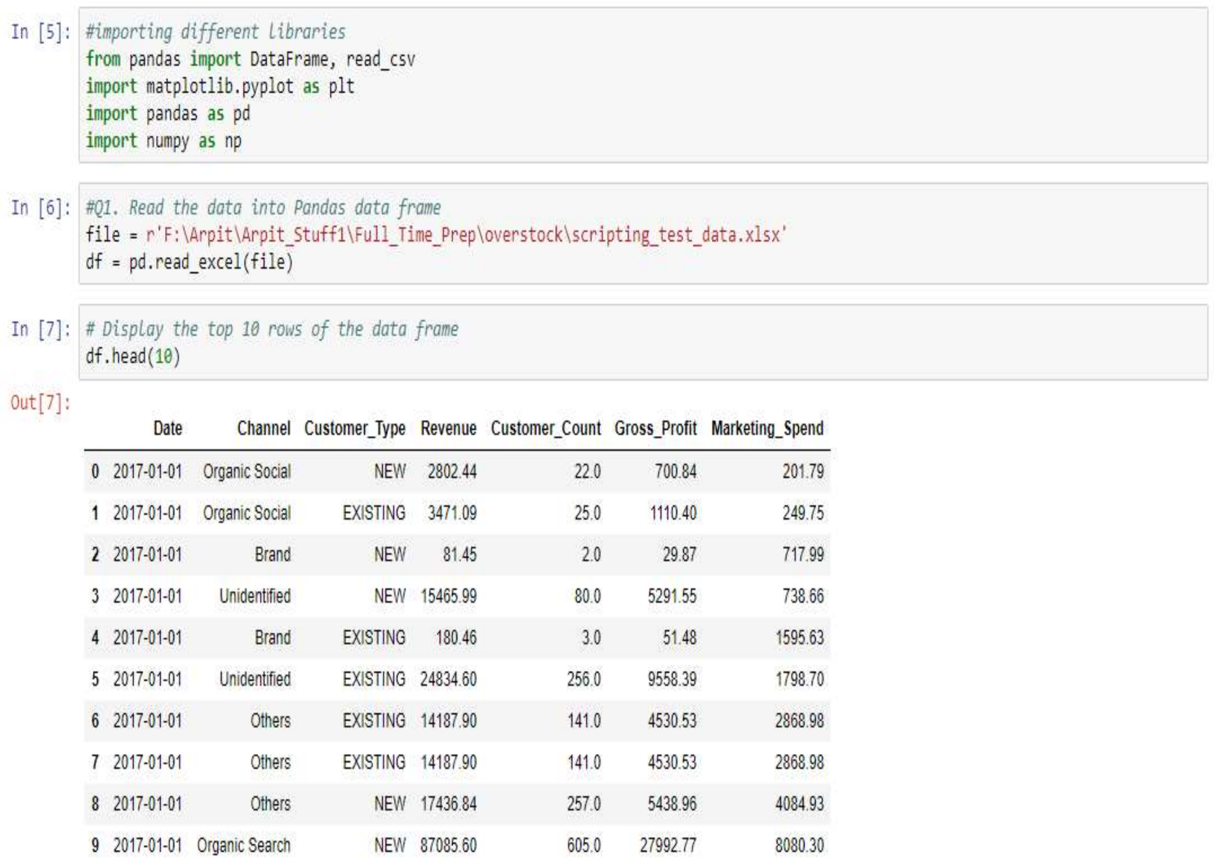
Report Scripting Test:

The objective of this exercise is to test your ability to read, clean, transform, visualize and interpret data using R or Python. You are provided a CSV file with some sample marketing sales data for an e-commerce website. You may choose either R or Python to complete the exercise. If you prefer, then you may also use R markdown or Python notebook format.

Q1. Read the data into an R or Pandas data frame. Display the top 10 rows of the data frame.

Solution:



In [5]:
```python
#importing different libraries
from pandas import DataFrame, read_csv
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

In [6]:
```python
#Q1. Read the data into Pandas data frame
file = r'F:\Arpit\Arpit_Stuff1\Full_Time_Prep\overstock\scripting_test_data.xlsx'
df = pd.read_excel(file)
```

In [7]:
```python
# Display the top 10 rows of the data frame
df.head(10)
```

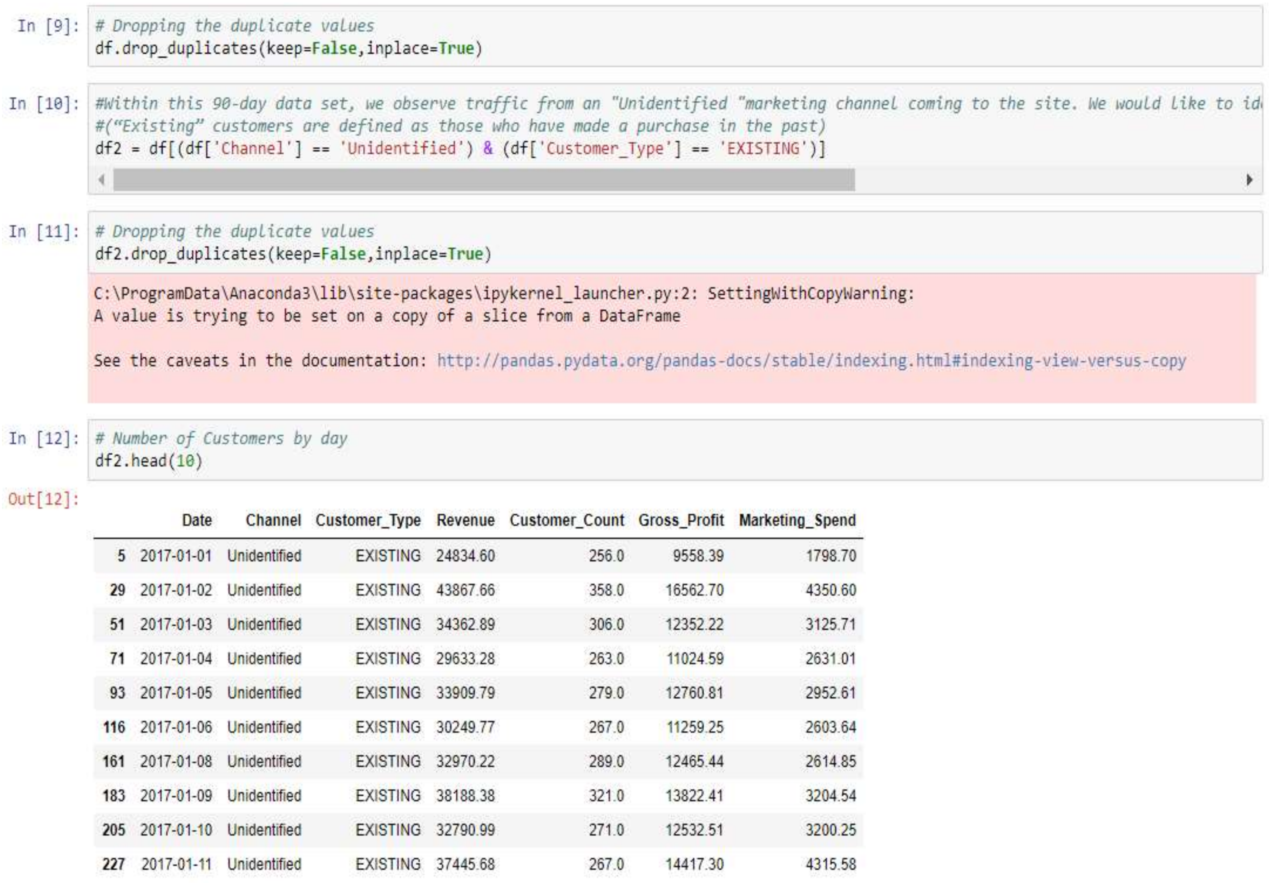
Out[7]:

	Date	Channel	Customer_Type	Revenue	Customer_Count	Gross_Profit	Marketing_Spend
0	2017-01-01	Organic Social	NEW	2802.44	22.0	700.84	201.79
1	2017-01-01	Organic Social	EXISTING	3471.09	25.0	1110.40	249.75
2	2017-01-01	Brand	NEW	81.45	2.0	29.87	717.99
3	2017-01-01	Unidentified	NEW	15465.99	80.0	5291.55	738.66
4	2017-01-01	Brand	EXISTING	180.46	3.0	51.48	1595.63
5	2017-01-01	Unidentified	EXISTING	24834.60	256.0	9558.39	1798.70
6	2017-01-01	Others	EXISTING	14187.90	141.0	4530.53	2868.98
7	2017-01-01	Others	EXISTING	14187.90	141.0	4530.53	2868.98
8	2017-01-01	Others	NEW	17436.84	257.0	5438.96	4084.93
9	2017-01-01	Organic Search	NEW	87085.60	605.0	27992.77	8080.30

Q2. Within this 90-day data set, we observe traffic from an "Unidentified "marketing channel coming to the site. We would like to identify **the number of "existing" customers by day** from this channel. ("Existing" customers are defined as those who have made a purchase in the past)

- Please demonstrate this result in a visualization. Are there any insights that you could derive from the visualization?
- What is the **total spend amount** and **daily average spend** on these existing customers?

Solution:



```
In [9]: # Dropping the duplicate values
        df.drop_duplicates(keep=False,inplace=True)
```

