In [1]:

```
import pandas as pd
import numpy as np
```

In [3]:

```
df = pd.read_csv("Sample - Superstore.csv", encoding='cp1252')
df
```

Out[3]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Cc
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	
...	...	...	...	...	...	...	...	...	
9989	9990	CA-2014-110422	1/21/2014	1/23/2014	Second Class	TB-21400	Tom Boeckenhauer	Consumer	
9990	9991	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	
9991	9992	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	
9992	9993	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	
9993	9994	CA-2017-119914	5/4/2017	5/9/2017	Second Class	CC-12220	Chris Cortes	Consumer	

9994 rows × 21 columns

In [4]:

```
df.shape
```

Out[4]:

```
(9994, 21)
```

In [5]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Row ID         9994 non-null   int64  
 1   Order ID       9994 non-null   object 
 2   Order Date     9994 non-null   object 
 3   Ship Date      9994 non-null   object 
 4   Ship Mode      9994 non-null   object 
 5   Customer ID    9994 non-null   object 
 6   Customer Name  9994 non-null   object 
 7   Segment        9994 non-null   object 
 8   Country        9994 non-null   object 
 9   City           9994 non-null   object 
 10  State          9994 non-null   object 
 11  Postal Code    9994 non-null   int64  
 12  Region         9994 non-null   object 
 13  Product ID     9994 non-null   object 
 14  Category       9994 non-null   object 
 15  Sub-Category   9994 non-null   object 
 16  Product Name   9994 non-null   object 
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64  
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: float64(3), int64(3), object(15)
memory usage: 1.6+ MB
```

In [6]:

```
df.columns
```

Out[6]:

```
Index(['Row ID', 'Order ID', 'Order Date', 'Ship Date', 'Ship Mode',
       'Customer ID', 'Customer Name', 'Segment', 'Country', 'City', 'Stat
e',
       'Postal Code', 'Region', 'Product ID', 'Category', 'Sub-Category',
       'Product Name', 'Sales', 'Quantity', 'Discount', 'Profit'],
      dtype='object')
```

In [7]:

```
df.describe()
```

Out[7]:

	Row ID	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500000	55190.379428	229.858001	3.789574	0.156203	28.656896
std	2885.163629	32063.693350	623.245101	2.225110	0.206452	234.260108
min	1.000000	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	90008.000000	209.940000	5.000000	0.200000	29.364000
max	9994.000000	99301.000000	22638.480000	14.000000	0.800000	8399.976000

In [8]:

```
df.isna().sum()
```

Out[8]:

```
Row ID           0
Order ID         0
Order Date       0
Ship Date        0
Ship Mode        0
Customer ID      0
Customer Name    0
Segment          0
Country          0
City             0
State            0
Postal Code      0
Region           0
Product ID       0
Category         0
Sub-Category     0
Product Name     0
Sales            0
Quantity         0
Discount         0
Profit           0
dtype: int64
```

In [9]:

```
df.duplicated().sum()
```

Out[9]:

```
0
```

In [10]:

```
df_cat = df[[ 'Ship Mode', 'Customer ID', 'Customer Name',
             'Segment', 'Country', 'City', 'State', 'Region',
             'Product ID', 'Category', 'Sub-Category', 'Product Name']]
```

In [11]:

```
df_cat.head()
```

Out[11]:

	Ship Mode	Customer ID	Customer Name	Segment	Country	City	State	Region	Produ
0	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	South	FUR-B 100017
1	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Kentucky	South	FUR-C 100004
2	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	California	West	OFF-L 100002
3	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	South	FUR-T 100005
4	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	Florida	South	OFF-S 100007

In [12]:

```
for feature in df_cat.columns:
    print(feature,':',df[feature].nunique())
```

```
Ship Mode : 4
Customer ID : 793
Customer Name : 793
Segment : 3
Country : 1
City : 531
State : 49
Region : 4
Product ID : 1862
Category : 3
Sub-Category : 17
Product Name : 1850
```

In [13]:

```
df['Order Date'].nunique()
```

Out[13]:

```
1237
```

In [14]:

```
df['Ship Date'].nunique()
```

Out[14]:

```
1334
```

In [15]:

```
df.head()
```

Out[15]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country
0	1	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States
1	2	CA-2016-152156	11/8/2016	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States
2	3	CA-2016-138688	6/12/2016	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States
3	4	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States
4	5	US-2015-108966	10/11/2015	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States

5 rows × 21 columns

In [16]:

```
df.tail()
```

Out[16]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Cou
9989	9990	CA-2014-110422	1/21/2014	1/23/2014	Second Class	TB-21400	Tom Boeckenhauer	Consumer	Un St
9990	9991	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	Un St
9991	9992	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	Un St
9992	9993	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	Un St
9993	9994	CA-2017-119914	5/4/2017	5/9/2017	Second Class	CC-12220	Chris Cortes	Consumer	Un St

5 rows × 21 columns

In [17]:

```
product_group = df.groupby(["Product Name"]).sum()["Sales"]
product_group.head()
```

Out[17]:

```
Product Name
"While you Were Out" Message Book, One Form per Page     25.228
#10 Gummed Flap White Envelopes, 100/Box                 41.300
#10 Self-Seal White Envelopes                           108.682
#10 White Business Envelopes,4 1/8 x 9 1/2              488.904
#10- 4 1/8" x 9 1/2" Recycled Envelopes                 286.672
Name: Sales, dtype: float64
```

In [18]:

```
top_selling_products = product_group.sort_values(ascending=False)
top_5_selling_products = pd.DataFrame(top_selling_products.head())
top_5_selling_products
```

Out[18]:

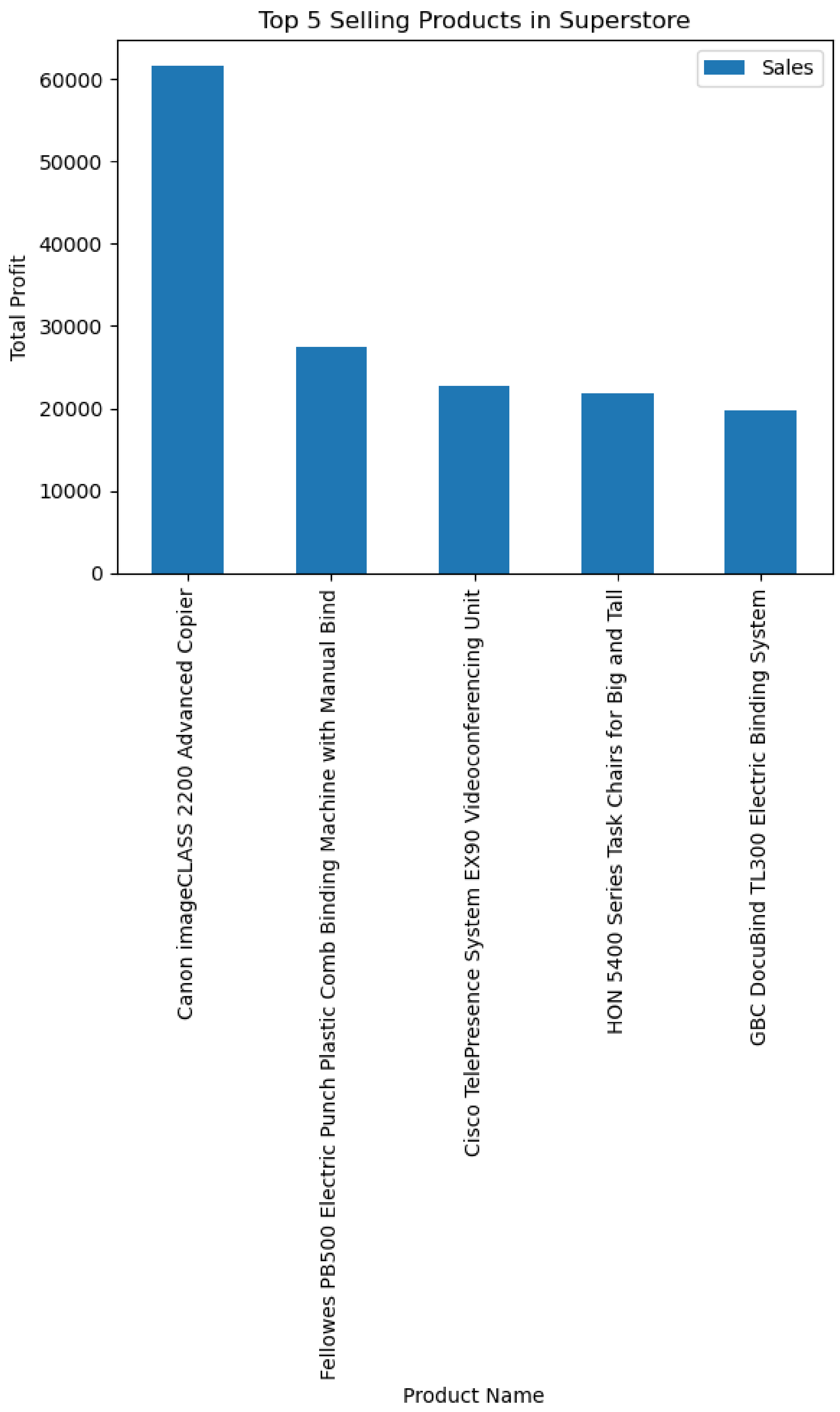
	Sales
Product Name	
Canon imageCLASS 2200 Advanced Copier	61599.824
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind	27453.384
Cisco TelePresence System EX90 Videoconferencing Unit	22638.480
HON 5400 Series Task Chairs for Big and Tall	21870.576
GBC DocuBind TL300 Electric Binding System	19823.479

In [19]:

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [20]:

```
top_5_selling_products.plot(kind="bar")

