Import necessary libraries

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

import warnings
warnings.filterwarnings('ignore')
```

Load & Explore data

In [2]:
```
data = pd.read_csv("D:\Projects\Python\All\project 1\Sales Data.csv")
data.head()
```

Out[2]:

	Unnamed: 0	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour
0	0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30 00:01:00	136 Church St, New York City, NY 10001	12	1700.00	New York City	0
1	1	295666	LG Washing Machine	1	600.00	2019-12-29 07:03:00	562 2nd St, New York City, NY 10001	12	600.00	New York City	7
2	2	295667	USB-C Charging Cable	1	11.95	2019-12-12 18:21:00	277 Main St, New York City, NY 10001	12	11.95	New York City	18
3	3	295668	27in FHD Monitor	1	149.99	2019-12-22 15:13:00	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco	15
4	4	295669	USB-C Charging Cable	1	11.95	2019-12-18 12:38:00	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta	12

In [3]:
```
data.drop('Unnamed: 0',axis=1,inplace=True)
data.head()
```

Out[3]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30 00:01:00	136 Church St, New York City, NY 10001	12	1700.00	New York City	0
1	295666	LG Washing Machine	1	600.00	2019-12-29 07:03:00	562 2nd St, New York City, NY 10001	12	600.00	New York City	7
2	295667	USB-C Charging Cable	1	11.95	2019-12-12 18:21:00	277 Main St, New York City, NY 10001	12	11.95	New York City	18
3	295668	27in FHD Monitor	1	149.99	2019-12-22 15:13:00	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco	15
4	295669	USB-C Charging Cable	1	11.95	2019-12-18 12:38:00	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta	12

In [4]:
```
data.info()
data.shape
data.duplicated().sum()
#class 'pandas.core.frame.DataFrame'>
#RangeIndex: 186850 entries, 0 to 186849
#Data columns (total 10 columns):
#Column Non-Null Count Dtype
#--- ------ -------------- -----
# 0 Order ID 185686 non-null int64
# 1 Product 185686 non-null object
# 2 Quantity Ordered 185686 non-null int64
# 3 Price Each 185686 non-null float64
# 4 Order Date 185686 non-null object
# 5 Purchase Address 185686 non-null object
# 6 Month 185686 non-null int64
# 7 Sales 185686 non-null float64
# 8 City 185686 non-null object
# 9 Hour 185686 non-null int64
#dtypes: float64(2), int64(4), object(4)
#memory usage: 14.2+ MB
```

In [5]:
```
data.describe()
```

Out[5]:

	Order ID	Quantity Ordered	Price Each	Month	Sales	Hour
count	185686.000000	185686.000000	185686.000000	185686.000000	185686.000000	185686.000000
mean	230417.568379	1.124383	184.399735	7.339140	185.490917	14.412981
std	51912.737110	0.442733	332.731330	3.502996	332.919771	5.425434
min	141234.000000	1.000000	2.990000	1.000000	2.990000	0.000000
25%	185831.250000	1.000000	11.950000	4.000000	11.950000	11.000000
50%	230497.500000	1.000000	14.950000	7.000000	14.950000	15.000000
75%	275535.750000	1.000000	150.000000	10.000000	150.000000	19.000000
max	319670.000000	9.000000	1700.000000	12.000000	3400.000000	23.000000

In [6]:
```
data.dtypes
```

Out[6]:
```
Order ID            int64
Product            object
Quantity Ordered    int64
Price Each        float64
Month               int64
Sales             float64
Purchase Address   object
Month              object
Sales              object
City               object
Hour               object
dtype: object
```

In [7]:
```
data[['Order Date', 'Order Time']] = data['Order Date'].str.split(' ', expand = True)
data['Order Date', 'Order Time'].head()
```

Out[7]:

	Order Date	Order Time
0	2019-12-30	00:01:00
1	2019-12-29	07:03:00
2	2019-12-12	18:21:00
3	2019-12-22	15:13:00
4	2019-12-18	12:38:00

In [8]:
```
data['Order Time'] = data['Order Time'].str.replace(':00', '')
data['Order Time'].head()
```

Out[8]:
```
0    00:01
1    07:03
2    18:21
3    15:13
4    12:38
Name: Order Time, dtype: object
```

In [9]:
```
data.head()
```

Out[9]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour	Order Time
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30	136 Church St, New York City, NY 10001	12	1700.00	New York City	0	00:01
1	295666	LG Washing Machine	1	600.00	2019-12-29	562 2nd St, New York City, NY 10001	12	600.00	New York City	7	07:03
2	295667	USB-C Charging Cable	1	11.95	2019-12-12	277 Main St, New York City, NY 10001	12	11.95	New York City	18	18:21
3	295668	27in FHD Monitor	1	149.99	2019-12-22	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco	15	15:13
4	295669	USB-C Charging Cable	1	11.95	2019-12-18	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta	12	12:38

In [10]:
```
data['Purchase Address'].apply(lambda x: x[:].split(',')[1].strip())
data.head()
```

Out[10]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour	Order Time
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30	136 Church St, New York City, NY 10001	12	1700.00	New York City	0	00:01
1	295666	LG Washing Machine	1	600.00	2019-12-29	562 2nd St, New York City, NY 10001	12	600.00	New York City	7	07:03
2	295667	USB-C Charging Cable	1	11.95	2019-12-12	277 Main St, New York City, NY 10001	12	11.95	New York City	18	18:21
3	295668	27in FHD Monitor	1	149.99	2019-12-22	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco	15	15:13
4	295669	USB-C Charging Cable	1	11.95	2019-12-18	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta	12	12:38

In [11]:
```
def get_city(address):
    return address.split(',')[1]
def get_state(address):
    return address.split(',')[2].split(' ')[1]