```
In [10]: #Within this 90-day data set, we observe traffic from an "Unidentified "marketing channel coming to the site. We would like to id
         #("Existing" customers are defined as those who have made a purchase in the past)
         df2 = df[(df['Channel'] == 'Unidentified') & (df['Customer_Type'] == 'EXISTING')]
```

```
In [11]: # Dropping the duplicate values
         df2.drop_duplicates(keep=False,inplace=True)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
```

```
In [12]: # Number of Customers by day
         df2.head(10)
```

Out[12]:

	Date	Channel	Customer_Type	Revenue	Customer_Count	Gross_Profit	Marketing_Spend
5	2017-01-01	Unidentified	EXISTING	24834.60	256.0	9558.39	1798.70
29	2017-01-02	Unidentified	EXISTING	43867.66	358.0	16562.70	4350.60
51	2017-01-03	Unidentified	EXISTING	34362.89	306.0	12352.22	3125.71
71	2017-01-04	Unidentified	EXISTING	29633.28	263.0	11024.59	2631.01
93	2017-01-05	Unidentified	EXISTING	33909.79	279.0	12760.81	2952.61
116	2017-01-06	Unidentified	EXISTING	30249.77	267.0	11259.25	2603.64
161	2017-01-08	Unidentified	EXISTING	32970.22	289.0	12465.44	2614.85
183	2017-01-09	Unidentified	EXISTING	38188.38	321.0	13822.41	3204.54
205	2017-01-10	Unidentified	EXISTING	32790.99	271.0	12532.51	3200.25
227	2017-01-11	Unidentified	EXISTING	37445.68	267.0	14417.30	4315.58

```
In [13]: #the total spend amount and daily average spend on these existing customers?
         df2['Total_spent'] = df2['Marketing_Spend'] + df2['Gross_Profit']
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py: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: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
```

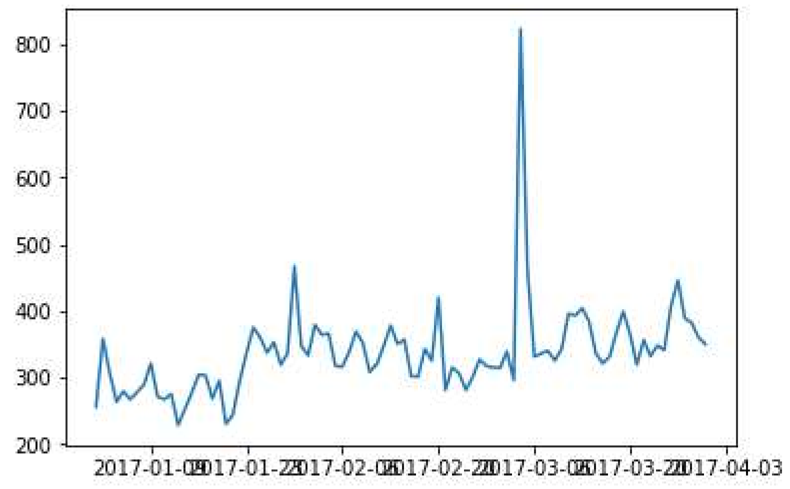
```
In [14]: # Display the total spend amount on these existing customers
         df2.head(10)
```

Out[14]:

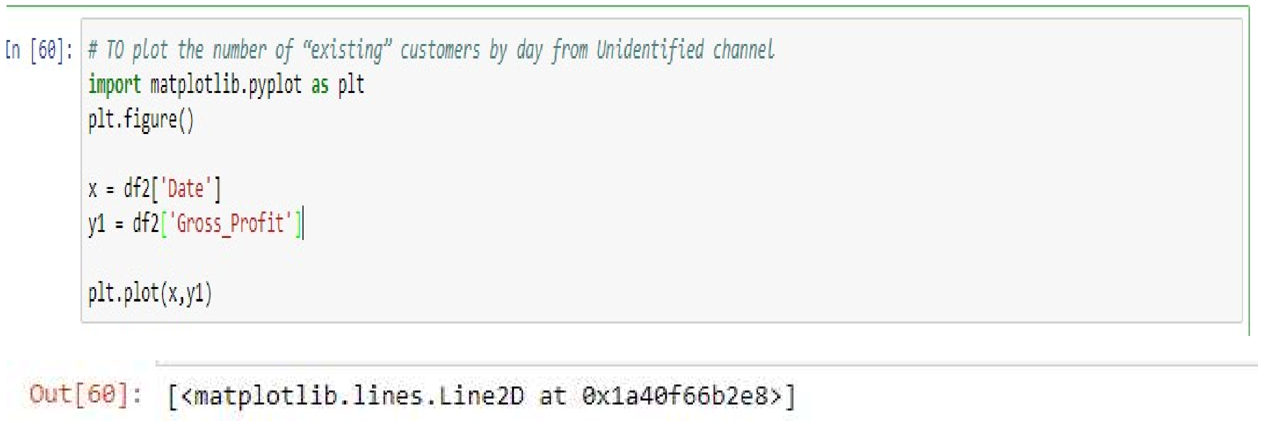
	Date	Channel	Customer_Type	Revenue	Customer_Count	Gross_Profit	Marketing_Spend	Total_spent
5	2017-01-01	Unidentified	EXISTING	24834.60	256.0	9558.39	1798.70	11357.09
29	2017-01-02	Unidentified	EXISTING	43867.66	358.0	16562.70	4350.60	20913.30
51	2017-01-03	Unidentified	EXISTING	34362.89	306.0	12352.22	3125.71	15477.93
71	2017-01-04	Unidentified	EXISTING	29633.28	263.0	11024.59	2631.01	13655.60
93	2017-01-05	Unidentified	EXISTING	33909.79	279.0	12760.81	2952.61	15713.42
116	2017-01-06	Unidentified	EXISTING	30249.77	267.0	11259.25	2603.64	13862.89
161	2017-01-08	Unidentified	EXISTING	32970.22	289.0	12465.44	2614.85	15080.29
183	2017-01-09	Unidentified	EXISTING	38188.38	321.0	13822.41	3204.54	17026.95
205	2017-01-10	Unidentified	EXISTING	32790.99	271.0	12532.51	3200.25	15732.76
227	2017-01-11	Unidentified	EXISTING	37445.68	267.0	14417.30	4315.58	18732.88

