In [5]:
```
# Import required library
import pandas as pd
import numpy as np
print("All libray imported")
```

```
All libray imported
```

In [6]:
```
# import data
data=pd.read_csv('UberDrives2016.csv.crdownload',skipfooter=1,engine='python')
```

In [7]:
```
data
```

Out[7]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
...	...	...	...	...	...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	Kar?chi	0.7	Meeting
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site

1155 rows × 7 columns

In [8]:
```
# shape of data
data.shape
```

Out[8]:
```
(1155, 7)
```

In [9]:
```
type(data)
```

Out[9]:
```
pandas.core.frame.DataFrame
```

In [10]:
```
# print first 5 rows of data
data.head()
```

Out[10]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit

In [11]:
```
data.head(15)
```

Out[11]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
5	1/6/2016 17:15	1/6/2016 17:19	Business	West Palm Beach	West Palm Beach	4.3	Meal/Entertain
6	1/6/2016 17:30	1/6/2016 17:35	Business	West Palm Beach	Palm Beach	7.1	Meeting
7	1/7/2016 13:27	1/7/2016 13:33	Business	Cary	Cary	0.8	Meeting
8	1/10/2016 8:05	1/10/2016 8:25	Business	Cary	Morrisville	8.3	Meeting
9	1/10/2016 12:17	1/10/2016 12:44	Business	Jamaica	New York	16.5	Customer Visit
10	1/10/2016 15:08	1/10/2016 15:51	Business	New York	Queens	10.8	Meeting
11	1/10/2016 18:18	1/10/2016 18:53	Business	Elmhurst	New York	7.5	Meeting
12	1/10/2016 19:12	1/10/2016 19:32	Business	Midtown	East Harlem	6.2	Meeting
13	1/11/2016 8:55	1/11/2016 9:21	Business	East Harlem	NoMad	6.4	Temporary Site
14	1/11/2016 11:56	1/11/2016 12:03	Business	Flatiron District	Midtown	1.6	Errand/Supplies

In [12]:
```
# print last 5 rows of data
data.tail()
```

Out[12]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	Kar?chi	0.7	Meeting
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site

In [13]:
```
# data types of all columns
data.dtypes
```

Out[13]:
```
START_DATE*     object
END_DATE*       object
CATEGORY*       object
START*          object
```

```
STOP*           object
MILES*         float64
PURPOSE*        object
dtype: object
```

In [14]:

```
# Print complete information about the data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1155 entries, 0 to 1154
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   START_DATE*  1155 non-null   object 
 1   END_DATE*    1155 non-null   object 
 2   CATEGORY*    1155 non-null   object 
 3   START*       1155 non-null   object 
 4   STOP*        1155 non-null   object 
 5   MILES*       1155 non-null   float64
 6   PURPOSE*     653 non-null    object 
dtypes: float64(1), object(6)
memory usage: 63.3+ KB
```

In [15]:

```
# Null value is present or not
data.isnull()
```

Out[15]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	True
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
1150	False	False	False	False	False	False	False
1151	False	False	False	False	False	False	False
1152	False	False	False	False	False	False	False
1153	False	False	False	False	False	False	False
1154	False	False	False	False	False	False	False

1155 rows × 7 columns

In [16]:

```
# Null values in every column
data.isnull().sum()
```

Out[16]:

```
START_DATE*      0
END_DATE*        0
CATEGORY*        0
START*           0
STOP*            0
MILES*           0
PURPOSE*       502
dtype: int64
```

In [17]:
```
# Summary of the data
data.describe()
```

Out[17]:

	MILES*
count	1155.000000
mean	10.566840
std	21.579106
min	0.500000
25%	2.900000
50%	6.000000
75%	10.400000
max	310.300000

In [18]:
```
# Summary of the data
data.describe(include="all")
```

Out[18]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
count	1155	1155	1155	1155	1155	1155.000000	653
unique	1154	1154	2	177	188	NaN	10
top	6/28/2016 23:34	6/28/2016 23:59	Business	Cary	Cary	NaN	Meeting
freq	2	2	1078	201	203	NaN	187
mean	NaN	NaN	NaN	NaN	NaN	10.566840	NaN
std	NaN	NaN	NaN	NaN	NaN	21.579106	NaN
min	NaN	NaN	NaN	NaN	NaN	0.500000	NaN
25%	NaN	NaN	NaN	NaN	NaN	2.900000	NaN
50%	NaN	NaN	NaN	NaN	NaN	6.000000	NaN
75%	NaN	NaN	NaN	NaN	NaN	10.400000	NaN
max	NaN	NaN	NaN	NaN	NaN	310.300000	NaN

In [19]:
```
# print the column names
data.columns
```

Out[19]:
```
Index(['START_DATE*', 'END_DATE*', 'CATEGORY*', 'START*', 'STOP*', 'MILES*',
       'PURPOSE*'],
      dtype='object')
```

In [20]:
```
data.index
```

Out[20]:
```
RangeIndex(start=0, stop=1155, step=1)
```

In [21]:
```
#print start date only
data['START_DATE*']
```

Out[21]:
```
0         1/1/2016 21:11
1          1/2/2016 1:25
                2016 20:25
```

```
3         1/5/2016 17:31
4         1/6/2016 14:42
              ...
1150     12/31/2016 1:07
1151    12/31/2016 13:24
1152    12/31/2016 15:03
1153    12/31/2016 21:32
1154    12/31/2016 22:08
Name: START_DATE*, Length: 1155, dtype: object
```

In [22]:

```
data['CATEGORY*']
```

Out[22]:

```
0       Business
1       Business
2       Business
3       Business
4       Business
          ...
1150    Business
1151    Business
1152    Business
1153    Business
1154    Business
Name: CATEGORY*, Length: 1155, dtype: object
```

In [23]:

```
# to fetch two columns at a time
data.loc[:,['START_DATE*','END_DATE*']]
```

Out[23]:

	START_DATE*	END_DATE*
0	1/1/2016 21:11	1/1/2016 21:17
1	1/2/2016 1:25	1/2/2016 1:37
2	1/2/2016 20:25	1/2/2016 20:38
3	1/5/2016 17:31	1/5/2016 17:45
4	1/6/2016 14:42	1/6/2016 15:49
...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14
1151	12/31/2016 13:24	12/31/2016 13:42
1152	12/31/2016 15:03	12/31/2016 15:38
1153	12/31/2016 21:32	12/31/2016 21:50
1154	12/31/2016 22:08	12/31/2016 23:51

1155 rows × 2 columns

In [24]:

```
# print start date, start, miles
data.loc[:,['START_DATE*','MILES*','START*']]
```

Out[24]:

	START_DATE*	MILES*	START*
0	1/1/2016 21:11	5.1	Fort Pierce
1	1/2/2016 1:25	5.0	Fort Pierce
2	1/2/2016 20:25	4.8	Fort Pierce
3	1/5/2016 17:31	4.7	Fort Pierce