data['City'] = data['Purchase Address'].apply(lambda x: f"{get_city(x)} ({get_state(x)})")
data.head()
```

Out[11]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour	Order Time
0	295665	Macbook Pro Laptop	1	1700.00	2019-12-30	136 Church St, New York City, NY 10001	12	1700.00	New York City (NY)	0	00:01
1	295666	LG Washing Machine	1	600.00	2019-12-29	562 2nd St, New York City, NY 10001	12	600.00	New York City (NY)	7	07:03
2	295667	USB-C Charging Cable	1	11.95	2019-12-12	277 Main St, New York City, NY 10001	12	11.95	New York City (NY)	18	18:21
3	295668	27in FHD Monitor	1	149.99	2019-12-22	410 6th St, San Francisco, CA 94016	12	149.99	San Francisco (CA)	15	15:13
4	295669	USB-C Charging Cable	1	11.95	2019-12-18	43 Hill St, Atlanta, GA 30301	12	11.95	Atlanta (GA)	12	12:38

In [12]:
```
data=data[['Order ID', 'Order Date', 'Month', 'Order Time', 'Hour', 'Product', 'Quantity Ordered', 'Price Each', 'Sales', 'Purchase Address', 'City']]
data.head()
```

Out[12]:

	Order ID	Order Date	Month	Order Time	Hour	Product	Quantity Ordered	Price Each	Sales	Purchase Address	City
0	295665	2019-12-30	12	00:01	0	Macbook Pro Laptop	1	1700.00	1700.00	136 Church St, New York City, NY 10001	New York City (NY)
1	295666	2019-12-29	12	07:03	7	LG Washing Machine	1	600.00	600.00	562 2nd St, New York City, NY 10001	New York City (NY)
2	295667	2019-12-12	12	18:21	18	USB-C Charging Cable	1	11.95	11.95	277 Main St, New York City, NY 10001	New York City (NY)
3	295668	2019-12-22	12	15:13	15	27in FHD Monitor	1	149.99	149.99	410 6th St, San Francisco, CA 94016	San Francisco (CA)
4	295669	2019-12-18	12	12:38	12	USB-C Charging Cable	1	11.95	11.95	43 Hill St, Atlanta, GA 30301	Atlanta (GA)

Exploratory Analysis

In [13]:
```
total_sales_per_month = data.groupby('Month')['Sales'].sum().reset_index()
plt.figure(figsize=(12,6))
ax = sns.barplot(x="Month", y="Sales",data=total_sales_per_month, errwidth=0, palette=sns.color_palette("viridis"))
plt.xlabel("Month")
plt.ylabel("Total Sales in USD ($)")
plt.title("Total Sales by Months")
for p in ax.patches:
    ax.annotate('{:.0f}'.format(p.get_height()),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', xytext=(0, 5), textcoords='offset points')

plt.show()
```


In [14]:
```
plt.figure(figsize=(12, 6))
months_counts=data['Month'].value_counts().sort_index()
ax = sns.barplot(x=months_counts.index, y=months_counts.values, palette=sns.color_palette("viridis"))
plt.xlabel("Month")
plt.ylabel("Orders")
ax.set_title('Total Orders by Months')
for p in ax.patches:
    ax.annotate('{:.0f}'.format(p.get_height()),
                (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', xytext=(0, 5), textcoords='offset points')

