Uber Data Analysis

June 5, 2023

1 Description

1.Imported libraries and loaded Uber ride dataset.

2.Examined data overview, including top and bottom rows, element count, and dimensions.

3.Identified and handled null values, cleaned column names.

4.Filtered records with missing ride purposes.

5.Extracted rides details for some locations.

6.Detected records with maximum miles traveled.

7.Removed records without stop location.

8.Analyzed unique start and stop locations.

9.Identified rides with same start and stop locations.

10.Categorized rides as business or personal.

11.Determined popular starting points and longest routes.

12.Analyzed monthly ride patterns and average distances.

==========

1.Imported libraries and loaded Uber ride dataset.

[148]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[149]:
```
df=pd.read_csv('D:\DS\python\case study\My Uber Drives - 2016.csv')
```

2.Examined data overview, including top and bottom rows, element count, and dimensions.

[150]:
```
print(' Get the top 7 rows of the dataset.\n')
df.head(7)
```

```
 Get the top 7 rows of the dataset.
```

[150]:

	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

[151]:
```
print('Get the last 5 rows of the dataset.\n')
df.tail(5)
```

```
Get the last 5 rows of the dataset.
```

[151]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
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	Totals	NaN	NaN	NaN	NaN	12204.7	NaN

[152]:
```
print('Get the total number of rows and columns in the dataset.\n',df.shape)
```

```
Get the total number of rows and columns in the dataset.
```

```
(1156, 7)
```

[153]:
```
print(' Get the total number of elements in the dataset\n',df.size)
```

```
Get the total number of elements in the dataset
8092
```

3.Identified and handled null values, cleaned column names.

[154]:
```
print(' Get the total number of NULL values across every column in the dataset.
 ↪\n')
df.isna().sum()
```

```
Get the total number of NULL values across every column in the dataset.
```

[154]:
```
START_DATE*      0
END_DATE*        1
CATEGORY*        1
START*           1
STOP*            1
MILES*           0
PURPOSE*       503
dtype: int64
```

[155]:
```
print('Get the total number of Non-NULL values across every column in the␣
 ↪dataset.\n')
df.notna().sum()
```

```
Get the total number of Non-NULL values across every column in the dataset.
```

[155]:
```
START_DATE*    1156
END_DATE*      1155
CATEGORY*      1155
START*         1155
STOP*          1155
MILES*         1156
PURPOSE*        653
dtype: int64
```

[180]:
```
print('Remove the * in every column name using the rename function.\n')
df.rename(columns={'START_DATE*':'START_DATE','END_DATE*':'END_DATE','MILES*':
 ↪'MILES','PURPOSE*':'PURPOSE','CATEGORY*':'CATEGORY','START*':'START','STOP*':
 ↪'STOP'})
```

```
Remove the * in every column name using the rename function.
```

[180]:
```
              START_DATE            END_DATE  CATEGORY        START  \
0    2016-01-01 21:11:00 2016-01-01 21:17:00  Business  Fort Pierce
1    2016-01-02 01:25:00 2016-01-02 01:37:00  Business  Fort Pierce
2    2016-01-02 20:25:00 2016-01-02 20:38:00  Business  Fort Pierce
3    2016-01-05 17:31:00 2016-01-05 17:45:00  Business  Fort Pierce
4    2016-01-06 14:42:00 2016-01-06 15:49:00  Business  Fort Pierce
…                    …                   …         …           …
1150 2016-12-31 01:07:00 2016-12-31 01:14:00  Business      Kar?chi
1151 2016-12-31 13:24:00 2016-12-31 13:42:00  Business      Kar?chi
1153 2016-12-31 21:32:00 2016-12-31 21:50:00  Business   Katunayake
1154 2016-12-31 22:08:00 2016-12-31 23:51:00  Business      Gampaha
1155                 NaT                  NaT       NaN          NaN

                    STOP    MILES          PURPOSE   DAY
0           Fort Pierce      5.1   Meal/Entertain   1.0
1           Fort Pierce      5.0              NaN   2.0
2           Fort Pierce      4.8  Errand/Supplies   2.0
3           Fort Pierce      4.7          Meeting   5.0
4       West Palm Beach     63.7   Customer Visit   6.0
…                   …      …               …   …
1150            Kar?chi      0.7          Meeting  31.0
1151  Unknown Location      3.9   Temporary Site  31.0
1153            Gampaha      6.4   Temporary Site  31.0
1154          Ilukwatta     48.2   Temporary Site  31.0
1155                NaN  12204.7              NaN   NaN

[1070 rows x 8 columns]
```