	START_DATE*	MILES*	START*
4	1/6/2016 14:42	63.7	Fort Pierce
...	...	...	...
1150	12/31/2016 1:07	0.7	Kar?chi
1151	12/31/2016 13:24	3.9	Kar?chi
1152	12/31/2016 15:03	16.2	Unknown Location
1153	12/31/2016 21:32	6.4	Katunayake
1154	12/31/2016 22:08	48.2	Gampaha

1155 rows × 3 columns

In [25]:
```
# to get range of colums when they are continuous
data.loc[:,'START_DATE*':'CATEGORY*']
```

Out[25]:

	START_DATE*	END_DATE*	CATEGORY*
0	1/1/2016 21:11	1/1/2016 21:17	Business
1	1/2/2016 1:25	1/2/2016 1:37	Business
2	1/2/2016 20:25	1/2/2016 20:38	Business
3	1/5/2016 17:31	1/5/2016 17:45	Business
4	1/6/2016 14:42	1/6/2016 15:49	Business
...	...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14	Business
1151	12/31/2016 13:24	12/31/2016 13:42	Business
1152	12/31/2016 15:03	12/31/2016 15:38	Business
1153	12/31/2016 21:32	12/31/2016 21:50	Business
1154	12/31/2016 22:08	12/31/2016 23:51	Business

1155 rows × 3 columns

In [26]:
```
# Range of rows also
data.loc[0:21,'START_DATE*':'CATEGORY*']
```

Out[26]:

	START_DATE*	END_DATE*	CATEGORY*
0	1/1/2016 21:11	1/1/2016 21:17	Business
1	1/2/2016 1:25	1/2/2016 1:37	Business
2	1/2/2016 20:25	1/2/2016 20:38	Business
3	1/5/2016 17:31	1/5/2016 17:45	Business
4	1/6/2016 14:42	1/6/2016 15:49	Business
5	1/6/2016 17:15	1/6/2016 17:19	Business
6	1/6/2016 17:30	1/6/2016 17:35	Business
7	1/7/2016 13:27	1/7/2016 13:33	Business
8	1/10/2016 8:05	1/10/2016 8:25	Business
9	1/10/2016 12:17	1/10/2016 12:44	Business

	START_DATE*	END_DATE*	CATEGORY*
10	1/10/2016 15:08	1/10/2016 15:51	Business
11	1/10/2016 18:18	1/10/2016 18:53	Business
12	1/10/2016 19:12	1/10/2016 19:32	Business
13	1/11/2016 8:55	1/11/2016 9:21	Business
14	1/11/2016 11:56	1/11/2016 12:03	Business
15	1/11/2016 13:32	1/11/2016 13:46	Business
16	1/11/2016 14:30	1/11/2016 14:43	Business
17	1/12/2016 12:33	1/12/2016 12:49	Business
18	1/12/2016 12:53	1/12/2016 13:09	Business
19	1/12/2016 14:42	1/12/2016 14:56	Business
20	1/12/2016 15:13	1/12/2016 15:28	Business
21	1/12/2016 15:42	1/12/2016 15:54	Business

In [27]:
```
# example of iloc method
data.iloc[:,[2,4,6]]
```

Out[27]:

	CATEGORY*	STOP*	PURPOSE*
0	Business	Fort Pierce	Meal/Entertain
1	Business	Fort Pierce	NaN
2	Business	Fort Pierce	Errand/Supplies
3	Business	Fort Pierce	Meeting
4	Business	West Palm Beach	Customer Visit
...	...	...	...
1150	Business	Kar?chi	Meeting
1151	Business	Unknown Location	Temporary Site
1152	Business	Unknown Location	Meeting
1153	Business	Gampaha	Temporary Site
1154	Business	Ilukwatta	Temporary Site

1155 rows × 3 columns

In [28]:
```
# to convert o/p into array
data.iloc[:,[2,4,6]].values
```

Out[28]:
```
array([['Business', 'Fort Pierce', 'Meal/Entertain'],
       ['Business', 'Fort Pierce', nan],
       ['Business', 'Fort Pierce', 'Errand/Supplies'],
       ...,
       ['Business', 'Unknown Location', 'Meeting'],
       ['Business', 'Gampaha', 'Temporary Site'],
       ['Business', 'Ilukwatta', 'Temporary Site']], dtype=object)
```

In [29]:
```
# condition
data['MILES*']<10
```

Out[29]:
```
0        True
1        True
2        True
3        True
4       False
        ...
1150     True
1151     True
1152    False
1153     True
1154    False
Name: MILES*, Length: 1155, dtype: bool
```

In [30]:
```
data[data['MILES*']<10]
```

Out[30]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
5	1/6/2016 17:15	1/6/2016 17:19	Business	West Palm Beach	West Palm Beach	4.3	Meal/Entertain
...	...	...	...	...	...	...	...
1148	12/30/2016 16:45	12/30/2016 17:08	Business	Kar?chi	Kar?chi	4.6	Meeting
1149	12/30/2016 23:06	12/30/2016 23:10	Business	Kar?chi	Kar?chi	0.8	Customer Visit
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	Kar?chi	0.7	Meeting
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site

837 rows × 7 columns

In [38]:
```
# number of rows satisfy condition
data[data['MILES*']<10].shape[0]
```

Out[38]:
```
837
```

In [32]:
```
# Print the number of rows where start statation is Cary
data[data['START*']=='Cary'].shape[0]
```

Out[32]:
```
201
```

In [33]:
```
data
```

Out[33]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
...	...	...	...	...	...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	Kar?chi	0.7	Meeting
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site

1155 rows × 7 columns

In [34]:
```
data.head()
```

Out[34]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit

today Work:- Create a new column in data See unique records Missing values in the data Convert categorical column in numerical data sorting on data Grouping Extarct month , day, year from date Concat & merge data

In [45]:
```
# create a new column
def fun(num):
    if num<5:
        return "Small"
    elif num>=5 and num<20:
        return "Normal"
    else:
        return "Long"
```

In [46]:
```
data['len_Miles']=data['MILES*'].apply(fun)
```

In [47]:
```
data
```

Out[47]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	Normal
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0	NaN	Normal

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	Small
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting	Small
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	Long
...	...	...	...	...	...	...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	Kar?chi	0.7	Meeting	Small
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site	Small
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting	Normal
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site	Normal
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site	Long

