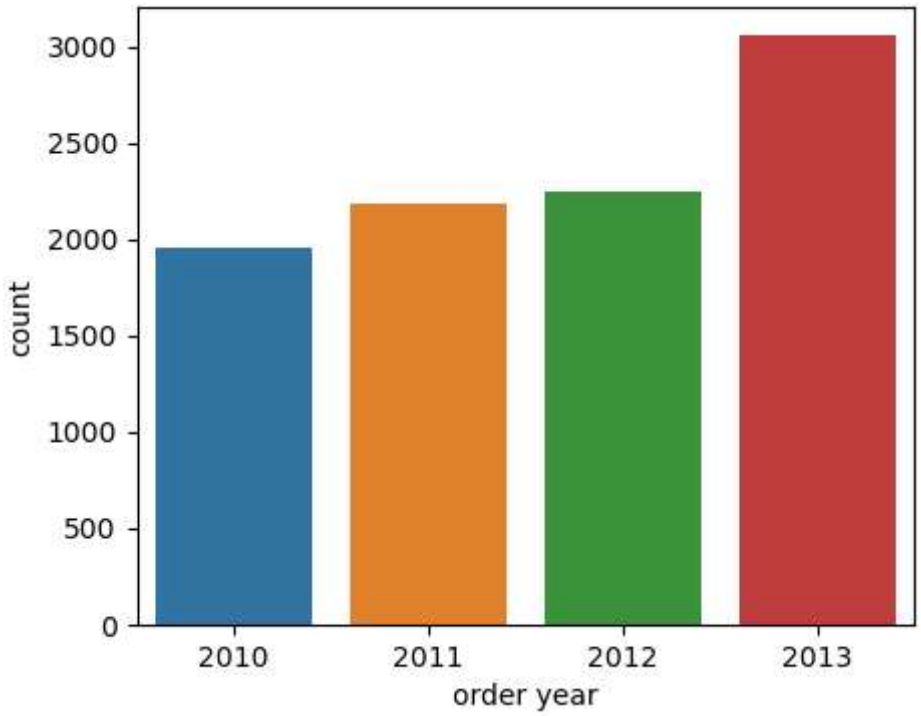
In [27]:
```
df["order year"]=df["Order Date"].dt.year
```

In [28]:
```
df["order year"].value_counts().plot(kind="bar")
```

Out[28]:
```
<Axes: xlabel='order year'>
```

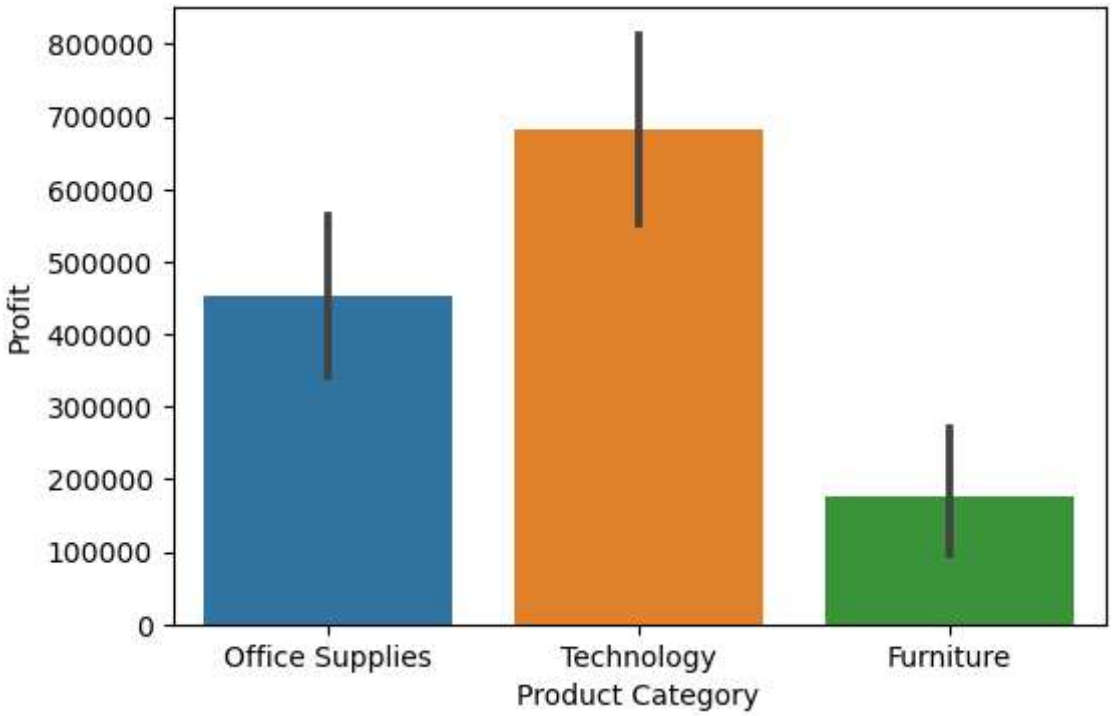


In [29]:
```
plt.figure(figsize=(5,4))
sns.countplot(x="order year",data=df)
plt.show()
```