4.Filtered records with missing ride purposes.

[156]:
```
print('Get the entries having NULL values in the Purpose column.\n')
df[df['PURPOSE*'].isna()]
```

```
Get the entries having NULL values in the Purpose column.
```

[156]:
```
           START_DATE*         END_DATE* CATEGORY*            START*  \
1         1/2/2016 1:25      1/2/2016 1:37  Business       Fort Pierce
32       1/19/2016 9:09     1/19/2016 9:23  Business       Whitebridge
85        2/9/2016 10:54     2/9/2016 11:07  Personal       Whitebridge
86        2/9/2016 11:43     2/9/2016 11:50  Personal        Northwoods
87        2/9/2016 13:36     2/9/2016 13:52  Personal        Tanglewood
…                    …                  …         …                 …
1066  12/19/2016 14:37   12/19/2016 14:50  Business  Unknown Location
1069  12/19/2016 19:05   12/19/2016 19:17  Business         Islamabad
1071   12/20/2016 8:49    12/20/2016 9:24  Business  Unknown Location
1143  12/29/2016 20:53   12/29/2016 21:42  Business           Kar?chi
1155            Totals                NaN       NaN               NaN
```

```
                    STOP*   MILES* PURPOSE*
1              Fort Pierce      5.0      NaN
32    Lake Wellingborough      7.2      NaN
85              Northwoods      5.3      NaN
86              Tanglewood      3.0      NaN
87                 Preston      5.1      NaN
…                     …      …       …
1066     Unknown Location      5.4      NaN
1069     Unknown Location      2.2      NaN
1071           Rawalpindi     12.0      NaN
1143     Unknown Location      6.4      NaN
1155                  NaN  12204.7      NaN

[503 rows x 7 columns]
```

[157]:
```
print('Get the entries having Non-NULL values in the Purpose. \n')
df[~df['PURPOSE*'].isna()]
```

```
Get the entries having Non-NULL values in the Purpose.
```

[157]:
```
           START_DATE*         END_DATE* CATEGORY*            START*  \
0       1/1/2016 21:11    1/1/2016 21:17  Business       Fort Pierce
2       1/2/2016 20:25    1/2/2016 20:38  Business       Fort Pierce
3       1/5/2016 17:31    1/5/2016 17:45  Business       Fort Pierce
4       1/6/2016 14:42    1/6/2016 15:49  Business       Fort Pierce
5       1/6/2016 17:15    1/6/2016 17:19  Business   West Palm Beach
…                  …                 …       …                 …
1150   12/31/2016 1:07   12/31/2016 1:14  Business           Kar?chi
1151  12/31/2016 13:24  12/31/2016 13:42  Business           Kar?chi
1152  12/31/2016 15:03  12/31/2016 15:38  Business  Unknown Location
1153  12/31/2016 21:32  12/31/2016 21:50  Business       Katunayake
1154  12/31/2016 22:08  12/31/2016 23:51  Business           Gampaha

                  STOP*  MILES*         PURPOSE*
0          Fort Pierce     5.1   Meal/Entertain
2          Fort Pierce     4.8  Errand/Supplies
3          Fort Pierce     4.7          Meeting
4      West Palm Beach    63.7   Customer Visit
5      West Palm Beach     4.3   Meal/Entertain
…                  …     …              …
1150           Kar?chi     0.7          Meeting
1151  Unknown Location     3.9   Temporary Site
1152  Unknown Location    16.2          Meeting
1153           Gampaha     6.4   Temporary Site
1154         Ilukwatta    48.2   Temporary Site
```

```
[653 rows x 7 columns]
```

5.Extracted rides details for some locations.

