In [1]:
```
import pandas as pds
import numpy as np
import matplotlib.pyplot as plt
df = pds.read_csv('G:\\Uber project\\uber.csv')
df
```

Out[1]:

	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	Meeting
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 8:33	1/11/2016 9:21	Business	East Harlem	Nolitad	6.4	Temporary Site
14	1/11/2016 11:56	1/11/2016 12:03	Business	Flatiron District	Midtown	1.6	Errand/Supplies
15	1/11/2016 12:32	1/11/2016 13:46	Business	Midtown	Midtown East	1.7	Meal/Entertain
16	1/11/2016 14:30	1/11/2016 14:43	Business	Midtown East	Midtown	1.9	Meal/Entertain
17	1/12/2016 12:33	1/12/2016 12:49	Business	Midtown	Hudson Square	1.9	Meal/Entertain
18	1/12/2016 13:53	1/12/2016 13:59	Business	Hudson Square	Lower Manhattan	4.0	Meal/Entertain
19	1/12/2016 14:42	1/12/2016 14:56	Business	Lower Manhattan	Hudson Square	1.8	Errand/Supplies
20	1/12/2016 15:13	1/12/01 15:28	Personal	Hudson Square	Hell's Kitchen	2.4	Customer Visit
21	1/12/2016 15:42	1/12/2016 15:54	Personal	Hell's Kitchen	Midtown	2.0	Errand/Supplies
22	1/12/2016 16:02	1/12/2016 17:00	Personal	New York	Queens Country	15.1	Meeting
23	1/13/2016 13:54	1/13/2016 14:07	Personal	Downtown	Gulfsteam	11.2	Meeting
24	1/13/2016 15:00	1/13/2016 15:28	Personal	Gulfton	Downtown	11.8	Meeting
25	1/14/2016 16:26	1/17/2016 17:05	Business	Houston	Houston	21.9	Customer Visit
26	1/14/2016 21:35	1/14/2016 21:45	Business	Eagan Park	Jamestown Court	3.9	Errand/Supplies
27	1/15/2016 0:41	1/15/2016 1:01	Business	Morrisville	Cary	8.0	Errand/Supplies
28	1/15/2016 11:43	1/15/2016 12:03	Business	Cary	Durham	10.4	Meal/Entertain

In [2]:
```
df.tail(10)
```

Out[2]:

	START_DATE	END_DATE	CATEGORY	START	STOP	MILES	PURPOSE
19	1/12/2016 14:42	1/12/2016 14:56	Business	Lower Manhattan	Hudson Square	1.8	Errand/Supplies
20	1/12/2016 15:13	1/12/01 15:28	Personal	Hudson Square	Hell's Kitchen	2.4	Customer Visit
21	1/12/2016 15:42	1/12/2016 15:54	Personal	Hell's Kitchen	Midtown	2.0	Errand/Supplies
22	1/12/2016 16:02	1/12/2016 17:00	Personal	New York	Queens Country	15.1	Meeting
23	1/13/2016 13:54	1/13/2016 14:07	Personal	Downtown	Gulfsteam	11.2	Meeting
24	1/13/2016 15:00	1/13/2016 15:28	Personal	Gulfton	Downtown	11.8	Meeting
25	1/14/2016 16:26	1/17/2016 17:05	Business	Houston	Houston	21.9	Customer Visit
26	1/14/2016 21:35	1/14/2016 21:45	Business	Eagan Park	Jamestown Court	3.9	Errand/Supplies
27	1/15/2016 0:41	1/15/2016 1:01	Business	Morrisville	Cary	8.0	Errand/Supplies
28	1/15/2016 11:43	1/15/2016 12:03	Business	Cary	Durham	10.4	Meal/Entertain

In [3]:
```
df.shape
```

Out[3]:
```
(29, 7)
```

In [4]:
```
df.size
```

Out[4]:
```
203
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29 entries, 0 to 28
Data columns (total 7 columns):
#   Column      Non-Null Count  Dtype
---  ------      --------------  -----
0   START_DATE  29 non-null     object
1   END_DATE    29 non-null     object
2   CATEGORY    29 non-null     object
3   START       29 non-null     object
4   STOP        29 non-null     object
5   MILES       29 non-null     float64
6   PURPOSE     28 non-null     object
dtypes: float64(1), object(6)
memory usage: 1.7+ KB
```

checking for missing values

In [6]:
```
df.isnull().values.any()
```

Out[6]:
```
True
```

how many missing values are present

In [7]:
```
df.isnull().values.sum()
```

Out[7]:
```
1
```

how to dropna in the missing values

In [8]:
```
dfs = df.dropna()
dfs.isnull().values.any()
```

Out[8]:
```
False
```

get the summary of the original data (before dropping the "na" values)

In [9]:
```
df.describe()
```

Out[9]:

	MILES
count	29.000000
mean	8.682759
std	11.734749
min	0.800000
25%	2.400000
50%	5.100000
75%	10.400000
max	63.700000

check the information of the new dataframe

In [10]:
```
dfs.describe()
```

Out[10]:

	MILES
count	28.000000
mean	8.778571
std	11.918580
min	0.900000
25%	2.300000
50%	5.650000
75%	10.500000
max	63.700000

Note: This question is based on the dataframe with no 'na' values in the 'start' variable

In [11]:
```
un_start_destination = dfs["START"].dropna()
unique_start = set(un_start_destination)
unique_start
```

Out[11]:
```
{' Midtown East',
 'Agabat',
 'Downtown',
 'Eagan Park',
 'East Harlem',
 'Elmhurst',
 'Flatiron District',
 'Fort Pierce',
 'Gulfton',
 'Hell's Kitchen',
 'Holyoke',
 'Hudson Square',
 'Jamaica',
 'Lower Manhattan',
 'Midtown',
 'Morrisville',
 'New York',
 'West Palm Beach'}
```

In [12]:
```
len(unique_start)
```

Out[12]:
```
18
```

In [13]:
```
stop_destination = dfs["STOP"].dropna()
unique_stop = set(stop_destination)
unique_stop
```

Out[13]:
```
{'Cary',
 'Downtown',
 'Durham',
 'East Harlem',
 'Fort Pierce',
 'Gulfstream',
 'Hell's Kitchen',
 'Houston',
 'Hudson Square',
 'Jamestown Court',
 'Lower Manhattan',
 'Midtown',
 'Midtown East',
 'Morrisville',
 'New York',
 'Nolita',
 'Palm Beach',
 'Queens Country',
 'West Palm Beach'}
```

In [14]:
```
len(unique_stop)
```

Out[14]:
```
20
```

Print all the uber trips that has the starting point of fort pierce

use the original dataframe without dropping the 'na' values.

In [15]:
```
dfs[dfs["START"]=="Fort Pierce"]
```

Out[15]:

	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 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

What is the most popular starting point for all the Uber drivers

this is based on the dataframe with no 'na' values in the 'start' variables

In [16]:
```
staring_point = dfs["START"].dropna()
df1 pds.DataFrame(staring_point.value_counts())
df.sort_values(["START"],ascending = False)
df = df.reset_index()
df1.rename(columns={'index':'starting_destination',"START":"count"})
df.loc[df["count"]==max(df["count"])]
#df.loc[df["count"]>=min(df["count"])]
```

Out[16]:

	starting_destination	count
0	Fort Pierce	4

In [17]:
```
stop_point = dfs["STOP"].dropna()
df = pds.DataFrame(stop_point.value_counts())
df.sort_values(["STOP"],ascending = False)
df = df.reset_index()
df.rename(columns={'index':"stop_destination","STOP":"count"})
df.loc[df["count"]==max(df["count"])]
#df.loc[df["count"]>=min(df["count"])]
```

Out[17]:

	stop_destination	count
0	Midtown	3
1	Fort Pierce	3

List the most frequent route taken by Uber drivers,

this is based on the dataframe with no'na' values.

In [18]:
```
df = dfs.dropna()
df = pds.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[18]:

		count
START	STOP	
Fort Pierce	Fort Pierce	3

Print all types of purposes for the trip in an array.

This is based on the dataframe with no 'na' values in the 'purpose' variable

In [19]:
```
print(np.array(dfs["PURPOSE"].dropna().unique()))
dfs["MILES"].groupby(dfs["PURPOSE"]).sum()
```

```
['Meal/Entertain' 'Errand/Supplies' 'Meeting' 'Customer Visit'
 'Temporary Site']
```

Out[19]:
```
PURPOSE
Customer Visit      104.5
Errand/Supplies      22.1
Meal/Entertain       29.3
Meeting              83.5
Temporary Site        6.4
Name: MILES, dtype: float64
```

Plot a bar graph of purposes vs distance

this is based on the original data frame

In [20]:
```
df = pds.DataFrame(dfs["MILES"].groupby(dfs["PURPOSE"]).sum())
df.plot(kind = "bar")
plt.show()
```

Print a dataframe of purposes and the distance travelled for that particular purpose

this is based on the original dataframe

In [21]:
```
df
```

Out[21]:

	MILES
PURPOSE	
Customer Visit	104.5
Errand/Supplies	22.1
Meal/Entertain	29.3
Meeting	83.5
Temporary Site	6.4

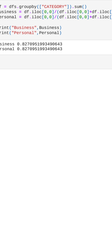
Plot number of trips vs category of trips.

this is based on the original dataframe

In [22]:
```
dfs.head()

df1pds.DataFrame(dfs["CATEGORY"].value_counts())
df1.reset_index()

df.plot(kind = "bar")
plt.show()
df
```



Out[22]:

CATEGORY	
Business	23
Personal	5

what is porportion of trips that is business and what is the proportion of trips that is personal?

Note : the proportion calculation is with respect to the 'miles' variable.

this question is based on the original dataframe

In [23]:
```
df = dfs.groupby("CATEGORY").sum()
Business = df.iloc[0,0]/df.iloc[0,0]+df.iloc[1,0]
Personal = df.iloc[0,0]/df.iloc[0,0]+df.iloc[1,0]

print("Business",Business)
print("Personal",Personal)
```

```
Business 0.8270051993469643
Personal 0.8270051993469643
```

In [ ]:
```

```