Instructions:

For all questions after 10th, Please only use the data specified in the note given just below the question You need to add answers in the same file i.e. PDS_UberDriveProject_Questions.ipynb' and rename that file as 'Name_Date.ipynb'.You can mention the date on which you will be uploading/submitting the file.For e.g. if you plan to submit your assignment on 1st January 2021, you can rename the file as 'STUDENTNAME_01-Jan-2021'

Load the necessary libraries. Import and load the dataset with a name uber_drives .

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# Get the Data
Uber_Data = pd.read_csv(r'C:\Users\Rishi Kalpa Mukherje\Downloads\Project_Python\uberdrives.csv')
```

Q1. Show the last 10 records of the dataset. (2 point)

In [3]:
```
Uber_Data.tail(10)
```

Out[3]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
1145	12/30/2016 10:15	12/30/2016 10:33	Business	Karachi	Karachi	2.8	Errand/Supplies
1146	12/30/2016 11:31	12/30/2016 11:56	Business	Karachi	Karachi	2.9	Errand/Supplies
1147	12/30/2016 15:41	12/30/2016 16:03	Business	Karachi	Karachi	4.6	Errand/Supplies
1148	12/30/2016 16:45	12/30/2016 17:08	Business	Karachi	Karachi	4.6	Meeting
1149	12/30/2016 23:06	12/30/2016 23:10	Business	Karachi	Karachi	0.8	Customer Visit
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Karachi	Karachi	0.7	Meeting
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Karachi	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

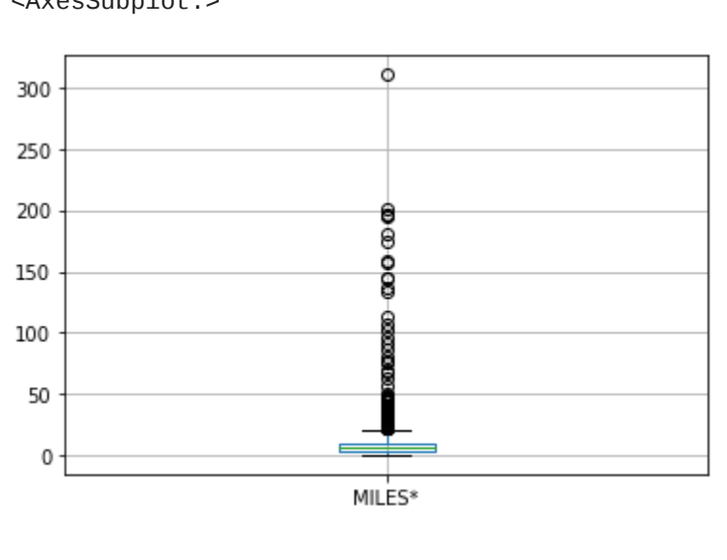
In [3]:
```
Uber_Data.describe().T
```

Out[3]:

	count	mean	std	min	25%	50%	75%	max
MILES*	1155.0	10.56684	21.579106	0.5	2.9	6.0	10.4	310.3

In [9]:
```
Uber_Data.boxplot()
```

Out[9]: <AxesSubplot:>



Q2. Show the first 10 records of the dataset. (2 points)

In [4]:
```
Uber_Data.head(10)
```

Out[4]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
0	01-01-2016 21:11	01-01-2016 21:17	Business	Fort Pierce	Fort Pierce	5.1	Meal/Entertain
1	01-02-2016 01:25	01-02-2016 01:37	Business	Fort Pierce	Fort Pierce	5.0	NaN
2	01-02-2016 20:25	01-02-2016 20:38	Business	Fort Pierce	Fort Pierce	4.8	Errand/Supplies
3	01-05-2016 17:31	01-05-2016 17:45	Business	Fort Pierce	Fort Pierce	4.7	Meeting
4	01-06-2016 14:42	01-06-2016 15:49	Business	Fort Pierce	West Palm Beach	63.7	Customer Visit
5	01-06-2016 17:15	01-06-2016 17:19	Business	West Palm Beach	West Palm Beach	4.3	Meal/Entertain
6	01-06-2016 17:30	01-06-2016 17:35	Business	West Palm Beach	Palm Beach	7.1	Meeting
7	01-07-2016 13:27	01-07-2016 13:33	Business	Cary	Cary	0.8	Meeting
8	01-10-2016 08:05	01-10-2016 08:25	Business	Cary	Morrisville	8.3	Meeting
9	01-10-2016 12:17	01-10-2016 12:44	Business	Jamaica	New York	16.5	Customer Visit

Q3. Show the dimension(number of rows and columns) of the dataset. (2 points)

In [5]:
```
Uber_Data.shape
```

Out[5]:
```
(1155, 7)
```

In [6]:
```
print(f"The number of Rows are: {Uber_Data.shape[0]}. \nThe number of Columns are: {Uber_Data.shape[1]}.")
```

