```
# importing libraries
import pandas as pd,numpy as np,os,matplotlib.pyplot as plt,seaborn as sns,io
from itertools import combinations
from collections import Counter
from google.colab import files
# uploading files on colab
uploaded = files.upload()
```

```
Choose Files 12 files
 • Sales_April_2019.csv(text/csv) - 1595953 bytes, last modified: 9/6/2020 - 100% done
 • Sales_August_2019.csv(text/csv) - 1043593 bytes, last modified: 9/6/2020 - 100% done
 • Sales_December_2019.csv(text/csv) - 2181642 bytes, last modified: 9/6/2020 - 100% done
 • Sales_February_2019.csv(text/csv) - 1046495 bytes, last modified: 9/6/2020 - 100% done
 • Sales_January_2019.csv(text/csv) - 843098 bytes, last modified: 9/6/2020 - 100% done
 • Sales_July_2019.csv(text/csv) - 1248753 bytes, last modified: 9/6/2020 - 100% done
 • Sales_June_2019.csv(text/csv) - 1182508 bytes, last modified: 9/6/2020 - 100% done
 • Sales_March_2019.csv(text/csv) - 1323497 bytes, last modified: 9/6/2020 - 100% done
 • Sales_May_2019.csv(text/csv) - 1443965 bytes, last modified: 9/6/2020 - 100% done
 • Sales_November_2019.csv(text/csv) - 1534677 bytes, last modified: 9/6/2020 - 100% done
 • Sales_October_2019.csv(text/csv) - 1770338 bytes, last modified: 9/6/2020 - 100% done
 • Sales_September_2019.csv(text/csv) - 1014958 bytes, last modified: 9/6/2020 - 100% done
Saving Sales_April_2019.csv to Sales_April_2019 (1).csv
Saving Sales_August_2019.csv to Sales_August_2019 (1).csv
Saving Sales_December_2019.csv to Sales_December_2019 (1).csv
Saving Sales_February_2019.csv to Sales_February_2019 (1).csv
Saving Sales_January_2019.csv to Sales_January_2019 (1).csv
Saving Sales_July_2019.csv to Sales_July_2019 (1).csv
Saving Sales_June_2019.csv to Sales_June_2019 (1).csv
Saving Sales_March_2019.csv to Sales_March_2019 (1).csv
Saving Sales_May_2019.csv to Sales_May_2019 (1).csv
Saving Sales_November_2019.csv to Sales_November_2019 (1).csv
Saving Sales_October_2019.csv to Sales_October_2019 (1).csv
Saving Sales_September_2019.csv to Sales_September_2019 (1).csv
```

```
#reading and merging files into one data set
all_month_data=pd.DataFrame()
df=pd.read_csv('Sales_April_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_August_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_December_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_February_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_January_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_July_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_June_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_March_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_May_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_November_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_October_2019.csv')
all_month_data=pd.concat([all_month_data,df])
df=pd.read_csv('Sales_September_2019.csv')
all_month_data=pd.concat([all_month_data,df])

#showing data
all_month_data.to_csv('final_data.csv',index=False)
df=pd.read_csv('final_data.csv')
df.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 |

```
# finding null values and removing them from dataset
df.isnull().sum()
remove_null=df.dropna(how='all')
df=remove_null.copy()
df.head()
df=df[df['Order Date'].str[:2]!='or']
df=df[df['Quantity Ordered']!='Quantity Ordered']

#converting data in numeric form & adding month , sales column for visualization
df['Quantity Ordered']=pd.to_numeric(df['Quantity Ordered'])
df['Price Each']=pd.to_numeric(df['Price Each'])
df['Month']=df['Order Date'].str[1:2]
df['Sales']=df['Quantity Ordered'] * df["Price Each"]

df.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 |

```
#What was the best month for Sales? How much was earned that month?
df.groupby(by=['Month'])['Quantity Ordered','Sales'].sum().sort_values("Sales",ascending=[False]).iloc[:1]
```

```
<ipython-input-77-eb610e3469c8>:2: FutureWarning: Indexing with multiple keys (implic
  df.groupby(by=['Month'])['Quantity Ordered','Sales'].sum().sort_values("Sales",asce
```

| | Quantity Ordered | Sales |
|---|---|---|
| **Month** | | |
| **2** | 41563 | 6815465.76 |

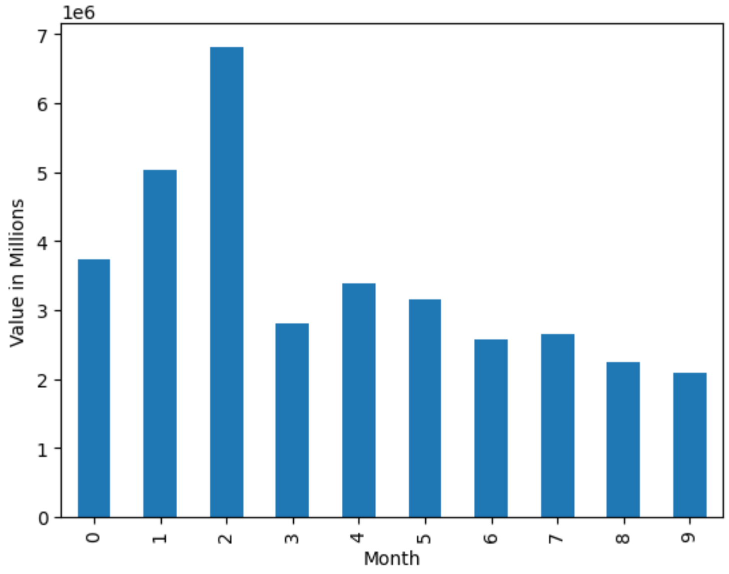
```
#Month of minimum sales
df.groupby(by=['Month'])['Quantity Ordered','Sales'].sum().sort_values("Sales",ascending=[True]).iloc[:1]
```

```
<ipython-input-78-b9c7824eaf25>:2: FutureWarning: Indexing with multiple keys (implic
  df.groupby(by=['Month'])['Quantity Ordered','Sales'].sum().sort_values("Sales",asce
```

