

```
import numpy as np
import pandas as pd
```

```
all_data = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/1686715083343_all_data.csv")
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
4	176562.0	USB-C Charging Cable	1.0	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016

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

```
all_data.shape
```

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

<bound method NDFrame.head of 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
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

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

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

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

## ▼ Add City Column

```
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	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)

## ▼ Data Exploration

Question 1 - What was the best month for sales and how much was earned in that month?

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

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

```
<ipython-input-12-dce0a735c05d>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy
all_data.groupby(['Month']).sum()
```

	Order ID	Quantity Ordered	Price Each	Sales
Month				
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

## ▼ Question 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 0x7fe2ce0137f0>
```

#### Q 4 Which 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-severa
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-17-7305ebdbe5d9>: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
df['Grouped'] = df.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))

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
```

#### Q 3 which products sold the most? Why do u 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
<ipython-input-20-ddc2ef51f24b>:2: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a fut
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

prices = 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 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
```

```
<ipython-input-22-ff49c55915e9>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a fu
prices = all_data.groupby('Product').mean()['Price Each']
```

