Superstore Sales (Kaggle)

1. Import dataset and check basic info

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv("Superstore.csv", encoding="ISO-8859-1")

print(df)
```

```
      Row ID        Order ID  Order Date   Ship Date       Ship Mode
\
0          1  CA-2016-152156   11/8/2016  11/11/2016    Second Class
1          2  CA-2016-152156   11/8/2016  11/11/2016    Second Class
2          3  CA-2016-138688   6/12/2016   6/16/2016    Second Class
3          4  US-2015-108966  10/11/2015  10/18/2015  Standard Class
4          5  US-2015-108966  10/11/2015  10/18/2015  Standard Class
...      ...             ...         ...         ...             ...
9989    9990  CA-2014-110422   1/21/2014   1/23/2014    Second Class
9990    9991  CA-2017-121258   2/26/2017    3/3/2017  Standard Class
9991    9992  CA-2017-121258   2/26/2017    3/3/2017  Standard Class
9992    9993  CA-2017-121258   2/26/2017    3/3/2017  Standard Class
9993    9994  CA-2017-119914    5/4/2017    5/9/2017    Second Class

     Customer ID     Customer Name    Segment        Country
City  \
0       CG-12520       Claire Gute   Consumer  United States
Henderson
1       CG-12520       Claire Gute   Consumer  United States
Henderson
2       DV-13045   Darrin Van Huff  Corporate  United States      Lo
s Angeles
3       SO-20335    Sean O'Donnell   Consumer  United States  Fort L
auderdale
4       SO-20335    Sean O'Donnell   Consumer  United States  Fort L
auderdale
...          ...               ...         ...           ...
...
9989    TB-21400  Tom Boeckenhauer   Consumer  United States
Miami
9990    DB-13060       Dave Brooks   Consumer  United States       C
osta Mesa
9991    DB-13060       Dave Brooks   Consumer  United States       C
osta Mesa
9992    DB-13060       Dave Brooks   Consumer  United States       C
osta Mesa
9993    CC-12220      Chris Cortes   Consumer  United States      We
stminster

      ... Postal Code  Region       Product ID         Category Sub-
Category  \
0     ...       42420   South  FUR-BO-10001798        Furniture    B
```

```
ookcases
1     ...       42420   South  FUR-CH-10000454        Furniture
Chairs
2     ...       90036    West  OFF-LA-10000240  Office Supplies
Labels
3     ...       33311   South  FUR-TA-10000577        Furniture
Tables
4     ...       33311   South  OFF-ST-10000760  Office Supplies
Storage
...   ...         ...     ...              ...              ...
...
9989  ...       33180   South  FUR-FU-10001889        Furniture  Fur
nishings
9990  ...       92627    West  FUR-FU-10000747        Furniture  Fur
nishings
9991  ...       92627    West  TEC-PH-10003645       Technology
Phones
9992  ...       92627    West  OFF-PA-10004041  Office Supplies
Paper
9993  ...       92683    West  OFF-AP-10002684  Office Supplies   Ap
pliances

                                          Product Name     Sales  Q
uantity  \
0                    Bush Somerset Collection Bookcase  261.9600
2
1     Hon Deluxe Fabric Upholstered Stacking Chairs,...  731.9400
3
2     Self-Adhesive Address Labels for Typewriters b...   14.6200
2
3         Bretford CR4500 Series Slim Rectangular Table  957.5775
5
4                        Eldon Fold 'N Roll Cart System   22.3680
2
...                                                 ...       ...
...
9989                           Ultra Door Pull Handle   25.2480
3
9990  Tenex B1-RE Series Chair Mats for Low Pile Car...   91.9600
2
9991                             Aastra 57i VoIP phone  258.5760
2
9992  It's Hot Message Books with Stickers, 2 3/4" x 5"   29.6000
4
9993  Acco 7-Outlet Masterpiece Power Center, Wihtou...  243.1600
2

      Discount    Profit
0         0.00   41.9136
1         0.00  219.5820
2         0.00    6.8714
3         0.45 -383.0310
4         0.20    2.5164
...        ...       ...
9989      0.20    4.1028
9990      0.00   15.6332
```

```
9991      0.20   19.3932
9992      0.00   13.3200
9993      0.00   72.9480

[9994 rows x 21 columns]
```

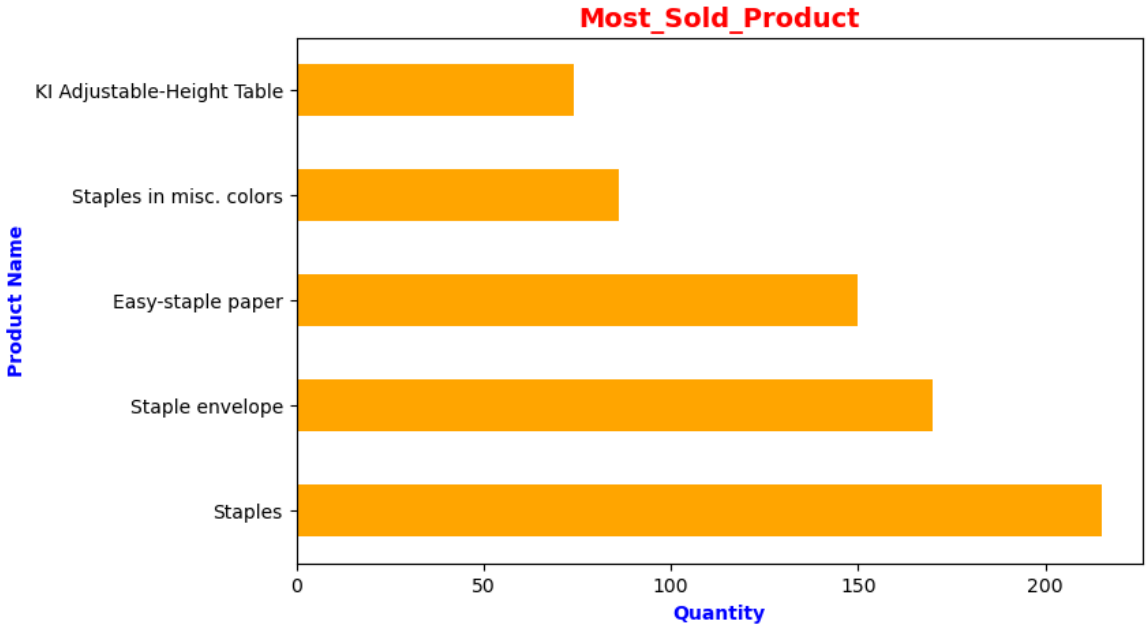
In [3]:
```
df.info()
```