1155 rows × 8 columns

In [48]:
```
# Column info
data['CATEGORY*'].unique()
```

Out[48]:
```
array(['Business', 'Personal'], dtype=object)
```

In [49]:
```
# Records in each category
data['CATEGORY*'].value_counts()
```

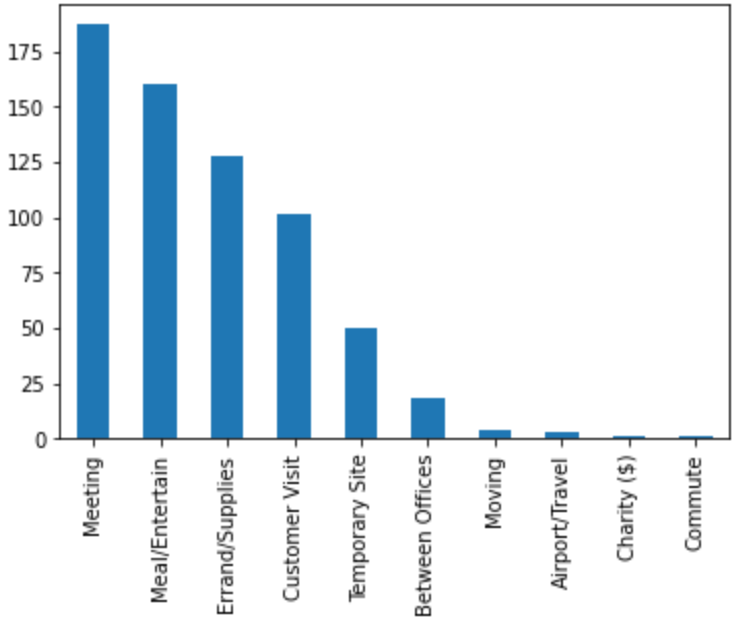
Out[49]:
```
Business    1078
Personal      77
Name: CATEGORY*, dtype: int64
```

In [50]:
```
#Purpose column
data['PURPOSE*'].value_counts()
```

Out[50]:
```
Meeting            187
Meal/Entertain     160
Errand/Supplies    128
Customer Visit     101
Temporary Site      50
Between Offices     18
Moving               4
Airport/Travel       3
Charity ($)          1
Commute              1
Name: PURPOSE*, dtype: int64
```

In [51]:
```
# pandas can visualize
data['PURPOSE*'].value_counts().plot(kind='bar')
```

Out[51]:
```
<AxesSubplot:>
```



In [52]:
```
#to check missing value
data.isnull().sum()
```

Out[52]:
```
START_DATE*      0
END_DATE*        0
CATEGORY*        0
START*           0
STOP*            0
MILES*           0
PURPOSE*       502
len_Miles        0
dtype: int64
```

In [53]:
```
# Percentage of missing value
(data.isnull().sum()/len(data))*100
```

Out[53]:
```
START_DATE*     0.000000
END_DATE*       0.000000
CATEGORY*       0.000000
START*          0.000000
STOP*           0.000000
MILES*          0.000000
PURPOSE*       43.463203
len_Miles       0.000000
dtype: float64
```

In [55]:
```
# to copy the data
df=data.copy()
```

In [56]:
```
# 1 way is to drop the null values
df.dropna()
```

Out[56]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	Normal
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	Small
3	1/5/2016 17:31	1/5/2016	Business	Fort Pierce	Fort Pierce	4.7	Meeting	Small

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
		17:45						
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	Long
5	1/6/2016 17:15	1/6/2016 17:19	Business	West Palm Beach	West Palm Beach	4.3	Meal/Entertain	Small
...	...	...	...	...	...	...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	Kar?chi	0.7	Meeting	Small
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9	Temporary Site	Small
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2	Meeting	Normal
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4	Temporary Site	Normal
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site	Long

653 rows × 8 columns

In [57]:
```
# drop the null values permanent
df.dropna(inplace=True)
```

In [59]:
```
df.isnull().sum()
```

Out[59]:
```
START_DATE*    0
END_DATE*      0
CATEGORY*      0
START*         0
STOP*          0
MILES*         0
PURPOSE*       0
len_Miles      0
dtype: int64
```

In [60]:
```
# fill missing values fillna()
# missing value in numerical col- mean, median,
# missing values in Catgorical- value-'NA', Mode- 'Business'-10
data['PURPOSE*'].fillna('NA')
```

Out[60]:
```
0        Meal/Entertain
1                    NA
2       Errand/Supplies
3               Meeting
4        Customer Visit
             ...
1150            Meeting
1151     Temporary Site
1152            Meeting
1153     Temporary Site
1154     Temporary Site
Name: PURPOSE*, Length: 1155, dtype: object
```

In [61]:
```
data['PURPOSE*'].fillna(method='ffill') #earlier will forward in empty cell
```

Out[61]:
```
0        Meal/Entertain
1        Meal/Entertain
2       Errand/Supplies
3               Meeting
4        Customer Visit
            ...
1150            Meeting
1151     Temporary Site
1152            Meeting
1153     Temporary Site
1154     Temporary Site
Name: PURPOSE*, Length: 1155, dtype: object
```

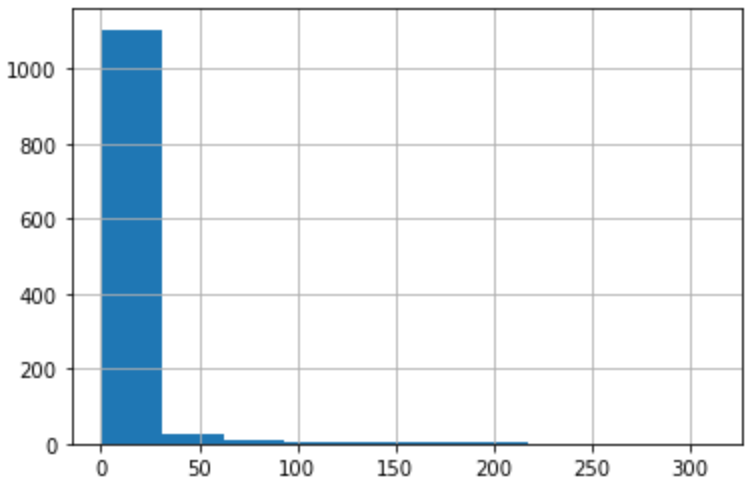
In [62]:
```
data['PURPOSE*'].fillna(method='bfill') # backward data will fill
```

Out[62]:
```
0        Meal/Entertain
1       Errand/Supplies
2       Errand/Supplies
3               Meeting
4        Customer Visit
            ...
1150            Meeting
1151     Temporary Site
1152            Meeting
1153     Temporary Site
1154     Temporary Site
Name: PURPOSE*, Length: 1155, dtype: object
```

In [64]:
```
data['PURPOSE*'].fillna('NA',inplace=True)#premanent change
```

In [65]:
```
# Histogram Miles -
data['MILES*'].hist()
```

Out[65]:
```
<AxesSubplot:>
```



In [66]:
```
data['MILES*'].fillna(data['MILES*'].median()) # as data is skewed i.e not normaly distrib
```

Out[66]:
```
0        5.1
1        5.0
2        4.8
3        4.7
4       63.7
        ...
1150     0.7
```

```
1152    16.2
1153     6.4
1154    48.2
Name: MILES*, Length: 1155, dtype: float64
```

In [45]:
```
#catagerical data to numerical data
df1=data.copy()
```

In [46]:
```
# Any Col has binary values- replace method : 1 ,0
df1['CATEGORY*'].replace({'Business':1,'Personal':0},inplace=True)
```