```
The number of Rows are: 1155.
The number of Columns are: 7.
```

Q4. Show the size (Total number of elements) of the dataset. (2 points)

In [7]:
```
Uber_Data.size
```

Out[7]:
```
8085
```

Q5. Display the information about all the variables of the data set. What can you infer from the output?(1 +2 points)

Hint: Information includes - Total number of columns,variable data-types, number of non-null values in a variable, and usage

In [8]:
```
Uber_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
```

Q6. Check for missing values. (2 points)

Note: Output should contain only one boolean value

In [9]:
```
print("Missing Value Status Column Wise \n")
for i in Uber_Data.isnull().sum().index:
    if Uber_Data.isnull().sum().get(i) != 0:
        print(f"{i} -- Yes")
    else:
        print(f"{i} -- No")
```

```
Missing Value Status Column Wise

START_DATE* -- No
END_DATE* -- No
CATEGORY* -- No
START* -- No
STOP* -- No
MILES* -- No
PURPOSE* -- Yes
```

In [18]:
```
Uber_Data.isna().values.any()
```

Out[18]:
```
True
```

Q7. How many missing values are present in the entire dataset? (2 points)

In [11]:
```
Uber_Data.isnull().sum().sum()
```

Out[11]:
```
502
```

Q8. Get the summary of the original data. (2 points).

Hint: Summary includes- Count,Mean, Std, Min, 25%,50%,75% and max

In [12]:
```
Uber_Data.describe()
```

Out[12]:

	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

Q9. Drop the missing values and store the data in a new dataframe (name it"df") (2-points)

Note: Dataframe "df" will not contain any missing value

In [13]:
```
df = Uber_Data.dropna()
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
START_DATE*    0
END_DATE*      0
CATEGORY*      0
START*         0
STOP*          0
MILES*         0
PURPOSE*       0
dtype: int64
```

In [15]:
```
df.isna().values.any()
```

Out[15]:
```
False
```

Q10. Check the information of the dataframe(df). (1 points)

Hint: Information includes - Total number of columns,variable data-types, number of non-null values in a variable, and usage

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 653 entries, 0 to 1154
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   START_DATE*  653 non-null    object
 1   END_DATE*    653 non-null    object
 2   CATEGORY*    653 non-null    object
 3   START*       653 non-null    object
 4   STOP*        653 non-null    object
 5   MILES*       653 non-null    float64
 6   PURPOSE*     653 non-null    object
dtypes: float64(1), object(6)
memory usage: 40.8+ KB
```

Q11 A. Get the unique start locations. (2 points)

