Name : Omkar Dhere

Roll No : 616

Batch : F1

PRN : 202201060030

Assignment 4

In [8]:
```
import numpy as np
import pandas as pd
all_data=pd.read_csv("/content/sample_data/all_data.csv")
all_data.head()
```

Out [8]:

	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

Clean up the data

In [9]:
```
all_data.shape
```

Out [9]: (69, 6)

Drop rows of NAN

In [10]:
```
#Find NAN
nan_df=all_data[all_data.isna().any(axis=1)]
display(nan_df.head())

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

	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

Out [10]:

	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

Get rid of text in order date column

In [11]:
```
all_data=all_data[all_data['Order Date'].str[0:2]!='Or']
print(all_data)
```

```
    Order ID                    Product  Quantity Ordered  Price Each  \
0   176559.0  Bose SoundSport Headphones               1.0       99.99
1   176560.0                Google Phone               1.0      600.00
2   176560.0            Wired Headphones               1.0       11.99
3   176561.0            Wired Headphones               1.0       11.99
4   176562.0        USB-C Charging Cable               1.0       11.95
..       ...                         ...               ...         ...
64  259329.0    Lightning Charging Cable               1.0       14.95
65  259330.0       AA Batteries (4-pack)               2.0        3.84
66  259331.0    Apple Airpods Headphones               1.0      150.00
67  259332.0    Apple Airpods Headphones               1.0      150.00
68  259333.0  Bose SoundSport Headphones               1.0       99.99

          Order Date                      Purchase Address
0   04-07-2019 22:30       682 Chestnut St, Boston, MA 02215
1   04-12-2019 14:38    669 Spruce St, Los Angeles, CA 90001
2   04-12-2019 14:38    669 Spruce St, Los Angeles, CA 90001
3      05/30/19 9:27       333 8th St, Los Angeles, CA 90001
4     04/29/19 13:03  381 Wilson St, San Francisco, CA 94016
..               ...                                    ...
64  09-05-2019 19:00       480 Lincoln St, Atlanta, GA 30301
65    09/25/19 22:01    763 Washington St, Seattle, WA 98101
66     09/29/19 7:00     770 4th St, New York City, NY 10001
67    09/16/19 19:21          782 Lake St, Atlanta, GA 30301
68    09/19/19 18:03   347 Ridge St, San Francisco, CA 94016

[67 rows x 6 columns]
```

Make columns correct type

In [12]:
```
all_data['Quantity Ordered']=pd.to_numeric(all_data['Quantity Ordered'])
all_data['Price each']=pd.to_numeric(all_data['Price Each'])
```

Argument data with additional columns

Add month column

In [13]:
```
all_data['Month']=all_data['Order Date'].str[0:2]
all_data['Month']=all_data['Month'].astype('int32')
all_data.head()
```

Out [13]:

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

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Price each	Month
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016	11.95	4

Add month column (Alternative method)

In [14]:
```
all_data['Month 2']=pd.to_datetime(all_data['Order Date']).dt.month
all_data.head()
```

Out [14]:

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

Add City Town

In [15]:
```
def get_city(address):
  return address.split(",")[1].strip(" ")

def get_state(address):
  return address.split(",")[2].split(" ")[1]

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

Out [15]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Price each	Month	Month 2	city
0	176559.0	Bose SoundSport Headphones	1.0	99.99	04-07-2019 22:30	682 Chestnut St, Boston, MA 02215	99.99	4	4	Boston (MA)
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	600.00	4	4	Los Angeles (CA)

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Price each	Month	Month 2	city
2	176560.0	Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	11.99	4	4	Los Angeles (CA)
3	176561.0	Wired Headphones	1.0	11.99	05/30/19 9:27	333 8th St, Los Angeles, CA 90001	11.99	5	5	Los Angeles (CA)
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016	11.95	4	4	San Francisco (CA)

Questions

Question 1:

What was the best month for sales? How much was earned that month?

In [16]:
```
all_data['Sales']=all_data['Quantity Ordered'].astype('int')*all_data['Price Ea
all_data.groupby(['Month']).sum()
```

```
<ipython-input-16-8ba29a3e5d2a>:2: FutureWarning: The default value of numeric_only in
DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False.
Either specify numeric_only or select only columns which should be valid for the function.
  all_data.groupby(['Month']).sum()
```

Out [16]:

Month	Order ID	Quantity Ordered	Price Each	Price each	Month 2	Sales
4	7335546.0	123.0	885.80	885.80	160	1210.76
5	353124.0	2.0	111.98	111.98	10	111.98
6	184076.0	1.0	14.95	14.95	6	14.95
8	726962.0	9.0	23.92	23.92	32	50.83
9	2378802.0	17.0	591.44	591.44	90	616.62
10	550924.0	11.0	10.67	10.67	30	39.69
11	740314.0	19.0	13.66	13.66	44	65.31
12	550635.0	17.0	8.97	8.97	36	50.83

Question 2:

What city sold the most product?

In [22]:
```
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 0x7fcc83d92b00>
```

In [34]:
```
from itertools import combinations
from collections import Counter

count = Counter()

for row in df2['Grouped']:
  row_list=row.split(',')
  count.update(Counter(combinations(row_list, 2)))

for key,value in count.most_common(10):
  print(key, value)
```

```
('Google Phone', 'Wired Headphones') 1
```

###Question 3:

What product sold the most?Wghy do you think it sold the most?

In [40]:
```
product_group=all_data.groupby('Product')
quantity_ordered=product_group.sum()['Quantity Ordered']
```

```
<ipython-input-40-11142b314e0e>:2: FutureWarning: The default value of numeric_only in
DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False.
Either specify numeric_only or select only columns which should be valid for the function.
  quantity_ordered=product_group.sum()['Quantity Ordered']
```

In [41]:
```
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
```

In [42]:
```
prices=all_data.groupby('Product').mean()['Price Each']
```

```
<ipython-input-42-1f4f73bca841>:1: FutureWarning: The default value of numeric_only in
DataFrameGroupBy.mean is deprecated. In a future version, numeric_only will default to False.
Either specify numeric_only or select only columns which should be valid for the function.
  prices=all_data.groupby('Product').mean()['Price Each']
```

In [43]:
```
print(prices)
```

```
Product
AA Batteries (4-pack)           3.84
AAA Batteries (4-pack)          2.99
Apple Airpods Headphones      150.00
Bose SoundSport Headphones     99.99
Google Phone                  600.00
Lightning Charging Cable       14.95
USB-C Charging Cable           11.95
Wired Headphones               11.99
Name: Price Each, dtype: float64
```

Question 4:

What city sold the most product?

In [28]:

```
df=all_data[all_data['Order ID'].duplicated(keep=False)]
df['Grouped']=df.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))
df2=df[['Order ID', 'Grouped']].drop_duplicates()
print(df['Grouped'])
```

```
1    Google Phone,Wired Headphones
2    Google Phone,Wired Headphones
Name: Grouped, dtype: object
```

```
<ipython-input-28-387d448b896d>:2: 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))
```