```
import pandas as pd
```

```
all_data = pd.read_csv("all_data.csv")

all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address |
|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 |
| 1 | NaN | NaN | NaN | NaN | NaN | NaN |
| 2 | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 04/07/19 22:30 | 682 Chestnut St, Boston, MA 02215 |
| | | | | | 04/12/19 | 669 Spruce St, Los |

**Delete NaN Value**

```
nan_data= all_data[all_data.isna().any(axis=1)]
nan_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address |
|---|---|---|---|---|---|---|
| 1 | NaN | NaN | NaN | NaN | NaN | NaN |
| 356 | NaN | NaN | NaN | NaN | NaN | NaN |
| 735 | NaN | NaN | NaN | NaN | NaN | NaN |
| 1433 | NaN | NaN | NaN | NaN | NaN | NaN |
| 1553 | NaN | NaN | NaN | NaN | NaN | NaN |

```
all_data= all_data.dropna()

all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address |
|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 |
| 2 | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 04/07/19 22:30 | 682 Chestnut St, Boston, MA 02215 |
| 3 | 176560 | Google Phone | 1 | 600 | 04/12/19 14:38 | 669 Spruce St, Los Angeles, CA 90001 |

**Delete 'Or'**

```
or_df= all_data[all_data['Order Date'].str[0:2] == 'Or']
or_df.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address |
|---|---|---|---|---|---|---|
| 519 | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address |
| 1149 | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address |
| 1155 | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address |

```
all_data= all_data[all_data['Order Date'].str[0:2] != 'Or']
```

**Add Month Column**

```
all_data['Month'] = all_data['Order Date'].str[0:2]
all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month |
|---|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 | 04 |

```
all_data['Month']= all_data['Month'].astype('int32')
all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month |
|---|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 | 4 |
| 2 | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 04/07/19 22:30 | 682 Chestnut St, Boston, MA 02215 | 4 |

**Convert Data Type**

```
#make to int
all_data['Quantity Ordered']= pd.to_numeric(all_data['Quantity Ordered'])
#make to float
all_data['Price Each']= pd.to_numeric(all_data['Price Each'])
```

**Add Sales Column**

```
all_data['Sales']= all_data['Quantity Ordered'] * all_data['Price Each']
all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month | Sales |
|---|---|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 | 4 | 23.90 |
| 2 | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 04/07/19 22:30 | 682 Chestnut St, Boston, MA 02215 | 4 | 99.99 |

## 1. **QUESTION:** What was the best month for sales? How much was earned that month?

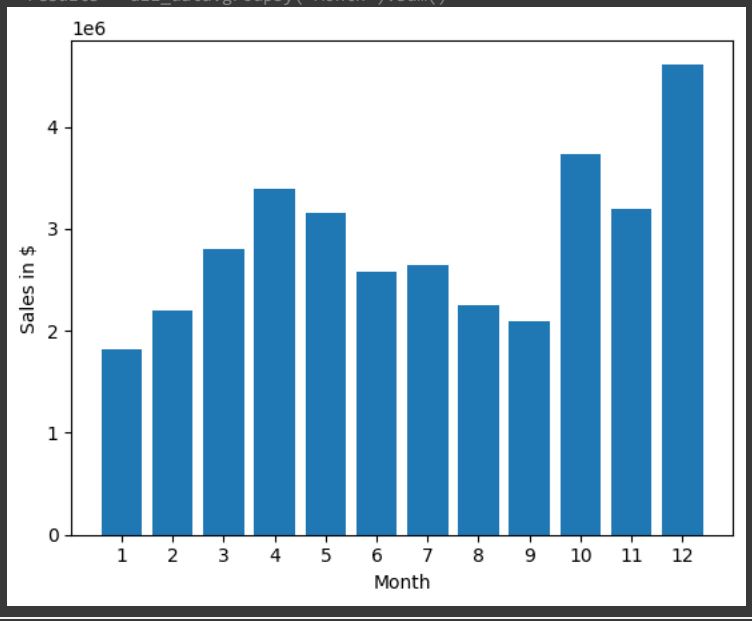
```
all_data.groupby('Month').sum().sort_values('Sales')
#the highest sales happened in month 12
```

```
<ipython-input-13-9dd1ad7f6c1d>:1: FutureWarning: The default value of numeric_only :
  all_data.groupby('Month').sum().sort_values('Sales')
```

| Month | Quantity Ordered | Price Each | Sales |
|---|---|---|---|
| 1 | 10903 | 1811768.38 | 1822256.73 |
| 9 | 13109 | 2084992.09 | 2097560.13 |
| 2 | 13449 | 2188884.72 | 2202022.42 |
| 8 | 13448 | 2230345.42 | 2244467.88 |
| 6 | 15253 | 2562025.61 | 2577802.26 |
| 7 | 16072 | 2632539.56 | 2647775.76 |
| 3 | 17005 | 2791207.83 | 2807100.38 |
| 5 | 18667 | 3135125.13 | 3152606.75 |
| 11 | 19798 | 3180600.68 | 3199603.20 |
| 4 | 20558 | 3367671.02 | 3390670.24 |
| 10 | 22703 | 3715554.83 | 3736726.88 |
| 12 | 28114 | 4588415.41 | 4613443.34 |