Note: This question is based on the dataframe with no 'NA' values

In [13]:
```
Uber_Data['START*'].unique()
```

Out[13]:
```
array(['Fort Pierce', 'West Palm Beach', 'Cary', 'Jamaica', 'New York',
       'Elmhurst', 'Midtown', 'East Harlem', 'Flatiron District',
       'Midtown East', 'Hudson Square', 'Lower Manhattan',
       "Hell's Kitchen", 'Downtown', 'Gulfton', 'Houston', 'Eagan Park',
       'Morrisville', 'Durham', 'Farmington Woods', 'Whitebridge',
       'Lake Wellingborough', 'Fayetteville Street', 'Raleigh',
       'Hazelwood', 'Fairmont', 'Meredith Townes', 'Apex', 'Chapel Hill',
       'Northwoods', 'Edgehill Farms', 'Tanglewood', 'Preston',
       'Eastgate', 'East Elmhurst', 'Jackson Heights', 'Long Island City',
       'Katunayaka', 'Unknown Location', 'Colombo', 'Nugegoda',
       'Islamabad', 'R?walpindi', 'Noorpur Shahan', 'Heritage Pines',
       'Westpark Place', 'Waverly Place', 'Wayne Ridge', 'Weston',
       'East Austin', 'West University', 'South Congress', 'The Drag',
       'Congress Ave District', 'Red River District', 'Georgian Acres',
       'North Austin', 'Coxville', 'Convention Center District', 'Austin',
       'Katy', 'Sharpstown', 'Sugar Land', 'Galveston', 'Port Bolivar',
       'Washington Avenue', 'Briar Meadow', 'Latta', 'Jacksonville',
       'Couples Glen', 'Kissimmee', 'Lake Reams', 'Orlando',
       'Sand Lake Commons', 'Sky Lake', 'Daytona Beach', 'Ridgeland',
       'Florence', 'Meredith', 'Holly Springs', 'Chessington', 'Burtrose',
       'Parkway', 'Mcvan', 'Capitol One', 'University District',
       'Seattle', 'Redmond', 'Bellevue', 'San Francisco', 'Palo Alto',
       'Sunnyvale', 'Newark', 'Menlo Park', 'Old City', 'Savon Height',
       'Kilarney Woods', 'Townes at Everett Crossing', 'Huntington Woods',
       'Seaport', 'Medical Centre', 'Rose Hill', 'Soho', 'Tribeca',
       'Financial District', 'Oakland', 'Emeryville', 'Berkeley',
       'Kenner', 'CBD', 'Lower Garden District', 'Lakeview', 'Storyville',
       'New Orleans', 'Metairie', 'Chalmette', 'Arabi',
       'Pontchartrain Shores', 'Marigny', 'Covington', 'Mandeville',
       'Jamestown Court', 'Summerwinds', 'Parkwood',
       'Pontchartrain Beach', 'St Thomas', 'Banner Elk', 'Elk Park',
       'Newland', 'Boone', 'Stonewater', 'Lexington Park at Amberly',
       'Arlington Park at Amberly', 'Arlington', 'Kalorama Triangle',
       'K Street', 'West End', 'Connecticut Avenue', 'Columbia Heights',
       'Washington', 'Wake Forest', 'Lahore', 'Karachi', 'SOMISSPO',
       'West Berkeley', 'North Berkeley Hills', 'San Jose', 'Eagle Rock',
       'Winston Salem', 'Asheville', 'Topton', 'Hayesville',
       'Bryson City', 'Almond', 'Mebane', 'Agnew', 'Cory', 'Renaissance',
       'Santa Clara', 'NOMA', 'Sunnyside', 'Ingleside', 'Central',
       'Tenderloin', 'College Avenue', 'South', 'Southside',
       'South Berkeley', 'Mountain View', 'El Cerrito', 'Krendle Woods',
       'Wake Co.', 'Fuquay-Varina', 'Rawalpindi', 'Katunayake', 'Gampaha'],
      dtype=object)
```

In [15]:
```
## Q11 B. Get the unique Stop locations. (2 points)
#### Note: This question is based on the dataframe with no 'NA' values
```