```
In [56]: # TO plot the number of “existing” customers by day from Unidentified channel
         import matplotlib.pyplot as plt
         plt.figure()

         x = df2['Date']
         y1 = df2['Customer_Count']

         plt.plot(x,y1)
```

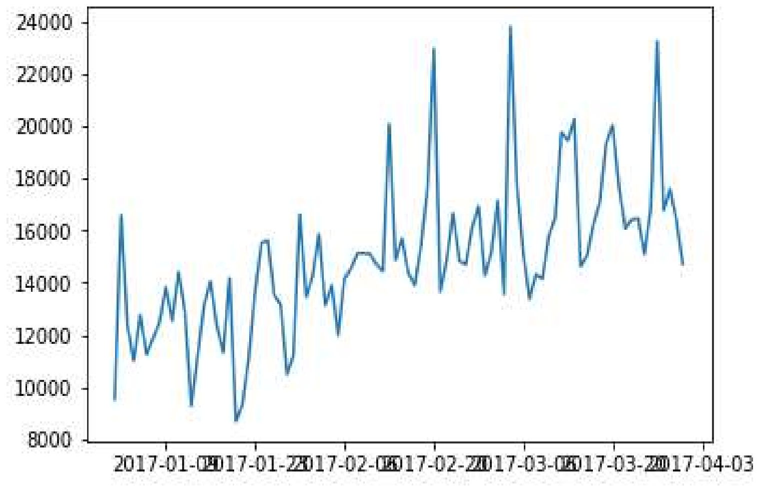
```
Out[56]: [<matplotlib.lines.Line2D at 0x1a40f51b828>]
```

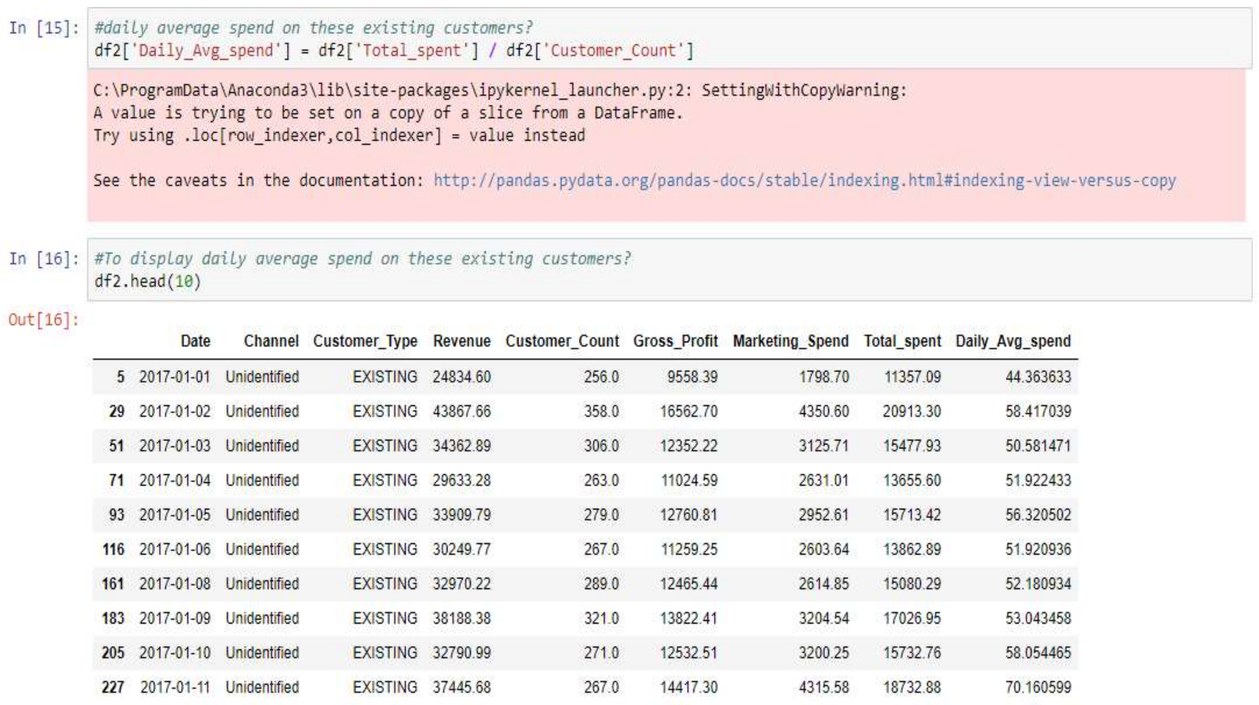


Insights: By above figure we found that there is spike in the count of existing customer coming from Unidentified marketing channel to the site in the later month of February which resulted in increase of gross profit as well it can be demonstrated by below graph:



```
In [60]: # TO plot the number of "existing" customers by day from Unidentified channel
         import matplotlib.pyplot as plt
         plt.figure()

         x = df2['Date']
         y1 = df2['Gross_Profit']

         plt.plot(x,y1)

Out[60]: [<matplotlib.lines.Line2D at 0x1a40f66b2e8>]
```





In [15]:
```
#daily average spend on these existing customers?
df2['Daily_Avg_spend'] = df2['Total_spent'] / df2['Customer_Count']
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py: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: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
```

In [16]:
```
#To display daily average spend on these existing customers?
df2.head(10)
```

Out[16]:

	Date	Channel	Customer_Type	Revenue	Customer_Count	Gross_Profit	Marketing_Spend	Total_spent	Daily_Avg_spend
5	2017-01-01	Unidentified	EXISTING	24834.60	256.0	9558.39	1798.70	11357.09	44.363633
29	2017-01-02	Unidentified	EXISTING	43867.66	358.0	16562.70	4350.60	20913.30	58.417039
51	2017-01-03	Unidentified	EXISTING	34362.89	306.0	12352.22	3125.71	15477.93	50.581471
71	2017-01-04	Unidentified	EXISTING	29633.28	263.0	11024.59	2631.01	13655.60	51.922433
93	2017-01-05	Unidentified	EXISTING	33909.79	279.0	12760.81	2952.61	15713.42	56.320502
116	2017-01-06	Unidentified	EXISTING	30249.77	267.0	11259.25	2603.64	13862.89	51.920936
161	2017-01-08	Unidentified	EXISTING	32970.22	289.0	12465.44	2614.85	15080.29	52.180934
183	2017-01-09	Unidentified	EXISTING	38188.38	321.0	13822.41	3204.54	17026.95	53.043458
205	2017-01-10	Unidentified	EXISTING	32790.99	271.0	12532.51	3200.25	15732.76	58.054465
227	2017-01-11	Unidentified	EXISTING	37445.68	267.0	14417.30	4315.58	18732.88	70.160599

Q3. We believe that the **Brand channel drives more New Customers than Existing Customers**. We would like to verify this statement using the data provided. Please provide the daily difference between Brand channel's New Customers and Existing Customers using a visualization. Are there any insights from the visualization?

```
In [17]: df3_a = df[(df['Channel'] == 'Brand') & (df['Customer_Type'] == 'NEW')]
```

```
In [18]: # To drop NULL values like NAN
         df3_a.dropna()
```

```
245  2017-01-12  Brand  NEW  338.22  2.0  96.23  2190.22
288  2017-01-14  Brand  NEW    9.22  1.0   4.33   901.25
311  2017-01-15  Brand  NEW   85.13  1.0  16.56  2288.74
331  2017-01-16  Brand  NEW   25.08  1.0   9.94   215.42
353  2017-01-17  Brand  NEW   99.98  1.0  29.35  1115.84
376  2017-01-18  Brand  NEW   75.90  1.0  32.73   595.29
398  2017-01-19  Brand  NEW   15.82  1.0  10.01   257.73
423  2017-01-20  Brand  NEW   99.17  1.0  26.49  2287.03
485  2017-01-23  Brand  NEW   45.87  1.0   7.74  2288.61
505  2017-01-24  Brand  NEW   25.95  1.0  10.00   786.94
529  2017-01-25  Brand  NEW  188.26  2.0  91.52  1993.83
572  2017-01-27  Brand  NEW   87.77  1.0  32.42  1913.77
634  2017-01-30  Brand  NEW   50.19  1.0  20.14   523.80
```

