

NAME:RUTUJA
CHAMLE
ROLL.NO:608
PRN:
202201070055
CLASS:F(1)

ASSIGNMENT 4

```
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()  
  
all_data.shape
```

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176559.0	Bose SoundSport Headphones	1.0	99.99	4/7/2019 22:30	682 Chestnut St, Boston, MA 02215
1	176560.0	Google Phone	1.0	600.00	4/12/2019 14:38	669 Spruce St, Los Angeles, CA 90001
2	176560.0	Wired Headphones	1.0	11.99	4/12/2019 14:38	669 Spruce St, Los Angeles, CA 90001
3	176561.0	Wired Headphones	1.0	11.99	5/30/2019 9:27	333 8th St, Los Angeles, CA 90001
4	176562.0	USB-C Charging Cable	1.0	11.95	4/29/2019 13:03	381 Wilson St, San Francisco, CA 94016

```
all_data.shape
```

0s

```
(69, 6)
```

Drop rows of 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')
```

```
all_data.head()
```

```
all_data.shape
```

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

```
(67, 6)
```

Get rid of text in order date column

```
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	4/7/2019 22:30	682 Chestnut St, Boston, MA 02215
1	4/12/2019 14:38	669 Spruce St, Los Angeles, CA 90001
2	4/12/2019 14:38	669 Spruce St, Los Angeles, CA 90001

```

3    5/30/2019 9:27    333 8th St, Los Angeles, CA 90001
4    4/29/2019 13:03   381 Wilson St, San Francisco, CA 94016
..    ...
64   9/5/2019 19:00    480 Lincoln St, Atlanta, GA 30301
65   9/25/2019 22:01   763 Washington St, Seattle, WA 98101
66   9/29/2019 7:00    770 4th St, New York City, NY 10001
67   9/16/2019 19:21    782 Lake St, Atlanta, GA 30301
68   9/19/2019 18:03   347 Ridge St, San Francisco, CA 94016

```

```
[69 rows x 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
```

```

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
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

```
Add city column
```

```
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)} ({get_state(x)})")  
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 4 Boston (MA)
1	176560.0 Google Phone	1.0	600.00	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001 4 Los Angeles (CA)
2	176560.0 Wired Headphones	1.0	11.99	04-12-2019 14:38	669 Spruce St, Los Angeles, CA 90001 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 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 4 San Francisco (CA)

```
# 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()
```

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

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

```
1  Google Phone,Wired Headphones
2  Google Phone,Wired Headphones
36         NaN
51         NaN
```

Name: Grouped, dtype: object

<ipython-input-10-be4b8fe819be>: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/10min_guide/indexing.html#returning-a-view-versus-a-copy

```
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)
```

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

```
quantity_ordered=product_group.sum()['Quantity Ordered']
<ipython-input-21-11142b314e0e>: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']
```

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

```
<ipython-input-14-11142b314e0e>: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']
```

```
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']
```

```
<ipython-input-19-1f4f73bca841>: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']
```

```
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	