In [16]:
```
Uber_Data['STOP*'].unique()
```

Out[16]:
```
array(['Fort Pierce', 'West Palm Beach', 'Palm Beach', 'Cary',
       'Morrisville', 'New York', 'Queens', 'East Harlem', 'NoMad',
       'Midtown', 'Midtown East', 'Hudson Square', 'Lower Manhattan',
       "Hell's Kitchen", 'Queens County', 'Gulfton', 'Downtown',
       'Houston', 'Jamestown Court', 'Durham', 'Whitebridge',
       'Lake Wellingborough', 'Raleigh', 'Umstead', 'Hazelwood',
       'Westpark Place', 'Meredith Townes', 'Leesville Hollow', 'Apex',
       'Chapel Hill', 'Williamsburg Manor', 'Macgregor Downs',
       'Edgehill Farms', 'Northwoods', 'Tanglewood', 'Preston',
       'Walnut Terrace', 'Jackson Heights', 'East Elmhurst',
       'Midtown West', 'Long Island City', 'Jamaica', 'Unknown Location',
       'Colombo', 'Nugegoda', 'Katunayaka', 'Islamabad', 'R?walpindi',
       'Noorpur Shahan', 'Heritage Pines', 'Waverly Place', 'Wayne Ridge',
       'Depot Historic District', 'Weston', 'West University',
       'South Congress', 'Arts District', 'Congress Ave District',
       'Red River District', 'The Drag', 'Convention Center District',
       'North Austin', 'Coxville', 'Katy', 'Alief', 'Sharpstown',
       'Sugar Land', 'Galveston', 'Port Bolivar', 'Washington Avenue',
       'Briar Meadow', 'Greater Greenspoint', 'Latta', 'Jacksonville',
       'Kissimmee', 'Isles of Buena Vista', 'Orlando', 'Lake Reams',
       'Vista East', 'Sky Lake', 'Sand Lake Commons', 'Daytona Beach',
       'Ridgeland', 'Florence', 'Cedar Hill', 'Holly Springs',
       'Harden Place', 'Chessington', 'Burtrose', 'Parkway',
       'Capitol One', 'University District', 'Redmond', 'Bellevue',
       'Seattle', 'Mcvan', 'Palo Alto', 'Sunnyvale', 'Newark',
       'Menlo Park', 'San Francisco', 'Parkway Museums', 'Hog Island',
       'Savon Height', 'Kildaire Farms', 'Kilarney Woods',
       'Gramercy-Flatiron', 'Tudor City', 'Soho', 'Tribeca',
       'Financial District', 'Kips Bay', 'Emeryville', 'Berkeley',
       'Oakland', 'Bay Farm Island', 'New Orleans',
       'Lower Garden District', 'Lakeview', 'Storyville',
       'Faubourg Marigny', 'Metairie', 'Kenner', 'Bywater', 'Chalmette',
       'Arabi', 'Pontchartrain Shores', 'Marigny', 'Covington',
       'Mandeville', 'Summerwinds', 'Parkwood', 'Pontchartrain Beach',
       'CBD', 'St Thomas', 'Banner Elk', 'Elk Park', 'Newland', 'Boone',
       'Stonewater', 'Lexington Park at Amberly',
       'Arlington Park at Amberly', 'Washington', 'K Street',
       'Kalorama Triangle', 'Northwest Rectangle', 'Columbia Heights',
       'Arlington', 'Farmington Woods', 'Wake Forest', 'Lahore',
       'Karachi', 'French Quarter', 'North Berkeley Hills', 'Southside',
       'San Jose', 'Eagle Rock', 'Huntington Woods', 'Winston Salem',
       'Asheville', 'Topton', 'Hayesville', 'Bryson City', 'Almond',
       'Mebane', 'Santa Clara', 'Cory', 'Agnew', 'Renaissance',
       'West Berkeley', 'Central', 'Sunnyside', 'Ingleside',
       'Potrero Flats', 'SOMISSPO', 'Tenderloin', 'College Avenue',
       'South', 'Southwest Berkeley', 'South Berkeley', 'Mountain View',
       'El Cerrito', 'Wake Co.', 'Fuquay-Varina', 'Rawalpindi', 'Gampaha',
       'Ilukwatta'], dtype=object)
```

Q12. What is the total number of unique start locations? (2 points)

Note: Use the original dataframe without dropping 'NA' values

In [20]:
```
Uber_Data['START*'].nunique()
```

Out[20]:
```
176
```

Q13. What is the total number of unique stop locations. (2 points)

Note: Use the original dataframe without dropping 'NA' values.

In [21]:
```
Uber_Data['STOP*'].nunique()
```

Out[21]:
```
187
```

Q14. Display all Uber trips that has the starting point as San Francisco. (2 points)

Note: Use the original dataframe without dropping the 'NA' values.

In [22]:
```
Uber_Data[Uber_Data['START*'] == "San Francisco"]
```

Out[22]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
362	05-09-2016 14:39	05-09-2016 15:06	Business	San Francisco	Palo Alto	20.5	Between Offices
440	6/14/2016 16:09	6/14/2016 16:39	Business	San Francisco	Emeryville	11.6	Meeting
836	10/19/2016 14:02	10/19/2016 14:31	Business	San Francisco	Berkeley	10.8	NaN
917	11-07-2016 19:17	11-07-2016 19:57	Business	San Francisco	Berkeley	13.2	Between Offices
919	11-08-2016 12:16	11-08-2016 12:49	Business	San Francisco	Berkeley	11.3	Meeting
927	11-09-2016 18:40	11-09-2016 19:17	Business	San Francisco	Oakland	12.7	Customer Visit
933	11-10-2016 15:17	11-10-2016 15:22	Business	San Francisco	Oakland	9.9	Temporary Site
966	11/15/2016 20:44	11/15/2016 21:00	Business	San Francisco	Berkeley	11.8	Temporary Site

Q15. What is the most popular starting point for the Uber drivers? (2 points)

Note: Use the original dataframe without dropping the 'NA' values.

