Final Project - Analyzing Sales Data

Date: 30 December 2021

Author: Kasidis Satangmongkol (Toy DataRockie)

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	Henderson
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson
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 Lauderdale
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale

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]
```

```python
# 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(10)
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...	Pos Coc
0	1	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	424
1	2	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	424
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	900
3	4	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	333
4	5	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	333
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	900
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	900
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	900
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	900
9	10	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	900

10 rows × 21 columns

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

```
11
```

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

df_postal_missing = df[ df["Postal Code"].isna() ]
df_postal_missing
```

	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 - 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
```

```
(9994, 21)
```

```
# TODO 02 - is there any missing values?, if there is, which colunm? how many nan v

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 03 - your friend ask for `California` data, filter it and export csv for hi

df_California = df[df['State'] == 'California'].dropna()
df_California.to_csv("df_California")
```

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in 2017

df_California_Eexas = df.query('State == "California" | State == "Texas"').dropna()
df_California_Eexas_2017 = df_California_Eexas[df_California_Eexas['Order Date'].dt
df_California_Eexas_2017.to_csv("df_California_Eexas_2017")
df_California_Eexas_2017
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...	P C
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9
9	10	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9
...	...	...	...	...	...	...	...	...	...	...	...	...
9885	9886	CA-2017-112291	2017-04-03	2017-04-08	Standard Class	KE-16420	Katrina Edelman	Corporate	United States	Los Angeles	...	9
9903	9904	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United States	Carrollton	...	7
9904	9905	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United States	Carrollton	...	7
9942	9943	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United States	Anaheim	...	9.
9943	9944	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United States	Anaheim	...	9.

632 rows × 21 columns

```
# TODO 05 - how much total sales, average sales, and standard deviation of sales yo


df2017_2  =  df[df["Order Date"].dt.year == 2017].dropna()
df2017_2['Sales'].agg(['sum', 'mean', 'std']).reindex()
```

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

```
# TODO 06 - which Segment has the highest profit in 2018

df2018 = df[df["Order Date"].dt.year == 2018].dropna()
df2018.groupby('Segment')["Profit"].sum().reindex()
```

```
Segment
Consumer       28281.3665
Corporate      19675.1978
Home Office    12470.1124
Name: Profit, dtype: float64
```

```
# TODO 07 - which top 5 States have the least total sales between 15 April 2019 - 3
df = df.dropna()
df04_12_2019 = df [ (df['Order Date'] >= "2019-04-15") & (df['Order Date'] <= "2019
df04_12_2019.groupby('State').sum('Sales').reset_index().sort_values('Sales', ascer
```

	State	Row ID	Postal Code	Sales	Quantity	Discount	Profit
26	New Hampshire	7208	6361.0	49.05	7	0.0	14.6469
28	New Mexico	23311	352880.0	64.08	11	0.6	24.9520
7	District of Columbia	11159	100080.0	117.07	18	0.0	50.2118
16	Louisiana	26138	281839.0	249.80	17	0.0	82.0472
36	South Carolina	58770	321531.0	502.48	42	0.0	144.1038

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 2019 e.g

#total sales

df2019_total = df[df["Order Date"].dt.year == 2019]['Sales'].sum()

#Total West & Central

wc_tabel = df.query('Region  == "West" | Region == "Central" ')
wc_value = wc_tabel[wc_tabel["Order Date"].dt.year == 2019]['Sales'].sum()
proportion = (wc_value /df2019_total )*100
proportion = proportion.round(3)
print(f'Proportion of total sales :{proportion}'+'%')
```

```
Proportion of total sales :55.244%
```

```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total sal

df_10popular = df[(df["Order Date"].dt.year == 2019) | (df["Order Date"].dt.year ==
.agg("sum")[['Quantity','Sales']].reset_index()
df_10popular["grand total"] = df_10popular["Quantity"] * df_10popular["Sales"]
df_10popular.sort_values('grand total', ascending = False).head(10).round(2)


## Note
# top_ten = df[df['Order Date'].dt.year.isin([2019,2020])] \
 #   .groupby('Product Name') \
  #  .agg('sum') \
   # .sort_values(by='Quantity', ascending=False) \
    #.round(decimals=2) \
#     .head(10) [['Quantity','Sales']]
```

	Product Name	Quantity	Sales	grand total
387	Canon imageCLASS 2200 Advanced Copier	20	61599.82	1231996.48
764	Hewlett Packard LaserJet 3310 Copier	31	16079.73	498471.69
410	Chromcraft Round Conference Tables	59	7965.05	469938.13
650	GBC Ibimaster 500 Manual ProClick Binding System	31	13621.54	422267.80
1309	Samsung Galaxy Mega 6.3	34	12263.71	416966.07
793	Hon Deluxe Fabric Upholstered Stacking Chairs,...	38	8222.13	312440.79
969	Logitech P710e Mobile Speakerphone	35	8806.16	308215.53
648	GBC DocuBind TL300 Electric Binding System	21	12737.26	267482.42
728	Global Troy Executive Leather Low-Back Tilter	25	10169.89	254247.35
745	HON 5400 Series Task Chairs for Big and Tall	21	11846.56	248777.80

```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)

## 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
 #1#5  Sub-Category    9994 non-null   object
 #16  Product Name    9994 non-null   object
 #17  Sales           9994 non-null   float64
 #18  Quantity        9994 non-null   int64
 #19  Discount        9994 non-null   float64
 #20  Profit

# df_State = df[1:].dropna().groupby("State")["Sales"].agg("sum")\
 #   .reset_index().sort_values('Sales', ascending = False)\
 #   .head(15).plot(kind = "bar", color = "Salmon");

df = df.drop(columns=['Row ID'])
```

```
# TODO Bonus - use np.where() to create new column in dataframe to help you answer
```