In [50]:
```
df1['START*'].replace({'Kar?chi':'Karachi'})
df1
```

Out[50]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	1/1/2016 21:11	1/1/2016 21:17	1	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	1/2/2016 1:25	1/2/2016 1:37	1	Fort Pierce	Fort Pierce	5.0	NaN
2	1/2/2016 20:25	1/2/2016 20:38	1	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	1/5/2016 17:31	1/5/2016 17:45	1	Fort Pierce	Fort Pierce	4.7	Meeting
4	1/6/2016 14:42	1/6/2016 15:49	1	Fort Pierce	West Palm Beach	63.7	Customer Visit
...	...	...	...	...	...	...	...
1150	12/31/2016 1:07	12/31/2016 1:14	1	Kar?chi	Kar?chi	0.7	Meeting
1151	12/31/2016 13:24	12/31/2016 13:42	1	Kar?chi	Unknown Location	3.9	Temporary Site
1152	12/31/2016 15:03	12/31/2016 15:38	1	Unknown Location	Unknown Location	16.2	Meeting
1153	12/31/2016 21:32	12/31/2016 21:50	1	Katunayake	Gampaha	6.4	Temporary Site
1154	12/31/2016 22:08	12/31/2016 23:51	1	Gampaha	Ilukwatta	48.2	Temporary Site

1155 rows × 7 columns

In [70]:
```
# Dummy Encoding
pd.get_dummies(df1['PURPOSE*'])
```

Out[70]:

	Airport/Travel	Between Offices	Charity ($)	Commute	Customer Visit	Errand/Supplies	Meal/Entertain	Meeting	Moving	NA
0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	1	0	0	0	0
3	0	0	0	0	0	0	0	1	0	0
4	0	0	0	0	1	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
1150	0	0	0	0	0	0	0	1	0	0
1151	0	0	0	0	0	0	0	0	0	0

	Airport/Travel	Between Offices	Charity ($)	Commute	Customer Visit	Errand/Supplies	Meal/Entertain	Meeting	Moving	NA
1152	0	0	0	0	0	0	0	1	0	0
1153	0	0	0	0	0	0	0	0	0	0
1154	0	0	0	0	0	0	0	0	0	0

1155 rows × 11 columns

In [71]:
```
# apply dummy encoding on whole data
df1['START_DATE*']=pd.to_datetime(df1['START_DATE*'])
df1['END_DATE*']=pd.to_datetime(df1['END_DATE*'])
```

In [72]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1155 entries, 0 to 1154
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype         
---  ------       --------------  -----         
 0   START_DATE*  1155 non-null   datetime64[ns]
 1   END_DATE*    1155 non-null   datetime64[ns]
 2   CATEGORY*    1155 non-null   int64         
 3   START*       1155 non-null   object        
 4   STOP*        1155 non-null   object        
 5   MILES*       1155 non-null   float64       
 6   PURPOSE*     1155 non-null   object        
 7   len_Miles    1155 non-null   object        
dtypes: datetime64[ns](2), float64(1), int64(1), object(4)
memory usage: 72.3+ KB
```

In [73]:
```
df2=pd.get_dummies(df1)
```

In [74]:
```
df2
```

Out[74]:

	START_DATE*	END_DATE*	CATEGORY*	MILES*	START*_Agnew	START*_Almond	START*_Apex	START*_A
0	2016-01-01 21:11:00	2016-01-01 21:17:00	1	5.1	0	0	0	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	1	5.0	0	0	0	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	1	4.8	0	0	0	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	1	4.7	0	0	0	
4	2016-01-06 14:42:00	2016-01-06 15:49:00	1	63.7	0	0	0	
...	...	...	...	...	...	...	...	
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	1	0.7	0	0	0	
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	1	3.9	0	0	0	
1152	2016-12-31 [illegible]00	2016-12-31 15:38:00	1	16.2	0	0	0	

	START_DATE*	END_DATE*	CATEGORY*	MILES*	START*_Agnew	START*_Almond	START*_Apex	START*_A
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	1	6.4	0	0	0	
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	1	48.2	0	0	0	

1155 rows × 383 columns

In [75]:
```
# Sorting the data
# sort_values()
data.sort_values(by=['MILES*'])
```

Out[75]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
420	6/8/2016 17:16	6/8/2016 17:18	Business	Soho	Tribeca	0.5	Errand/Supplies	Small
44	1/26/2016 17:27	1/26/2016 17:29	Business	Cary	Cary	0.5	Errand/Supplies	Small
120	2/17/2016 16:38	2/17/2016 16:43	Business	Katunayaka	Katunayaka	0.5	Errand/Supplies	Small
1111	12/25/2016 0:10	12/25/2016 0:14	Business	Lahore	Lahore	0.6	Errand/Supplies	Small
1110	12/24/2016 22:04	12/24/2016 22:09	Business	Lahore	Lahore	0.6	Errand/Supplies	Small
...	...	...	...	...	...	...	...	...
546	7/14/2016 16:39	7/14/2016 20:05	Business	Morrisville	Banner Elk	195.3	NA	Long
776	9/27/2016 21:01	9/28/2016 2:37	Business	Unknown Location	Unknown Location	195.6	NA	Long
881	10/30/2016 15:22	10/30/2016 18:23	Business	Asheville	Mebane	195.9	NA	Long
270	3/25/2016 22:54	3/26/2016 1:39	Business	Jacksonville	Kissimmee	201.0	Meeting	Long
269	3/25/2016 16:52	3/25/2016 22:22	Business	Latta	Jacksonville	310.3	Customer Visit	Long

1155 rows × 8 columns

In [76]:
```
data.sort_values(by=['MILES*'],ascending=False)
```

Out[76]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
269	3/25/2016 16:52	3/25/2016 22:22	Business	Latta	Jacksonville	310.3	Customer Visit	Long
270	3/25/2016 22:54	3/26/2016 1:39	Business	Jacksonville	Kissimmee	201.0	Meeting	Long
881	10/30/2016 15:22	10/30/2016 18:23	Business	Asheville	Mebane	195.9	NA	Long
776	9/27/2016 21:01	9/28/2016 2:37	Business	Unknown Location	Unknown Location	195.6	NA	Long
546	7/14/2016 16:39	7/14/2016 20:05	Business	Morrisville	Banner Elk	195.3	NA	Long

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
...	...	...	...	...	...	...	...	...
945	11/12/2016 13:46	11/12/2016 13:50	Business	Central	West Berkeley	0.6	NA	Small
1121	12/27/2016 12:53	12/27/2016 12:57	Business	Kar?chi	Kar?chi	0.6	Meal/Entertain	Small
420	6/8/2016 17:16	6/8/2016 17:18	Business	Soho	Tribeca	0.5	Errand/Supplies	Small
44	1/26/2016 17:27	1/26/2016 17:29	Business	Cary	Cary	0.5	Errand/Supplies	Small
120	2/17/2016 16:38	2/17/2016 16:43	Business	Katunayaka	Katunayaka	0.5	Errand/Supplies	Small

1155 rows × 8 columns

In [77]:
```
# sort the data on two col
data.sort_values(by=['START*','MILES*']) # in this according to start miles is sorted
```

Out[77]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
908	11/5/2016 8:34	11/5/2016 8:43	Business	Agnew	Renaissance	2.2	NA	Small
910	11/5/2016 19:20	11/5/2016 19:28	Business	Agnew	Agnew	2.2	NA	Small
911	11/6/2016 10:50	11/6/2016 11:04	Business	Agnew	Renaissance	2.4	NA	Small
906	11/4/2016 21:04	11/4/2016 21:20	Business	Agnew	Cory	4.3	NA	Small
879	10/30/2016 12:58	10/30/2016 13:18	Business	Almond	Bryson City	15.2	NA	Normal
...	...	...	...	...	...	...	...	...
321	4/19/2016 17:44	4/19/2016 18:08	Business	Whitebridge	Wayne Ridge	8.2	Meal/Entertain	Normal
64	2/2/2016 13:04	2/2/2016 13:23	Business	Whitebridge	Williamsburg Manor	8.3	Meeting	Normal
511	7/4/2016 17:31	7/4/2016 17:49	Business	Whitebridge	Summerwinds	8.8	Meeting	Normal
72	2/4/2016 18:04	2/4/2016 18:31	Business	Whitebridge	Macgregor Downs	9.0	Meeting	Normal
870	10/28/2016 18:13	10/28/2016 20:07	Business	Winston Salem	Asheville	133.6	Meeting	Long

1155 rows × 8 columns

In [79]:
```
# sort the data on two col
data.sort_values(by=['START*','MILES*'],ascending=[False,True])
```

Out[79]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
870	10/28/2016 18:13	10/28/2016 20:07	Business	Winston Salem	Asheville	133.6	Meeting	Long
516	7/5/2016 16:48	7/5/2016 16:52	Business	Whitebridge	Whitebridge	0.6	Errand/Supplies	Small

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
890	11/1/2016 19:14	11/1/2016 19:20	Business	Whitebridge	Whitebridge	1.0	NA	Small
889	11/1/2016 17:35	11/1/2016 17:42	Business	Whitebridge	Whitebridge	1.2	NA	Small
263	3/22/2016 19:12	3/22/2016 19:25	Personal	Whitebridge	Whitebridge	1.4	NA	Small
...	...	...	...	...	...	...	...	...
879	10/30/2016 12:58	10/30/2016 13:18	Business	Almond	Bryson City	15.2	NA	Normal
908	11/5/2016 8:34	11/5/2016 8:43	Business	Agnew	Renaissance	2.2	NA	Small
910	11/5/2016 19:20	11/5/2016 19:28	Business	Agnew	Agnew	2.2	NA	Small
911	11/6/2016 10:50	11/6/2016 11:04	Business	Agnew	Renaissance	2.4	NA	Small
906	11/4/2016 21:04	11/4/2016 21:20	Business	Agnew	Cory	4.3	NA	Small

1155 rows × 8 columns

In [80]:
```
# groupby- Data is grouped- various operation
data.groupby('START*')
```

Out[80]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000025E6A4FED00>
```

