Final Project - Analyzing Sales Data

Date: 28 December 2022

Author: Pongphisan Pisuttiwat (Seng)

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	City
0	1	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderso
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderso
2	3	CA-2019-138688	6/12/2019	6/16/2019	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles
3	4	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdal
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdal

5 rows × 21 columns

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

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...	P C
0	1	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	4
1	2	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	4
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	9
3	4	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	3
4	5	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	3

5 rows × 21 columns

```
# TODO - count nan in postal code column
df['Postal Code'].isna().sum()
df.isna().sum()
```

```
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
dtype: int64
```

```
# 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/Region	City	...
2234	2235	CA-2020-104066	2020-12-05	2020-12-10	Standard Class	QJ-19255	Quincy Jones	Corporate	United States	Burlington	...
5274	5275	CA-2018-162887	2018-11-07	2018-11-09	Second Class	SV-20785	Stewart Visinsky	Consumer	United States	Burlington	...
8798	8799	US-2019-150140	2019-04-06	2019-04-10	Standard Class	VM-21685	Valerie Mitchum	Home Office	United States	Burlington	...
9146	9147	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United States	Burlington	...
9147	9148	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United States	Burlington	...
9148	9149	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United States	Burlington	...
9386	9387	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9387	9388	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9388	9389	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9389	9390	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9741	9742	CA-2018-117086	2018-11-08	2018-11-12	Standard Class	QJ-19255	Quincy Jones	Corporate	United States	Burlington	...

11 rows × 21 columns

```
# TODO - How many order in each State
df.groupby('State')['Order ID'].count().sort_values(ascending = False)
```

```
State
California              2001
New York                1128
Texas                    985
Pennsylvania             587
Washington               506
Illinois                 492
Ohio                     469
Florida                  383
Michigan                 255
North Carolina           249
Virginia                 224
Arizona                  224
Georgia                  184
Tennessee                183
Colorado                 182
Indiana                  149
Kentucky                 139
Massachusetts            135
New Jersey               130
Oregon                   124
Wisconsin                110
Maryland                 105
Delaware                  96
Minnesota                 89
Connecticut               82
Oklahoma                  66
Missouri                  66
Alabama                   61
Arkansas                  60
Rhode Island              56
Utah                      53
Mississippi               53
South Carolina            42
Louisiana                 42
Nevada                    39
Nebraska                  38
New Mexico                37
Iowa                      30
New Hampshire             27
Kansas                    24
Idaho                     21
Montana                   15
South Dakota              12
Vermont                   11
District of Columbia      10
Maine                      8
North Dakota               7
West Virginia              4
Wyoming                    1
Name: Order ID, dtype: int64
```

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
```

```
(9994, 21)
```

```
# TODO 02 - is there any missing values?, if there is, which colunm? how many nan
df.isna().sum().sort_values(ascending = False).reset_index()
```

	index	0
0	Postal Code	11
1	Row ID	0
2	Discount	0
3	Quantity	0
4	Sales	0
5	Product Name	0
6	Sub-Category	0
7	Category	0
8	Product ID	0
9	Region	0
10	State	0
11	Order ID	0
12	City	0
13	Country/Region	0
14	Segment	0
15	Customer Name	0
16	Customer ID	0
17	Ship Mode	0
18	Ship Date	0
19	Order Date	0
20	Profit	0

```
# TODO 03 - your friend ask for `California` data, filter it and export csv for h
df.query('State == "California"').to_csv('California data.csv')
```

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in 201
California_Texas_2017_Data = df[((df['State'] == 'California') | (df['State'] ==
                                 & (df['Order Date'] > '2016-12-31') & (df
California_Texas_2017_Data.to_csv('California_Texas_2017_Data.csv')
```

```
# TODO 05 - how much total sales, average sales, and standard deviation of sales
df[(df['Order Date'] > '2016-12-31') & (df['Order Date'] < '2018-01-01')]\
    ['Sales'].agg(['sum', 'mean', 'std']).reset_index()
```

	index	Sales
0	sum	484247.498100
1	mean	242.974159
2	std	754.053357

```
# TODO 06 - which Segment has the highest profit in 2018
df.groupby('Segment')['Profit'].sum().sort_values(ascending = False).reset_index(
```

	Segment	Profit
0	Consumer	134119.2092
1	Corporate	91979.1340
2	Home Office	60298.6785

```
# TODO 07 - which top 5 States have the least total sales between 15 April 2019 -
df[(df['Order Date'] > '2019-04-15') & (df['Order Date'] < '2019-12-31')]\
    .groupby('State')['Sales'].sum().sort_values().reset_index().head(5)
```

	State	Sales
0	New Hampshire	49.05
1	New Mexico	64.08
2	District of Columbia	117.07
3	Louisiana	249.80
4	South Carolina	502.48

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 2019 e
data2019 = df[(df['Order Date'] > '2018-12-31') & (df['Order Date'] < '2020-01-01
West_Central_sales = data2019[(data2019['Region'] == "Central") | (data2019['Regi
Total_sales = data2019['Sales'].sum()
print(str(round((West_Central_sales/Total_sales*100),2))+'%')
```

```
54.97%
```