```
<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
```

2. Show top 5 most sold products.

In [4]:
```
# Most sold products (by quantity)
most_sold_product = df.groupby("Product Name")["Quantity"].sum().so
print(most_sold_product)
most_sold_product.plot(kind="barh", figsize=(8,5), color='orange')
plt.title("Most_Sold_Product", size= '14', color='red', fontweight=
plt.xlabel("Quantity",color='blue',fontweight='bold')
plt.ylabel("Product Name",color='blue',fontweight='bold')
plt.show()
```

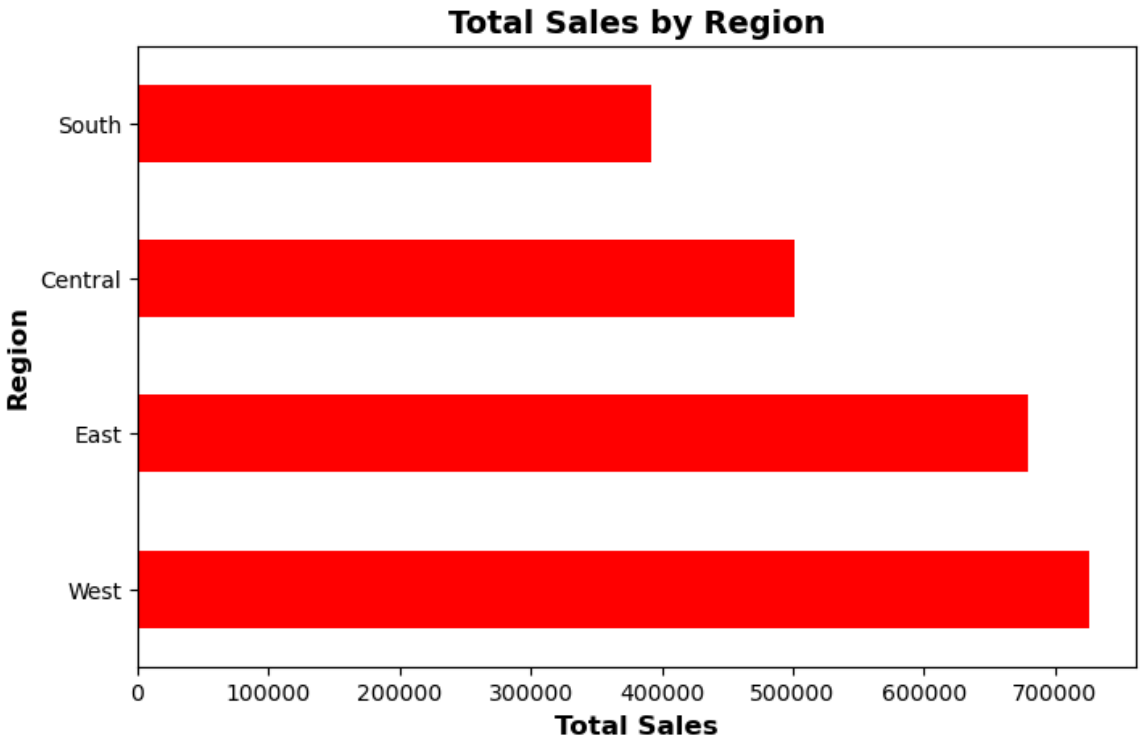
```
Product Name
Staples                        215
Staple envelope                170
Easy-staple paper              150
Staples in misc. colors         86
KI Adjustable-Height Table      74
Name: Quantity, dtype: int64
```



3. Find total sales by region.

In [5]:
```
# Correct aggregation: Sales sum by Region
total_sales_region = df.groupby("Region")["Sales"].sum().sort_value
print(total_sales_region)

# Barh plot
total_sales_region.plot(kind="barh", figsize=(8,5), color='red')

plt.title("Total Sales by Region", fontsize=14, fontweight='bold')
plt.xlabel("Total Sales", fontsize=12, fontweight='bold')
plt.ylabel("Region", fontsize=12, fontweight='bold')

plt.show()
```

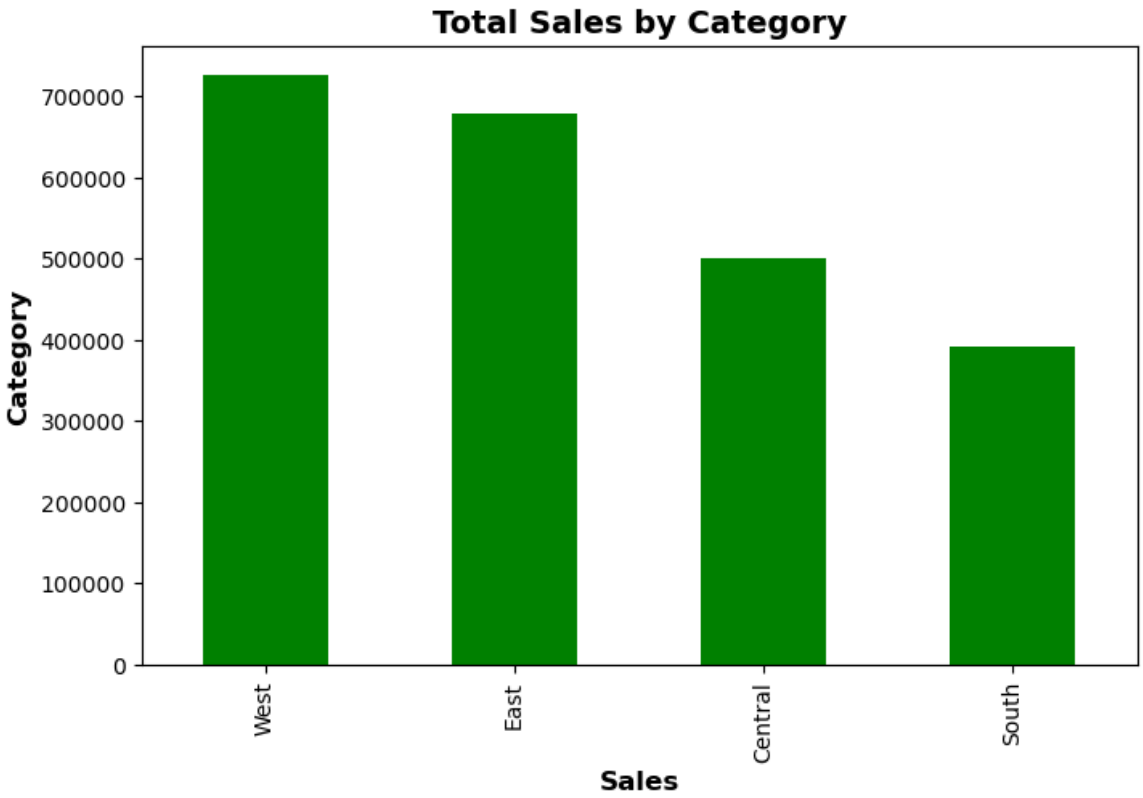
```
Region
West       725457.8245
East       678781.2400
Central    501239.8908
South      391721.9050
Name: Sales, dtype: float64
```



4. Find total sales by category.

```
In [6]: total_sales_category = df.groupby("Sales")["Category"].sum().sort_v
        print(total_sales_category)

        total_sales_region.plot(kind="bar", figsize=(8,5), color='green')

        plt.title("Total Sales by Category", fontsize=14, fontweight='bold'
        plt.xlabel("Sales", fontsize=12, fontweight='bold')
        plt.ylabel("Category", fontsize=12, fontweight='bold')

        plt.show()
```

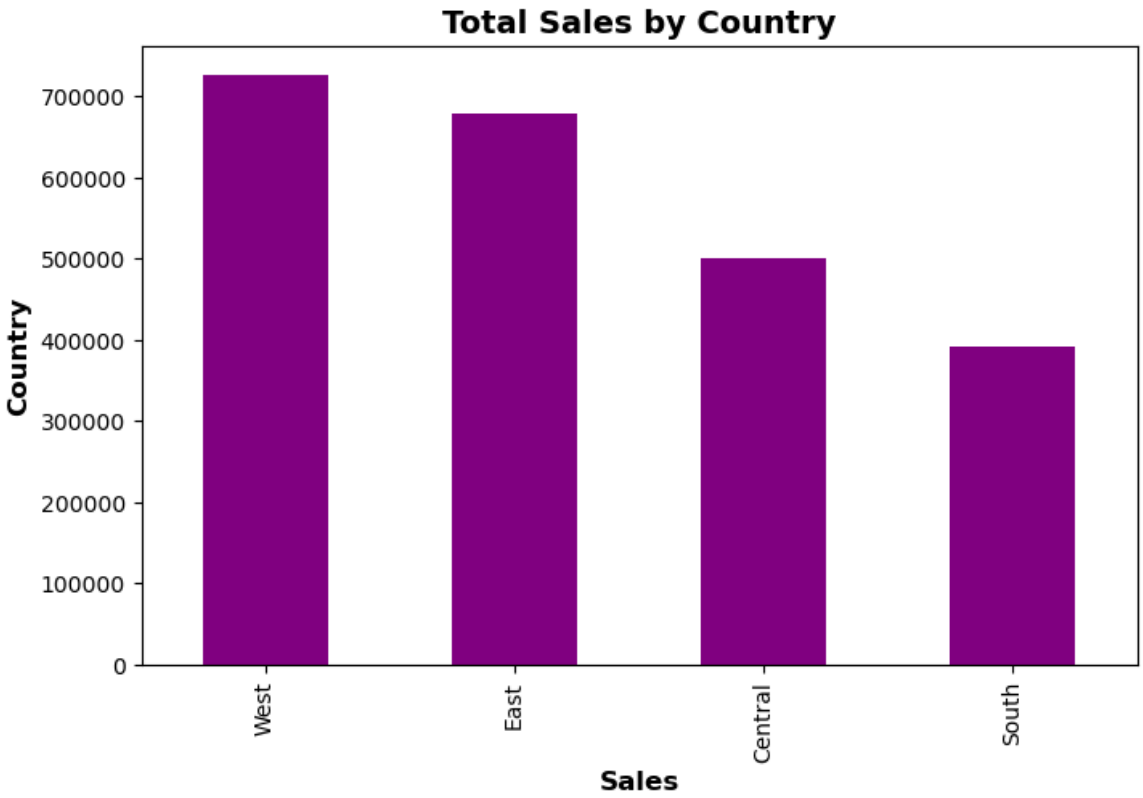
```
Sales
71.976     TechnologyTechnologyTechnologyTechnologyTechno...
47.984     TechnologyTechnologyTechnologyTechnologyTechno...
89.970     TechnologyTechnologyTechnologyTechnologyTechno...
239.970    TechnologyTechnologyTechnologyTechnologyTechno...
99.980     TechnologyTechnologyTechnologyTechnologyTechno...
                                ...
35.568                                          Furniture
239.372                                         Furniture
239.358                                         Furniture
12.070                                          Furniture
95.200                                          Furniture
Name: Category, Length: 5825, dtype: object
```



5.find total sales by country

In [7]:
```
total_sales_country = df.groupby("Sales")["Country"].sum().sort_val
print(total_sales_country)