In [81]:
```
data.groupby('START*').count()
```

Out[81]:

START*	START_DATE*	END_DATE*	CATEGORY*	STOP*	MILES*	PURPOSE*	len_Miles
Agnew	4	4	4	4	4	4	4
Almond	1	1	1	1	1	1	1
Apex	17	17	17	17	17	17	17
Arabi	1	1	1	1	1	1	1
Arlington	1	1	1	1	1	1	1
...	...	...	...	...	...	...	...
West University	2	2	2	2	2	2	2
Weston	2	2	2	2	2	2	2
Westpark Place	17	17	17	17	17	17	17
Whitebridge	68	68	68	68	68	68	68
Winston Salem	1	1	1	1	1	1	1

177 rows × 7 columns

In [82]:
```
data.groupby('START*')['MILES*'].count()
```

Out[82]:
```
START*
Agnew                4
```

```
Almond              1
Apex               17
Arabi               1
Arlington           1
                   ..
West University     2
Weston              2
Westpark Place     17
Whitebridge        68
Winston Salem       1
Name: MILES*, Length: 177, dtype: int64
```

In [83]:
```
data.groupby('START*')['MILES*'].sum() #sum of miles for all records for that group
```

Out[83]:
```
START*
Agnew               11.1
Almond              15.2
Apex                90.8
Arabi               17.0
Arlington            4.9
                    ...
West University      4.4
Weston               8.0
Westpark Place      37.1
Whitebridge        273.4
Winston Salem      133.6
Name: MILES*, Length: 177, dtype: float64
```

In [84]:
```
data.groupby('START*')['MILES*'].agg(["mean","sum","max","min","count"])
```

Out[84]:

START*	mean	sum	max	min	count
Agnew	2.775000	11.1	4.3	2.2	4
Almond	15.200000	15.2	15.2	15.2	1
Apex	5.341176	90.8	9.0	1.0	17
Arabi	17.000000	17.0	17.0	17.0	1
Arlington	4.900000	4.9	4.9	4.9	1
...	...	...	...	...	...
West University	2.200000	4.4	2.3	2.1	2
Weston	4.000000	8.0	4.2	3.8	2
Westpark Place	2.182353	37.1	4.2	1.7	17
Whitebridge	4.020588	273.4	9.0	0.6	68
Winston Salem	133.600000	133.6	133.6	133.6	1

177 rows × 5 columns

In [51]:
```
data.groupby(['CATEGORY*','STOP*','PURPOSE*']).count() # more than two columns get groupe
```

Out[51]:

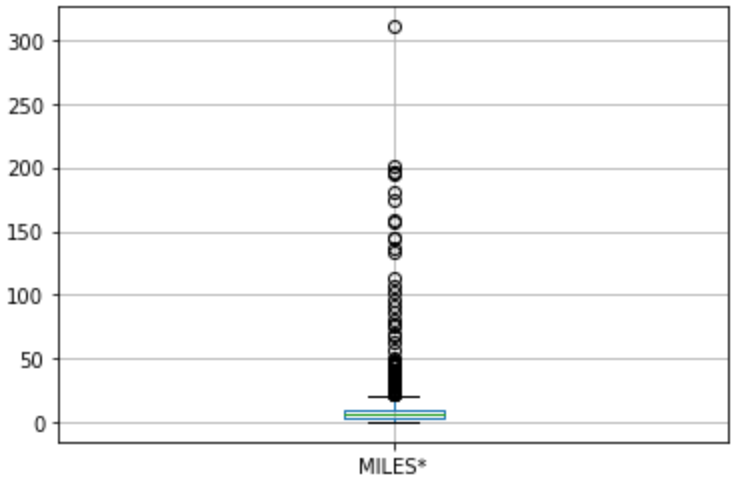
CATEGORY*	STOP*	PURPOSE*	START_DATE*	END_DATE*	START*	MILES*
Business	**Agnew**	**Meeting**	1	1	1	1