# Add a title to the plot
plt.title("Top 5 Selling Products in Superstore")

# Add labels to the x and y axes
plt.xlabel("Product Name")
plt.ylabel("Total Profit")

# Show the plot
plt.show()
```


Top 5 Selling Products in Superstore
Sales
Total Profit
60000
50000
40000
30000
20000
10000
0
Canon imageCLASS 2200 Advanced Copier
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind
Cisco TelePresence System EX90 Videoconferencing Unit
HON 5400 Series Task Chairs for Big and Tall
GBC DocuBind TL300 Electric Binding System
Product Name

In [21]:

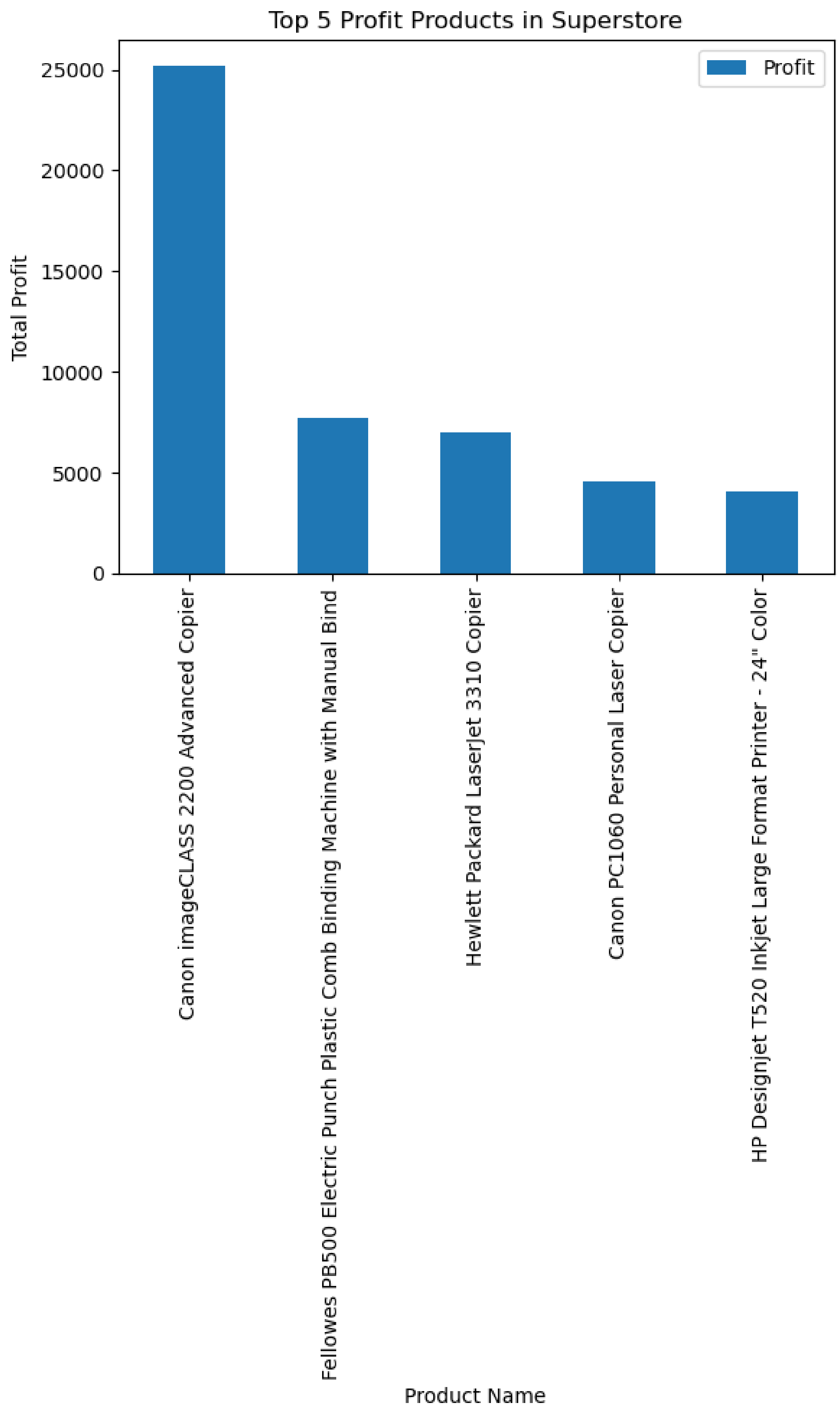
```
product_group = df.groupby(["Product Name"]).sum()["Profit"]

top_profit_products = product_group.sort_values(ascending=False)

top_5_profit_products =pd.DataFrame(top_profit_products[:5])
top_5_profit_products
```

Out[21]:

	Profit
Product Name	
Canon imageCLASS 2200 Advanced Copier	25199.9280
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind	7753.0390
Hewlett Packard LaserJet 3310 Copier	6983.8836
Canon PC1060 Personal Laser Copier	4570.9347
HP Designjet T520 Inkjet Large Format Printer - 24" Color	4094.9766

In [22]:

```
top_5_profit_products.plot(kind="bar")

plt.title("Top 5 Profit Products in Superstore")

plt.xlabel("Product Name")
plt.ylabel("Total Profit")

plt.show()
```


Top 5 Profit Products in Superstore
Profit
Total Profit
25000
20000
15000
10000
5000
0
Canon imageCLASS 2200 Advanced Copier
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind
Hewlett Packard LaserJet 3310 Copier
Canon PC1060 Personal Laser Copier
HP Designjet T520 Inkjet Large Format Printer - 24" Color
Product Name

In [23]:

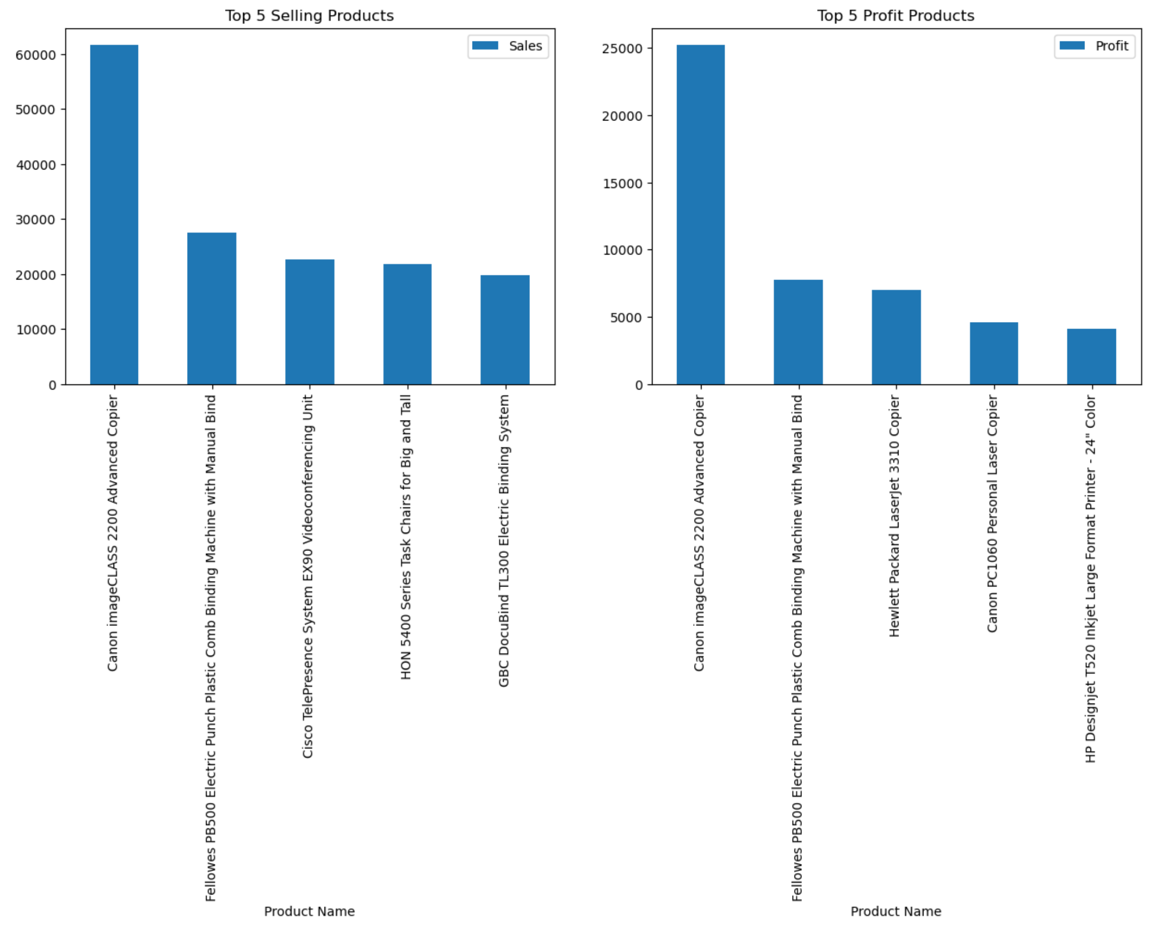
```
top_5_profit_products.index == top_5_selling_products.index
```

Out[23]:

```
array([ True,  True, False, False, False])
```

In [24]:

```
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(15,5))

# Plot the top 5 selling products in the first column
top_5_selling_products.plot(kind="bar", y="Sales", ax=ax1)

# Set the title for the first plot
ax1.set_title("Top 5 Selling Products")

# Plot the top 5 profit products in the second column
top_5_profit_products.plot(kind="bar", y="Profit", ax=ax2)

# Set the title for the second plot
ax2.set_title("Top 5 Profit Products")

# Show the plot
plt.show()
```



In [25]:

```
list(top_5_profit_products.index)
```

Out[25]:

```
['Canon imageCLASS 2200 Advanced Copier',
 'Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual B
ind',
 'Hewlett Packard LaserJet 3310 Copier',
 'Canon PC1060 Personal Laser Copier',
 'HP Designjet T520 Inkjet Large Format Printer - 24" Color']
```

In [26]:

```
list(top_5_selling_products.index)
```

Out[26]:

