Final Project - Analyzing Sales Data

Date: 16 August 2023

Author: Napat Teekasuk

Course: `Pandas Foundation`

```
# import data
import pandas as pd
df = pd.read_csv("sample-store.csv")
```

```
# preview top 5 rows
df.head()
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region
0	1	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States
2	3	CA-2019-138688	6/12/2019	6/16/2019	Second Class	DV-13045	Darrin Van Huff	Corporate	United States
3	4	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States

5 rows × 21 columns

```
# shape of dataframe
# shape of dataframe
df.shape
```

```
(9994, 21)
```

```
# see data frame information using .info()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Row ID           9994 non-null   int64  
 1   Order ID         9994 non-null   object 
 2   Order Date       9994 non-null   object 
 3   Ship Date        9994 non-null   object 
 4   Ship Mode        9994 non-null   object 
 5   Customer ID      9994 non-null   object 
 6   Customer Name    9994 non-null   object 
 7   Segment          9994 non-null   object 
 8   Country/Region   9994 non-null   object 
 9   City             9994 non-null   object 
 10  State            9994 non-null   object 
 11  Postal Code      9983 non-null   float64
 12  Region           9994 non-null   object 
 13  Product ID       9994 non-null   object 
 14  Category         9994 non-null   object 
```

We can use `pd.to_datetime()` function to convert columns 'Order Date' and 'Ship Date' to datetime.

```
# example of pd.to_datetime() function
pd.to_datetime(df['Order Date'].head(), format='%m/%d/%Y')
```

```
0   2019-11-08
1   2019-11-08
2   2019-06-12
3   2018-10-11
4   2018-10-11
Name: Order Date, dtype: datetime64[ns]
```

```
# TODO - convert order date and ship date to datetime in the original datafra
```

```
df['Order Date'] = pd.to_datetime(df['Order Date'], format='%m/%d/%Y')
df['Ship Date'] = pd.to_datetime(df['Order Date'], format='%m/%d/%Y')

print(df['Order Date'].dtype)
print(df['Ship Date'].dtype)
```

```
datetime64[ns]
datetime64[ns]
```

```
# TODO - count nan in postal code column
```

```
df['Postal Code'].isna().sum()
```

```
11
```

```
# TODO - filter rows with missing values
```

```
df[ df['Postal Code'].isna() ]
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Regi
2234	2235	CA-2020-104066	12/5/2020	12/10/2020	Standard Class	QJ-19255	Quincy Jones	Corporate	United State
5274	5275	CA-2018-162887	11/7/2018	11/9/2018	Second Class	SV-20785	Stewart Visinsky	Consumer	United State
8798	8799	US-2019-150140	4/6/2019	4/10/2019	Standard Class	VM-21685	Valerie Mitchum	Home Office	United State
9146	9147	US-2019-165505	1/23/2019	1/27/2019	Standard Class	CB-12535	Claudia Bergmann	Corporate	United State
9147	9148	US-2019-165505	1/23/2019	1/27/2019	Standard Class	CB-12535	Claudia Bergmann	Corporate	United State
9148	9149	US-2019-165505	1/23/2019	1/27/2019	Standard Class	CB-12535	Claudia Bergmann	Corporate	United State

9386	9387	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9387	9388	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9388	9389	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9389	9390	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9741	9742	CA-2018-117086	11/8/2018	11/12/2018	Standard Class	QJ-19255	Quincy Jones	Corporate	United State

11 rows × 21 columns

```
# TODO - Explore this dataset on your owns, ask your own questions
```

Data Analysis Part

Answer 10 below questions to get credit from this course. Write `pandas` code to find answers.

```
# TODO 01 - how many columns, rows in this dataset
df.shape
## ANS rows == 9994, colums == 21
```

```
(9994, 21)
```

```
# TODO 02 - is there any missing values?, if there is, which colunm? how many
df.isna().sum()
## ANS Postal Code = 11
```

```
Row ID           0
Order ID         0
Order Date       0
Ship Date        0
Ship Mode        0
Customer ID      0
Customer Name    0
```

```
Segment            0
Country/Region     0
City               0
State              0
Postal Code       11
Region             0
Product ID         0
Category           0
Sub-Category       0
Product Name       0
Sales              0
Quantity           0
Discount           0
Profit             0
Profit_Check       0
dtype: int64
```

```
# TODO 03 - your friend ask for `California` data, filter it and export csv fo
df_California = df[ df['State'] == 'California']

df_California.to_csv('California.csv')

## ANS File name is California.csv
```