Hint:Popular means the place that is visited the most

In [36]:
```
Starting_Point = Uber_Data['START*'].dropna()
Uber_Data = pd.DataFrame(Starting_Point.value_counts())
Uber_Data.sort_values(['START*'], ascending=False)
Uber_Data = Uber_Data.reset_index()
Uber_Data = Uber_Data.rename(columns={'index':'Start_Location', 'START*': 'Count'})
Uber_Data.loc[Uber_Data['Count'] == max(Uber_Data['Count'])]
```

Out[36]:

	Start_Location	Count
0	Cary	201

In [17]:
```
Uber_Data['START*'].value_counts()
```

Out[17]:
```
Cary                 201
Unknown Location     148
Morrisville           85
Whitebridge           68
Islamabad             57
                    ...
Ridgeland              1
Daytona Beach          1
Sky Lake               1
Sand Lake Commons      1
Gampaha                1
Name: START*, Length: 176, dtype: int64
```

Q16. What is the most popular dropping point for the Uber drivers? (2 points)

Note: Use the original dataframe without dropping the 'NA' values.

Hint: Popular means the place that is visited the most

In [33]:
```
Dropping_Point = Uber_Data['STOP*'].dropna()
Uber_Data = pd.DataFrame(Dropping_Point.value_counts())
Uber_Data.sort_values(['STOP*'], ascending=False)
Uber_Data = Uber_Data.reset_index()
Uber_Data = Uber_Data.rename(columns={'index':'Drop_Location', 'STOP*': 'Count'})
Uber_Data.loc[Uber_Data['Count'] == max(Uber_Data['Count'])]
```

Out[33]:

	Drop_Location	Count
0	Cary	203

In [18]:
```
Uber_Data['STOP*'].value_counts()
```

Out[18]:
```
Cary                 203
Unknown Location     149
Morrisville           84
Whitebridge           65
Islamabad             58
                    ...
Daytona Beach          1
Sand Lake Commons      1
Sky Lake               1
Vista East             1
Ilukwatta              1
Name: STOP*, Length: 187, dtype: int64
```

Q17. What is the most frequent route taken by Uber drivers. (3 points)

Note: This question is based on the new dataframe with no 'na' values.

Hint-Print the most frequent route taken by Uber drivers (Route= combination of START & END points present in the Data set).

In [39]:
```
df = Uber_Data.dropna()
df = pd.DataFrame(df.groupby(['START*','STOP*']).size())
df = df.rename(columns={0:'Count'})
df = df.sort_values(['Count'], ascending=False)
df.loc[df['Count'] == max(df['Count'])]
```

Out[39]:

START*	STOP*	Count
Cary	Morrisville	52

In [40]:
```
df = Uber_Data.dropna()
```

In [42]:
```
df = Uber_Data.groupby(['START*','STOP*']).size().sort_values(ascending=False).head()
df
```

Out[42]:
```
START*            STOP*
Unknown Location  Unknown Location    86
Morrisville       Cary                75
Cary              Morrisville         67
                  Cary                53
                  Durham              36
dtype: int64
```

In [ ]:
```
# Most frequent route taken by Uber drivers - Start Location - "Cary" and Stop Location - "Morrisville"
```

Q18. Display all types of purposes for the trip in an array. (2 points)

Note: This question is based on the new dataframe with no 'NA' values.

In [40]:
```
print(np.array(Uber_Data["PURPOSE*"].dropna().unique()))
```

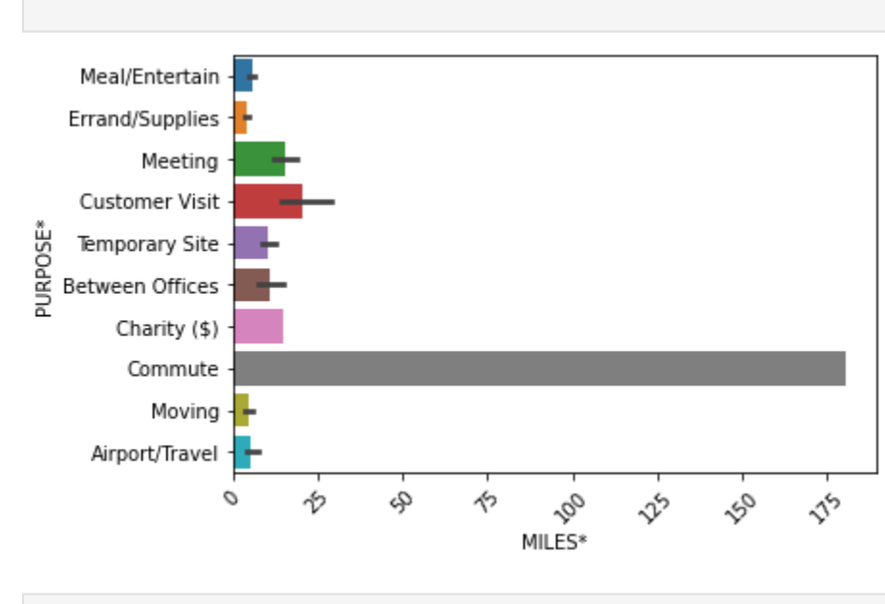
```
['Meal/Entertain' 'Errand/Supplies' 'Meeting' 'Customer Visit'
 'Temporary Site' 'Between Offices' 'Charity ($)' 'Commute' 'Moving'
 'Airport/Travel']
```