[159]:
```
print(' Get the entries in the data where the START location is Fort Pierce\n')
df[df['START*']=='Fort Pierce']
```

```
 Get the entries in the data where the START location is Fort Pierce
```

[159]:

	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

[160]:
```
print(' Get the entries in the data where the STOP location is Fort Pierce\n')
df[df['STOP*']=='Fort Pierce']
```

```
 Get the entries in the data where the STOP location is Fort Pierce
```

[160]:

	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

6.Detected records with maximum miles traveled.

[161]:
```
print('Sort the entries in the data in descending order of the MILES column.\n')
df.sort_values('MILES*',ascending=False)
```

```
Sort the entries in the data in descending order of the MILES column.
```

[161]:
```
           START_DATE*         END_DATE* CATEGORY*            START*  \
1155            Totals               NaN       NaN               NaN
269    3/25/2016 16:52   3/25/2016 22:22  Business             Latta
270    3/25/2016 22:54    3/26/2016 1:39  Business      Jacksonville
881   10/30/2016 15:22  10/30/2016 18:23  Business         Asheville
776    9/27/2016 21:01    9/28/2016 2:37  Business  Unknown Location
…                  …                 …       …                 …
1121  12/27/2016 12:53  12/27/2016 12:57  Business           Kar?chi
1110  12/24/2016 22:04  12/24/2016 22:09  Business            Lahore
44     1/26/2016 17:27   1/26/2016 17:29  Business              Cary
420     6/8/2016 17:16    6/8/2016 17:18  Business              Soho
120    2/17/2016 16:38   2/17/2016 16:43  Business       Katunayaka

                   STOP*   MILES*         PURPOSE*
1155               NaN  12204.7              NaN
269       Jacksonville    310.3   Customer Visit
270          Kissimmee    201.0          Meeting
881             Mebane    195.9              NaN
776   Unknown Location    195.6              NaN
…                  …      …                 …
1121           Kar?chi      0.6   Meal/Entertain
1110            Lahore      0.6  Errand/Supplies
44                Cary      0.5  Errand/Supplies
420             Tribeca      0.5  Errand/Supplies
120          Katunayaka      0.5  Errand/Supplies

[1156 rows x 7 columns]
```

7.Removed records without stop location.

[162]:
```
# 17. Write a code to drop all the rows where there are NULL values in the STOP␣
 ↪column.
print('Drop all the rows where there are NULL values in the STOP column.\n')
df[df['STOP*'].isna()==False]
```

```
Drop all the rows where there are NULL values in the STOP column.
```

[162]:
```
           START_DATE*         END_DATE* CATEGORY*            START*  \
0       1/1/2016 21:11    1/1/2016 21:17  Business       Fort Pierce
1        1/2/2016 1:25     1/2/2016 1:37  Business       Fort Pierce
2       1/2/2016 20:25    1/2/2016 20:38  Business       Fort Pierce
3       1/5/2016 17:31    1/5/2016 17:45  Business       Fort Pierce
4       1/6/2016 14:42    1/6/2016 15:49  Business       Fort Pierce
…                  …                 …       …                 …
1150   12/31/2016 1:07   12/31/2016 1:14  Business           Kar?chi
1151  12/31/2016 13:24  12/31/2016 13:42  Business           Kar?chi
1152  12/31/2016 15:03  12/31/2016 15:38  Business  Unknown Location
```

```
1153  12/31/2016 21:32  12/31/2016 21:50  Business        Katunayake
1154  12/31/2016 22:08  12/31/2016 23:51  Business           Gampaha

                  STOP*  MILES*         PURPOSE*
0          Fort Pierce     5.1   Meal/Entertain
1          Fort Pierce     5.0              NaN
2          Fort Pierce     4.8  Errand/Supplies
3          Fort Pierce     4.7          Meeting
4       West Palm Beach    63.7   Customer Visit
…                  …     …               …
1150           Kar?chi     0.7          Meeting
1151  Unknown Location     3.9   Temporary Site
1152  Unknown Location    16.2          Meeting
1153           Gampaha     6.4   Temporary Site
1154         Ilukwatta    48.2   Temporary Site

[1155 rows x 7 columns]
```

[163]:
```
# 18. Get the Statistical Properties about the numerical columns in the data.
print(df.describe())
```

```
            MILES*
count   1156.000000
mean      21.115398
std      359.299007
min        0.500000
25%        2.900000
50%        6.000000
75%       10.400000
max    12204.700000
```

8.Analyzed unique start and stop locations.

[164]:
```
print('\n\n\nthe unique and total number of unique values in the␣
  ↪START\n',df['START*'].unique(),'\n\n\nand STOP column of the␣
  ↪data\n\n\n',df['STOP*'].unique())
```

```
the unique and total number of unique values in the START
 ['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' 'Kar?chi' 'Katunayake' 'Gampaha' nan]
```

and STOP column of the data

```
 ['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' 'Kar?chi' 'Gampaha' 'Ilukwatta'
nan]
```

9.Identified rides with same start and stop locations.

[165]:
```
print('rides where we have the same START and STOP locations\n')
df[df['START*']==df['STOP*']]
```

```
rides where we have the same START and STOP locations
```

[165]:

	START_DATE*	END_DATE*	CATEGORY*	START*	\
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	
5	1/6/2016 17:15	1/6/2016 17:19	Business	West Palm Beach	
…	…	…	…	…	
1147	12/30/2016 15:41	12/30/2016 16:03	Business	Kar?chi	
1148	12/30/2016 16:45	12/30/2016 17:08	Business	Kar?chi	
1149	12/30/2016 23:06	12/30/2016 23:10	Business	Kar?chi	
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	

	STOP*	MILES*	PURPOSE*
0	Fort Pierce	5.1	Meal/Entertain

```
1          Fort Pierce     5.0              NaN
2          Fort Pierce     4.8  Errand/Supplies
3          Fort Pierce     4.7          Meeting
5      West Palm Beach     4.3   Meal/Entertain
…                  …     …              …
1147           Kar?chi     4.6  Errand/Supplies
1148           Kar?chi     4.6          Meeting
1149           Kar?chi     0.8   Customer Visit
1150           Kar?chi     0.7          Meeting
1152  Unknown Location    16.2          Meeting

[288 rows x 7 columns]
```

10.Categorized rides as business or personal.

[166]:
```
print(df['CATEGORY*'].value_counts())
```

```
Business    1078
Personal      77
Name: CATEGORY*, dtype: int64
```

11.Determined popular start and stop points and longest routes.

[167]:
```
print('favorite starting point according the the total number of MILES covered.
 ↪\n')
df.groupby('START*').agg({'MILES*':'count'}).
 ↪sort_values('MILES*',ascending=False)
```

```
favorite starting point according the the total number of MILES covered.
```

[167]:

```
                   MILES*
START*
Cary                  201
Unknown Location      148
Morrisville            85
Whitebridge            68
Islamabad              57
…                     …
Flatiron District       1
Florence                1
Fuquay-Varina           1
Gampaha                 1
Winston Salem           1

[177 rows x 1 columns]
```

[168]:
```
print('starting point for the ride where maximum miles are covered\n\n\n')
df.groupby('START*').agg({'MILES*':'sum'}).sort_values('MILES*',ascending=False)
```

starting point for the ride where maximum miles are covered

[168]:

START*	MILES*
Unknown Location	1976.5
Cary	1791.3
Morrisville	671.7
Raleigh	433.0
Islamabad	401.2
…	…
South Berkeley	0.9
Congress Ave District	0.8
Sunnyside	0.7
Medical Centre	0.7
Soho	0.5

[177 rows x 1 columns]

[169]:
```
print('most popular START-STOP pair according to the total number of rides␣
 ↪covered\n\n\n')
df.groupby(['START*','STOP*']).count().
 ↪sort_values('START_DATE*',ascending=False)['START_DATE*'].reset_index()
```

most popular START-STOP pair according to the total number of rides covered

[169]:

	START*	STOP*	START_DATE*
0	Unknown Location	Unknown Location	86
1	Morrisville	Cary	75
2	Cary	Morrisville	67
3	Cary	Cary	53
4	Cary	Durham	36
..	…	…	…
358	Houston	Galveston	1
359	Heritage Pines	Whitebridge	1
360	Heritage Pines	Edgehill Farms	1
361	Hell's Kitchen	Midtown	1
362	Winston Salem	Asheville	1

[363 rows x 3 columns]

[170]:
```
# 27. Check the data types of all the columns in the dataset.
print('data types of all the columns in the dataset.\n')
```

```
df.info()
```

data types of all the columns in the dataset.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1156 entries, 0 to 1155
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   START_DATE*  1156 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*       1156 non-null   float64
 6   PURPOSE*     653 non-null    object
dtypes: float64(1), object(6)
memory usage: 63.3+ KB
```

7.Removed records without start and stop location.

[171]:
```
df=df[(df['STOP*']!='Unknown Location') | (df['START*']!='Unknown Location')]
df
```

[171]:

	START_DATE*	END_DATE*	CATEGORY*	START*	\
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	
…	…	…	…	…	
1150	12/31/2016 1:07	12/31/2016 1:14	Business	Kar?chi	
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	
1155	Totals	NaN	NaN	NaN	

	STOP*	MILES*	PURPOSE*
0	Fort Pierce	5.1	Meal/Entertain
1	Fort Pierce	5.0	NaN
2	Fort Pierce	4.8	Errand/Supplies
3	Fort Pierce	4.7	Meeting
4	West Palm Beach	63.7	Customer Visit
…	…	…	…
1150	Kar?chi	0.7	Meeting
1151	Unknown Location	3.9	Temporary Site
1153	Gampaha	6.4	Temporary Site
1154	Ilukwatta	48.2	Temporary Site

```
1155                NaN  12204.7                NaN

[1070 rows x 7 columns]
```

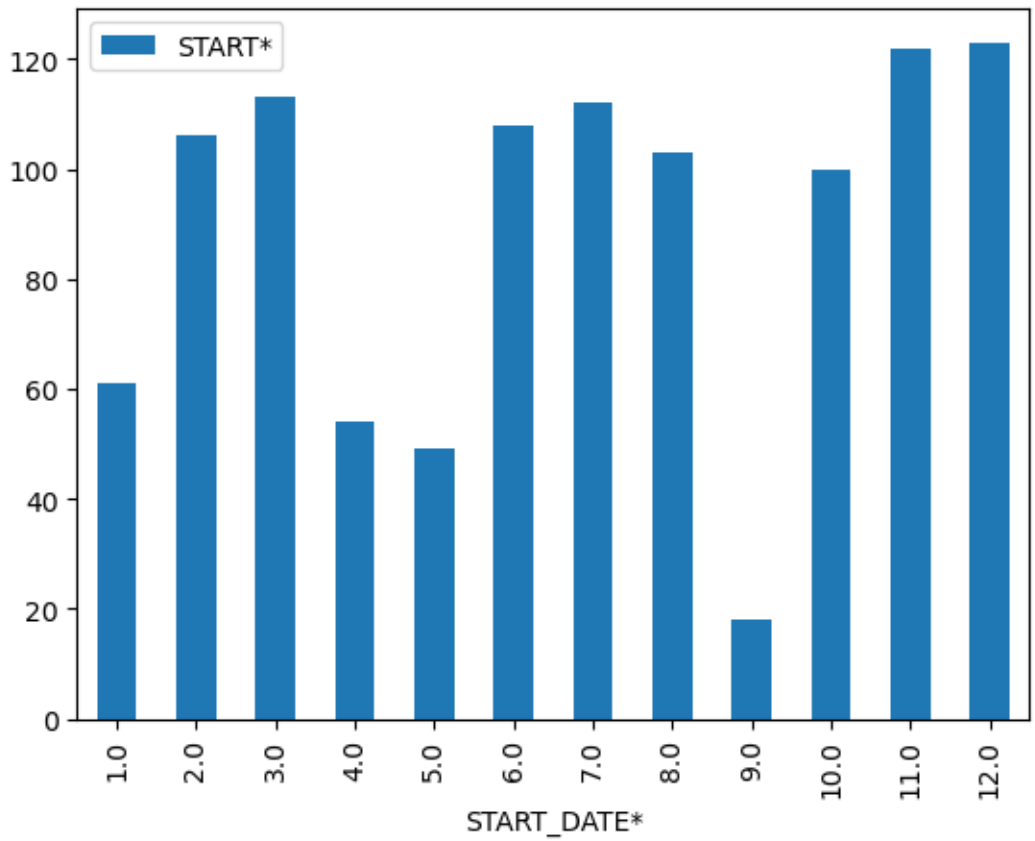
[172]:
```
print('Convert the datatypes of START_DATE and END_DATE columns to datetime.')
df['START_DATE*'] = pd.to_datetime(df['START_DATE*'],errors='coerce')
df['END_DATE*'] = pd.to_datetime(df['END_DATE*'], errors='coerce')
```

```
Convert the datatypes of START_DATE and END_DATE columns to datetime.
```

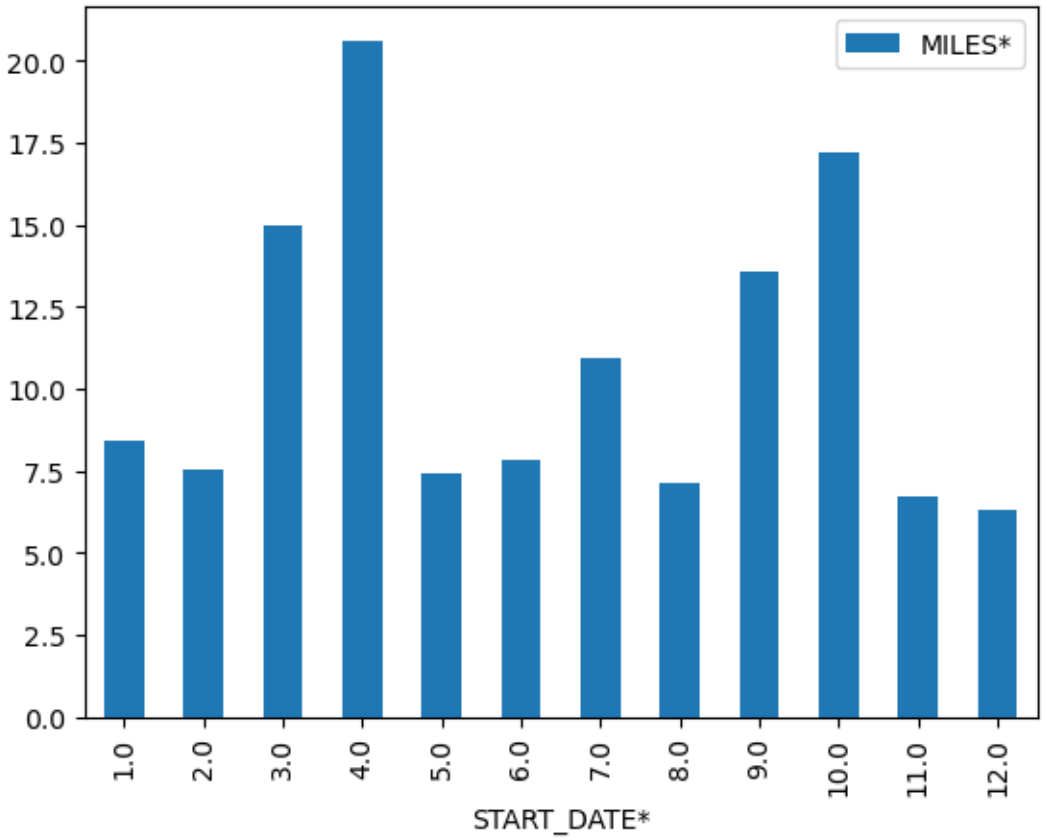
12.Analyzed monthly ride patterns and average distances.

[173]:
```
print('Extract the month from START_DATE and try to get the proportion of rides␣
 ↪of different months\n\n\n')
df.pivot_table(index=df['START_DATE*'].dt.month,aggfunc={'START*':'count'}).
 ↪plot(kind='bar')
plt.show()
```

```
Extract the month from START_DATE and try to get the proportion of rides of
different months
```



[174]:
```
df.pivot_table(index=df['START_DATE*'].dt.month,aggfunc={'MILES*':'mean'}).
  ↪plot(kind='bar')
plt.show()
```



[175]:
```
df['DAY'] = df['START_DATE*'].dt.day
print('Extract the day from the START_DATE column\n\n\n')
df
```

```
Extract the day from the START_DATE column
```

[175]:

	START_DATE*	END_DATE*	CATEGORY*	START*	\
0	2016-01-01 21:11:00	2016-01-01 21:17:00	Business	Fort Pierce	
1	2016-01-02 01:25:00	2016-01-02 01:37:00	Business	Fort Pierce	
2	2016-01-02 20:25:00	2016-01-02 20:38:00	Business	Fort Pierce	
3	2016-01-05 17:31:00	2016-01-05 17:45:00	Business	Fort Pierce	
4	2016-01-06 14:42:00	2016-01-06 15:49:00	Business	Fort Pierce	

```
…                   …                   …       …           …
1150 2016-12-31 01:07:00 2016-12-31 01:14:00  Business      Kar?chi
1151 2016-12-31 13:24:00 2016-12-31 13:42:00  Business      Kar?chi
1153 2016-12-31 21:32:00 2016-12-31 21:50:00  Business   Katunayake
1154 2016-12-31 22:08:00 2016-12-31 23:51:00  Business      Gampaha
1155                 NaT                 NaT       NaN          NaN

                  STOP*   MILES*         PURPOSE*   DAY
0          Fort Pierce      5.1   Meal/Entertain   1.0
1          Fort Pierce      5.0              NaN   2.0
2          Fort Pierce      4.8  Errand/Supplies   2.0
3          Fort Pierce      4.7          Meeting   5.0
4      West Palm Beach     63.7   Customer Visit   6.0
…                  …      …               …   …
1150           Kar?chi      0.7          Meeting  31.0
1151  Unknown Location      3.9   Temporary Site  31.0
1153           Gampaha      6.4   Temporary Site  31.0
1154         Ilukwatta     48.2   Temporary Site  31.0
1155               NaN  12204.7              NaN   NaN

[1070 rows x 8 columns]
```

10.Categorized rides as business or personal.

[176]:
```
print('the total miles covered per category per purpose.\n\n\n')
df1=df.pivot_table(index=df['CATEGORY*'],aggfunc={'MILES*':'sum'}).reset_index()
df1
```

```
the total miles covered per category per purpose.
```

[176]:

	CATEGORY*	MILES*
0	Business	10128.7
1	Personal	715.2

35. Find the percentage of Business Miles covered and Personal mIles covered.

[179]:
```
df1['MILES*']=pd.to_numeric(df1['MILES*'],errors='coerce')
df1['MILES*']=df1['MILES*'].apply(lambda x:x*100/df1['MILES*'].sum())
print('Find the percentage of Business Miles covered and Personal mIles␣
 ↪covered\n\n\n')
df1.reset_index()
```

```
Find the percentage of Business Miles covered and Personal mIles covered
```

[179]:

	index	CATEGORY*	MILES*
0	0	Business	93.404587
1	1	Personal	6.595413

2 Thank you