

NAME: SHRUTI JADHAV

ROLL NO. 620

PRN: 202201050041

BATCH:F1

o 176559.0 1.0 99.99 682 Chestnut St, Boston, MA 02215
Headphones 22:30

04-12-2019

1 176560.0 Google Phone 1.0 600.00 669 Spruce St, Los Angeles, CA 90001
14:38

04-12-2019

2 176560.0 Wired Headphones 1.0 11.99 669 Spruce St, Los Angeles, CA 90001
14:38

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 data

2 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	Quantity	Price	Order	Purchase
Product				
IDOrdered	Each	Date	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	Quantity	Price	Order
-------	----------	-------	-------

	ID	Ordered	Product	Each	Date	Purchase Address
0	176559.0		Bose SoundSport 1.0 Headphones	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
1	all_data.shape					669 Spruce St, Los Angeles, CA 90001
2	176560.0					
	(67 , 6)					
3	176561.0		Wired Headphones 1.0	11.99	04-12-2019 14:38	333 8 th St, Los Angeles, CA 90001
			Wired Headphones 1.0	11.99	05/30/19 9:27	CA 90001

```
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 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]

```

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

```

ID	Ordered	Each	Date
----	---------	------	------

```

1 all_data['Month']= all_data['Order Date'].str[0:2]
2 all_data['Month']= all_data['Month'].astype('int32')
3 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
						381 Wilson St	

Order Quantity Price Order Product
Purchase Address

```

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

```

Order ID	Quantity Ordered	Product	Price Each	Order Date	Purchase Address
176559.0	1.0	Bose SoundSport Headphones	99.99	2019-04-07 14:38	Chestnut St, Boston, MA 02215
176560.0	1.0	Los Angeles Headphones	600.00	2019-04-12 14:38	Los Angeles, CA 90001
176560.0	1.0	Los Angeles Headphones	11.99	2019-04-12 14:38	Los Angeles, CA 90001
		Wired		05/30/19	Los Angeles, CA 90001

```
3 176561.0      1.0    11.99  5      Angeles
      H      d h    9 27  A      I
```

```
1 #waht was the best month for sales?how much was earned that month?
```

```
2 all_data['Sales']=all_data['Quantity Ordered'].astype('int')*all_data['Price Each'].astype('float')    3
all_data.groupby(['Month']).sum()
```

```
4
```

```
< ipython-input-11-8fec2581ce34>:3: FutureWarning: The default value of
numeric_onl 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
```

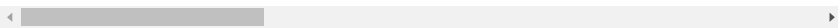
```
8 726962.0 9.0 23.92 50.83 9 2378802.0
```

```
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
```



```
1 #2)WHICH CITY SOLD
```

```
THE MOST PRODUCT? 2
```

```
Dummyscity=all_data.groupby
y(['city'])
```

```
3 print( Dummyscity )
```

```
4 #city_max=all_data.groupby(['city']).sum()
```

```
5 #print(max(city_max))
```

```
< pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f62dbe6fd 00>
```

```
1 #waht products are most often sold together
```

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

```
3 df['Grouped']=df.groupby('Order ID')['Product'].transform(lambda x:', '.join(x))
```

```
4 df2=df[['Order ID','Grouped']] .drop_duplicates ()
5 print(df['Grouped'])
```

```
1 Google Phone,Wired Headphones
```

```
2 Google Phone,Wired Headphones
```

```
Name: 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-copy
```

```
df['Grouped']=df.groupby('Order ID')['Product'].transform(lambda x:','.join(x))
```

```
1 from itertools import combinations
```

```
2 from collections import Counter
```

```
3
```

```
4 count=Counter()
```

```
5
```

```
6 for row in df2['Grouped']:
```

```
7 row_list=row.split(',')
```

```
8 count.update(Counter(combinations(row_list,2)))
```

```
9
```

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

```
11 print( key,value )
```

```
12
```

```
13
```

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

```
1 product_group=all_data.groupby('Product')
```

```
2 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']
```

```
1 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

```
1 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. E

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

```
1 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