Q19. Plot a bar graph of Purpose vs Miles(Distance). What can you infer from the plot(2 +2 points)

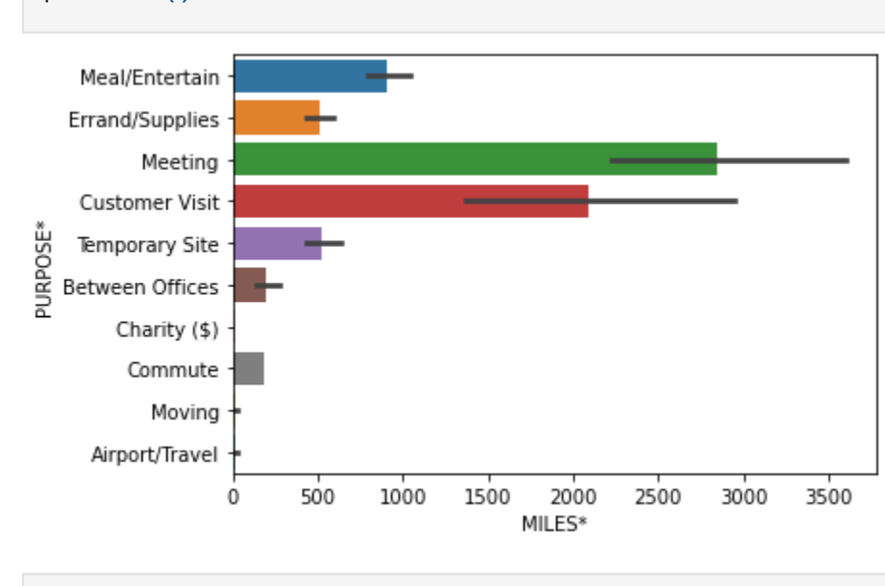
Note: Use the original dataframe without dropping the 'NA' values.

Hint:You have to plot total/sum miles per purpose

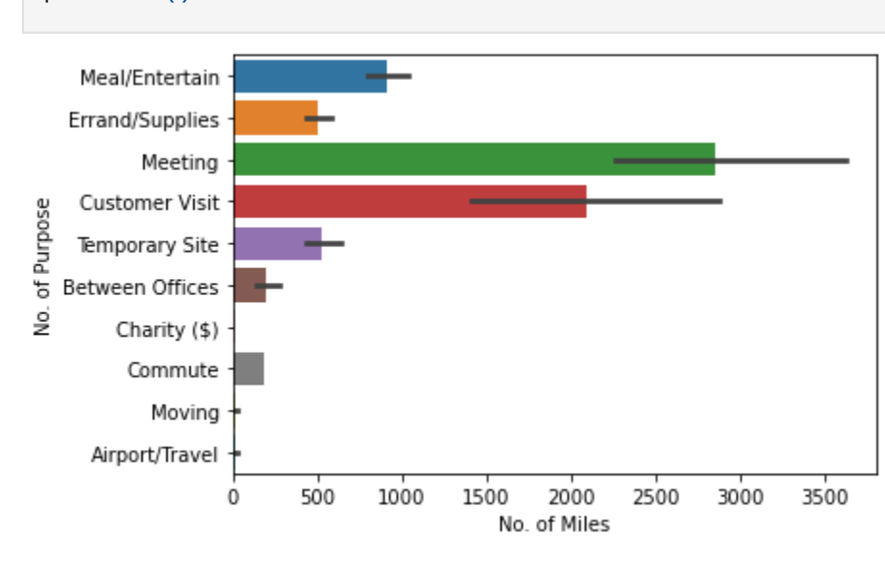
In [41]:
```
sns.barplot(x=Uber_Data['MILES*'], y=Uber_Data['PURPOSE*'])
plt.xticks(rotation=45)
plt.show()
```



In [14]:
```
sns.barplot(x=Uber_Data['MILES*'],y=Uber_Data['PURPOSE*'],estimator=sum)
plt.show()
```



In [10]:
```
sns.barplot(x=Uber_Data['MILES*'],y=Uber_Data['PURPOSE*'],estimator=np.sum)
plt.xlabel('No. of Miles')
plt.ylabel('No. of Purpose')
plt.show()
```



Q20. Display a dataframe of Purpose and the total distance travelled for that particular Purpose. (3 points)

Note: Use the original dataframe without dropping "NA" values

In [42]:
```
Uber_Data.groupby(['PURPOSE*']).agg({'MILES*' : ['sum']})
```

Out[42]:

	MILES*
	sum
PURPOSE*	
Airport/Travel	16.5
Between Offices	197.0
Charity ($)	15.1
Commute	180.2
Customer Visit	2089.5
Errand/Supplies	508.0
Meal/Entertain	911.7
Meeting	2851.3
Moving	18.2
Temporary Site	523.7

In [86]:
```
Uber_Data.groupby('PURPOSE*').sum()
```

Out[86]:

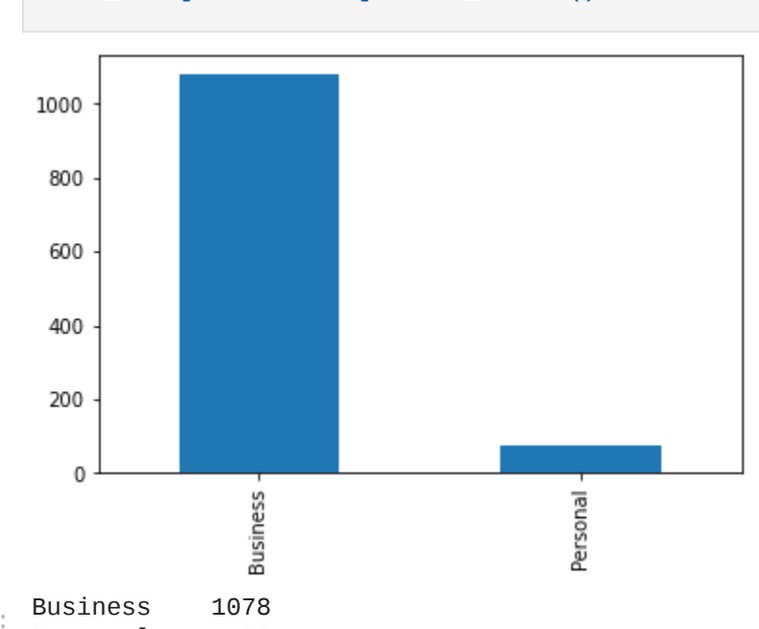
	MILES*
PURPOSE*	
Airport/Travel	16.5
Between Offices	197.0
Charity ($)	15.1
Commute	180.2
Customer Visit	2089.5
Errand/Supplies	508.0
Meal/Entertain	911.7
Meeting	2851.3
Moving	18.2
Temporary Site	523.7

Q21. Generate a plot showing count of trips vs category of trips. What can you infer from the plot (2 +1 points)

Note: Use the original dataframe without dropping the 'NA' values.

In [64]:
```
Uber_Data["CATEGORY*"].value_counts().plot(kind = "bar")
plt.show()

Uber_Data["CATEGORY*"].value_counts()
```



Out[64]:
```
Business    1078
Personal      77
Name: CATEGORY*, dtype: int64
```

Q22. What percentage of Miles were clocked under Business Category and what percentage of Miles were clocked under Personal Category ? (3 points)

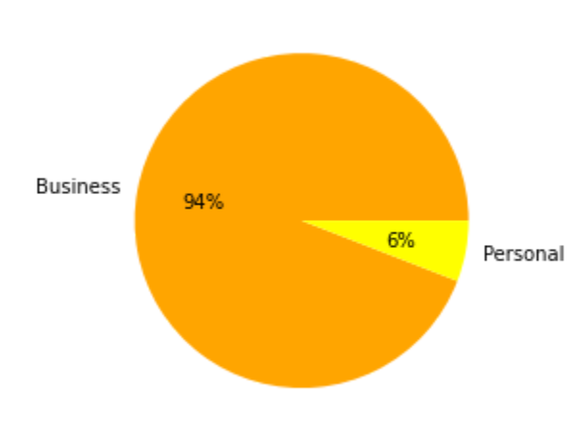
Note:Use the original dataframe without dropping the 'NA' values.

