NAME: GUNGUN WAHANE
ROLL NO: 666
PRN: 202201050029
BATCH: F4

ASSIGNMENT 4:

```
import numpy as np
import pandas as pd
all_data=pd.read_csv("/content/asst 4 dataset.csv")
all_data
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
36	NaN	NaN	NaN	NaN	NaN	NaN
51	NaN	NaN	NaN	NaN	NaN	NaN

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	City
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215	4	Boston (A)
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4	Los Angeles (A)
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4	Los Angeles (A)
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001	5	Los Angeles (A)
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016	4	San Francisco (A)

```
all_data.shape
```

```
(69, 6)
```

Drop rows of NAN

```
#find NAN
nan_df = all_data[all_data.isna().any(axis=1)]
display(nan_df.head())

all_data.shape

all_data=all_data.dropna(how='all')
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
36	NaN	NaN	NaN	NaN	NaN	NaN
51	NaN	NaN	NaN	NaN	NaN	NaN

```
all_data.shape  #dropped 2 column
(67, 6)
```

get rid of the text in order date column

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

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016
...	...	...	...	...	...	...
64	259329.0	Lightning Charging Cable	1.0	14.95	09-05-2019 19:00	480 Lincoln St, Atlanta, GA 30301
65	259330.0	AA Batteries (4-pack)	2.0	3.84	09/25/19 22:01	763 Washington St, Seattle, WA 98101
66	259331.0	Apple Airpods Headphones	1.0	150.00	09/29/19 7:00	770 4th St, New York City, NY 10001
67	259332.0	Apple Airpods Headphones	1.0	150.00	09/16/19 19:21	782 Lake St, Atlanta, GA 30301
68	259333.0	Bose SoundSport Headphones	1.0	99.99	09/19/19 18:03	347 Ridge St, San Francisco, CA 94016

67 rows × 6 columns

Make columns correct type

```
all_data['Quantity Ordered'] = pd.to_numeric(all_data['Quantity Ordered'])
all_data['Price Each'] = pd. to_numeric(all_data['Price Each'])
```

Augment data with additional columns

Add month column

```
all_data['Month']= all_data['Order Date'].str[0:2]
all_data['Month']= all_data['Month'].astype('int32')
all_data.head()
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215	4
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001	5
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016	4

```
def get_city(address):
    return address.split(",")[1].strip(" ")

def get_state(address):
    return address.split(",")[2].strip(" ")[1]
all_data['City'] = all_data['Purchase Address'].apply(lambda x:
f"{get_city(x)} ({get_state(x)})")
all_data.head()
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215	4
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001	5
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016	4

Q 1. what was the best month for sales? How much was earned that month

```
all_data['Sales'] = all_data['Quantity Ordered'].astype('int') *
all_data['Price Each'].astype('float')
```

```
all_data.groupby(['Month']).sum()
```

```
<ipython-input-17-dce0a735c05d>:1: FutureWarning: The default value of numeric_o
  all_data.groupby(['Month']).sum()
```

Month	Order ID	Quantity Ordered	Price Each	Sales
4	7335546.0	123.0	885.80	1210.76
5	353124.0	2.0	111.98	111.98
6	184076.0	1.0	14.95	14.95
8	726962.0	9.0	23.92	50.83
9	2378802.0	17.0	591.44	616.62
10	550924.0	11.0	10.67	39.69
11	740314.0	19.0	13.66	65.31
12	550635.0	17.0	8.97	50.83

2.What city sold the most product

```
Dummycity=all_data.groupby(['City'])
print(Dummycity)
#city_max=all_data.groupby(['City']).sum()
#print(max(city_max))
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7fcb2cbfbd90>
```

Q.3 what product sold the most? why do you think it sold the most?

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

```
print(quantity_ordered)
```

```
Product
AA Batteries (4-pack)          64.0
AAA Batteries (4-pack)        109.0
Apple Airpods Headphones        3.0
Bose SoundSport Headphones      3.0
Google Phone                    1.0
Lightning Charging Cable        4.0
USB-C Charging Cable            8.0
Wired Headphones                7.0
Name: Quantity Ordered, dtype: float64
```

Q 4. what products are most often sold together?

```
df=all_data[all_data['Order ID'].duplicated(keep=False)]

#referenced: https://stackoverflow.com/questions/27298178/concatenate-strings-from-several-rows-using-pandas-groupby
df['Grouped'] = df.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))
print(df['Grouped'])
```

```
1    Google Phone,Wired Headphones
2    Google Phone,Wired Headphones
Name: Grouped, dtype: object
<ipython-input-25-c49218ac08af>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Grouped'] = df.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))
```