```
In [19]: # Dropping the duplicate values
         df3_a.drop_duplicates(keep=False,inplace=True)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
```

```
In [20]: df3_a.sort_index(inplace=True)
```

```
In [21]: df3_b = df[(df['Channel'] == 'Brand') & (df['Customer_Type'] == 'EXISTING')]
In [22]: # To drop NULL values like NAN
         df3_b.dropna()
```

1741	2017-03-21	Brand	EXISTING	652.77	3.0	194.52	6558.94
1763	2017-03-22	Brand	EXISTING	1301.77	3.0	422.69	6474.28
1785	2017-03-23	Brand	EXISTING	550.23	4.0	130.87	4873.88
1808	2017-03-24	Brand	EXISTING	655.32	5.0	232.54	5752.67
1828	2017-03-25	Brand	EXISTING	325.81	3.0	105.65	3831.79
1851	2017-03-26	Brand	EXISTING	536.96	4.0	177.69	6233.74
1870	2017-03-27	Brand	EXISTING	408.59	4.0	127.10	3731.70
1892	2017-03-28	Brand	EXISTING	164.36	2.0	58.14	2820.61
1917	2017-03-29	Brand	EXISTING	282.65	4.0	88.90	5125.06
1939	2017-03-30	Brand	EXISTING	590.71	4.0	188.67	4293.00
1957	2017-03-31	Brand	EXISTING	83.15	1.0	29.17	1264.08

80 rows × 7 columns

```
In [23]: # Dropping the duplicate values
         df3_b.drop_duplicates(keep=False,inplace=True)

         C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: SettingWithCopyWarning:
         A value is trying to be set on a copy of a slice from a DataFrame

         See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy

In [24]: df3_b.sort_index(inplace=True)

In [25]: #the total number of Existing customers Brand channel drives
         df3_b['Customer_Count'].sum()

Out[25]: 197.0

In [26]: #the total number of New customers Brand channel drives
         df3_a['Customer_Count'].sum()

Out[26]: 171.0
```

In [27]:
```
#the daily difference between Brand channel’s New Customers and Existing Customers
#df3_a['Daily_Diff_Customers'] = df3_b['Customer_Count'] - df3_a['Customer_Count']
```

In [28]:
```
df3_a1=df3_a.rename(columns={"Customer_Count":"New_Cust_Count"})
```

In [29]:
```
df3_a1.head()
```

Out[29]:

	Date	Channel	Customer_Type	Revenue	New_Cust_Count	Gross_Profit	Marketing_Spend
2	2017-01-01	Brand	NEW	81.45	2.0	29.87	717.99
26	2017-01-02	Brand	NEW	496.48	1.0	52.66	1791.89
49	2017-01-03	Brand	NEW	28.05	1.0	9.67	1580.03
70	2017-01-04	Brand	NEW	54.07	1.0	12.95	2286.60
90	2017-01-05	Brand	NEW	16.99	1.0	2.47	408.24

In [30]:
```
df3_a1.reset_index(drop=True)
```

Out[30]:

	Date	Channel	Customer_Type	Revenue	New_Cust_Count	Gross_Profit	Marketing_Spend
0	2017-01-01	Brand	NEW	81.45	2.0	29.87	717.99
1	2017-01-02	Brand	NEW	496.48	1.0	52.66	1791.89
2	2017-01-03	Brand	NEW	28.05	1.0	9.67	1580.03
3	2017-01-04	Brand	NEW	54.07	1.0	12.95	2286.60
4	2017-01-05	Brand	NEW	16.99	1.0	2.47	408.24
5	2017-01-06	Brand	NEW	18.81	1.0	4.63	584.24
6	2017-01-07	Brand	NEW	188.57	2.0	36.29	1301.63
7	2017-01-08	Brand	NEW	NaN	NaN	NaN	NaN
8	2017-01-10	Brand	NEW	164.51	1.0	68.69	2291.66
9	2017-01-12	Brand	NEW	338.22	2.0	96.23	2190.22
10	2017-01-14	Brand	NEW	9.22	1.0	4.33	901.25

In [31]:
```
df3_b1=df3_b.rename({"Customer_Count":"Ext_Cust_Count"},axis=1)
```