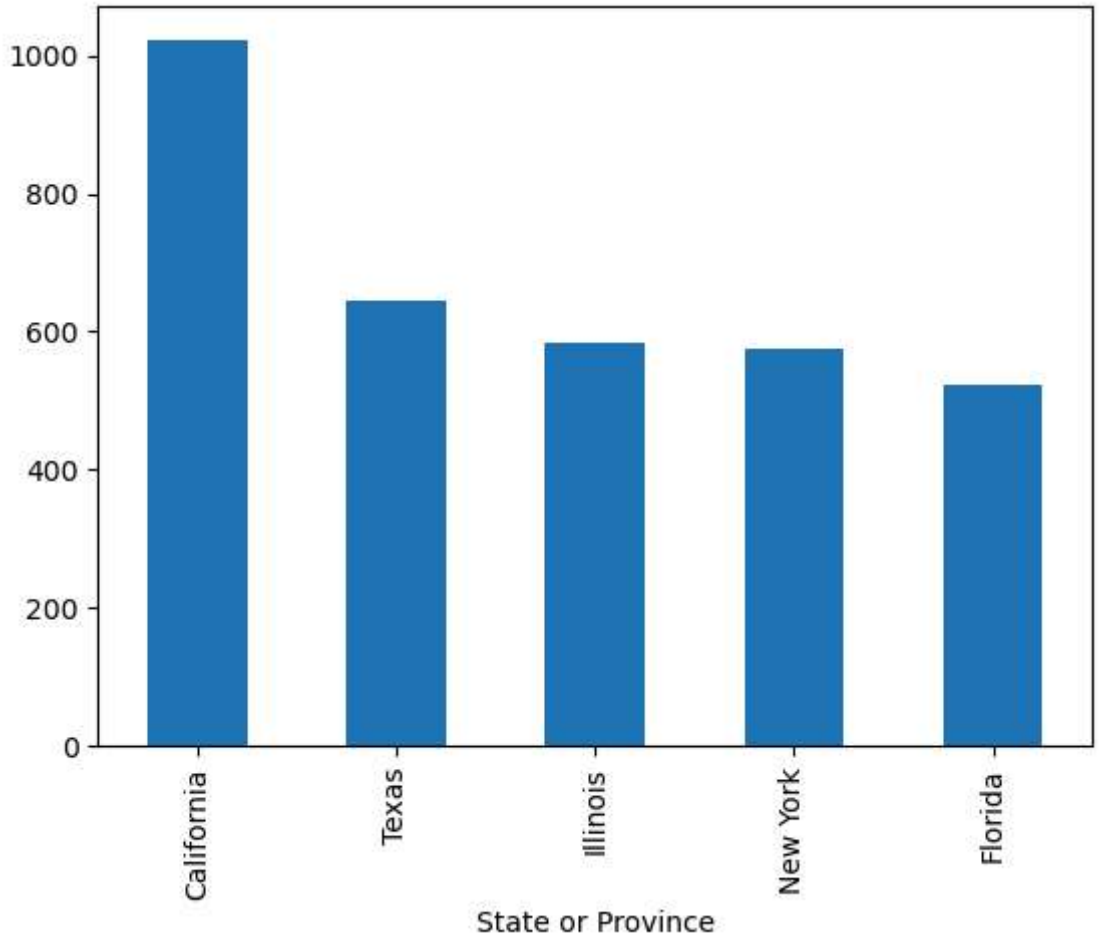
Profit analysis

In [32]:
```
plt.figure(figsize=(6,4))
sns.barplot(x="Product Category",y="Profit",data=df, estimator="sum")
plt.show()
```



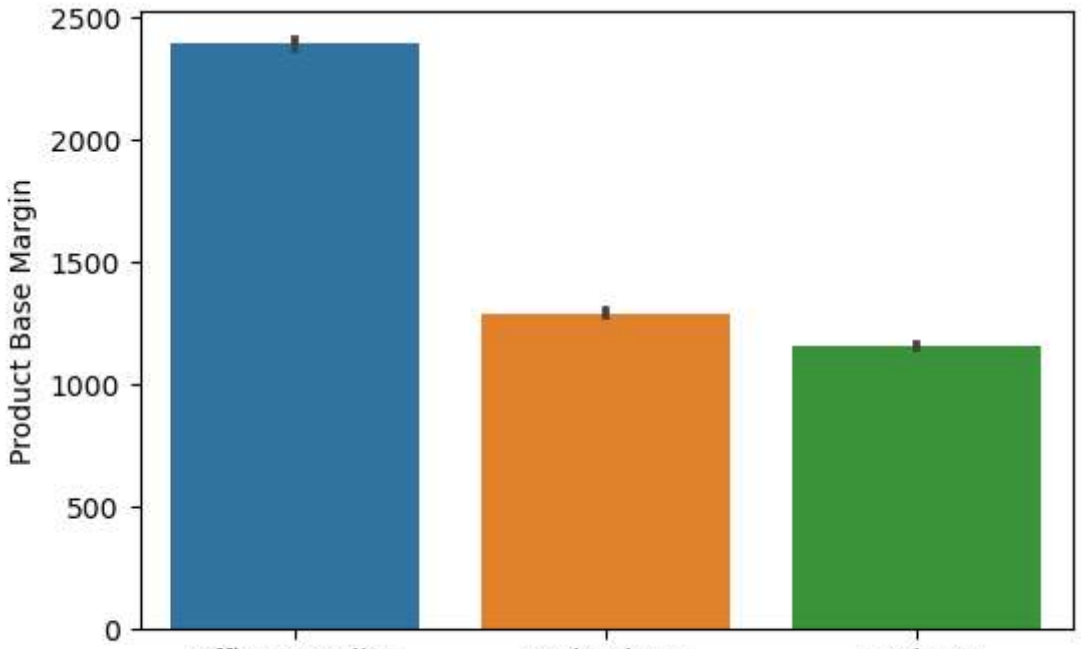
In [42]:
```
df["State or Province"].value_counts().head(5).plot(kind="bar")
# df["State or Province"].value_counts()[:5].plot(kind="bar"), both are same c
```

Out[42]: <Axes: xlabel='State or Province'>



Profit base Margin

In [45]:
```
plt.figure(figsize=(6,4))
sns.barplot(x="Product Category",y="Product Base Margin",data=df, estimator="s
plt.show()
```



Thank you