DataFrames With Pandas Part 2 :

Changing data types:

If we want to convert many data into date and time then we can use list.

In [4]:

```
pd.to_datetime(["12-Jun-2012","23-Nov-2000"])
```

Out[4]:

```
DatetimeIndex(['2012-06-12', '2000-11-23'], dtype='datetime64[ns]', freq=None)
```

To convert a particular column datatype:

In [5]:

```
temp = pd.DataFrame({"A":["1","2","3"],"B":[11,12,13],"C": ["12-06-2012","13-06-2015","15-06-2017"]
})
temp
```

Out[5]:

	A	B	C
0	1	11	12-06-2012
1	2	12	13-06-2015
2	3	13	15-06-2017

In [6]:

```
temp.dtypes
```

Out[6]:

```
A    object
B     int64
C    object
dtype: object
```

In [7]:

```
temp["C"] = pd.to_datetime(temp["C"])
temp["A"] = pd.to_numeric(temp["A"])
temp.dtypes
```

Out[7]:

```
A             int64
B             int64
C    datetime64[ns]
dtype: object
```

In [8]:

```
pd.to_datetime(["12-Jun-2000","abc"],errors = "coerce")
```

Out[8]:

```
DatetimeIndex(['2000-06-12', 'NaT'], dtype='datetime64[ns]', freq=None)
```

NaT - Not Any Time

In [9]:

```
temp["A"] = temp["A"].astype(str)
temp.dtypes
```

Out[9]:

```
A            object
B             int64
C    datetime64[ns]
dtype: object
```

In [1]:

```
from pandas import read_csv
```

In [2]:

```
import pandas as pd
df = pd.read_csv("Uber Drives 2016.csv")
```

In [3]:

```
df
```

Out[3]:

	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
...	...	...	...	...	...	...	...
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

1156 rows × 7 columns

To convert data into datetime format:

In [10]:

```
pd.to_datetime(df["END_DATE*"],format='%m/%d/%Y %H:%M')
```

Out[10]:

```
0       2016-01-01 21:17:00
1       2016-01-02 01:37:00
2       2016-01-02 20:38:00
3       2016-01-05 17:45:00
4       2016-01-06 15:49:00
               ...
```

```
1151   2016-12-31 13:42:00
1152   2016-12-31 15:38:00
1153   2016-12-31 21:50:00
1154   2016-12-31 23:51:00
1155                   NaT
Name: END_DATE*, Length: 1156, dtype: datetime64[ns]
```

Dataset Summarization Methods :

describe() :

It will describe a dataframe.

In [11]:

```
df.describe(include = "all")
```

Out[11]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*	PURPOSE*
count	1156	1155	1155	1155	1155	1156.000000	653
unique	1155	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	21.115398	NaN
std	NaN	NaN	NaN	NaN	NaN	359.299007	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	12204.700000	NaN

value_counts() :

It is used to count how many times a particular value is repeated in a column.

In [12]:

```
# count of unique start locations
df["START*"].value_counts()
```

Out[12]:

```
Cary                201
Unknown Location    148
Morrisville          85
Whitebridge          68
Islamabad            57
                   ...
Summerwinds           1
Ridgeland             1
Long Island City      1
Fuquay-Varina         1
Seaport               1
Name: START*, Length: 177, dtype: int64
```

In [13]:

```
df["START*"].value_counts().head()
```

Out[13]:

```
Cary                201
Unknown Location    148
Morrisville          85
Whitebridge          68
Islamabad            57
Name: START*, dtype: int64
```

Common data manipulation tasks :

There are 5 verbs of data manipulations:

- Selecting / Indexing
- Filtering
- Sorting
- Mutating / Conditionally adding columns
- Groupby / Summarize

Selecting / Indexing :

```
There are 2 methods:


    * iloc
    * loc
```

iloc:

```
Accessing row or a column using number.
```

In [14]:

```
df.head()
```

Out[14]:

	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

In [15]:

```
df.iloc[0:5,0:4]
```

Out[15]:

	START_DATE*	END_DATE*	CATEGORY*	START*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce

	START_DATE	END_DATE	CATEGORY	START
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

In [16]:

```
df.iloc[0:3,[0,3]]
```

Out[16]:

	START_DATE*	START*
0	1/1/2016 21:11	Fort Pierce
1	1/2/2016 1:25	Fort Pierce
2	1/2/2016 20:25	Fort Pierce

If we want all the columns and particular rows :

In [17]:

```
df.iloc[0:4]
```

Out[17]:

	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

If we want all the rows and particular column :

In [18]:

```
df.iloc[:,0:3]
```

Out[18]:

	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
...	...	...	...
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	Totals	NaN	NaN

1156 rows × 3 columns

If we want all the rows and all the

If we want all the columns and all the rows except last column :

In [19]:

```
data = df.iloc[:,: -1]
data
```

Out[19]:

	START_DATE*	END_DATE*	CATEGORY*	START*	STOP*	MILES*
0	1/1/2016 21:11	1/1/2016 21:17	Business	Fort Pierce	Fort Pierce	5.1
1	1/2/2016 1:25	1/2/2016 1:37	Business	Fort Pierce	Fort Pierce	5.0
2	1/2/2016 20:25	1/2/2016 20:38	Business	Fort Pierce	Fort Pierce	4.8
3	1/5/2016 17:31	1/5/2016 17:45	Business	Fort Pierce	Fort Pierce	4.7
4	1/6/2016 14:42	1/6/2016 15:49	Business	Fort Pierce	West Palm Beach	63.7
...	...	...	...	...	...	...
1151	12/31/2016 13:24	12/31/2016 13:42	Business	Kar?chi	Unknown Location	3.9
1152	12/31/2016 15:03	12/31/2016 15:38	Business	Unknown Location	Unknown Location	16.2
1153	12/31/2016 21:32	12/31/2016 21:50	Business	Katunayake	Gampaha	6.4
1154	12/31/2016 22:08	12/31/2016 23:51	Business	Gampaha	Ilukwatta	48.2
1155	Totals	NaN	NaN	NaN	NaN	12204.7

1156 rows × 6 columns

In [20]:

```
print(data.shape)
print(df.shape)
```

```
(1156, 6)
(1156, 7)
```

{It will not affect the orginal dataframe}

loc:

It is used for accessing rows and columns using name.

In [21]:

```
# label based indexing
df.loc[:,["START*","STOP*"]]
```

Out[21]:

	START*	STOP*
0	Fort Pierce	Fort Pierce
1	Fort Pierce	Fort Pierce
2	Fort Pierce	Fort Pierce
3	Fort Pierce	Fort Pierce
4	Fort Pierce	West Palm Beach
...	...	...
1151	Kar?chi	Unknown Location
1152	Unknown Location	Unknown Location

	START*	STOP*
1153	Katunayake	Gampaha
1154	Gampaha	Ilukwatta
1155	NaN	NaN

1156 rows × 2 columns

In [22]:

```
df.loc[:,["START*","STOP*"]].head()
```

Out[22]:

	START*	STOP*
0	Fort Pierce	Fort Pierce
1	Fort Pierce	Fort Pierce
2	Fort Pierce	Fort Pierce
3	Fort Pierce	Fort Pierce
4	Fort Pierce	West Palm Beach

In [23]:

```
df[["START*","STOP*"]]
```

Out[23]:

	START*	STOP*
0	Fort Pierce	Fort Pierce
1	Fort Pierce	Fort Pierce
2	Fort Pierce	Fort Pierce
3	Fort Pierce	Fort Pierce
4	Fort Pierce	West Palm Beach
...	...	...
1151	Kar?chi	Unknown Location
1152	Unknown Location	Unknown Location
1153	Katunayake	Gampaha
1154	Gampaha	Ilukwatta
1155	NaN	NaN

1156 rows × 2 columns

In [24]:

```
a = df.loc[:,"START*"]
```

In [25]:

```
type(a)
```

Out[25]:

```
pandas.core.series.Series
```

In [26]:

```
a = df.loc[:,["START*"]]
```

In [27]:

```
type(a)
```

Out[27]:

```
pandas.core.frame.DataFrame
```

In [30]:

```
a = df.loc[:,("START*")]
a
```

Out[30]:

```
0             Fort Pierce
1             Fort Pierce
2             Fort Pierce
3             Fort Pierce
4             Fort Pierce
               ...
1151             Kar?chi
1152    Unknown Location
1153          Katunayake
1154             Gampaha
1155                 NaN
Name: START*, Length: 1156, dtype: object
```

In [31]:

```
type(a)
```

Out[31]:

```
pandas.core.series.Series
```

{Brackets place an important role in datatype}