In [12]:
```
GroupBy_Data = Uber_Data[["CATEGORY*","MILES*"]].groupby(['CATEGORY*']).agg({'MILES*': ['sum']})

data = GroupBy_Data.values.flatten()
labels = GroupBy_Data.index

Colors = ["orange" , "yellow"]

#create pie chart
plt.pie(data, labels = labels, colors = Colors, autopct='%.0f%%',)
plt.show()
```



In [70]:
```
Uber_Data.columns
```

Out[70]:
```
Index(['START_DATE*', 'END_DATE*', 'CATEGORY*', 'START*', 'STOP*', 'MILES*',
       'PURPOSE*'],
      dtype='object')
```

In [81]:
```
Uber_Data.groupby("CATEGORY*").sum().dropna()
```

Out[81]:

	MILES*
CATEGORY*	
Business	11487.0
Personal	717.7

In [85]:
```
Uber_Data.groupby("MILES*").sum().dropna()
```

Out[85]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	PURPOSE*
MILES*						
0.5	1/26/2016 17:272/17/2016 16:3806-08-2016 17:16	1/26/2016 17:292/17/2016 16:4306-08-2016 17:18	BusinessBusinessBusiness	CaryKatunayakaSoho	CaryKatunayakaTribeca	Errand/SuppliesErrand/SuppliesErrand/Supplies
0.6	07-05-2016 16:4811-12-2016 13:4612/24/2016 22:...	07-05-2016 16:5211-12-2016 13:5012/24/2016 22:...	BusinessBusinessBusinessBusinessBusiness	WhitebridgeCentralLahoreLahoreKarachi	WhitebridgeWest BerkeleyLahoreLahoreKarachi	Errand/SuppliesErrand/SuppliesErrand/SuppliesM...
0.7	04-01-2016 16:5206-08-2016 14:319/13/2016 16:5...	04-01-2016 16:5706-08-2016 14:379/13/2016 17:0...	PersonalBusinessBusinessBusinessBusine...	KissimmeeMedical CentreUnknown LocationUnknown...	KissimmeeTudor CityUnknown LocationUnknown Loc...	Errand/SuppliesErrand/SuppliesMeeting
0.8	01-07-2016 13:2703-04-2016 16:1603-11-2016 10:...	01-07-2016 13:3303-04-2016 16:2203-11-2016 10:...	BusinessBusinessBusinessBusinessBusiness	CaryFayetteville StreetCongress Ave DistrictWe...	CaryDepot Historic DistrictDowntownCentralSout...	MeetingErrand/SuppliesCustomer Visit
0.9	3/21/2016 16:0506-08-2016 17:597/18/2016 20:28...	3/21/2016 16:1306-08-2016 18:057/18/2016 20:32...	BusinessBusinessBusinessBusinessBusine...	DowntownTribecaHazelwoodLahoreUnknown Location...	MidtownFinancial DistrictWestonLahoreUnknown L...	Meal/EntertainErrand/SuppliesErrand/Supplies
...	...	...	...	...	...	...
195.3	7/14/2016 16:39	7/14/2016 20:05	Business	Morrisville	Banner Elk	0
195.6	9/27/2016 21:01	9/28/2016 2:37	Business	Unknown Location	Unknown Location	0
195.9	10/30/2016 15:22	10/30/2016 18:23	Business	Asheville	Mebane	0
201.0	3/25/2016 22:54	3/26/2016 1:39	Business	Jacksonville	Kissimmee	Meeting
310.3	3/25/2016 16:52	3/25/2016 22:22	Business	Latta	Jacksonville	Customer Visit

256 rows × 6 columns

In [83]:
```
Uber_Data.groupby("MILES*").sum()/ Uber_Data.groupby("CATEGORY*").sum().dropna()
```

Out[83]:

	CATEGORY*	END_DATE*	MILES*	PURPOSE*	START*	START_DATE*	STOP*
0.5	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.6	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.7	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.8	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.9	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...
195.9	NaN	NaN	NaN	NaN	NaN	NaN	NaN
201.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
310.3	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Business	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Personal	NaN	NaN	NaN	NaN	NaN	NaN	NaN

258 rows × 7 columns

THE END