			START_DATE*	END_DATE*	START*	MILES*
CATEGORY*	STOP*	PURPOSE*				
	Alief	Meal/Entertain	1	1	1	1
	Apex	Errand/Supplies	5	5	5	5
		Meal/Entertain	3	3	3	3
		Meeting	3	3	3	3
...	...	...	...	...	...	...
Personal	Cary	Commute	1	1	1	1
		Moving	1	1	1	1
	Morrisville	Moving	1	1	1	1
	Preston	Moving	1	1	1	1
	Whitebridge	Moving	1	1	1	1

226 rows × 4 columns

In [88]:
```
#Outlier
# box plot
data.boxplot()
```

Out[88]:
```
<AxesSubplot:>
```



In [89]:
```
df2=data.copy()
```

In [90]:
```
# 1.5 IQR rule
IQR=df2['MILES*'].quantile(0.75)-df2['MILES*'].quantile(0.25)
IQR
```

Out[90]:
```
7.5
```

In [91]:
```
lower=df2['MILES*'].quantile(0.25)-(1.5*IQR)
upper=df2['MILES*'].quantile(0.75)+(1.5*IQR)
```

In [92]:
```
print(lower)
print(upper)
```

```
-8.35
21.65
```

In [93]:
```
df2[df2['MILES*']<-8.35]
```

Out[93]:

START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles

In [95]:
```
df2[df2['MILES*']>21.65]
```

Out[95]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	Long
25	1/14/2016 16:29	1/14/2016 17:05	Business	Houston	Houston	21.9	Customer Visit	Long
36	1/20/2016 13:25	1/20/2016 14:19	Business	Raleigh	Cary	40.2	Customer Visit	Long
62	2/1/2016 12:10	2/1/2016 12:43	Business	Chapel Hill	Cary	23.3	Customer Visit	Long
108	2/16/2016 3:21	2/16/2016 4:13	Business	Katunayaka	Unknown Location	43.7	Customer Visit	Long
...	...	...	...	...	...	...	...	...
979	11/20/2016 10:27	11/20/2016 11:32	Business	Cary	Cary	39.2	Between Offices	Long
1088	12/21/2016 20:56	12/21/2016 23:42	Business	Rawalpindi	Unknown Location	103.0	Meeting	Long
1089	12/22/2016 15:40	12/22/2016 16:38	Business	Unknown Location	Unknown Location	32.3	Meeting	Long
1092	12/22/2016 17:56	12/22/2016 18:29	Business	Unknown Location	Unknown Location	23.2	Meeting	Long
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2	Temporary Site	Long

77 rows × 8 columns

In [96]:
```
df2[df2['MILES*']>21.65].count()
```

Out[96]:
```
START_DATE*    77
END_DATE*      77
CATEGORY*      77
START*         77
STOP*          77
MILES*         77
PURPOSE*       77
len_Miles      77
dtype: int64
```

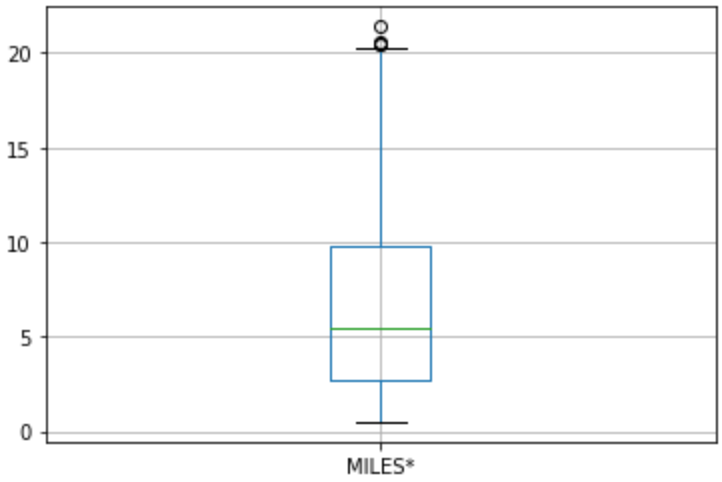
In [97]:
```
# drop data where outlier found
# drop the data where i have outlier
df2.drop(df2[df2['MILES*']>21.65].index,inplace=True)
```

In [98]:
```
df2.shape
```

Out[98]: (1078, 8)

In [99]:
```
df2.boxplot()
```

Out[99]: <AxesSubplot:>



In [100…
```
# handle the outlier
df3=data.copy()
```

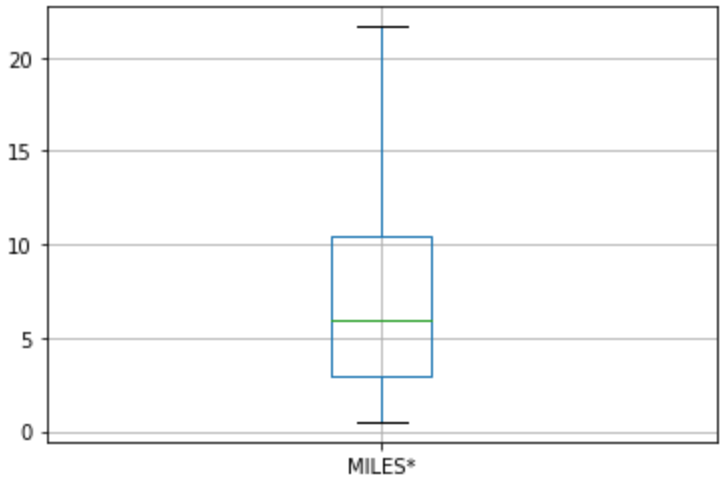
In [104…
```
# how to impute the outlier
# values more than 21.65 are outlier if i make all values more than 21.65 as 21.65
df3.loc[df3['MILES*']>21.65,'MILES*']=21.65
```

In [102…
```
df3[df3['MILES*']>21.65].count()
```

Out[102…
```
START_DATE*    0
END_DATE*      0
CATEGORY*      0
START*         0
STOP*          0
MILES*         0
PURPOSE*       0
len_Miles      0
dtype: int64
```

In [105…
```
df3.boxplot()
```

Out[105… <AxesSubplot:>



In [107…

```
# 1.5 IQR rule is stat. way to find outlier
# Business value less than 5 percentile -
# 5 - 95 //// another example
data['MILES*'].quantile(0.95)
data[data['MILES*']>28.1].count()
```

Out[107…

```
START_DATE*    57
END_DATE*      57
CATEGORY*      57
START*         57
STOP*          57
MILES*         57
PURPOSE*       57
len_Miles      57
dtype: int64
```

In [109…

```
# Date issue
df4=data.copy()
```

In [111…

```
df4['START_DATE*']=pd.to_datetime(df4['START_DATE*'])
df4['END_DATE*']=pd.to_datetime(df4['END_DATE*'])
df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1155 entries, 0 to 1154
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   START_DATE*  1155 non-null   datetime64[ns]
 1   END_DATE*    1155 non-null   datetime64[ns]
 2   CATEGORY*    1155 non-null   object
 3   START*       1155 non-null   object
 4   STOP*        1155 non-null   object
 5   MILES*       1155 non-null   float64
 6   PURPOSE*     1155 non-null   object
 7   len_Miles    1155 non-null   object
dtypes: datetime64[ns](2), float64(1), object(5)
memory usage: 72.3+ KB
```