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in

df_cali_tex = df[ (df['State'] == 'California' ) | (df['State'] == 'Texas') ]

df_cali_tex_2017 = df_cali_tex[(df_cali_tex['Order Date'] >= '2017-01-01') &
    .reset_index()

df_cali_tex_2017.to_csv('California_Texas_2017.csv')

## ANS File name is California_Texas_2017.csv
```

```
# TODO 05 - how much total sales, average sales, and standard deviation of sa
df_2017 = df[(df['Order Date'] >= '2017-01-01') & (df['Order Date'] <= '2017-1

df_2017['Sales'].agg(['sum', 'mean', 'std'])
## ANS total sales = 484247.498, average sales = 242.974, and standard deviati
```

```
sum     484247.498100
mean       242.974159
std        754.053357
Name: Sales, dtype: float64
```

```
# TODO 06 - which Segment has the highest profit in 2018
df_2018 = df[(df['Order Date'] >= '2018-01-01') & (df['Order Date'] <= '2018-1

df_2018.groupby('Segment')['Profit'].sum().sort_values(ascending = False)
## ANS Consumer has the highest profit in 2018
```

```
Segment
Consumer       28460.1665
Corporate      20688.3248
Home Office    12470.1124
Name: Profit, dtype: float64
```

```
# TODO 07 - which top 5 States have the least total sales between 15 April 20
df_2019 = df[(df['Order Date'] >= '2019-04-15') & (df['Order Date'] <= '2019-1

df_2019.groupby('State')['Sales'].sum().sort_values(ascending=True).head(5)
## ANS New Hampshire, New Mexico, District of Columbia, Louisiana and South Ca
```

```
State
New Hampshire            49.05
New Mexico               64.08
District of Columbia    117.07
Louisiana               249.80
South Carolina          502.48
Name: Sales, dtype: float64
```

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 20
df_total_2019 = df[(df['Order Date'] >= '2019-01-01') & (df['Order Date'] <=

total_sale_2019 = df_total_2019['Sales'].sum()

west_central_sale_2019 = df_total_2019[(df_total_2019['Region'] == 'West') |
    .sum()

result = (west_central_sale_2019 / total_sale_2019) * 100
result

## the proportion of total sales (%) in West + Central in 2019 is 54.97 %
```

```
54.97479891837763
```

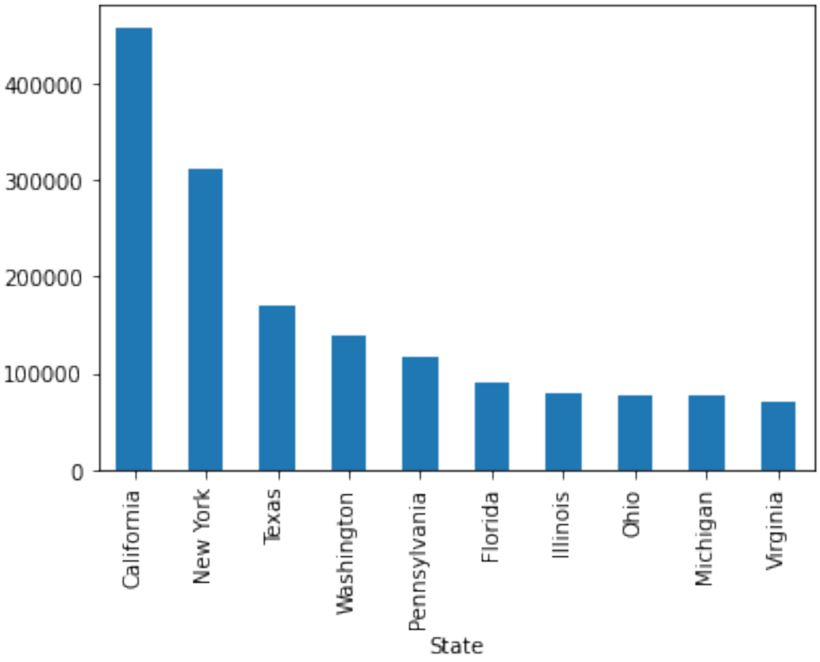
```
# TODO 09 - find top 10 popular products in terms of number of orders vs. tota
df_2019to2020 = df[(df['Order Date'] >= '2019-01-01') & (df['Order Date'] <=

df_2019to2020[['Product Name', 'Sales', 'Quantity']].groupby('Product Name')[
    .sum().sort_values('Quantity', ascending=False).head(10)
```