total_sales_region.plot(kind="bar", figsize=(8,5), color='purple')

plt.title("Total Sales by Country", fontsize=14, fontweight='bold')
plt.xlabel("Sales", fontsize=12, fontweight='bold')
plt.ylabel("Country", fontsize=12, fontweight='bold')

plt.show()
```

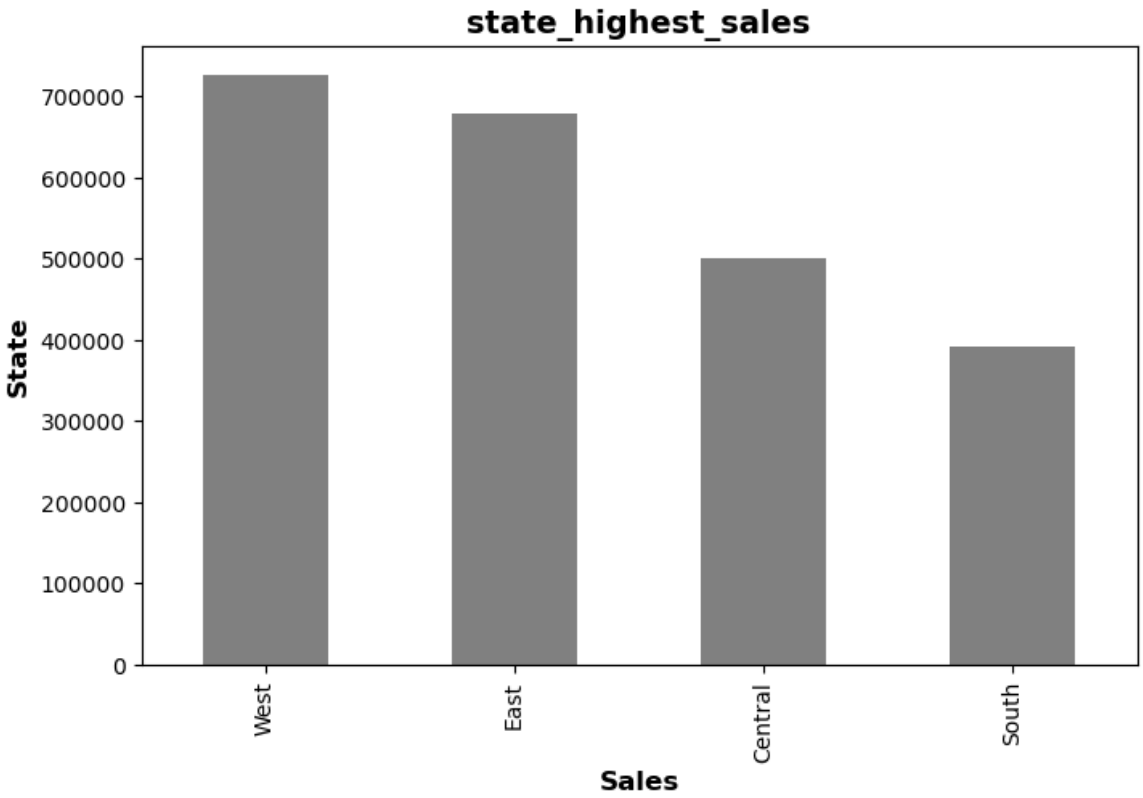
```
Sales
12.960       United StatesUnited StatesUnited StatesUnited ...
19.440       United StatesUnited StatesUnited StatesUnited ...
15.552       United StatesUnited StatesUnited StatesUnited ...
25.920       United StatesUnited StatesUnited StatesUnited ...
10.368       United StatesUnited StatesUnited StatesUnited ...
                                    ...
62.296                                          United States
62.352                                          United States
62.376                                          United States
62.400                                          United States
22638.480                                       United States
Name: Country, Length: 5825, dtype: object
```



6. Find the state with highest sales.

In [8]:
```
state_highest_sales = df.groupby("State")["Sales"].sum().sort_value
print(state_highest_sales)

total_sales_region.plot(kind="bar", figsize=(8,5), color='grey')

plt.title("state_highest_sales", fontsize=14, fontweight='bold')
plt.xlabel("Sales", fontsize=12, fontweight='bold')
plt.ylabel("State", fontsize=12, fontweight='bold')

plt.show()
```

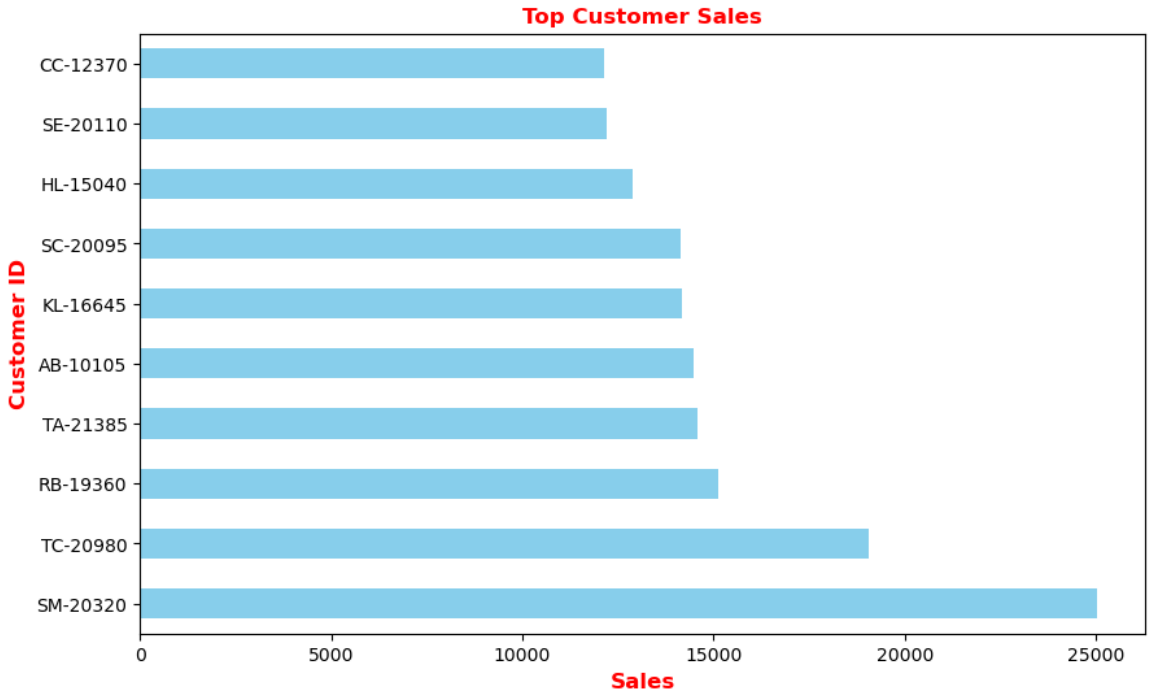
```
State
California              457687.6315
New York                310876.2710
Texas                   170188.0458
Washington              138641.2700
Pennsylvania            116511.9140
Florida                  89473.7080
Illinois                 80166.1010
Ohio                     78258.1360
Michigan                 76269.6140
Virginia                 70636.7200
North Carolina           55603.1640
Indiana                  53555.3600
Georgia                  49095.8400
Kentucky                 36591.7500
New Jersey               35764.3120
Arizona                  35282.0010
Wisconsin                32114.6100
Colorado                 32108.1180
Tennessee                30661.8730
Minnesota                29863.1500
Massachusetts            28634.4340
Delaware                 27451.0690
Maryland                 23705.5230
Rhode Island             22627.9560
Missouri                 22205.1500
Oklahoma                 19683.3900
Alabama                  19510.6400
Oregon                   17431.1500
Nevada                   16729.1020
Connecticut              13384.3570
Arkansas                 11678.1300
Utah                     11220.0560
Mississippi              10771.3400
Louisiana                 9217.0300
Vermont                   8929.3700
South Carolina            8481.7100
Nebraska                  7464.9300
New Hampshire             7292.5240
Montana                   5589.3520
New Mexico                4783.5220
Iowa                      4579.7600
Idaho                     4382.4860
Kansas                    2914.3100
District of Columbia      2865.0200
Wyoming                   1603.1360
South Dakota              1315.5600
Maine                     1270.5300
West Virginia             1209.8240
North Dakota               919.9100
Name: Sales, dtype: float64
```



7. Find top 10 customers by sales amount.

In [9]:
```
top_customer_sales=df.groupby("Customer ID")["Sales"].sum().sort_va
print(top_customer_sales)