```
['Canon imageCLASS 2200 Advanced Copier',
 'Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual B
ind',
 'Cisco TelePresence System EX90 Videoconferencing Unit',
 'HON 5400 Series Task Chairs for Big and Tall',
 'GBC DocuBind TL300 Electric Binding System']
```

In [27]:

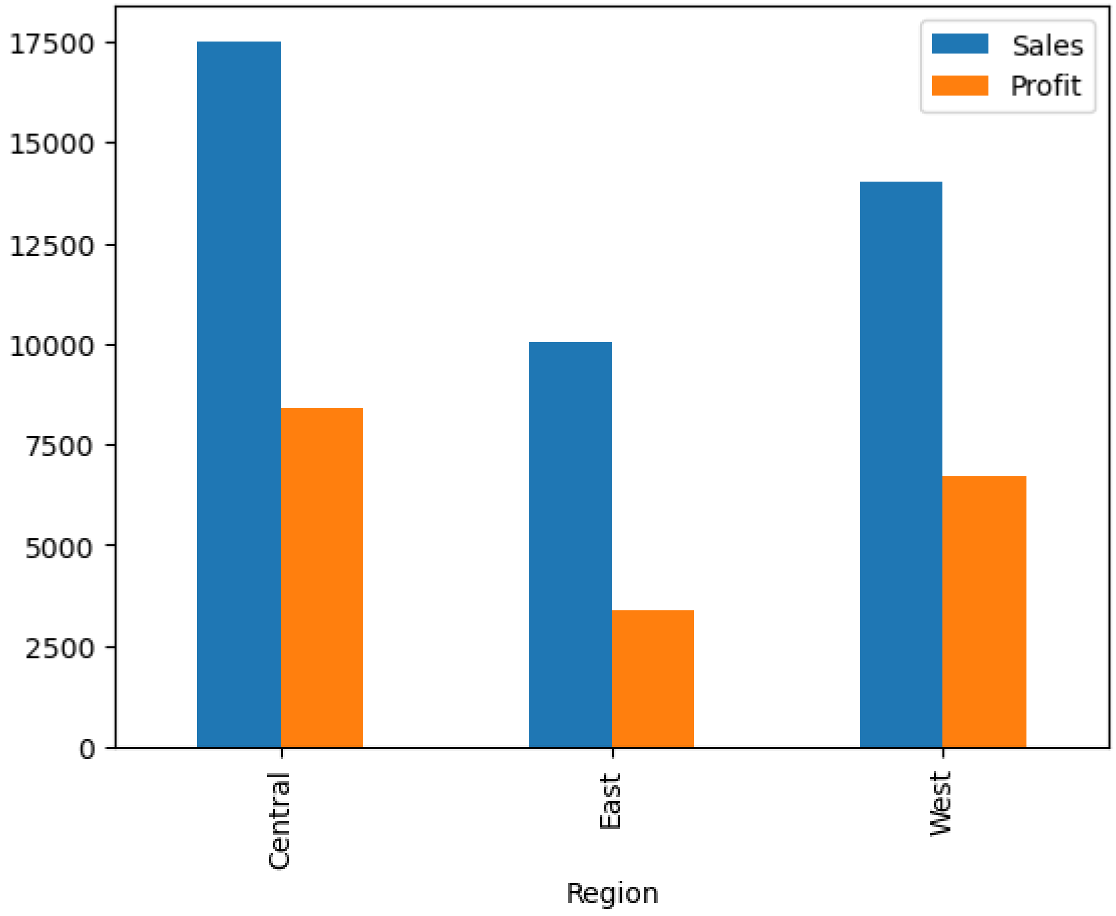
```
df.Region.value_counts()
```

Out[27]:

```
West       3203
East       2848
Central    2323
South      1620
Name: Region, dtype: int64
```

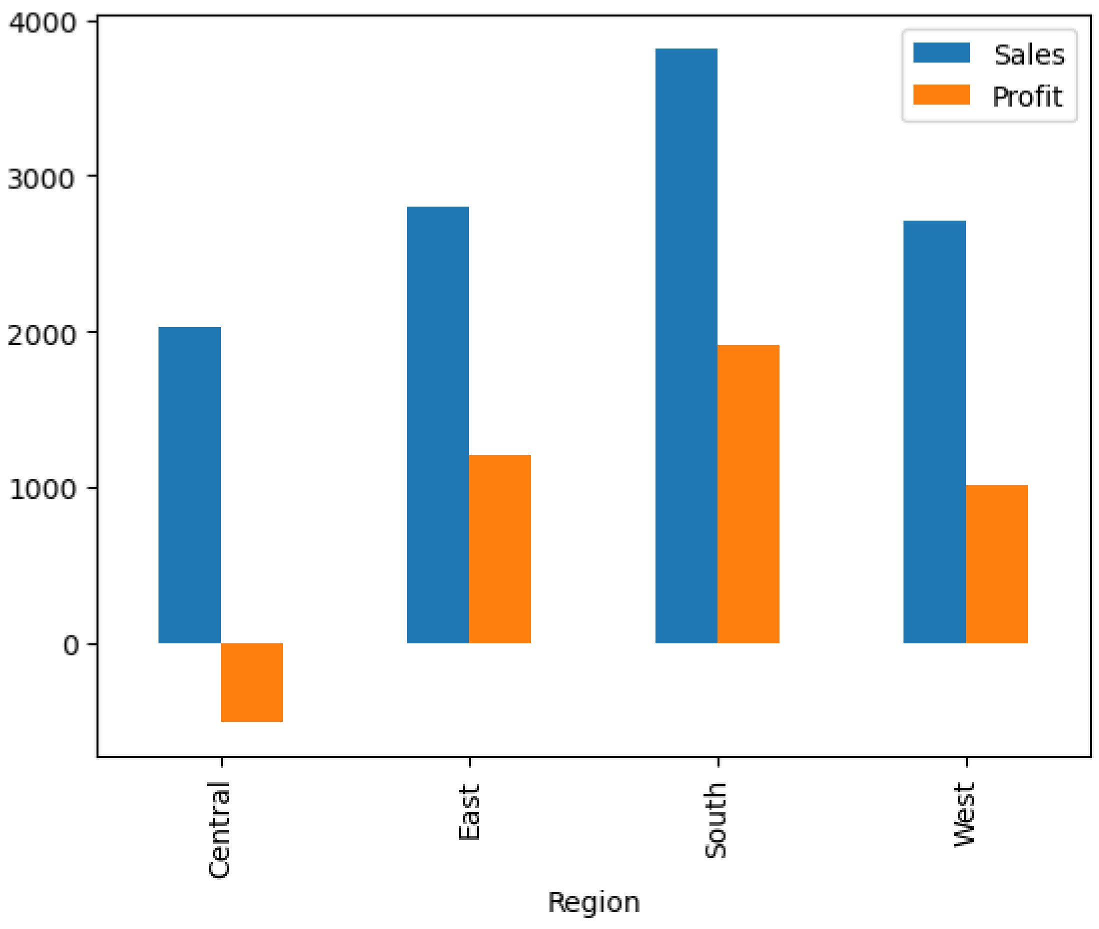
In [28]:

```
product = df[df["Product Name"] == "Canon imageCLASS 2200 Advanced Copier"]

# Group the data by Region
region_group = product.groupby(["Region"]).mean()[["Sales", "Profit"]]

# Ploting
region_group.plot(kind="bar")

plt.show()
```



In [29]:

```
product = df[df["Product Name"] == "Fellowes PB500 Electric Punch Plastic Comb Binding M

# Group the data by Region
region_group = product.groupby(["Region"]).mean()[["Sales", "Profit"]]

# Plot the average sales and profit by region
region_group.plot(kind="bar")

# Show the plot
plt.show()
```



In [30]:

```
product = df[(df["Product Name"] == "Fellowes PB500 Electric Punch Plastic Comb Binding 
product
```

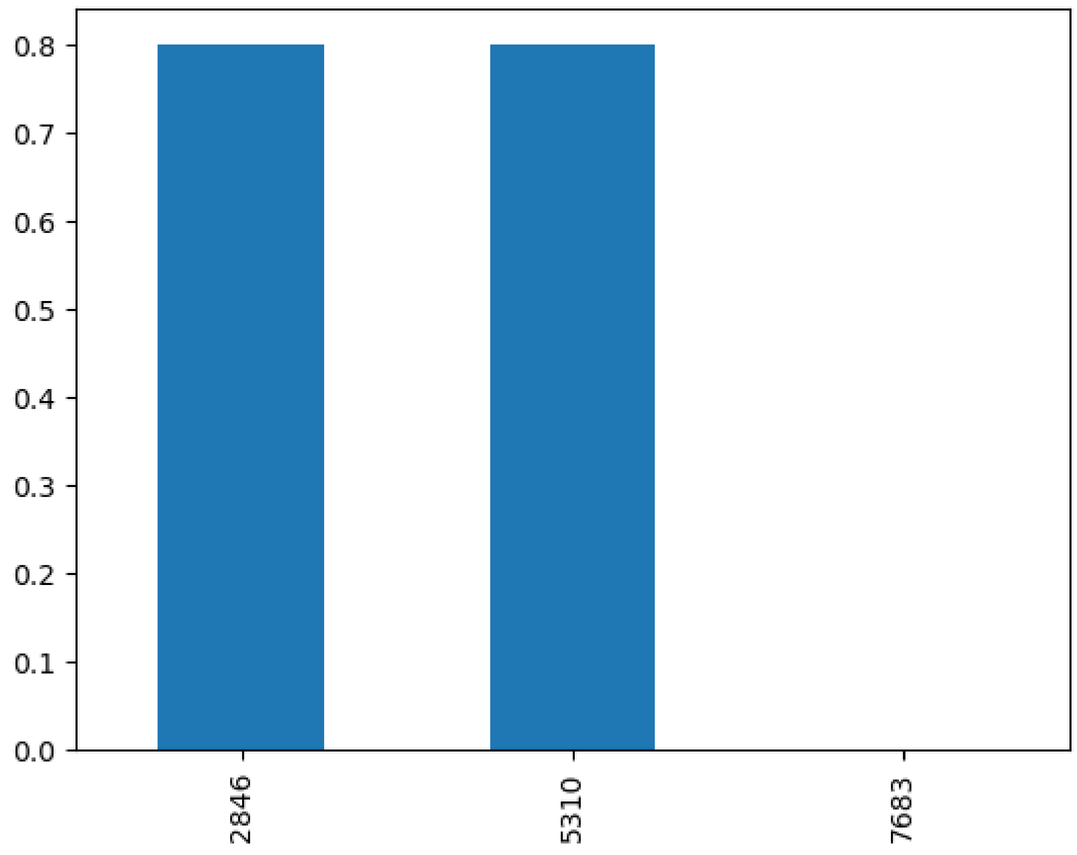
Out[30]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Count
2846	2847	CA-2017-152093	9/10/2017	9/15/2017	Standard Class	SN-20560	Skye Norling	Home Office	Unit Stat
5310	5311	CA-2017-131254	11/19/2017	11/21/2017	First Class	NC-18415	Nathan Cano	Consumer	Unit Stat
7683	7684	CA-2015-120782	4/28/2015	5/1/2015	First Class	SD-20485	Shirley Daniels	Home Office	Unit Stat

3 rows × 21 columns

In [31]:

```
product = df[(df["Product Name"] == "Fellowes PB500 Electric Punch Plastic Comb Binding 

# Plot a histogram of the discounts offered for the product in the central region
product["Discount"].plot(kind="bar")

# Show the plot
plt.show()
product
```



Out[31]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Count
2846	2847	CA-2017-152093	9/10/2017	9/15/2017	Standard Class	SN-20560	Skye Norling	Home Office	Unite Stat
5310	5311	CA-2017-131254	11/19/2017	11/21/2017	First Class	NC-18415	Nathan Cano	Consumer	Unite Stat
7683	7684	CA-2015-120782	4/28/2015	5/1/2015	First Class	SD-20485	Shirley Daniels	Home Office	Unite Stat

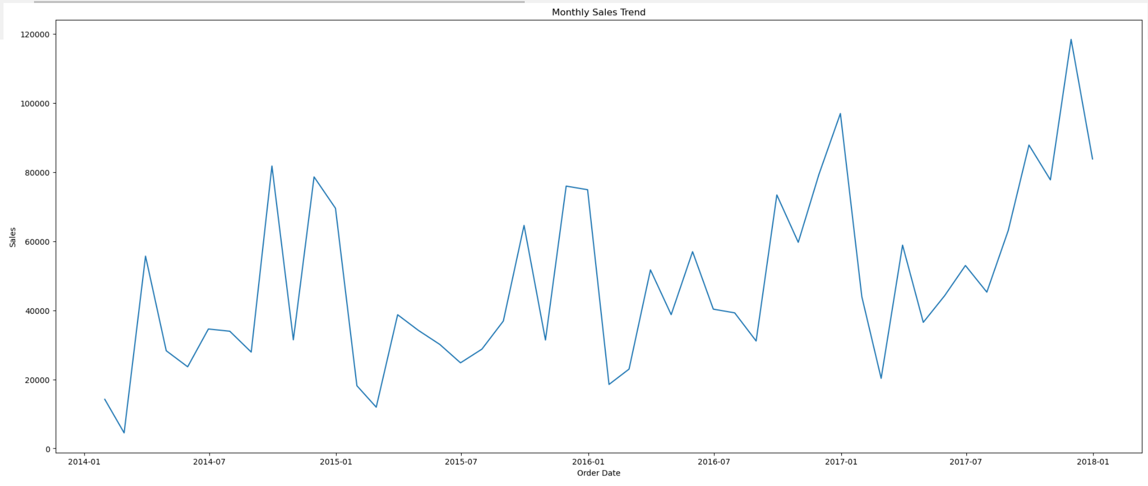
3 rows × 21 columns

In [32]:

```
df['Order Date'] = pd.to_datetime(df['Order Date'])
```

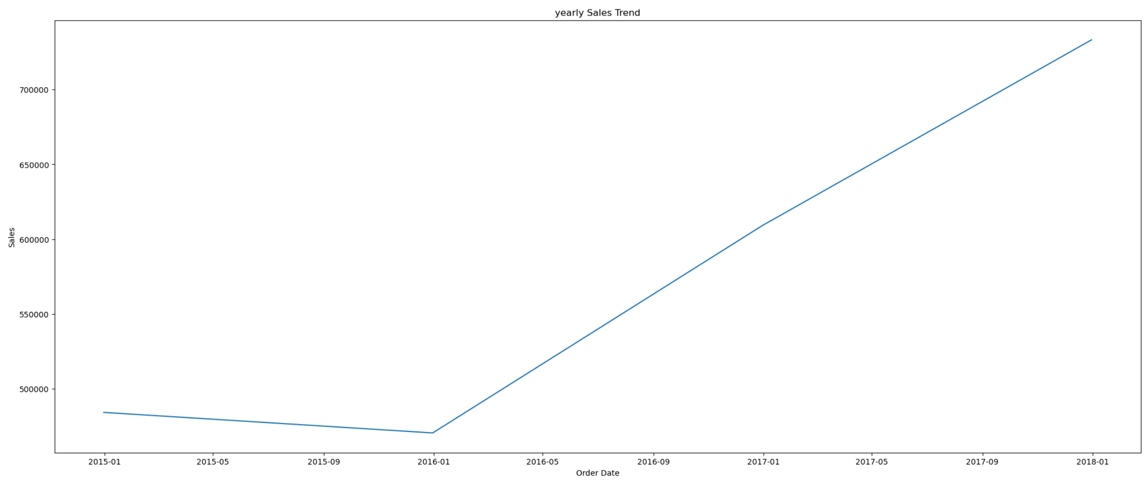
In [33]:

```
monthly_sales = df.groupby(['Order Date'], as_index=False).sum()

# Set the Order Date column as the index of the dataframe
monthly_sales = monthly_sales.set_index('Order Date')
# Resample the data into monthly intervals
monthly_sales = monthly_sales.resample('M').sum() # M for month

# Plot
plt.figure(figsize=(25,10))
plt.plot(monthly_sales['Sales'])
plt.xlabel("Order Date")
plt.ylabel("Sales")
plt.title("Monthly Sales Trend")
plt.show()
```



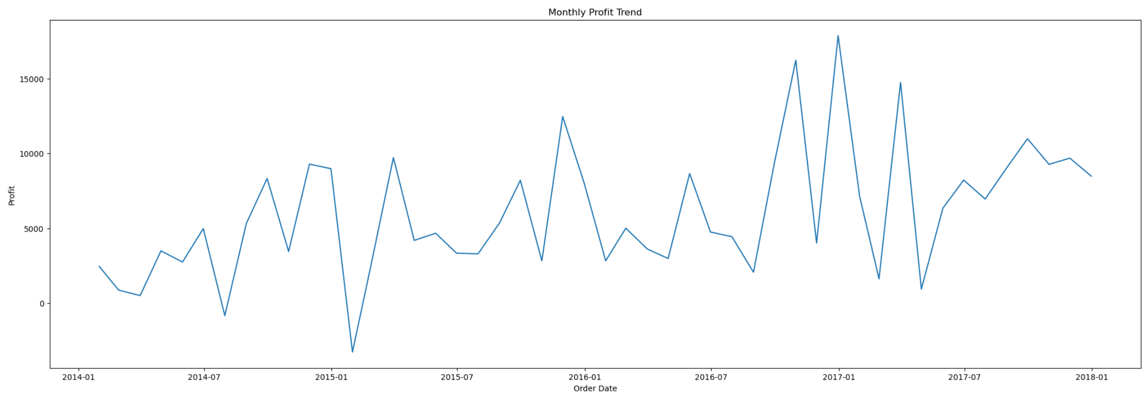
In [34]:

```
yearly_sales = monthly_sales.resample('Y').sum()


plt.figure(figsize=(25,10))
plt.plot(yearly_sales['Sales'])
plt.xlabel("Order Date")
plt.ylabel("Sales")
plt.title("yearly Sales Trend")
plt.show()
```



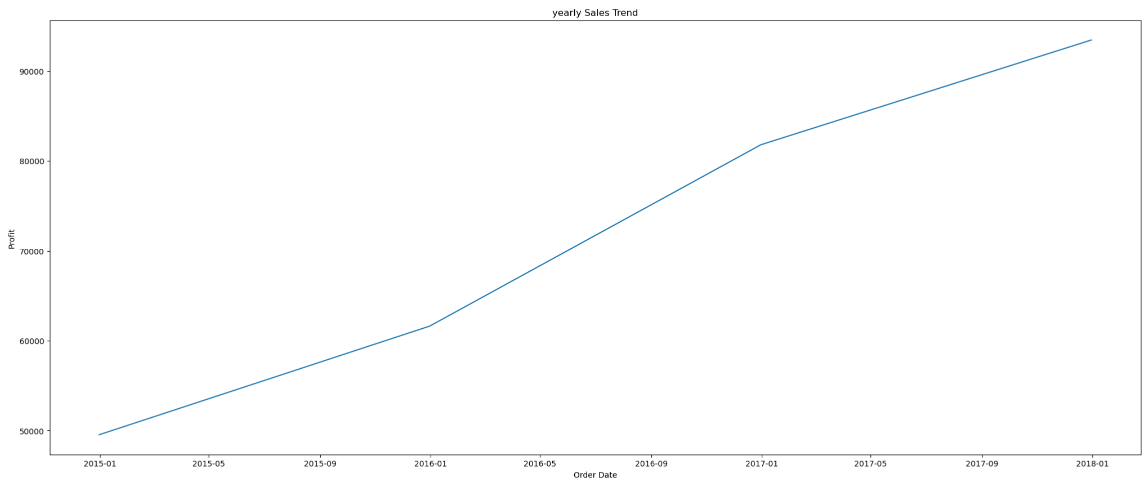
In [35]:

```
monthly_sales = df.groupby(['Order Date'], as_index=False).sum()

# Set the Order Date column as the index of the dataframe
monthly_sales = monthly_sales.set_index('Order Date')

# Resample the data into monthly intervals
monthly_sales = monthly_sales.resample('M').sum() # M for month

# Plot
plt.figure(figsize=(25,8))
plt.plot(monthly_sales['Profit'])
plt.xlabel("Order Date")
plt.ylabel("Profit")
plt.title("Monthly Profit Trend")
plt.show()
```



In [36]:

```
yearly_sales = monthly_sales.resample('Y').sum()


plt.figure(figsize=(25,10))
plt.plot(yearly_sales['Profit'])
plt.xlabel("Order Date")
plt.ylabel("Profit")
plt.title("yearly Sales Trend")
plt.show()
```



In [37]:

```
df_places = df[['Country','City','State','Region']]
df_places.head()
```

Out[37]:

	Country	City	State	Region
0	United States	Henderson	Kentucky	South
1	United States	Henderson	Kentucky	South
2	United States	Los Angeles	California	West
3	United States	Fort Lauderdale	Florida	South
4	United States	Fort Lauderdale	Florida	South

In [38]:

```
for place in df_places.columns:
    print(place,':',df_places[place].nunique())
```

```
Country : 1
City : 531
State : 49
Region : 4
```

In [39]:

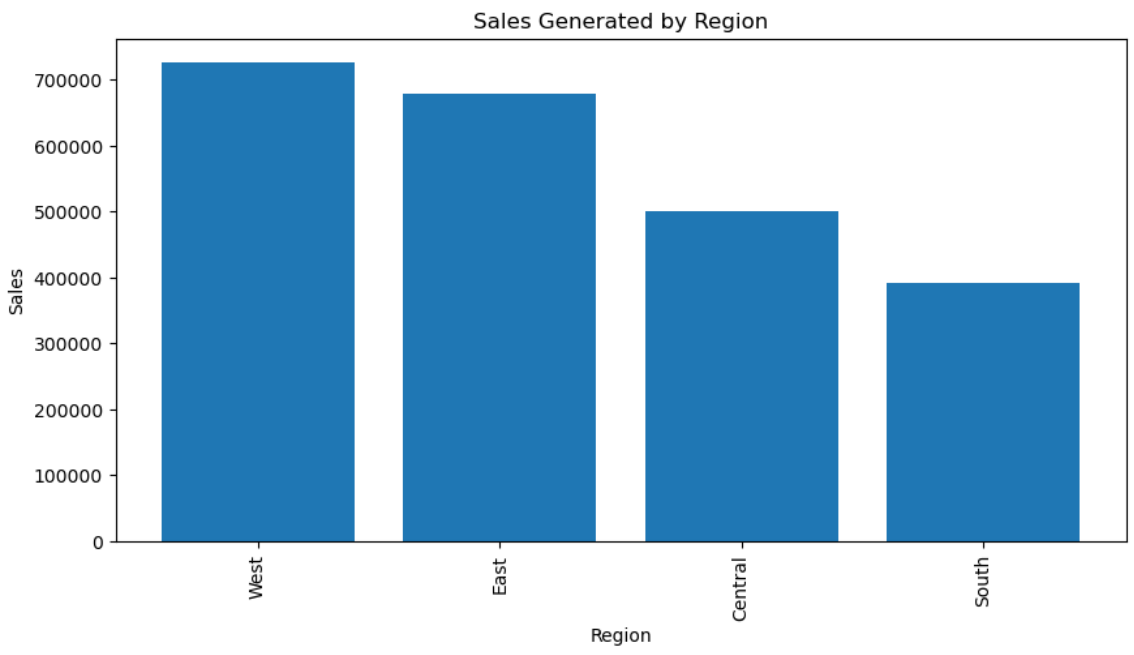
```
df_places = df[['City','State','Region','Sales','Profit']]
df_places.head()
```

Out[39]:

```
             City       State  Region     Sales     Profit
0       Henderson    Kentucky   South  261.9600    41.9136
1       Henderson    Kentucky   South  731.9400   219.5820
2     Los Angeles  California    West   14.6200     6.8714
3  Fort Lauderdale     Florida   South  957.5775  -383.0310
4  Fort Lauderdale     Florida   South   22.3680     2.5164
```

In [40]:

```
region_sales = df_places.groupby(['Region'], as_index=False).sum()
region_sales.sort_values(by='Sales', ascending=False, inplace=True)

# Plot the total sales geProfitnerated by each region and city
plt.figure(figsize=(10,5))
plt.bar(region_sales['Region'], region_sales['Sales'], align='center',)
plt.xlabel("Region")
plt.ylabel("Sales")
plt.title("Sales Generated by Region")
plt.xticks(rotation=90)
plt.show()
region_sales
```

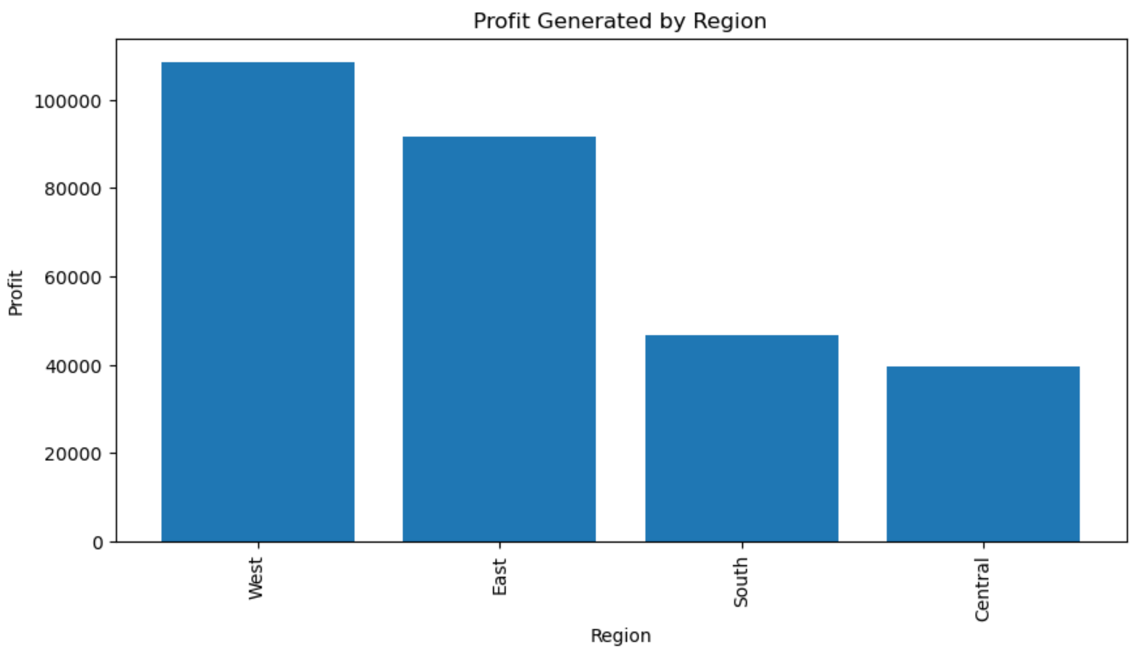


Out[40]:

	Region	Sales	Profit
3	West	725457.8245	108418.4489
1	East	678781.2400	91522.7800
0	Central	501239.8908	39706.3625
2	South	391721.9050	46749.4303

In [41]:

```
region_profit = df_places.groupby(['Region'], as_index=False).sum()
region_profit.sort_values(by='Profit', ascending=False, inplace=True)

# Plot the total sales generated by each region and city
plt.figure(figsize=(10,5))
plt.bar(region_profit['Region'], region_profit['Profit'], align='center',)
plt.xlabel("Region")
plt.ylabel("Profit")
plt.title("Profit Generated by Region")
plt.xticks(rotation=90)
plt.show()
region_profit
```

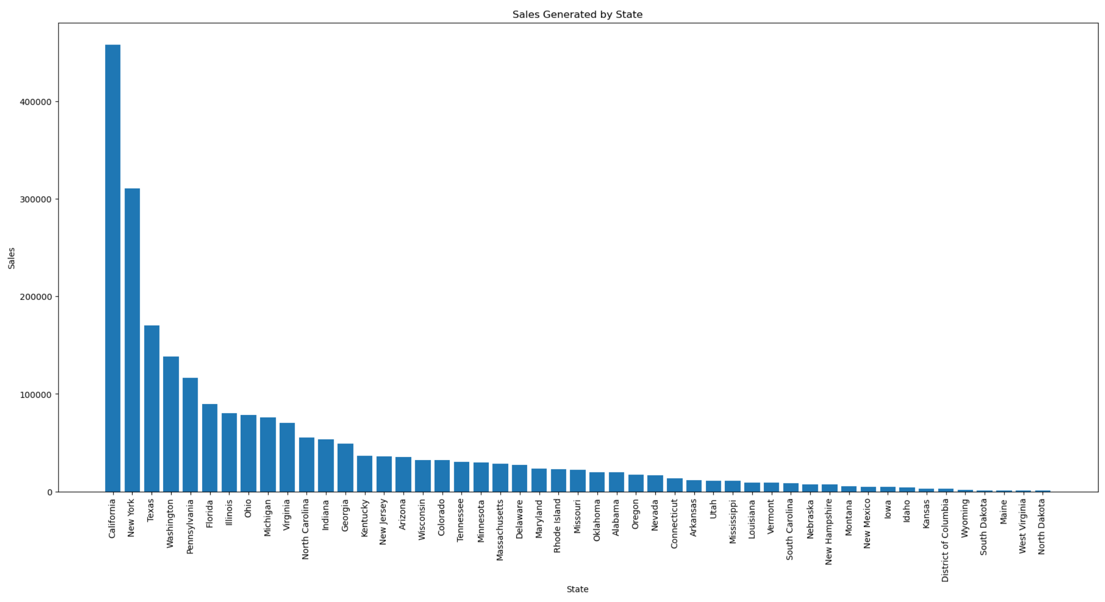


Out[41]:

	Region	Sales	Profit
3	West	725457.8245	108418.4489
1	East	678781.2400	91522.7800
2	South	391721.9050	46749.4303
0	Central	501239.8908	39706.3625

In [42]:

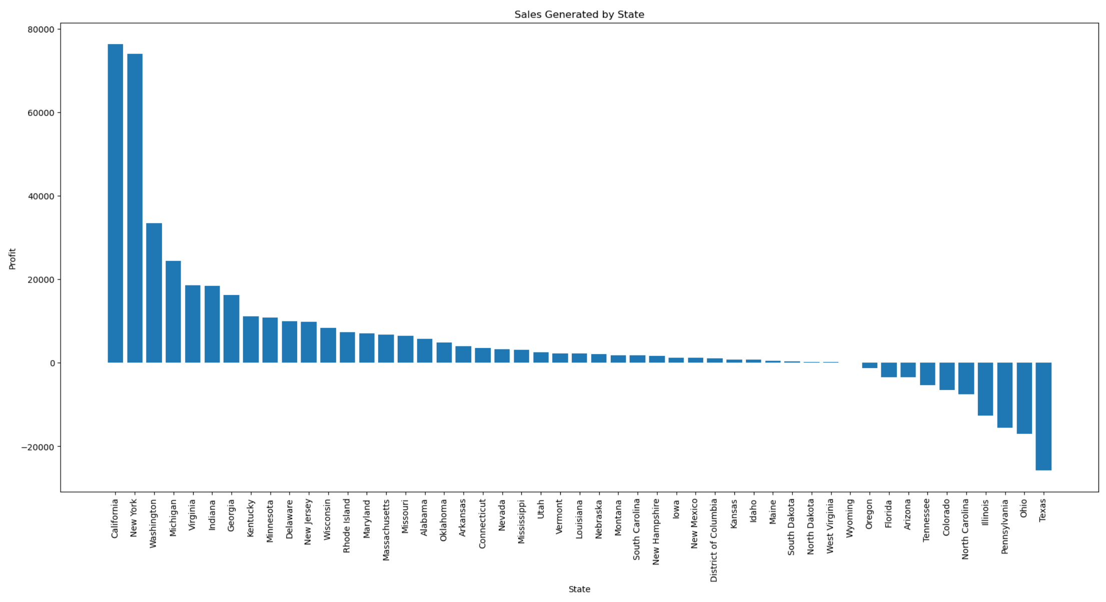
```
state_sales = df_places.groupby(['State'], as_index=False).sum()
state_sales.sort_values(by='Sales', ascending=False, inplace=True)


plt.figure(figsize=(22,10))
plt.bar(state_sales['State'], state_sales['Sales'], align='center',)
plt.xlabel("State")
plt.ylabel("Sales")
plt.title("Sales Generated by State")
plt.xticks(rotation=90)

plt.show()
state_sales
```



Out[42]:

In [43]:

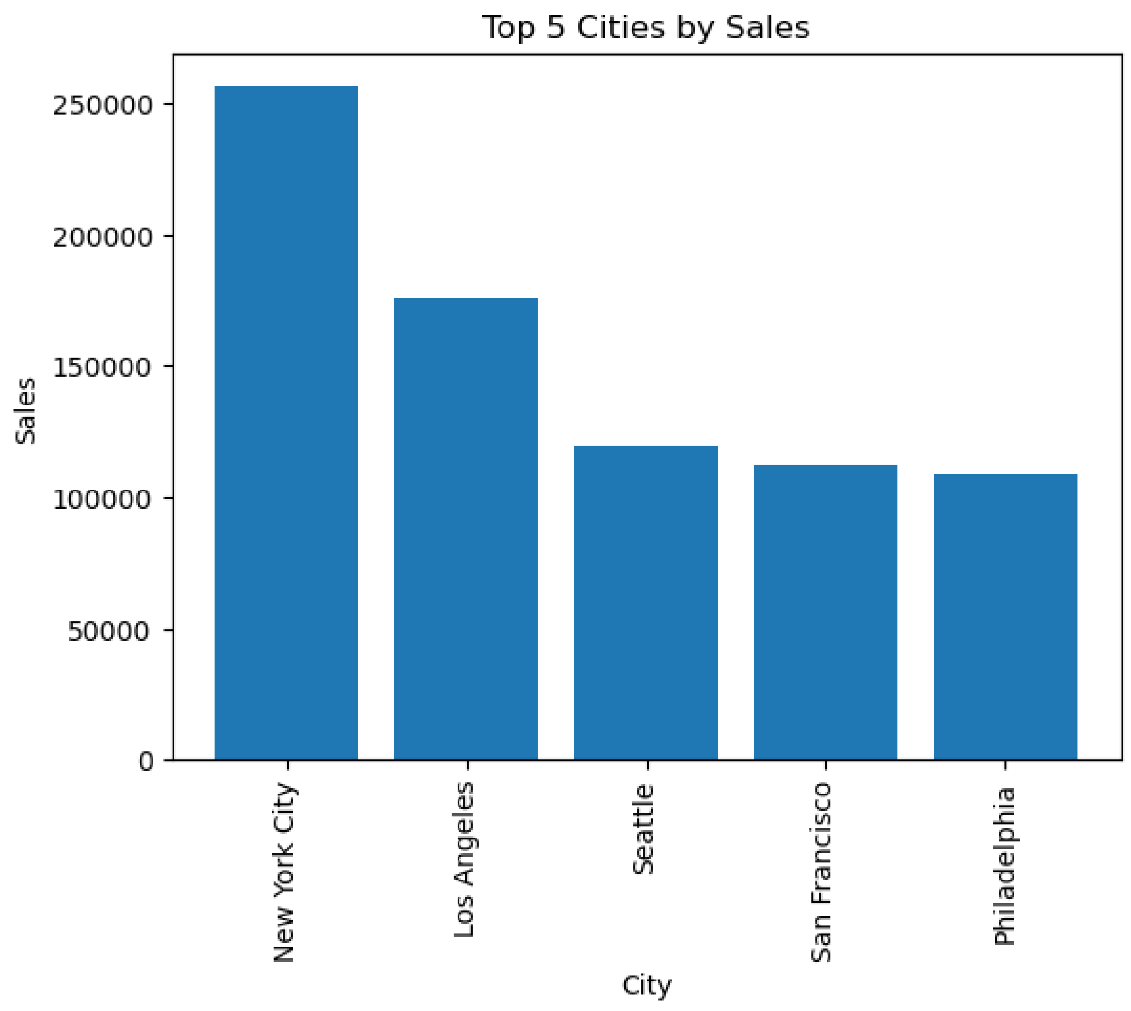
```
state_profit = df_places.groupby(['State'], as_index=False).sum()
state_profit.sort_values(by='Profit', ascending=False, inplace=True)


plt.figure(figsize=(22,10))
plt.bar(state_profit['State'], state_profit['Profit'], align='center',)
plt.xlabel("State")
plt.ylabel("Profit")
plt.title("Sales Generated by State")
plt.xticks(rotation=90)

plt.show()
state_profit
```



Out[43]:

In [44]:

```
city_sales = df_places.groupby('City', as_index=False).sum()

# Sort the data by Sales in descending order
city_sales.sort_values(by='Sales', ascending=False, inplace=True)

# Select the top 5 cities
top_5_cities_sales = city_sales.head()

plt.bar(top_5_cities_sales['City'], top_5_cities_sales['Sales'], align='center')
plt.xlabel("City")
plt.ylabel("Sales")
plt.title("Top 5 Cities by Sales")
plt.xticks(rotation=90)
plt.show()
top_5_cities_sales
```

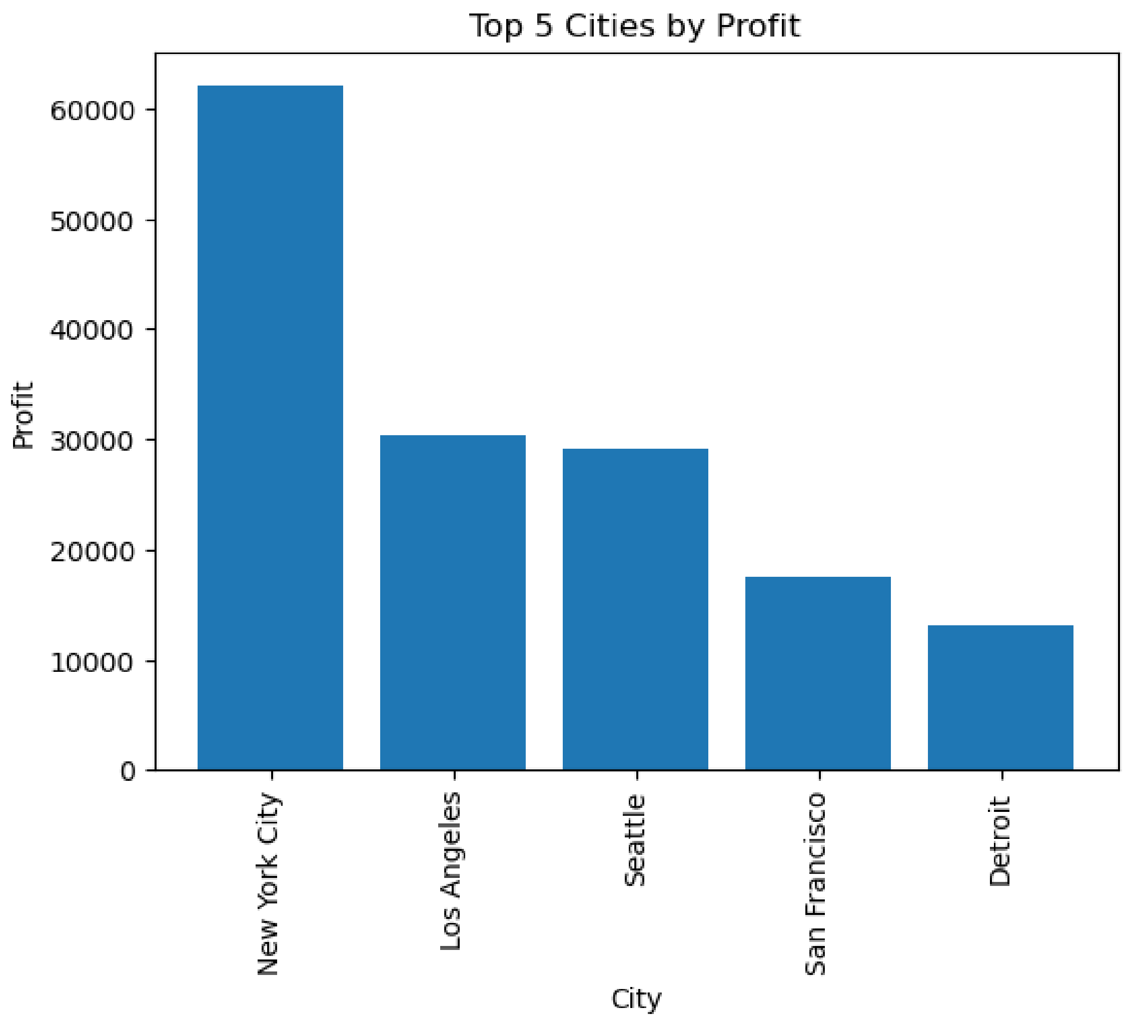


Out[44]:

	City	Sales	Profit
329	New York City	256368.161	62036.9837
266	Los Angeles	175851.341	30440.7579
452	Seattle	119540.742	29156.0967
438	San Francisco	112669.092	17507.3854
374	Philadelphia	109077.013	-13837.7674

In [45]:

```
city_profit = df_places.groupby('City', as_index=False).sum()

# Sort the data by Sales in descending order
city_profit.sort_values(by='Profit', ascending=False, inplace=True)

# Select the top 5 cities
top_5_cities_profit =city_profit.head()

plt.bar(top_5_cities_profit['City'], top_5_cities_profit['Profit'], align='center')
plt.xlabel("City")
plt.ylabel("Profit")
plt.title("Top 5 Cities by Profit")
plt.xticks(rotation=90)

plt.show()
top_5_cities_profit
```



Out[45]:

	City	Sales	Profit
329	New York City	256368.161	62036.9837
266	Los Angeles	175851.341	30440.7579
452	Seattle	119540.742	29156.0967
438	San Francisco	112669.092	17507.3854
123	Detroit	42446.944	13181.7908

In [46]:

```
df.head()
```

Out[46]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	
0	1	CA-2016-152156	2016-11-08	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Hen
1	2	CA-2016-152156	2016-11-08	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Hen
2	3	CA-2016-138688	2016-06-12	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	A
3	4	US-2015-108966	2015-10-11	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Lauc
4	5	US-2015-108966	2015-10-11	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Lauc

5 rows × 21 columns

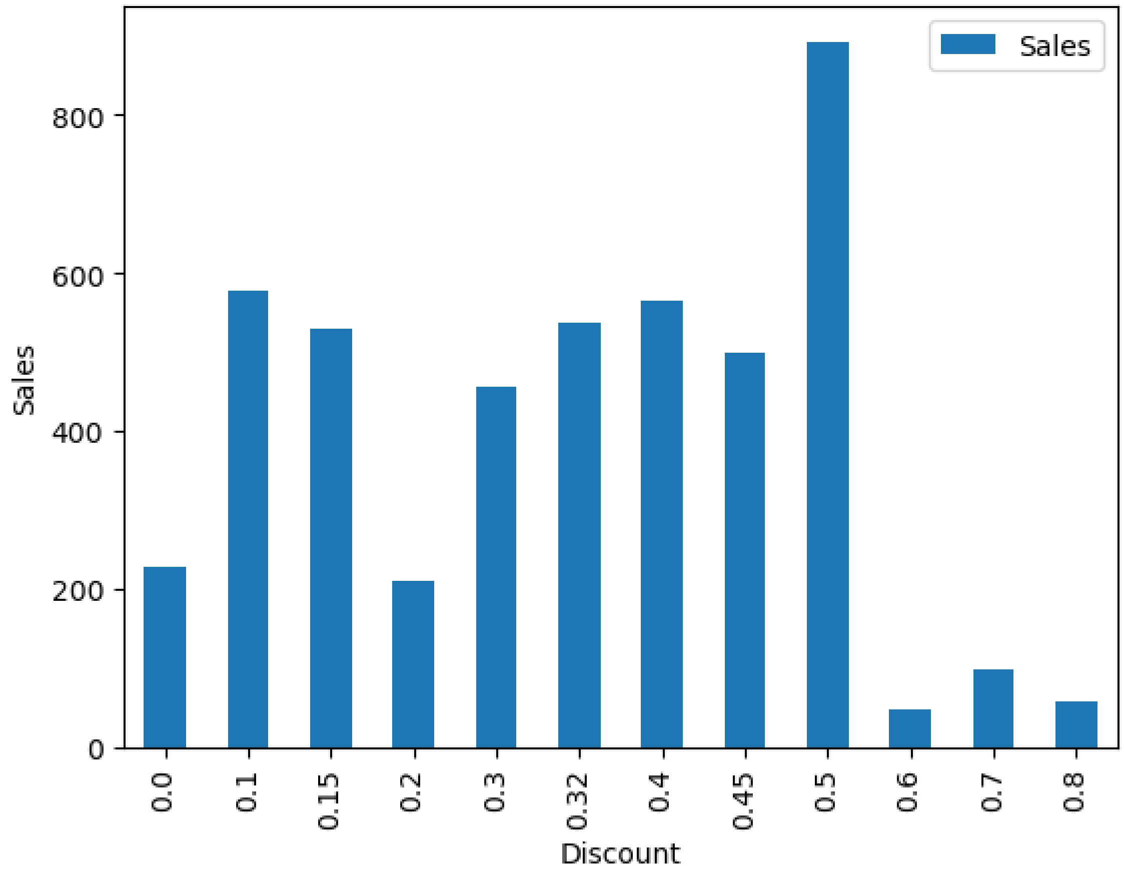
In [47]:

```
df.Discount.value_counts()
```

Out[47]:

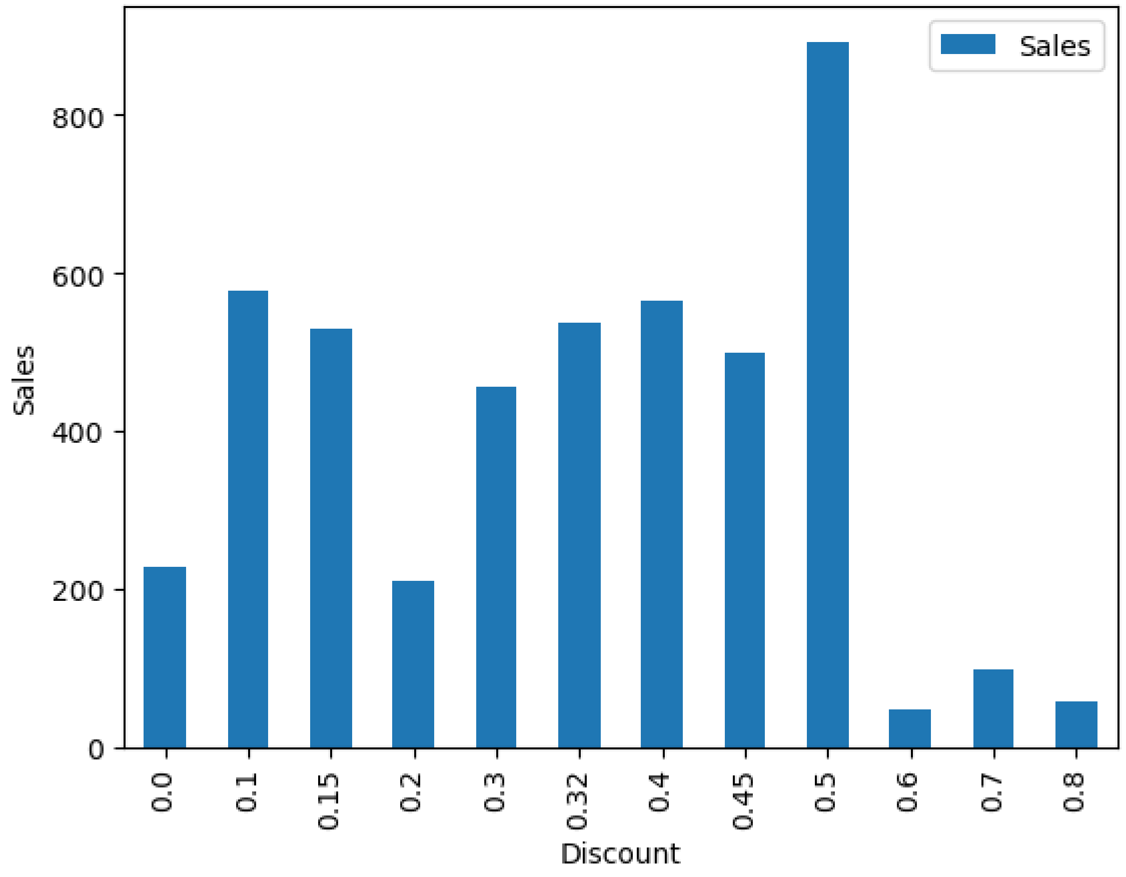
```
0.00    4798
0.20    3657
0.70     418
0.80     300
0.30     227
0.40     206
0.60     138
0.10      94
0.50      66
0.15      52
0.32      27
0.45      11
Name: Discount, dtype: int64
```

In [48]:
```
discount_group = df.groupby(["Discount"]).mean()[["Sales"]]

ax = discount_group.plot(kind="bar")

ax.set_ylabel("Sales")

plt.show()
```



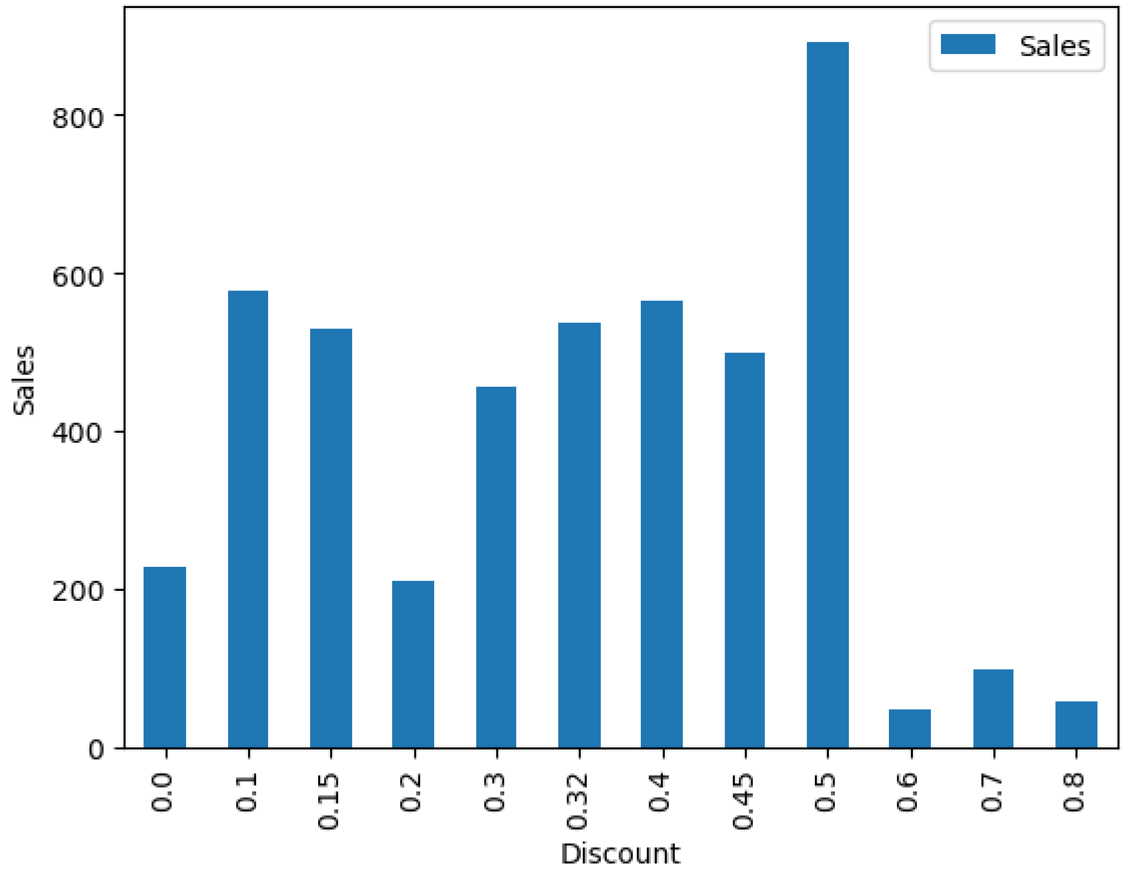
In [49]:

```
discount_group = df.groupby(["Discount"]).mean()[["Sales"]]

ax = discount_group.plot(kind="bar")

ax.set_ylabel("Sales")

plt.show()
```



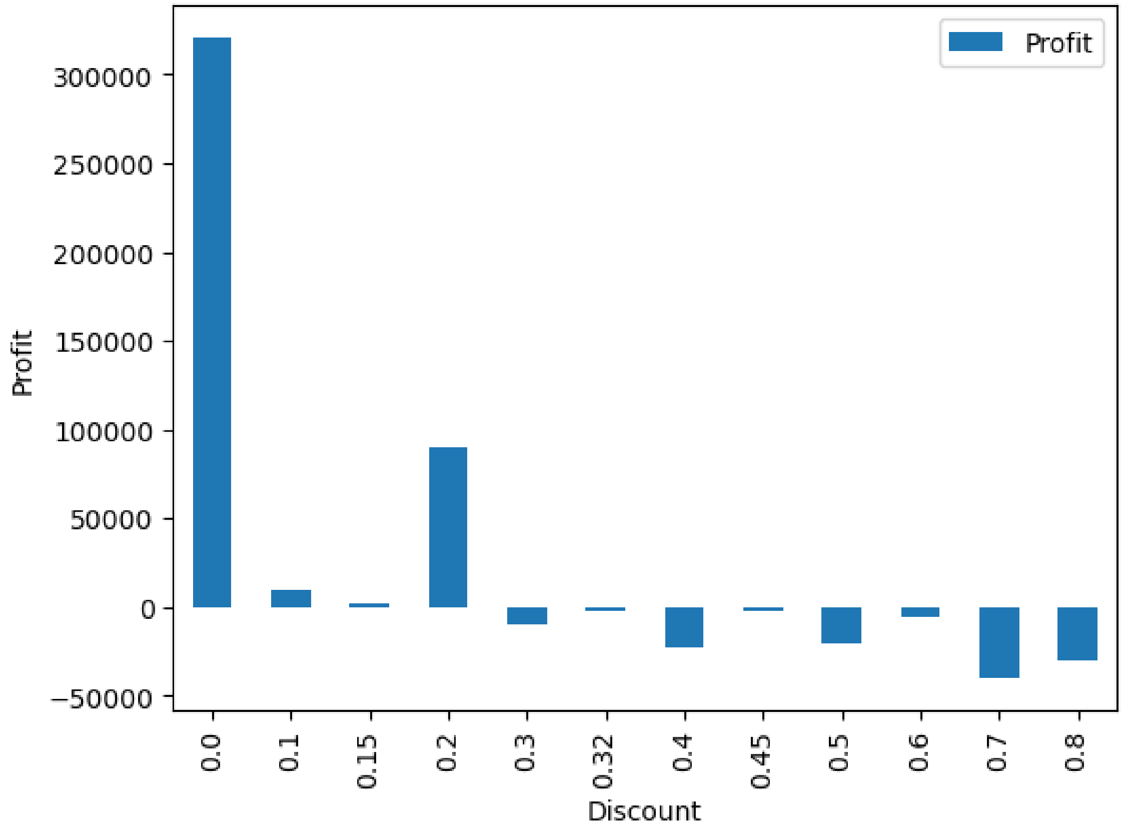
In [50]:

```
discount_group = df.groupby(["Discount"]).mean()[["Sales"]]

ax = discount_group.plot(kind="bar")

ax.set_ylabel("Sales")

plt.show()
```



In [51]:

```
discount_group = df.groupby(["Discount"]).sum()[["Profit"]]

ax = discount_group.plot(kind="bar")

ax.set_ylabel("Profit")

plt.show()
```



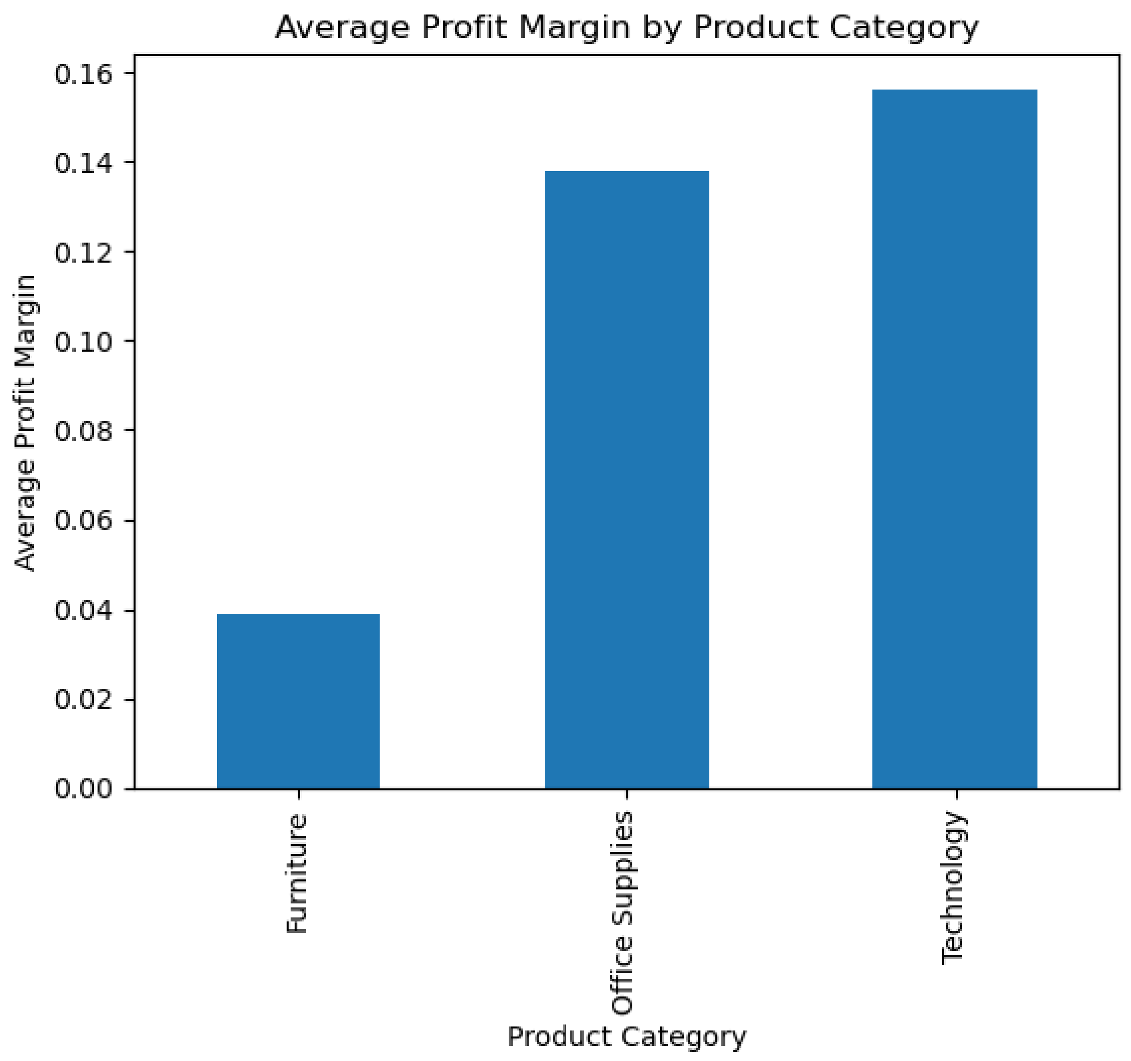
In [52]:

```
avg_profit_margin_by_category = df.groupby('Category')['Profit'].sum()

print(avg_profit_margin_by_category)
```

```
Category
Furniture           18451.2728
Office Supplies    122490.8008
Technology         145454.9481
Name: Profit, dtype: float64
```

In [53]:

```
df['Profit Margin'] = df['Profit'] / df['Sales']

# Group the data by product category and calculate the average profit margin for each ca
avg_profit_margin_by_category = df.groupby('Category')['Profit Margin'].mean()

# Plot the average profit margin for each category as a bar chart
avg_profit_margin_by_category.plot(kind='bar')

# Add a title and labels to the chart
plt.title("Average Profit Margin by Product Category")
plt.xlabel("Product Category")
plt.ylabel("Average Profit Margin")

plt.show()
```



In [54]:

```
df.head()
```

Out[54]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	
0	1	CA-2016-152156	2016-11-08	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Hen
1	2	CA-2016-152156	2016-11-08	11/11/2016	Second Class	CG-12520	Claire Gute	Consumer	United States	Hen
2	3	CA-2016-138688	2016-06-12	6/16/2016	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	A
3	4	US-2015-108966	2015-10-11	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Lauc
4	5	US-2015-108966	2015-10-11	10/18/2015	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Lauc

5 rows × 22 columns

In [55]:

```
df.Segment.value_counts()
```

Out[55]:

```
Consumer       5191
Corporate      3020
Home Office    1783
Name: Segment, dtype: int64
```

In [56]:

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

Out[56]:

```
Standard Class    5968
Second Class      1945
First Class       1538
Same Day           543
Name: Ship Mode, dtype: int64
```

In [ ]:

```
```