```
results = all_data.groupby('Month').sum()
import matplotlib.pyplot as plt
months= range(1, 13)
plt.bar(months, results['Sales'])
plt.xticks(months)
plt.ylabel('Sales in $')
plt.xlabel('Month')
plt.show()
```

```
<ipython-input-14-7064818d7ab7>:1: FutureWarning: The default value of numeric_only :
  results = all_data.groupby('Month').sum()
```




## 2. **Question:** What City had the highest number of sales

```
all_data.tail()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month | Sales |
|---|---|---|---|---|---|---|---|---|
| **186845** | 259353 | AAA Batteries (4-pack) | 3 | 2.99 | 09/17/19 20:56 | 840 Highland St, Los Angeles, CA 90001 | 9 | 8.97 |
| **186846** | 259354 | iPhone | 1 | 700.00 | 09/01/19 | 216 Dogwood St, San | 9 | 700.00 |

**Add City Column**

```
all_data['City']= all_data['Purchase Address'].apply(lambda x: x.split(',')[1])
#.apply(lambda x: x.split(',')[1]) ---- [1] is the index order if we use ',' as divider
#---- x = is the cell content
all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month | Sales | City |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 | 4 | 23.90 | Dallas |
| **2** | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 04/07/19 22:30 | 682 Chestnut St, Boston, MA 02215 | 4 | 99.99 | Boston |
| | | | | | | 669 | | | |

```
all_data['City']= all_data['City'] + ' ' + all_data['Purchase Address'].str[-8:-5]
```

```
all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month | Sales | City |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 | 4 | 23.90 | Dallas TX |
| 2 | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 04/07/19 22:30 | 682 Chestnut St, Boston, MA 02215 | 4 | 99.99 | Boston MA |
| | | | | | | 669 | | | |

**City with The Highest Sales**

```
results_city= all_data.groupby('City').sum().sort_values(by='Sales')
```

```
<ipython-input-19-64d97379de1c>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a fut
  results_city= all_data.groupby('City').sum().sort_values(by='Sales')
```

```
results_city
#San Fransisco is the city with the highest number of sales
```

| City | Quantity Ordered | Price Each | Month | Sales |
|---|---|---|---|---|
| Portland ME | 2750 | 447189.25 | 17144 | 449758.27 |
| Austin TX | 11153 | 1809873.61 | 69829 | 1819581.75 |
| Portland OR | 11303 | 1860558.22 | 70621 | 1870732.34 |
| Seattle WA | 16553 | 2733296.01 | 104941 | 2747755.48 |
| Dallas TX | 16730 | 2752627.82 | 104620 | 2767975.40 |
| Atlanta GA | 16602 | 2779908.20 | 104794 | 2795498.58 |
| Boston MA | 22528 | 3637409.77 | 141112 | 3661642.01 |
| New York City NY | 27932 | 4635370.83 | 175741 | 4664317.43 |
| Los Angeles CA | 33289 | 5421435.23 | 208325 | 5452570.80 |
| San Francisco CA | 50239 | 8211461.74 | 315520 | 8262203.91 |

## **3. Question:** What time should we display advertisements to maximize likelihood of customer's buying product?

```
all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month | Sales | City |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 04/19/19 08:46 | 917 1st St, Dallas, TX 75001 | 4 | 23.90 | Dallas TX |
| 2 | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 04/07/19 22:30 | 682 Chestnut St, Boston, MA 02215 | 4 | 99.99 | Boston MA |
| | | | | | | 669 | | | |

**Change Order Date to Date Time Data Type**

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

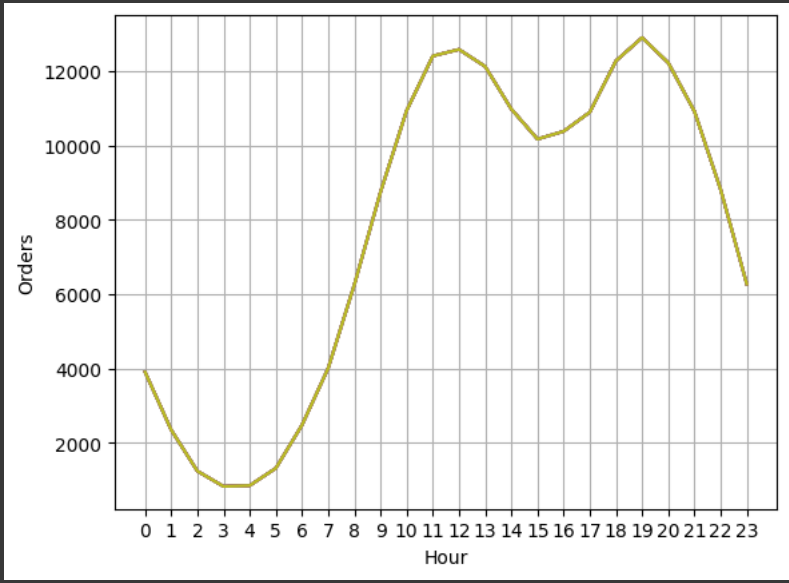
```
#then we can make a new column for the 'Hour', using .dt.hour
all_data['Hour']= all_data['Order Date'].dt.hour
```

```
all_data.head()
```

| | Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month | Sales | City | Ho |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 176558 | USB-C Charging Cable | 2 | 11.95 | 2019-04-19 08:46:00 | 917 1st St, Dallas, TX 75001 | 4 | 23.90 | Dallas TX | |
| 2 | 176559 | Bose SoundSport Headphones | 1 | 99.99 | 2019-04-07 22:30:00 | 682 Chestnut St, Boston, MA 02215 | 4 | 99.99 | Boston MA | |
| | | | | | | 669 | | | | |

```
hours = [hour for hour, df in all_data.groupby('Hour')]

plt.plot(hours, all_data.groupby(['Hour']).count())
plt.xticks(hours)
plt.xlabel('Hour')
plt.ylabel('Orders')
plt.grid()
plt.show()
#based on the chart we could consider the peak time around at 11 am or 7 pm
```




## 5. **Question:** What products sold the most? Why do you think it sold the most?

```
all_data.head()
```

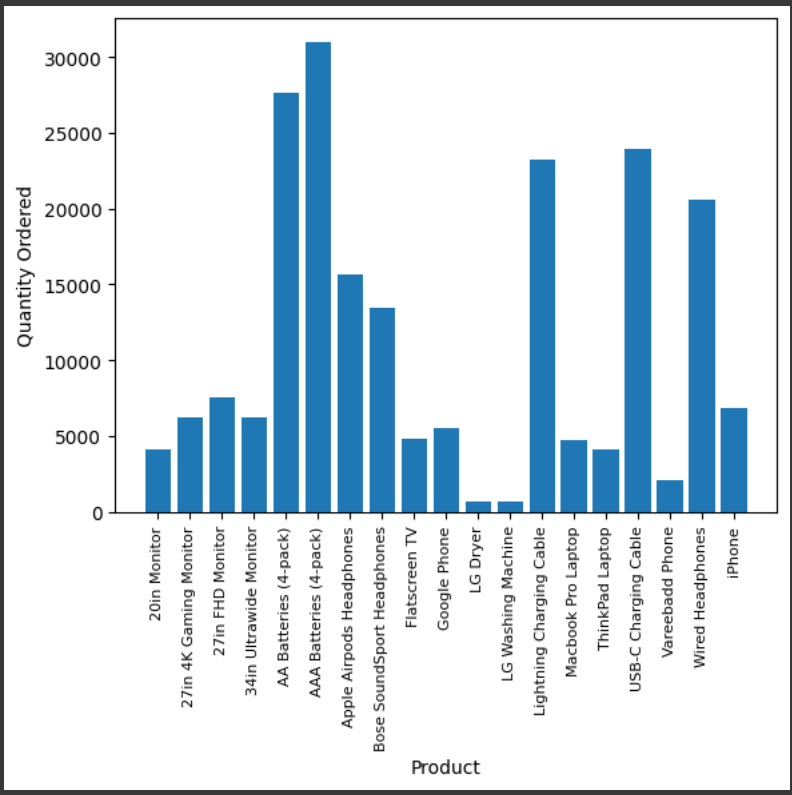
| Order ID | Product | Quantity Ordered | Price Each | Order Date | Purchase Address | Month | Sales | City | Hour |
|---|---|---|---|---|---|---|---|---|---|

```
product_group = all_data.groupby('Product')
quantity_ordered = product_group.sum()['Quantity Ordered']

products = [product for product, df in product_group]
```

```
<ipython-input-32-1ec6facaf08a>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a fut
  quantity_ordered = product_group.sum()['Quantity Ordered']
```

```
plt.bar(products, quantity_ordered)
#Rotate the label
plt.ylabel('Quantity Ordered')
plt.xlabel('Product')
plt.xticks(products, rotation='vertical', size=8)
plt.show()
```




based on the graph, it shows that the top 5 items with the highest number of order are **"AA Batteries, AAA Batteries, Lightning Charging Cable, USB-C Charging Cable"**

Hypothesis: Because the prices are cheaper than other product in the store

**Prove the Hypothesis**

```
prices = all_data.groupby('Product').mean()['Price Each']


#make subplot
fig, ax1 = plt.subplots()

ax2 = ax1.twinx()
ax1.bar(products, quantity_ordered)
ax2.plot(products, prices, 'r-')

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)

plt.show()
```

```
<ipython-input-37-cf831df431d6>:1: FutureWarning: The default value of numeric_only :
  prices = all_data.groupby('Product').mean()['Price Each']
<ipython-input-37-cf831df431d6>:14: UserWarning: FixedFormatter should only be used t
  ax1.set_xticklabels(products, rotation='vertical', size=8)
```