top_customer_sales.plot(kind="barh",figsize=(10,6), color="skyblue"
plt.title("Top Customer Sales", fontsize=12, fontweight='bold', col
plt.xlabel("Sales", fontsize=12, fontweight='bold', color="red")
plt.ylabel("Customer ID", fontsize=12, fontweight='bold', color="re
plt.show()
```

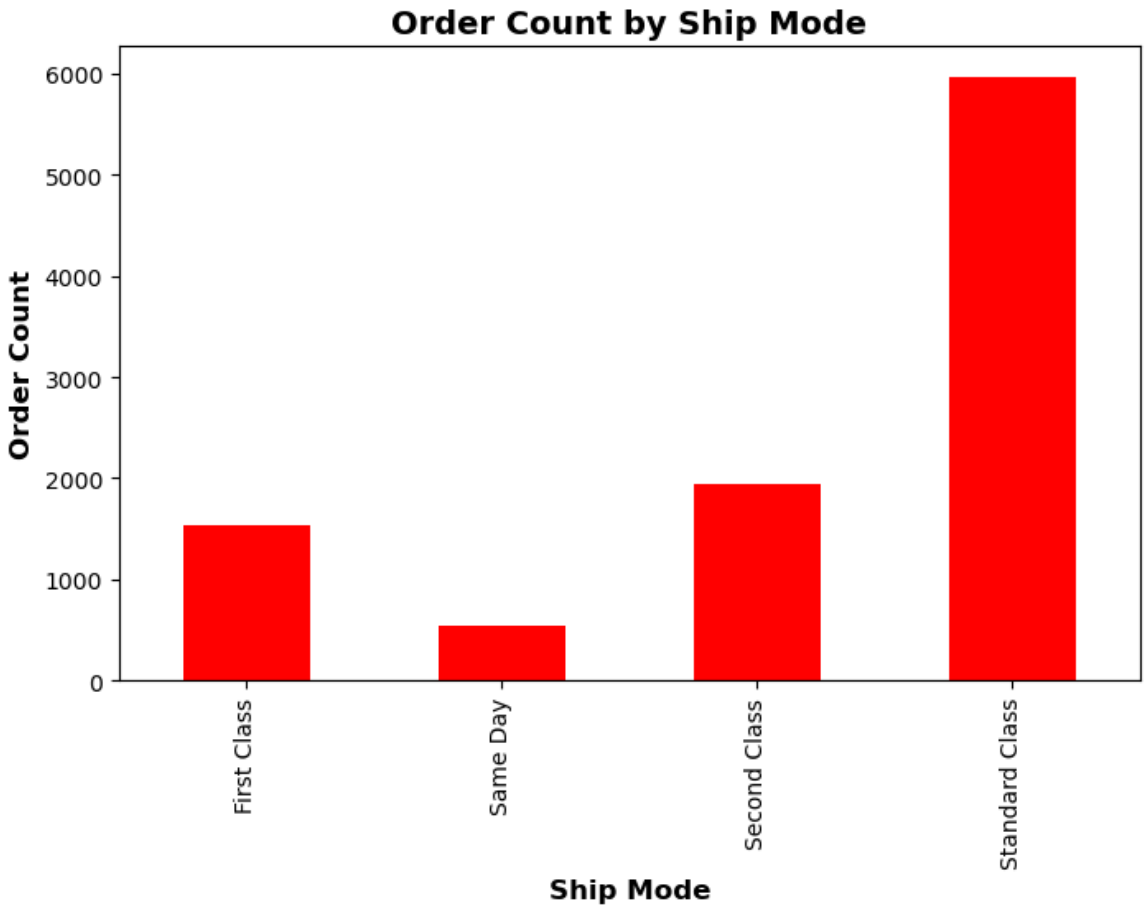
```
Customer ID
SM-20320    25043.050
TC-20980    19052.218
RB-19360    15117.339
TA-21385    14595.620
AB-10105    14473.571
KL-16645    14175.229
SC-20095    14142.334
HL-15040    12873.298
SE-20110    12209.438
CC-12370    12129.072
Name: Sales, dtype: float64
```



8. Calculate order count by ship mode.

In [10]:
```
order_count_ship_mode = df.groupby("Ship Mode")["Order ID"].count()
print(order_count_ship_mode)

# Plot
order_count_ship_mode.plot(kind="bar", figsize=(8,5), color="red")
plt.title("Order Count by Ship Mode", fontsize=14, fontweight="bold
plt.xlabel("Ship Mode", fontsize=12, fontweight="bold")
plt.ylabel("Order Count", fontsize=12, fontweight="bold")
plt.show()
```

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



9. Find month with highest sales.

In [11]:
```
# Convert 'Order Date' to datetime if not already
df['Order Date'] = pd.to_datetime(df['Order Date'])
print(df['Order Date'])
```

```
0      2016-11-08
1      2016-11-08
2      2016-06-12
3      2015-10-11
4      2015-10-11
          ...
9989   2014-01-21
9990   2017-02-26
9991   2017-02-26
9992   2017-02-26
9993   2017-05-04
Name: Order Date, Length: 9994, dtype: datetime64[ns]
```

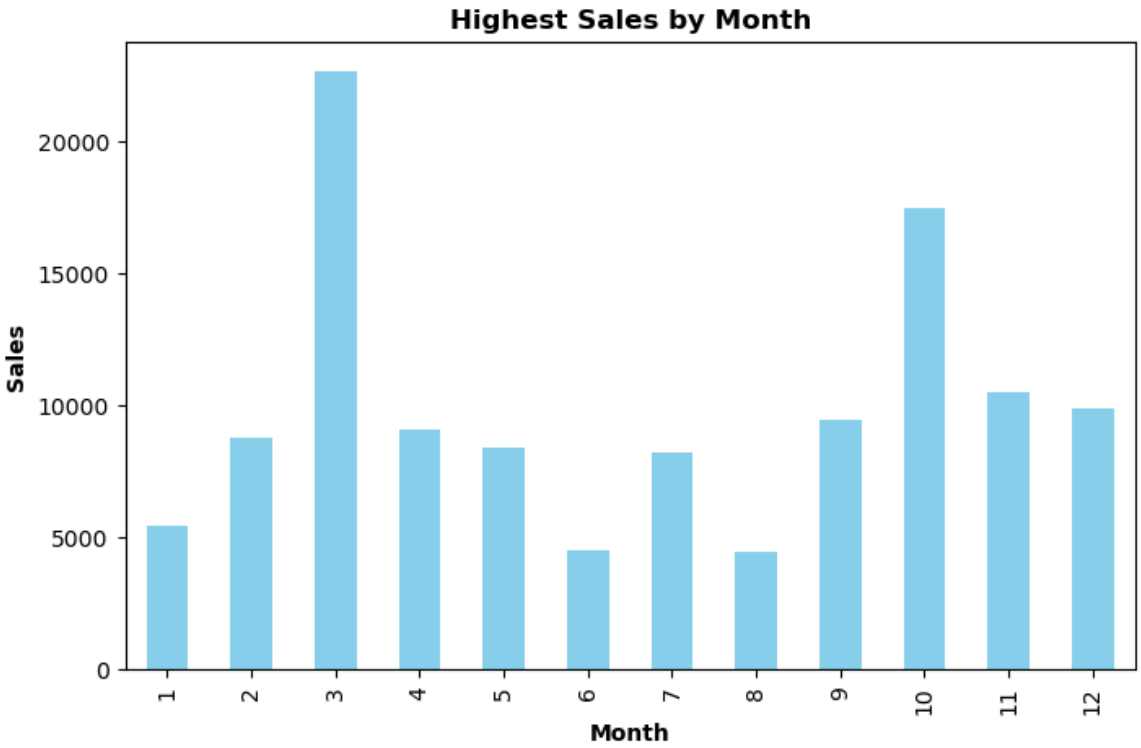
In [12]:
```
# Create a 'month' column
df['month'] = df['Order Date'].dt.month
print(df['month'])
```

```
0       11
1       11
2        6
3       10
4       10
        ..
9989     1
9990     2
9991     2
9992     2
9993     5
Name: month, Length: 9994, dtype: int32
```

In [13]:
```
month_highest_sales = df.groupby('month')['Sales'].max()
print(month_highest_sales)
```

```
month
1      5443.960
2      8749.950
3     22638.480
4      9099.930
5      8399.976
6      4476.800
7      8187.650
8      4416.174
9      9449.950
10    17499.950
11    10499.970
12     9892.740
Name: Sales, dtype: float64
```

In [14]:
```
month_highest_sales.plot(kind='bar', figsize=(8,5), color='skyblue'

plt.title("Highest Sales by Month", fontweight='bold')
plt.xlabel("Month", fontweight='bold')
plt.ylabel("Sales",fontweight='bold')
plt.show()
```



10. find year with highest sales

In [15]:
```
df['Order Date']=pd.to_datetime(df['Order Date'])
print(df['Order Date'])
```

```
0      2016-11-08
1      2016-11-08
2      2016-06-12
3      2015-10-11
4      2015-10-11
          ...
9989   2014-01-21
9990   2017-02-26
9991   2017-02-26
9992   2017-02-26
9993   2017-05-04
Name: Order Date, Length: 9994, dtype: datetime64[ns]
```

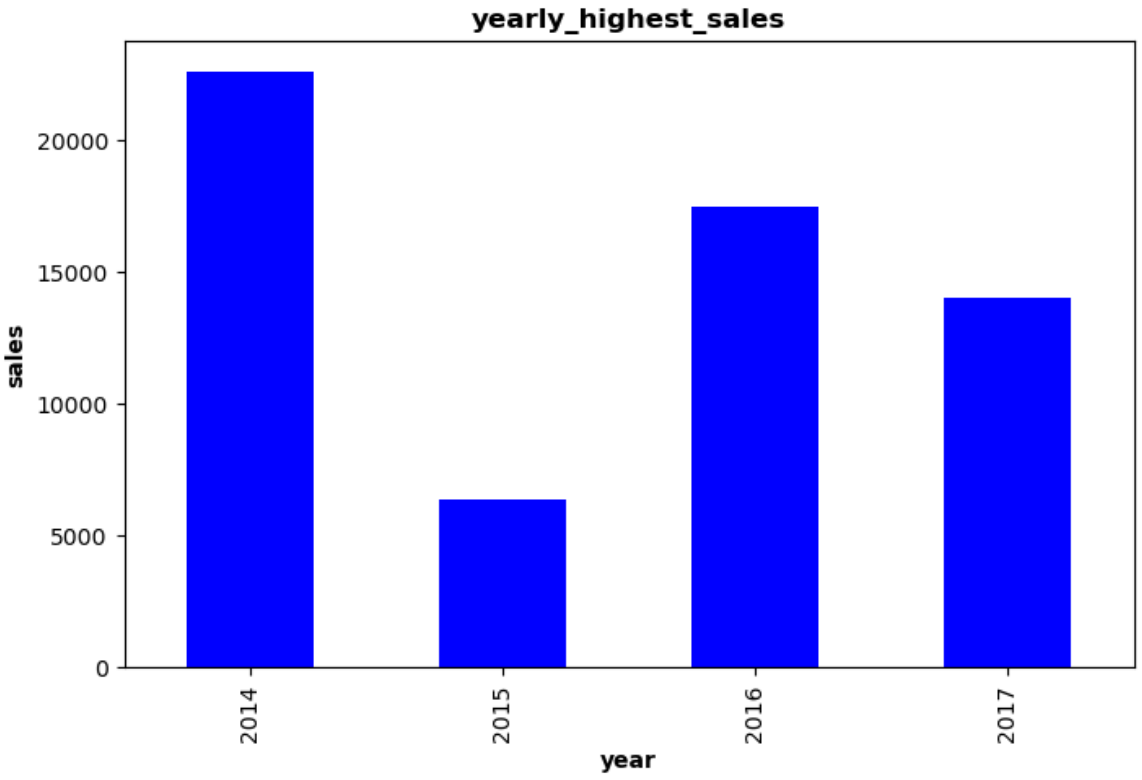
In [16]:
```
df['year']=df['Order Date'].dt.year
print(df['year'])
```