```
In [32]: df3_b1.head()
```

Out[32]:

	Date	Channel	Customer_Type	Revenue	Ext_Cust_Count	Gross_Profit	Marketing_Spend
4	2017-01-01	Brand	EXISTING	180.46	3.0	51.48	1595.63
25	2017-01-02	Brand	EXISTING	139.40	2.0	74.09	499.66
47	2017-01-03	Brand	EXISTING	12.63	1.0	2.65	709.78
92	2017-01-05	Brand	EXISTING	78.20	2.0	30.13	1879.37
115	2017-01-06	Brand	EXISTING	54.60	1.0	23.94	1703.65

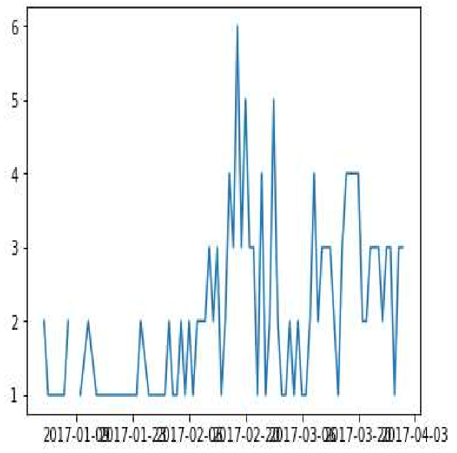
```
In [33]: #Ext_cust_column = df3_b1['Ext_Cust_Count']
         #df3_a1 = pd.concat([df3_a1,Ext_cust_column],axis=1)
```

```
In [34]: df3_b1.reset_index(drop=True)
```

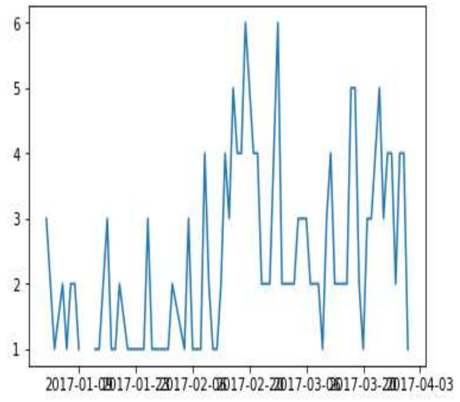
8	2017-01-12	Brand	EXISTING	NaN	NaN	NaN	NaN
9	2017-01-13	Brand	EXISTING	24.09	1.0	2.59	2286.35
10	2017-01-14	Brand	EXISTING	14.17	1.0	7.63	1387.12
11	2017-01-16	Brand	EXISTING	245.80	3.0	80.05	2082.48
12	2017-01-17	Brand	EXISTING	119.48	1.0	48.58	1179.63
13	2017-01-18	Brand	EXISTING	220.08	1.0	29.83	1701.80
14	2017-01-19	Brand	EXISTING	124.58	2.0	25.09	2031.98
15	2017-01-21	Brand	EXISTING	128.35	1.0	51.26	2287.27
16	2017-01-22	Brand	EXISTING	26.72	1.0	5.85	2286.36
17	2017-01-24	Brand	EXISTING	49.50	1.0	22.65	1499.77
18	2017-01-25	Brand	EXISTING	28.80	1.0	9.73	307.02
19	2017-01-26	Brand	EXISTING	358.02	3.0	81.67	2305.26
20	2017-01-27	Brand	EXISTING	17.12	1.0	13.53	373.31

```
In [55]: # TO plot number of New customer for Brand channel on a daily basis for 90 days
         import matplotlib.pyplot as plt
         plt.figure()

         x = df3_a1['Date']
         y1 = df3_a1['New_Cust_Count']

         plt.plot(x,y1)
```

```
Out[55]: [<matplotlib.lines.Line2D at 0x1a40f4afcf8>]
```



```
In [54]: # TO plot number of existing customer for Brand channel on a daily basis for 90 days
         import matplotlib.pyplot as plt
         plt.figure()

         x = df3_b1['Date']
         y1 = df3_b1['Ext_Cust_Count']

         plt.plot(x,y1)

Out[54]: [<matplotlib.lines.Line2D at 0x1a40f489fd0>]
```



Insights:

- So the null hypothesis which says that **Brand channel drives more New Customers than Existing Customers is wrong as we derived the numbers we found that Brand channel is able to drive more Existing Customers of total 197 than New Customers of total 171**
- Also, through above graph we found that both the count of existing customers and new customers are getting increased in middle of the February month spike can be seen in both the graph.
- Counts for both the new customer and existing customer is constant in the month of January.

Q4. Please calculate 'Marketing_Contribution' using formula: ***Marketing_Contribution = Gross_Profit - Marketing_Spend***. Make a plot of the daily Marketing_Contribution at the aggerated level (for all channels and customer types together)

In [37]:
```
# To calculate 'Marketing_Contribution' using formula: Marketing_Contribution = Gross_Profit - Marketing_Spend

df4 = df.copy()
```

In [38]:
```
df4.head()
```

Out[38]:

	Date	Channel	Customer_Type	Revenue	Customer_Count	Gross_Profit	Marketing_Spend
0	2017-01-01	Organic Social	NEW	2802.44	22.0	700.84	201.79
1	2017-01-01	Organic Social	EXISTING	3471.09	25.0	1110.40	249.75
2	2017-01-01	Brand	NEW	81.45	2.0	29.87	717.99
3	2017-01-01	Unidentified	NEW	15465.99	80.0	5291.55	738.66
4	2017-01-01	Brand	EXISTING	180.46	3.0	51.48	1595.63

In [39]:
```
# To calculate 'Marketing_Contribution' using formula: Marketing_Contribution = Gross_Profit - Marketing_Spend

df4['Marketing_Contribution'] = df4['Gross_Profit'] / df4['Marketing_Spend']
```

In [40]:
```
df4.head()
```

Out[40]:

	Date	Channel	Customer_Type	Revenue	Customer_Count	Gross_Profit	Marketing_Spend	Marketing_Contribution
0	2017-01-01	Organic Social	NEW	2802.44	22.0	700.84	201.79	3.473116
1	2017-01-01	Organic Social	EXISTING	3471.09	25.0	1110.40	249.75	4.446046
2	2017-01-01	Brand	NEW	81.45	2.0	29.87	717.99	0.041602
3	2017-01-01	Unidentified	NEW	15465.99	80.0	5291.55	738.66	7.163715
4	2017-01-01	Brand	EXISTING	180.46	3.0	51.48	1595.63	0.032263

In [41]:
```
import matplotlib.pyplot as plt
```

In [44]:
```
import seaborn as sns
%matplotlib inline
#to plot the graphs inline on jupyter noteboo
```

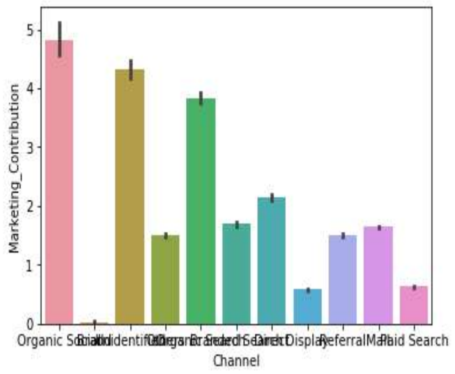
In [46]:
```
df4.head()
```

Out[46]:

	Date	Channel	Customer_Type	Revenue	Customer_Count	Gross_Profit	Marketing_Spend	Marketing_Contribution
0	2017-01-01	Organic Social	NEW	2802.44	22.0	700.84	201.79	3.473116
1	2017-01-01	Organic Social	EXISTING	3471.09	25.0	1110.40	249.75	4.446046
2	2017-01-01	Brand	NEW	81.45	2.0	29.87	717.99	0.041602
3	2017-01-01	Unidentified	NEW	15465.99	80.0	5291.55	738.66	7.163715
4	2017-01-01	Brand	EXISTING	180.46	3.0	51.48	1595.63	0.032263

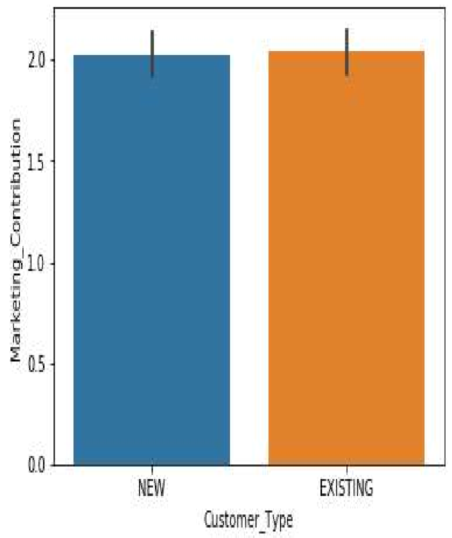
In [47]:
```
# plot of the daily Marketing_Contribution at the aggerated level for all channel
sns.barplot(x=df4.Channel, y=df4.Marketing_Contribution)
```

Out[47]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a40f156eb8>
```



In [48]:
```
# plot of the daily Marketing_Contribution at the aggerated level for all channel

sns.barplot(x=df4.Customer_Type, y=df4.Marketing_Contribution)
```

Out[48]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a40f167a90>
```



Insights:

- From the above figure we can find that for both customer type new and existing we have almost similar Marketing_contribution value which means that both types of customers are contributing equally.
- From the above figure we found that Organic social channel is having the highest marketing_contribution among all whereas Referral channel is the least contributor to the marketing.