	Sales	Quantity
Product Name		
Staples	462.068	124
Easy-staple paper	1481.728	89
Staple envelope	644.936	73
Staples in misc. colors	357.164	60
Chromcraft Round Conference Tables	7965.053	59
Storex Dura Pro Binders	176.418	49
Situations Contoured Folding Chairs, 4/Set	2612.064	47
Wilson Jones Clip & Carry Folder Binder Tool for Ring Binders, Clear	178.060	44
Avery Non-Stick Binders	122.128	43
Eldon Wave Desk Accessories	215.924	42

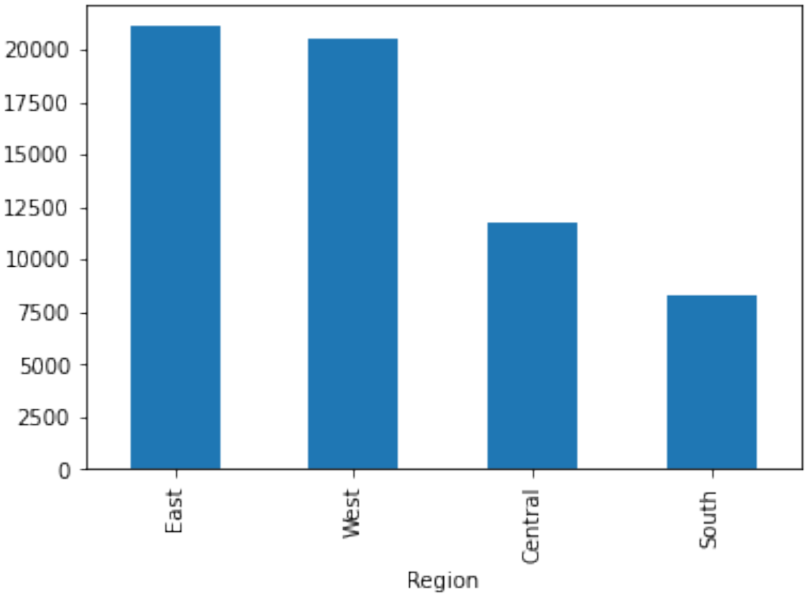
```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)
df.groupby('State')['Sales'].sum().sort_values(ascending=False).head(10).plot
```

Download



```
df_2018.groupby('Region')['Profit'].sum().sort_values(ascending = False).plot
```

Download



```
# TODO Bonus - use np.where() to create new column in dataframe to help you ar
import numpy as np
df['Profit_Check'] = np.where(df['Profit'] > 0, 'Profit', 'Loss')
df
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/
0	1	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United S
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United S
2	3	CA-2019-138688	6/12/2019	6/16/2019	Second Class	DV-13045	Darrin Van Huff	Corporate	United S
3	4	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United S
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United S
...	...	...	...	...	...	...	...	...	...
9989	9990	CA-2017-110422	1/21/2017	1/23/2017	Second Class	TB-21400	Tom Boeckenhauer	Consumer	United S

9990	9991	CA-2020-121258	2/26/2020	3/3/2020	Standard Class	DB-13060	Dave Brooks	Consumer	United S
9991	9992	CA-2020-121258	2/26/2020	3/3/2020	Standard Class	DB-13060	Dave Brooks	Consumer	United S
9992	9993	CA-2020-121258	2/26/2020	3/3/2020	Standard Class	DB-13060	Dave Brooks	Consumer	United S
9993	9994	CA-2020-119914	5/4/2020	5/9/2020	Second Class	CC-12220	Chris Cortes	Consumer	United S

9994 rows × 22 columns