plt.show()
```


In [15]:
```
fig, ax = plt.subplots(1, 2, figsize=(16, 5))

total_sales_per_city = data.groupby('City')['Sales'].sum().reset_index()
total_sales_per_city = total_sales_per_city.sort_values(by='Sales', ascending=False)
sns.barplot(x="Sales", y="City", data=total_sales_per_city, palette=sns.color_palette('plasma'),ax=ax[0])
ax[0].set_xlabel('Sales in USD ($)')
ax[0].set_ylabel('City')
ax[0].set_title('Total Sales by Cities')
for p in ax[0].patches:
    ax[0].annotate('{:.0f}'.format(p.get_width()), (p.get_width(), p.get_y()+0.6))

total_orders_per_city = data.groupby('City')['Sales'].count().reset_index()
total_orders_per_city = total_orders_per_city.sort_values(by='Sales', ascending=False)
sns.barplot(x="Sales", y="City", data=total_orders_per_city, palette=sns.color_palette('viridis'),ax=ax[1])
ax[1].set_xlabel('Orders')
ax[1].set_ylabel('City')
ax[1].set_title('Total Orders by Cities')
for p in ax[1].patches:
    ax[1].annotate('{:.0f}'.format(p.get_width()), (p.get_width(), p.get_y()+0.6))

plt.tight_layout()
plt.show()
```


In [16]:
```
hourly_orders = data.groupby('Hour')['Order ID'].count()
hourly_sales = data.groupby('Hour')['Sales'].sum()

fig, ax = plt.subplots(1, 2, figsize=(16, 5))

ax[0].plot(hourly_orders.index, hourly_orders.values, marker='o', color='blue', markerfacecolor='red', markersize=5)
ax[0].grid(linestyle='--')
ax[0].set_xlabel('Number of orders')
ax[0].set_xticks(hourly_orders.index)
ax[0].set_ylabel('Number of orders by hours')
ax[0].set_title('Number of orders by hours')

ax[1].plot(hourly_sales.index, hourly_sales.values, marker='o', color='blue', markerfacecolor='red', markersize=5)
ax[1].grid(linestyle='--')
ax[1].set_xlabel('Total Sales')
ax[1].set_xticks(hourly_sales.index)
ax[1].set_ylabel('Total Sales by hours')
ax[1].set_title('Total Sales by hours')

plt.tight_layout()
plt.show()
```


In [17]:
```
product_sales = data.groupby('Product')['Sales'].sum()
plt.figure(figsize=(12, 6))
ax = sns.barplot(x=product_sales.index, y=product_sales.values, palette='viridis')
plt.xlabel('Product')
plt.ylabel('Sales in USD ($)')
plt.title('Total Sales by Products')
plt.xticks(rotation=45)
plt.show()
```


In [18]:
```
product_group = data.groupby('Product')
quantity_ordered = product_group.sum()['Quantity Ordered']
prices = product_group.mean()['Price Each']
products = [product for product, data in product_group]

fig, ax1 = plt.subplots(figsize=(15,6))
ax2 = ax1.twinx()
ax1.bar(products, quantity_ordered, color='g')
ax2.plot(products, prices, 'b-')
ax1.set_xlabel('Product Name')
ax1.set_ylabel('Quantity Ordered', color='g')
ax2.set_ylabel('Price ($)', color='b')
ax1.set_xticklabels(products, rotation='vertical', size=8)

fig.show()
```


In [19]:
```
product_orders = data.groupby('Product')['Order ID'].count()
top_qty_products = product_orders.nlargest(10)
product_qty = data.groupby('Product')['Quantity Ordered'].sum()
top_qty_products = product_qty.nlargest(10)

fig, ax = plt.subplots(1, 2, figsize=(16, 5))

top_orders_plot=sns.barplot(x=top_orders.index,y=top_orders.values,palette='viridis',ax=ax[0])
ax[0].set_xlabel('Product')
ax[0].set_ylabel('Orders')
ax[0].set_title('Top 10 Products Based on Number of Orders')

top_qty_products_plot=sns.barplot(x=top_qty_products.index,y=top_qty_products.values,palette='viridis',ax=ax[1])
ax[1].set_xlabel('Product')
ax[1].set_ylabel('Quantity')
ax[1].set_title('Top 10 Products Based on Quantity Ordered')

plt.show()
```


Coorelation Matrix

In [20]:
```
corr = data.corr()
```

In [21]:
```
plt.figure(figsize=(8,5))
sns.heatmap(corr,annot=True,fmt='.1f', cmap='cmo.colorcet_palette("viridis")')
plt.show()
```


In [ ]:
```

```