In [112…

```
# Year column created
df4.insert(loc=2,column='Year',value=df4['START_DATE*'].dt.year) # loc - location of creat
```

In [113…

```
df4
```

Out[113…

	START_DATE*	END_DATE*	Year	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_Miles
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	Normal
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	Business	Fort Pierce	Fort Pierce	5.0	NA	Normal
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	Small
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	Business	Fort Pierce	Fort Pierce	4.7	Meeting	Small
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	Long
...	...	...	...	...	...	...	...	...	...
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	Business	Kar?chi	Kar?chi	0.7	Meeting	Small
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	Business	Kar?chi	Unknown Location	3.9	Temporary Site	Small
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	Business	Unknown Location	Unknown Location	16.2	Meeting	Normal
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	Business	Katunayake	Gampaha	6.4	Temporary Site	Normal
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	Business	Gampaha	Ilukwatta	48.2	Temporary Site	Long

1155 rows × 9 columns

In [114…

```
# Month column created by extracting month data
df4.insert(loc=3,column='Month',value=df4['START_DATE*'].dt.month)
```

In [115…

```
df4
```

Out[115…

	START_DATE*	END_DATE*	Year	Month	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_M
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	1	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain	No
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	1	Business	Fort Pierce	Fort Pierce	5.0	NA	No
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	1	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies	S
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	1	Business	Fort Pierce	Fort Pierce	4.7	Meeting	S
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	1	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit	L
...	...	...	...	...	...	...	...	...	...	
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	12	Business	Kar?chi	Kar?chi	0.7	Meeting	S
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	12	Business	Kar?chi	Unknown Location	3.9	Temporary Site	S
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	12	Business	Unknown Location	Unknown Location	16.2	Meeting	No

	START_DATE*	END_DATE*	Year	Month	CATEGORY*	START*	STOP*	MILES*	PURPOSE*	len_M
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	12	Business	Katunayake	Gampaha	6.4	Temporary Site	No
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	12	Business	Gampaha	Ilukwatta	48.2	Temporary Site	I

1155 rows × 10 columns

In [118…

```
# same with date
df4.insert(loc=4,column='Date',value=df4['START_DATE*'].dt.date)# loc ic location of colu
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_32260/3297512386.py in <module>
      1 # same with date
----> 2 df4.insert(loc=4,column='Date',value=df4['START_DATE*'].dt.date)# loc ic location
of column to br created

~\anaconda3\lib\site-packages\pandas\core\frame.py in insert(self, loc, column, value, all
ow_duplicates)
   4412         if not allow_duplicates and column in self.columns:
   4413             # Should this be a different kind of error??
-> 4414             raise ValueError(f"cannot insert {column}, already exists")
   4415         if not isinstance(loc, int):
   4416             raise TypeError("loc must be int")

ValueError: cannot insert Date, already exists
```

In [117…

```
df4
```

Out[117…

	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	1	2016-01-01	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	1	2016-01-02	Business	Fort Pierce	Fort Pierce	5.0	NA
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	1	2016-01-02	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	1	2016-01-05	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	1	2016-01-06	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
...	...	...	...	...	...	...	...	...	...	...
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	12	2016-12-31	Business	Kar?chi	Kar?chi	0.7	Meeting
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	12	2016-12-31	Business	Kar?chi	Unknown Location	3.9	Temporary Site
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	12	2016-12-31	Business	Unknown Location	Unknown Location	16.2	Meeting
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	12	2016-12-31	Business	Katunayake	Gampaha	6.4	Temporary Site
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	12	2016-12-31	Business	Gampaha	Ilukwatta	48.2	Temporary Site

lumns

In [121…
```
# Month col
df4.insert(loc=7,column='Weekday',value=df4['START_DATE*'].dt.day_name())
```

In [122…
```
df4
```

Out[122…

	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Weekday	Weekday1	STOP*
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	1	2016-01-01	Business	Fort Pierce	Friday	Friday	Fort Pierce
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	Saturday	Fort Pierce
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	Saturday	Fort Pierce
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	1	2016-01-05	Business	Fort Pierce	Tuesday	Tuesday	Fort Pierce
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	1	2016-01-06	Business	Fort Pierce	Wednesday	Wednesday	West Palm Beach
...	...	...	...	...	...	...	...	...	...	...
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	Saturday	Kar?chi
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	Saturday	Unknown Location
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	12	2016-12-31	Business	Unknown Location	Saturday	Saturday	Unknown Location
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	12	2016-12-31	Business	Katunayake	Saturday	Saturday	Gampaha
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	12	2016-12-31	Business	Gampaha	Saturday	Saturday	Ilukwatta

1155 rows × 13 columns

In [123…
```
# hour
df4.insert(loc=8,column='Hour',value=df4['START_DATE*'].dt.hour)
```

In [124…
```
df4
```

Out[124…

	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Weekday	Hour	Weekday1
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	1	2016-01-01	Business	Fort Pierce	Friday	21	Friday
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	1	Saturday
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	20	Saturday
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	1	2016-01-05	Business	Fort Pierce	Tuesday	17	Tuesday
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	1	2016-01-06	Business	Fort Pierce	Wednesday	14	Wednesday
...	...	...	...	...	...	...	...	...	...	...

	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Weekday	Hour	Weekday1	
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	1	Saturday	
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	13	Saturday	U L
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	12	2016-12-31	Business	Unknown Location	Saturday	15	Saturday	U L
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	12	2016-12-31	Business	Katunayake	Saturday	21	Saturday	Ga
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	12	2016-12-31	Business	Gampaha	Saturday	22	Saturday	Il

1155 rows × 14 columns

In [125…
```
# drop unnecessary col
df4.drop(['Weekday1'],axis=1,inplace=True)
```

In [126…
```
df4
```

Out[126…

	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Weekday	Hour	STOP*	MILE
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	1	2016-01-01	Business	Fort Pierce	Friday	21	Fort Pierce	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	1	Fort Pierce	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	20	Fort Pierce	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	1	2016-01-05	Business	Fort Pierce	Tuesday	17	Fort Pierce	
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	1	2016-01-06	Business	Fort Pierce	Wednesday	14	West Palm Beach	6
...	...	...	...	...	...	...	...	...	...	...	
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	1	Kar?chi	
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	13	Unknown Location	
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	12	2016-12-31	Business	Unknown Location	Saturday	15	Unknown Location	1
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	12	2016-12-31	Business	Katunayake	Saturday	21	Gampaha	
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	12	2016-12-31	Business	Gampaha	Saturday	22	Ilukwatta	4

1155 rows × 13 columns

In [130…
```
df4.groupby(['Weekday']).count()
```

Out[130…

Weekday	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Hour	STOP*	MILES*	PURPOS
	206	206	206	206	206	206	206	206	206	206	2