```
0       2016
1       2016
2       2016
3       2015
4       2015
        ...
9989    2014
9990    2017
9991    2017
9992    2017
9993    2017
Name: year, Length: 9994, dtype: int32
```

In [17]:
```
yearly_highest_sales=df.groupby('year')['Sales'].max()
print(yearly_highest_sales)
```

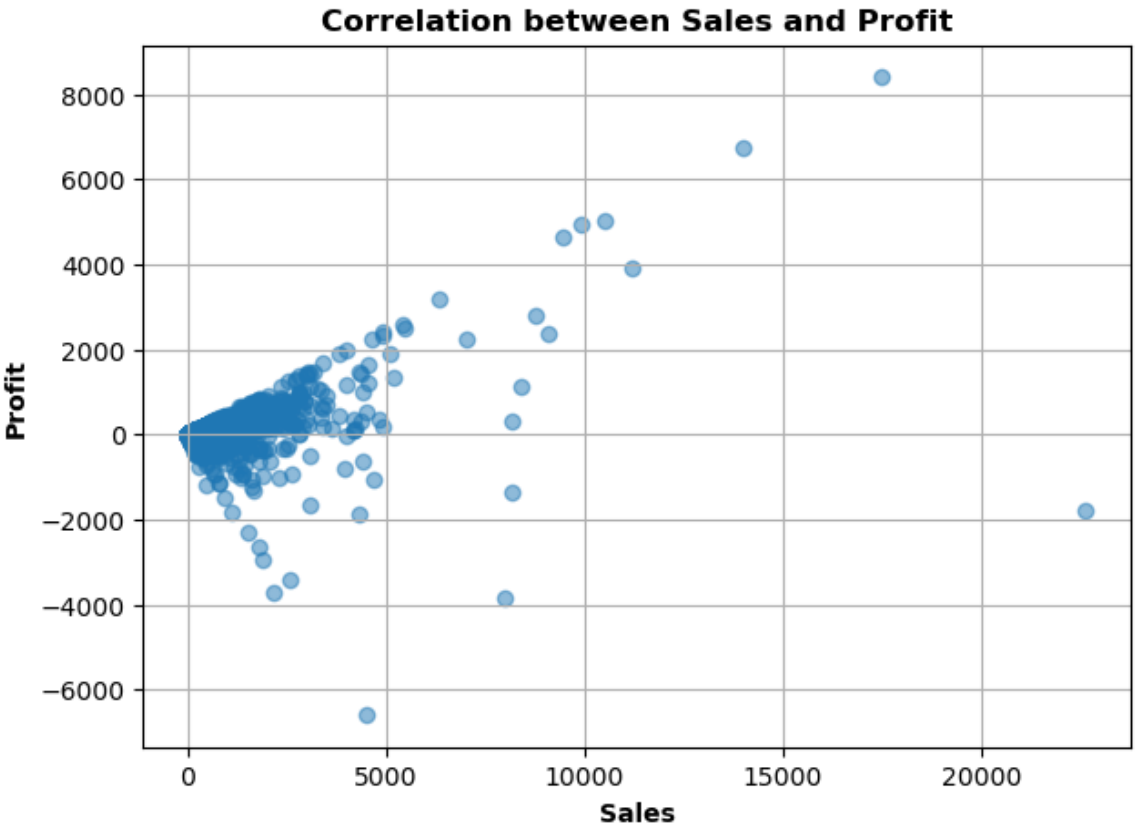
```
year
2014    22638.48
2015     6354.95
2016    17499.95
2017    13999.96
Name: Sales, dtype: float64
```

In [18]:
```
yearly_highest_sales.plot(kind='bar', figsize=(8,5), color='blue')
plt.title('yearly_highest_sales', fontweight='bold')
plt.xlabel("year",fontweight='bold')
plt.ylabel('sales', fontweight='bold')
plt.show()
```



10. Find correlation between sales and profit.

In [19]:
```
df['Sales'].corr(df['Profit'])
plt.figure(figsize=(7,5))
plt.scatter(df['Sales'], df['Profit'], alpha=0.5)

plt.title("Correlation between Sales and Profit", fontweight='bold'
plt.xlabel("Sales", fontweight='bold')
plt.ylabel("Profit", fontweight='bold')
plt.grid(True)
plt.show()
```



11. Group data by category and sub-category to calculate total sales.

```
In [20]: total_sales=df.groupby(['Category', 'Sub-Category'])['Sales'].sum()
         print(total_sales)
```

```
Category         Sub-Category
Furniture        Bookcases       114879.9963
                 Chairs          328449.1030
                 Furnishings      91705.1640
                 Tables          206965.5320
Office Supplies  Appliances      107532.1610
                 Art              27118.7920
                 Binders         203412.7330
                 Envelopes        16476.4020
                 Fasteners         3024.2800
                 Labels           12486.3120
                 Paper            78479.2060
                 Storage         223843.6080
                 Supplies         46673.5380
Technology       Accessories     167380.3180
                 Copiers         149528.0300
                 Machines        189238.6310
                 Phones          330007.0540
Name: Sales, dtype: float64
```

12. Find percentage contribution of each category in total sales.

```
In [21]: # Group sales by category
         category_sales = df.groupby('Category')['Sales'].sum()
         print(category_sales)
```

```
Category
Furniture          741999.7953
Office Supplies    719047.0320
Technology         836154.0330
Name: Sales, dtype: float64
```

In [22]:
```
# Calculate percentage contribution
percentage = (category_sales / category_sales.sum()) * 100

print(percentage)
```

```
Category
Furniture          32.300171
Office Supplies    31.301008
Technology         36.398821
Name: Sales, dtype: float64
```

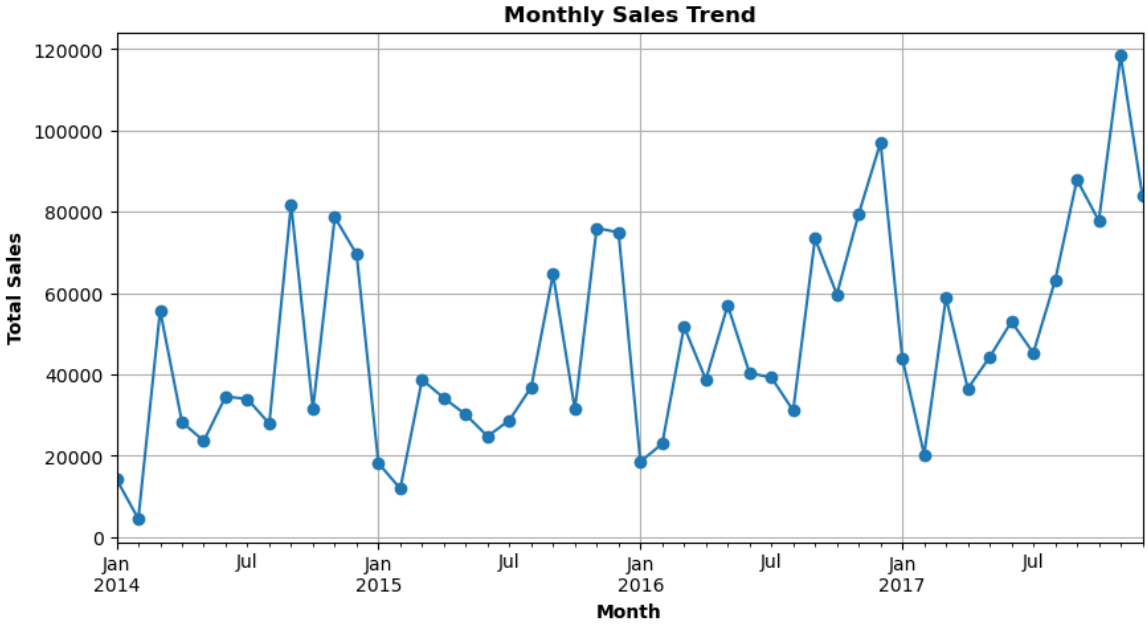
13. Create pivot table showing sales by region and category.

In [23]:
```
pivot_table = pd.pivot_table(df,
                             values='Sales',
                             index='Region',
                             columns='Category',
                             aggfunc='sum')

print(pivot_table)
```

```
Category    Furniture  Office Supplies  Technology
Region
Central   163797.1638       167026.415  170416.312
East      208291.2040       205516.055  264973.981
South     117298.6840       125651.313  148771.908
West      252612.7435       220853.249  251991.832
```

14. Show trend of sales (monthly line chart).

In [24]:
```
# Ensure Order Date is datetime
df['Order Date'] = pd.to_datetime(df['Order Date'])

# Group sales by month
monthly_sales = df.groupby(df['Order Date'].dt.to_period('M'))['Sal

# Plot line chart
monthly_sales.plot(kind='line', figsize=(10,5), marker='o')
plt.title("Monthly Sales Trend", fontweight='bold')
plt.xlabel("Month",fontweight='bold')
plt.ylabel("Total Sales", fontweight='bold')
plt.grid(True)
plt.show()
```



15. Identify products sold at a loss (profit < 0).

In [25]:
```
loss_products = df[df['Profit'] < 0]
print(loss_products)
```

```
      Row ID        Order ID Order Date   Ship Date       Ship Mode
\
3          4  US-2015-108966 2015-10-11  10/18/2015  Standard Class
14        15  US-2015-118983 2015-11-22  11/26/2015  Standard Class
15        16  US-2015-118983 2015-11-22  11/26/2015  Standard Class
23        24  US-2017-156909 2017-07-16   7/18/2017    Second Class
27        28  US-2015-150630 2015-09-17   9/21/2015  Standard Class
...      ...             ...        ...         ...             ...
9920    9921  CA-2016-149272 2016-03-15   3/19/2016  Standard Class
9921    9922  CA-2014-111360 2014-11-24  11/30/2014  Standard Class
9931    9932  CA-2015-104948 2015-11-13  11/17/2015  Standard Class
9937    9938  CA-2016-164889 2016-06-03    6/6/2016    Second Class
9962    9963  CA-2015-168088 2015-03-19   3/22/2015     First Class

     Customer ID     Customer Name      Segment        Country  \
3       SO-20335    Sean O'Donnell     Consumer  United States
14      HP-14815     Harold Pawlan  Home Office  United States
15      HP-14815     Harold Pawlan  Home Office  United States
23      SF-20065   Sandra Flanagan     Consumer  United States
27      TB-21520   Tracy Blumstein     Consumer  United States
...          ...               ...         ...            ...
9920    MY-18295   Muhammed Yedwab    Corporate  United States
9921    AT-10435       Alyssa Tate  Home Office  United States
9931    KH-16510     Keith Herrera     Consumer  United States
9937    CP-12340    Christine Phan    Corporate  United States
9962    CM-12655  Corinna Mitchell  Home Office  United States

                City  ...       Product ID         Category Sub-Cat
egory  \
3     Fort Lauderdale  ...  FUR-TA-10000577        Furniture       T
ables
14         Fort Worth  ...  OFF-AP-10002311  Office Supplies   Appli
```

```
ances
15         Fort Worth  ...  OFF-BI-10000756  Office Supplies      Bi
nders
23       Philadelphia  ...  FUR-CH-10002774        Furniture       C
hairs
27       Philadelphia  ...  FUR-BO-10004834        Furniture    Book
cases
...               ...  ...              ...              ...
...
9920            Bryan  ...  OFF-BI-10004233  Office Supplies      Bi
nders
9921            Akron  ...  OFF-BI-10003350  Office Supplies      Bi
nders
9931   San Bernardino  ...  FUR-BO-10004357        Furniture    Book
cases
9937      Los Angeles  ...  FUR-TA-10001676        Furniture       T
ables
9962          Houston  ...  FUR-BO-10004218        Furniture    Book
cases

                                          Product Name      Sales Q
uantity  \
3          Bretford CR4500 Series Slim Rectangular Table   957.5775
5
14    Holmes Replacement Filter for HEPA Air Cleaner...    68.8100
5
15     Storex DuraTech Recycled Plastic Frosted Binders     2.5440
3
23                   Global Deluxe Stacking Chair, Gray    71.3720
2
27    Riverside Palais Royal Lawyers Bookcase, Royal...  3083.4300
7
...                                                 ...        ...
...
9920  GBC Pre-Punched Binding Paper, Plastic, White,...    22.3860
7
9921                     Acco Expandable Hanging Binders     5.7420
3
9931     O'Sullivan Living Dimensions 3-Shelf Bookcases   683.3320
4
9937       Hon 61000 Series Interactive Training Tables    71.0880
2
9962  Bush Heritage Pine Collection 5-Shelf Bookcase...   383.4656
4

     Discount     Profit  month  year
3        0.45  -383.0310     10  2015
14       0.80  -123.8580     11  2015
15       0.80    -3.8160     11  2015
23       0.30    -1.0196      7  2017
27       0.50 -1665.0522      9  2015
...       ...         ...    ...   ...
9920     0.80   -35.8176      3  2016
9921     0.70    -4.5936     11  2014
9931     0.15   -40.1960     11  2015
9937     0.20    -1.7772      6  2016
```

```
9962     0.32   -67.6704      3  2015

[1871 rows x 23 columns]
```

16. Find top 5 products with highest profit.

```
In [26]: highest_product_profit = (df.groupby('Product Name')['Profit'].sum(

         print(highest_product_profit)
```

```
Product Name
Canon imageCLASS 2200 Advanced Copier
25199.9280
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manu
al 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
Name: Profit, dtype: float64
```

17. Count number of unique customers.

```
In [27]: df['Customer ID'].nunique()
```

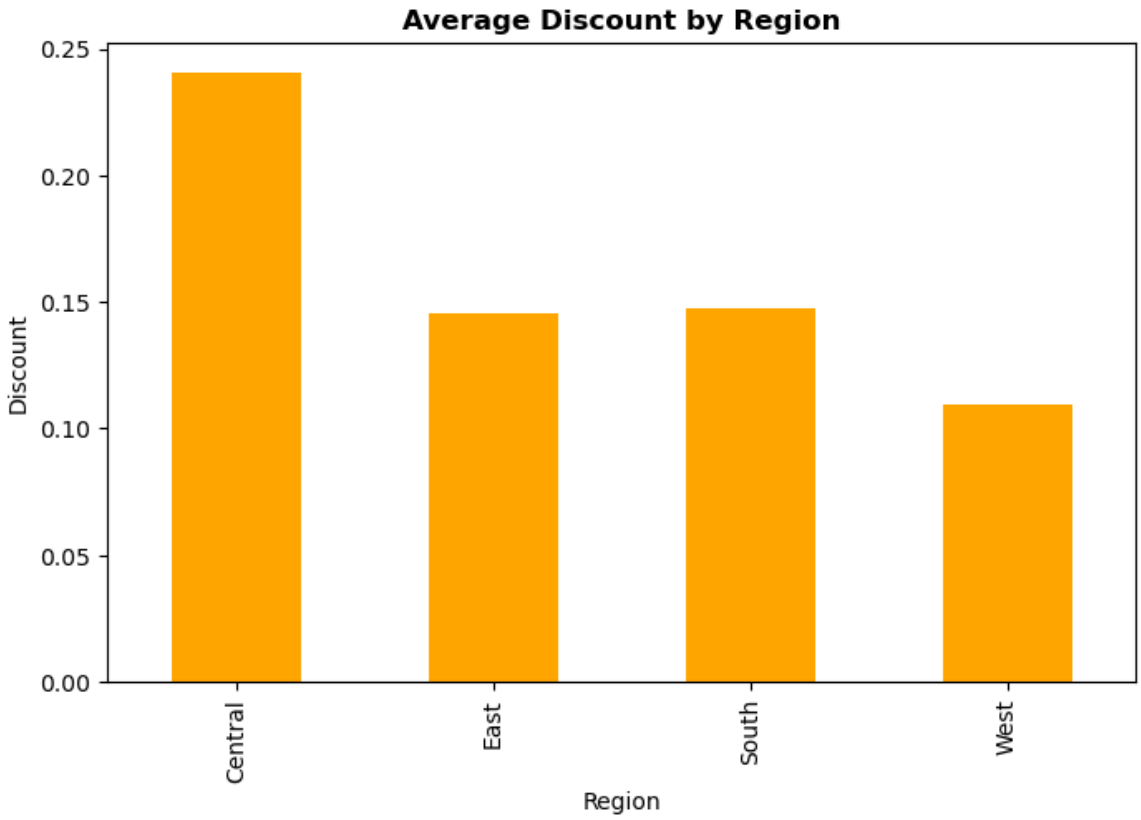
```
Out[27]: 793
```

18. Find average discount given by region.

```
In [28]: avg_discount_region = df.groupby('Region')['Discount'].mean()
         print(avg_discount_region)
```

```
Region
Central    0.240353
East       0.145365
South      0.147253
West       0.109335
Name: Discount, dtype: float64
```

```
In [36]: avg_discount_region.plot(kind='bar', figsize=(8,5), color='orange')
         plt.title('Average Discount by Region', fontweight='bold')
         plt.xlabel('Region')
         plt.ylabel('Discount')
         plt.show()
```



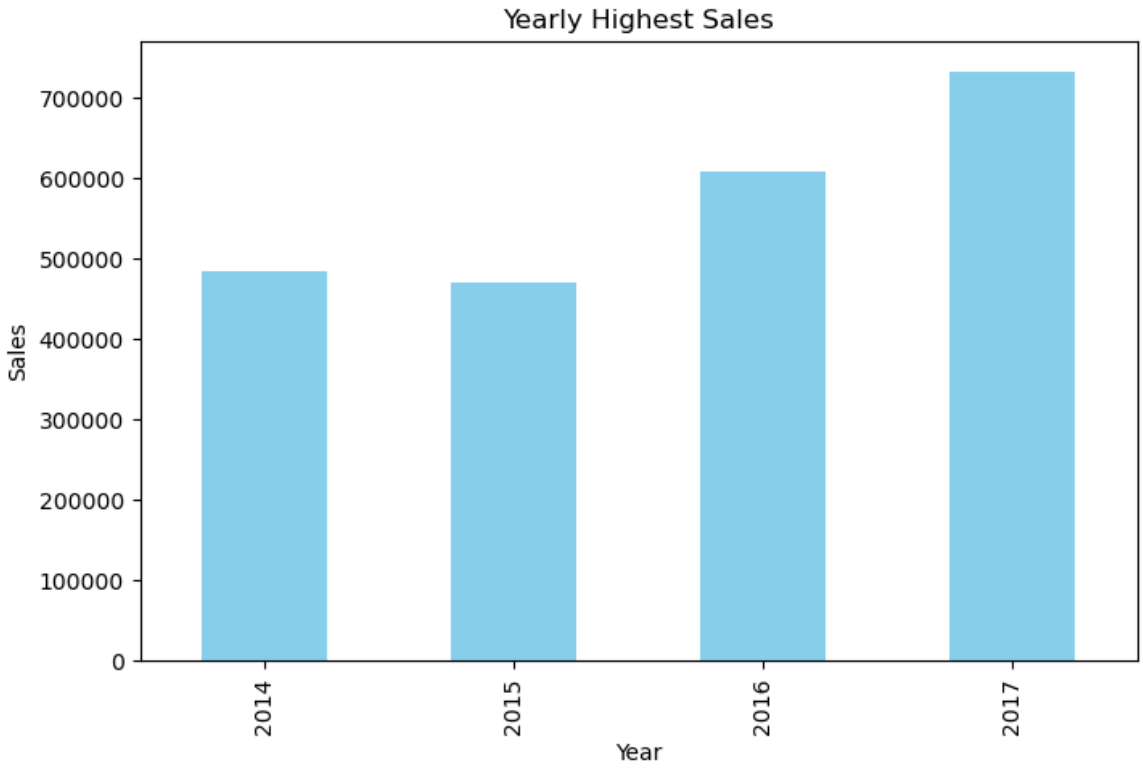
19. Find year with highest sales.

In [30]:
```
df['Order Date']=pd.to_datetime(df['Order Date'])
print(df['Order Date'])

df['Year'] = df['Order Date'].dt.year          # Extract year
print(df['Year'])

Yearly_Sales = df.groupby('Year')['Sales'].sum()
print('Yearly_Sales')

yearly_sales = df.groupby('Year')['Sales'].sum()
```

```
0      2016-11-08
1      2016-11-08
2      2016-06-12
3      2015-10-11
4      2015-10-11
          ...
9989   2014-01-21
9990   2017-02-26
9991   2017-02-26
9992   2017-02-26
9993   2017-05-04
Name: Order Date, Length: 9994, dtype: datetime64[ns]
0       2016
1       2016
2       2016
3       2015
4       2015
        ...
9989    2014
9990    2017
9991    2017
9992    2017
9993    2017
Name: Year, Length: 9994, dtype: int32
Yearly_Sales
```

In [39]:
```
yearly_highest_sales= Yearly_Sales.idxmax
print(yearly_highest_sales)

yearly_sales.plot(kind='bar', figsize=(8,5), color='skyblue')
plt.title('Yearly Highest Sales')
plt.xlabel('Year')
plt.ylabel('Sales')
plt.show()
```

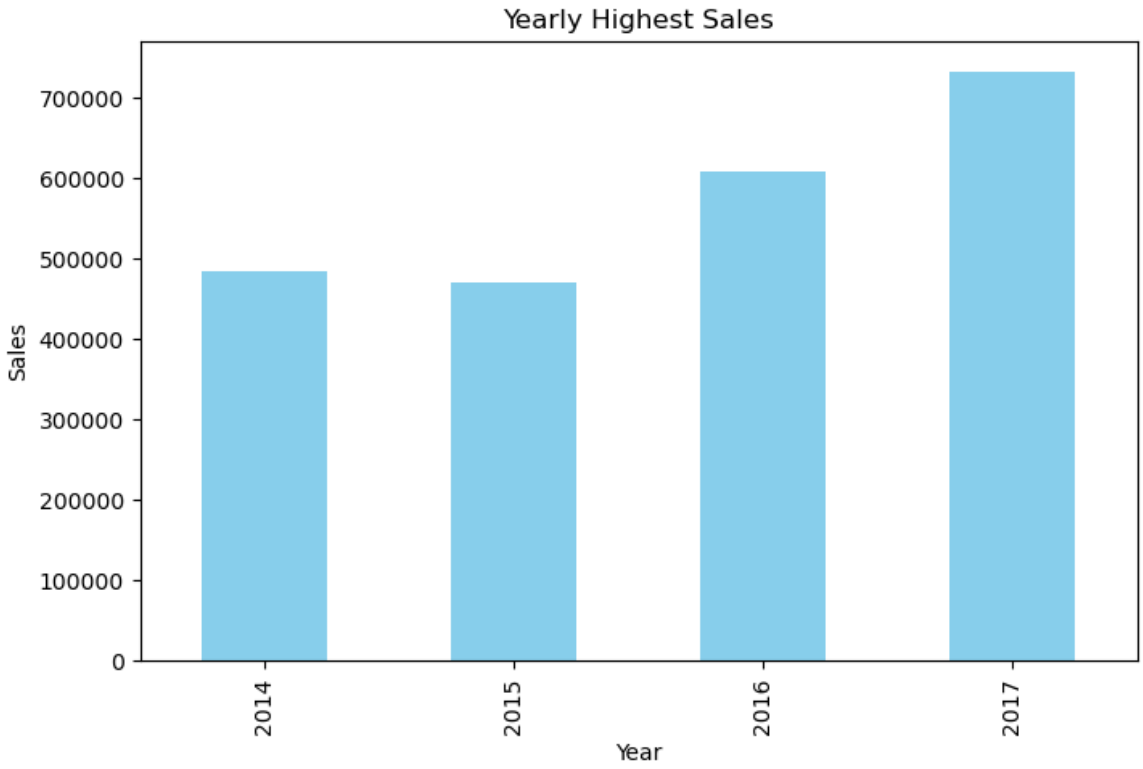
```
<bound method Series.idxmax of Year
2014    484247.4981
2015    470532.5090
2016    609205.5980
2017    733215.2552
Name: Sales, dtype: float64>
```



20. find year with lowest sale

In [43]:
```
yearly_lowest_sales= Yearly_Sales.idxmin
print(yearly_lowest_sales)

yearly_sales.plot(kind='bar', figsize=(8,5), color='skyblue')
plt.title('Yearly Highest Sales')
plt.xlabel('Year')
plt.ylabel('Sales')
plt.show()
```

```
<bound method Series.idxmin of Year
2014    484247.4981
2015    470532.5090
2016    609205.5980
2017    733215.2552
Name: Sales, dtype: float64>
```



20. Create customer segmentation based on sales (High/Medium/Low).

In [49]:
```
# Step 1: Calculate total sales per customer
customer_sales = df.groupby('Customer ID')['Sales'].sum().reset_ind

# Step 2: Define segmentation bins
# Example: High (> 70th percentile), Medium (40–70th), Low (< 40th)
customer_sales['Segment'] = pd.qcut(customer_sales['Sales'],
                                    q=[0, 0.4, 0.7, 1.0],
                                    labels=['Low', 'Medium', 'High'

print(customer_sales.head())

# Step 3: Count customers in each segment
segment_counts = customer_sales['Segment'].value_counts()
print(segment_counts)
customer_sales.set_index('Customer ID')['Sales'].plot(kind='line',
plt.title('Total Sales per Customer')
plt.xlabel('Customer ID')
plt.ylabel('Sales')
plt.show()
```

```
  Customer ID     Sales Segment
0    AA-10315  5563.560    High
1    AA-10375  1056.390     Low
2    AA-10480  1790.512  Medium
3    AA-10645  5086.935    High
4    AB-10015   886.156     Low
Segment
Low       317
Medium    238
High      238
Name: count, dtype: int64
```