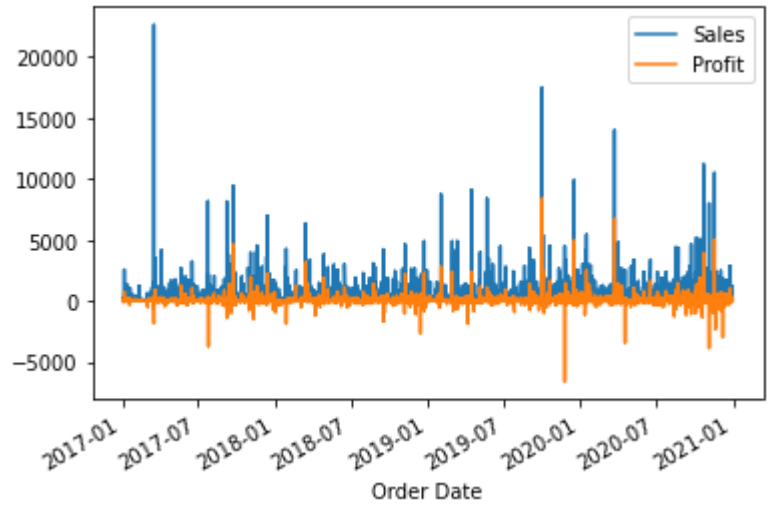
```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total s
PopularByOrder = df[(df['Order Date'] > '2018-12-31') & (df['Order Date'] < '2021
            ['Order ID'].count().sort_values(ascending = False).head(10).reset_in
PopularBySales = df[(df['Order Date'] > '2018-12-31') & (df['Order Date'] < '2021
            ['Sales'].sum().sort_values(ascending = False).head(10).reset_index()
PopularByOrder
PopularBySales
```

	Product Name	Sales
0	Canon imageCLASS 2200 Advanced Copier	61599.824
1	Hewlett Packard LaserJet 3310 Copier	16079.732
2	3D Systems Cube Printer, 2nd Generation, Magenta	14299.890
3	GBC Ibimaster 500 Manual ProClick Binding System	13621.542
4	GBC DocuBind TL300 Electric Binding System	12737.258
5	GBC DocuBind P400 Electric Binding System	12521.108
6	Samsung Galaxy Mega 6.3	12263.708
7	HON 5400 Series Task Chairs for Big and Tall	11846.562
8	Martin Yale Chadless Opener Electric Letter Op...	11825.902
9	Global Troy Executive Leather Low-Back Tilter	10169.894

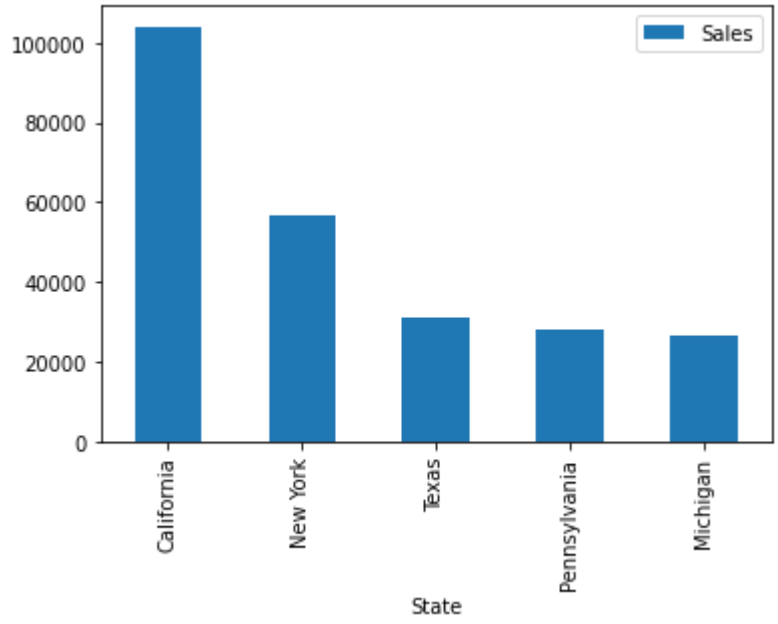
```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)
df[['Sales', 'Order Date', 'Profit']].plot(x='Order Date', y=['Sales', 'Profit'],

df[(df['Order Date'] > '2019-04-15') & (df['Order Date'] < '2019-12-31')]\
    .groupby('State')['Sales'].sum().sort_values(ascending= False).reset_index().
    [['State', 'Sales']].plot(x='State', y='Sales', kind='bar')
```

```
<AxesSubplot:xlabel='State'>
```

Download



Download




```
# TODO Bonus - use np.where() to create new column in dataframe to help you answe
import numpy as np
df['OrderResult']= np.where(df['Profit']<0,'loss', 'Profit')
df.groupby('OrderResult')['OrderResult'].count()
```

```
OrderResult
Profit    8123
loss      1871
Name: OrderResult, dtype: int64
```

```
0
```