Weekday	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Hour	STOP*	MILES*	PURPOS
Monday	174	174	174	174	174	174	174	174	174	174	1
Saturday	150	150	150	150	150	150	150	150	150	150	1
Sunday	148	148	148	148	148	148	148	148	148	148	1
Thursday	154	154	154	154	154	154	154	154	154	154	1
Tuesday	176	176	176	176	176	176	176	176	176	176	1
Wednesday	147	147	147	147	147	147	147	147	147	147	1

In [131…
```
df4.rename(columns={'Weekday':'Week_day'},inplace=True)
```

In [132…
```
df4
```

Out[132…

	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Week_day	Hour	STOP*	MILI
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	1	2016-01-01	Business	Fort Pierce	Friday	21	Fort Pierce	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	1	Fort Pierce	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	20	Fort Pierce	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	1	2016-01-05	Business	Fort Pierce	Tuesday	17	Fort Pierce	
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	1	2016-01-06	Business	Fort Pierce	Wednesday	14	West Palm Beach	6
...	...	...	...	...	...	...	...	...	...	...	
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	1	Kar?chi	
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	13	Unknown Location	
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	12	2016-12-31	Business	Unknown Location	Saturday	15	Unknown Location	1
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	12	2016-12-31	Business	Katunayake	Saturday	21	Gampaha	
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	12	2016-12-31	Business	Gampaha	Saturday	22	Ilukwatta	4

1155 rows × 13 columns

In [133…
```
# Final save the output in csv file
# to_csv()
df4.to_csv('Uber_New_Analysis.csv')
```

In [134…
```
df4.to_csv('Uber_New_Analysis.csv',index=False)
```

In [135…
```
new=pd.read_csv('Uber_New_Analysis.csv')
```

In [136…
```
new
```

Out[136…

	START_DATE*	END_DATE*	Year	Month	Date	CATEGORY*	START*	Week_day	Hour	STOP*	MILI
0	2016-01-01 21:11:00	2016-01-01 21:17:00	2016	1	2016-01-01	Business	Fort Pierce	Friday	21	Fort Pierce	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	1	Fort Pierce	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	2016	1	2016-01-02	Business	Fort Pierce	Saturday	20	Fort Pierce	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	2016	1	2016-01-05	Business	Fort Pierce	Tuesday	17	Fort Pierce	
4	2016-01-06 14:42:00	2016-01-06 15:49:00	2016	1	2016-01-06	Business	Fort Pierce	Wednesday	14	West Palm Beach	6
...	...	...	...	...	...	...	...	...	...	...	
1150	2016-12-31 01:07:00	2016-12-31 01:14:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	1	Kar?chi	
1151	2016-12-31 13:24:00	2016-12-31 13:42:00	2016	12	2016-12-31	Business	Kar?chi	Saturday	13	Unknown Location	
1152	2016-12-31 15:03:00	2016-12-31 15:38:00	2016	12	2016-12-31	Business	Unknown Location	Saturday	15	Unknown Location	1
1153	2016-12-31 21:32:00	2016-12-31 21:50:00	2016	12	2016-12-31	Business	Katunayake	Saturday	21	Gampaha	
1154	2016-12-31 22:08:00	2016-12-31 23:51:00	2016	12	2016-12-31	Business	Gampaha	Saturday	22	Ilukwatta	4

1155 rows × 13 columns

In [139…
```
# Merge two dataframes
first=pd.DataFrame({'id':[1,2,3,4,5],'Name':['Alex','Amy','Hari','Alice','Nancy'],
                    'subid':['sub1','sub2','sub4','sub6','sub5']})
```

In [140…
```
first
```

Out[140…

	id	Name	subid
0	1	Alex	sub1
1	2	Amy	sub2
2	3	Hari	sub4
3	4	Alice	sub6
4	5	Nancy	sub5

In [141…
```
second=pd.DataFrame({'id':[1,2,3,4,5],'Name':['x','y','z','p','q'],
                     'subid':['sub2','sub4','sub3','sub6','sub5']})
```

In [142…
```
second
```

Out[142…

	id	Name	subid
0	1	x	sub2
1	2	y	sub4
2	3	z	sub3
3	4	p	sub6
4	5	q	sub5

In [143…

```
new=pd.merge(first,second,on='id')
new
```

Out[143…

	id	Name_x	subid_x	Name_y	subid_y
0	1	Alex	sub1	x	sub2
1	2	Amy	sub2	y	sub4
2	3	Hari	sub4	z	sub3
3	4	Alice	sub6	p	sub6
4	5	Nancy	sub5	q	sub5

In [144…

```
#pass two id for merging
new=pd.merge(first,second,on=['id','subid'])
new
```

Out[144…

	id	Name_x	subid	Name_y
0	4	Alice	sub6	p
1	5	Nancy	sub5	q

Types of Merge- left,right,inner,outer left- use keys from first Dataframe right- use keys from second Dataframe outer- union of keys inner- intersection of keys

In [145…

```
new=pd.merge(first,second,on='subid',how='left')
new
```

Out[145…

	id_x	Name_x	subid	id_y	Name_y
0	1	Alex	sub1	NaN	NaN
1	2	Amy	sub2	1.0	x
2	3	Hari	sub4	2.0	y
3	4	Alice	sub6	4.0	p
4	5	Nancy	sub5	5.0	q

In [147…

```
new=pd.merge(first,second,on='subid',how='right')
new
```

Out[147…

	id_x	Name_x	subid	id_y	Name_y
0	2.0	Amy	sub2	1	x
1	3.0	Hari	sub4	2	y

	id_x	Name_x	subid	id_y	Name_y
2	NaN	NaN	sub3	3	z
3	4.0	Alice	sub6	4	p
4	5.0	Nancy	sub5	5	q

In [149…
```
new=pd.merge(first,second,on='subid',how='inner')
new
```

Out[149…

	id_x	Name_x	subid	id_y	Name_y
0	2	Amy	sub2	1	x
1	3	Hari	sub4	2	y
2	4	Alice	sub6	4	p
3	5	Nancy	sub5	5	q

In [150…
```
new=pd.merge(first,second,on='subid',how='outer')
new
```

Out[150…

	id_x	Name_x	subid	id_y	Name_y
0	1.0	Alex	sub1	NaN	NaN
1	2.0	Amy	sub2	1.0	x
2	3.0	Hari	sub4	2.0	y
3	4.0	Alice	sub6	4.0	p
4	5.0	Nancy	sub5	5.0	q
5	NaN	NaN	sub3	3.0	z

In [151…
```
pd.concat([first,second])
```

Out[151…

	id	Name	subid
0	1	Alex	sub1
1	2	Amy	sub2
2	3	Hari	sub4
3	4	Alice	sub6
4	5	Nancy	sub5
0	1	x	sub2
1	2	y	sub4
2	3	z	sub3
3	4	p	sub6
4	5	q	sub5

In [152…
```
pd.concat([first,second],ignore_index=True)
```

Out[152…

	id	Name	subid
0	1	Alex	sub1
1	2	Amy	sub2
2	3	Hari	sub4
3	4	Alice	sub6
4	5	Nancy	sub5
5	1	x	sub2
6	2	y	sub4
7	3	z	sub3
8	4	p	sub6
9	5	q	sub5

In [ ]: