1 # NAME :Pratiksha Ranmare
2 # ROLL NO : 654
3 # PRN NO : 202201070056
4 # BATCH : F3

```
7 import numpy as
np8 import pandas
as pd
9 all_data=pd.read_csv("/content/1686715083343 all data (7).csv")
10 all_data.head()
```

	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
						381 Wilson St San Francisco CA

```
1
```

```
1 #clean up the
data2
all_data.shape
```

```
(69, 6)
```

```
1 # drop rows of nana
2 nan_df=all_data[all_data.isna().any(axis=1)]
3 display(nan_df.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

```
      1
all_data.shape
```

```
   (69, 6)
```

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

	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, LosAngeles, CA 90001
3	176561.0	Wired	1.0	11.99	05/30/19	333 8th St, Los

1
all_data.shape

(67, 6)

```
1 #get rid of text order date column
2 all_data=all_data[all_data['Order Date'].str[0:2]!='Or']
3 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        763 Washington St, Seattle, WA 98101
      22:01
66     09/29/19        770 4th St, New York City, NY 10001
       7:00
67    09/16/19             782 Lake St, Atlanta, GA 30301
      19:21
68    09/19/19       347 Ridge St, San Francisco, CA 94016
      18:03

[67 rows x 6 columns]
```

```
#make column correct type
all_data['Quantity Ordered']=pd.to_numeric(all_data['Quantity Ordered'])3 all_data['Price Each']=pd.to_numeric(all_data['Price Each'])
all_data.head()
```

```
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
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
1	176560.0	Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4
						381 Wilson St	

```
#Add city column
def get_city(address):
  return address.split(",")[1].strip(" ")
def get_state(address):
  return address.split(",")[2].strip("
")[1]6
all_data['city']=all_data['Purchase Address'].apply(lambda x:f"{get_city(x)}
({get_state(x)})")8 all_data.head()
9
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	city
0	176559.0	Bose SoundSport Headphones	1	1	04-07-17	682	6	Google Phone

2	176560.0	Wired Headphones	1.0	99.99	04/29/2019 22:30	669 Spruce St, Los Angeles, CA 90001	4	Los Angeles (A)
			1.0	600.00	04-12-2019 14:38			
			1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001	4	Los Angeles (A)
3	176561.0	Wired	1.0	11.99	05/30/19	333 8th St, Los	5	Los Angeles

```
#waht 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-11-8fec2581ce34>:3: FutureWarning: The default value of
  numeric_onlall_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)WHICH 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 0x7f62dbe6fd00>
```

```
#waht products are most often sold together
df=all_data[all_data['Order ID'].duplicated(keep=False)]
df['Grouped']=df.groupby('Order ID')['Product'].transform(lambda
x:','.join(x))4 df2=df[['Order ID','Grouped']].drop_duplicates()
print(df['Grouped'])
```

```
1    Google Phone,Wired Headphones
2    Google Phone,Wired
HeadphonesName: Grouped,
dtype: object
<ipython-input-18-1970be6762a6>:3: 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-copydf['Grouped']=df.groupby('Order ID')['Product'].transform(lambda x:','.join(x))
```

```
from itertools import
combinations2 from collections
import Counter
3
count=Counter()
```

```
 5
 6 for row in df2['Grouped']:
 7   row_list=row.split(',')
 8   count.update(Counter(combinations(row_lis
 t,2)))9
10 for key,value in count.most_common(10):
11   print(key,value)
```

```
12
13
```

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

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

```
<ipython-input-20-11142b314e0e>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only
  will default to False. Ei 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             3.0
Headphones
Google Phone                1.0
Lightning Charging Cable    4.0
USB-C Charging Cable        8.0
Wired Headphones            7.0
Name: Quantity Ordered, dtype: float64
```

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

```
<ipython-input-22-1f4f73bca841>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future version, numeric_only
  will default to False. Eprices=all_data.groupby('Product').mean()['Price Each']
```

```
print(prices)
```

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