| | Quantity Ordered | Sales |
|---|---|---|
| **Month** | | |
| **9** | 13109 | 2097560.13 |

```
# Visualizing data in bar chart
df.groupby('Month')['Sales'].sum().plot(kind="bar")
plt.ylabel('Value in Millions')
```

```
Text(0, 0.5, 'Value in Millions')
```



```
# Most ordered product
(df.groupby(by=['Product'])['Quantity Ordered','Sales'].sum()).sort_values("Quantity Ordered",ascending=[False]).iloc[:1]
```

```
<ipython-input-80-31d87abec668>:2: FutureWarning: Indexing with multiple keys (implic
  (df.groupby(by=['Product'])['Quantity Ordered','Sales'].sum()).sort_values("Quantit
```

| | Quantity Ordered | Sales |
|---|---|---|
| **Product** | | |
| **AAA Batteries (4-pack)** | 31017 | 92740.83 |

```
# least ordered product
(df.groupby(by=['Product'])['Quantity Ordered','Sales'].sum()).sort_values("Quantity Ordered",ascending=[True]).iloc[:1]
```

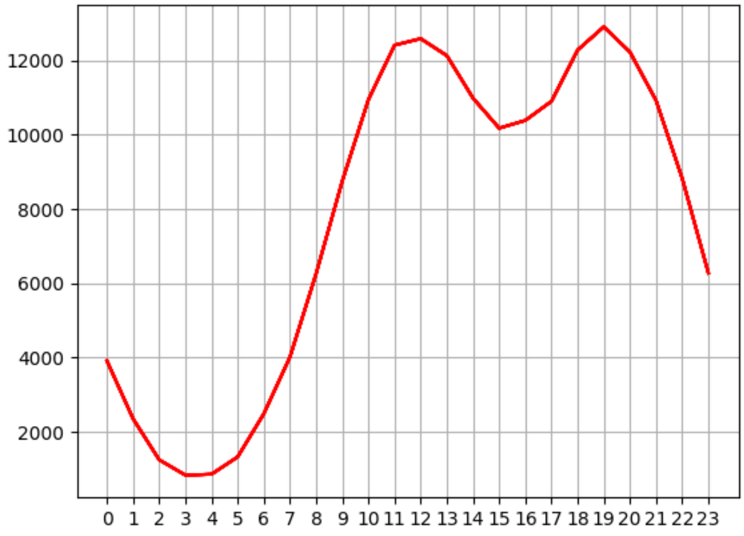
```
<ipython-input-81-98b635af3eff>:2: FutureWarning: Indexing with multiple keys (implic
  (df.groupby(by=['Product'])['Quantity Ordered','Sales'].sum()).sort_values("Quantit
```

| | Quantity Ordered | Sales |
|---|---|---|
| **Product** | | |
| **LG Dryer** | 646 | 387600.0 |

```
#Best time for advertisements.

#Adding hor minute column
df['Order Date']=pd.to_datetime(df['Order Date'])
df['Hour']=df['Order Date'].dt.hour
df['Minute']=df['Order Date'].dt.minute

#Visualization
hours=[hour for hour,df in df.groupby('Hour')]
plt.plot(hours,df.groupby(['Hour']).count(),color='red')
plt.xticks(hours)
plt.grid()
plt.show()
```



```
#pairs that can be stacked together by the customer
abc=df[df['Order ID'].duplicated(keep=False)]
abc['Grouped']=abc.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))
abc=abc[['Order ID','Grouped']].drop_duplicates()
abc.head()
```

```
<ipython-input-85-c163022f32ca>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
```

```
#more appropriate results
count=Counter()
keys=[]
values=[]
for row in abc['Grouped']:
  row_list=row.split(',')
  count.update(Counter(combinations(row_list,2)))
for key,value in count.most_common(10):
  keys.append(key)
  values.append(value)

d = {'Items': keys , 'Purchasing frequency' : values }
res=pd.DataFrame(d)
res.head(10)
```

| | Items | Purchasing frequency |
|---|---|---|
| **0** | (iPhone, Lightning Charging Cable) | 1005 |
| **1** | (Google Phone, USB-C Charging Cable) | 987 |
| **2** | (iPhone, Wired Headphones) | 447 |
| **3** | (Google Phone, Wired Headphones) | 414 |
| **4** | (Vareebadd Phone, USB-C Charging Cable) | 361 |
| **5** | (iPhone, Apple Airpods Headphones) | 360 |
| **6** | (Google Phone, Bose SoundSport Headphones) | 220 |
| **7** | (USB-C Charging Cable, Wired Headphones) | 160 |
| **8** | (Vareebadd Phone, Wired Headphones) | 143 |
| **9** | (Lightning Charging Cable, Wired Headphones) | 92 |