In [1]:

```
pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\dilip\anaconda3\lib\site-packages (1.3.
5)
Requirement already satisfied: numpy>=1.17.3; platform_machine != "aarch64" and platform_
machine != "arm64" and python_version < "3.10" in c:\users\dilip\anaconda3\lib\site-packa
ges (from pandas) (1.19.2)
Requirement already satisfied: pytz>=2017.3 in c:\users\dilip\anaconda3\lib\site-packages
(from pandas) (2020.1)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\dilip\anaconda3\lib\sit
e-packages (from pandas) (2.8.1)
Requirement already satisfied: six>=1.5 in c:\users\dilip\anaconda3\lib\site-packages (fr
om python-dateutil>=2.7.3->pandas) (1.15.0)
Note: you may need to restart the kernel to use updated packages.
```

Creating Pandas Series

In [1]:

```
import pandas as pd
import numpy as np

# Creating empty series
ser = pd.Series()
#print("Pandas Series: ", ser)

# simple array
data = np.array(['g', 'p', 'r', 'e', 'c'])

ser = pd.Series(data)
print("Pandas Series:\n", ser)
```

```
Pandas Series:
 0    g
1    p
2    r
3    e
4    c
dtype: object
```

```
<ipython-input-1-6e5d90e0c662>:5: DeprecationWarning: The default dtype for empty Series
will be 'object' instead of 'float64' in a future version. Specify a dtype explicitly to
silence this warning.
  ser = pd.Series()
```

In [7]:

```
import numpy as np
import pandas as pd

#info= np.array(['p','a','n','d','a','s'])
#print(info)
ser=pd.Series(info)
print(ser)
```

```
0    p
1    a
2    n
3    d
4    a
5    s
dtype: object
```

Accessing data from Series

In [10]:

```
# import pandas and numpy
import pandas as pd
import numpy as np

# creating simple array
data = np.array(['g','p','r','e','c','k', 'u','r','n','o','o','l'])
ser = pd.Series(data)
#print(ser)


#retrieve the first element
print(ser[:5])
```

```
0    g
1    p
2    r
3    e
4    c
dtype: object
```

Creating DataFrame

- **creating an empty dataframe**

In [21]:

```
# import pandas as pd
import pandas as pd

df = pd.DataFrame()

print(df)
```

```
Empty DataFrame
Columns: []
Index: []
```

In [2]:

```
# import pandas as pd
import pandas as pd

# list of strings
lst = ['Assam', 'Andhra Pradesh', 'Bhopal', 'Delhi',
            'Maharastra', 'Tamilnadu', 'Karnataka']

df = pd.DataFrame(lst)
print(df)
```

```
                0
0           Assam
1  Andhra Pradesh
2          Bhopal
3           Delhi
4      Maharastra
5       Tamilnadu
6       Karnataka
```

In [4]:

```
# DataFrame from dictonary  / lists


import pandas as pd

# initialise data of lists.
```

```
data = {'Name':['Tom', 'nick', 'krish', 'jack'], 'Age':[20, 21, 19, 18]}

# Create DataFrame
df = pd.DataFrame(data)

# Print the output.
print(df)
```

```
    Name  Age
0    Tom   20
1   nick   21
2  krish   19
3   jack   18
```

In [11]:

```
data['Age']
```

Out[11]:

```
[20, 21, 19, 18]
```

In [27]:

```
# importing pandas as pd
import pandas as pd

# dictionary of lists
dict = {'name':["aparna", "pankaj", "sudhir", "Geeku"],
        'degree': ["MBA", "BCA", "M.Tech", "MBA"],
        'score':[90, 40, 80, 98]}

df = pd.DataFrame(dict)

print(df)
```

```
     name  degree  score
0  aparna     MBA     90
1  pankaj     BCA     40
2  sudhir  M.Tech     80
3   Geeku     MBA     98
```

In [28]:

```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

#load data into a DataFrame object:
df = pd.DataFrame(data)

print(df)
```

```
   calories  duration
0       420        50
1       380        40
2       390        45
```

Giving names to index

In [12]:

```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
```

```
}

df = pd.DataFrame(data, index = ["day1", "day2", "day3"])

print(df)
```

```
      calories  duration
day1       420        50
day2       380        40
day3       390        45
```

In [33]:

```
import pandas as pd
data=pd.read_csv("cocoa.csv")
data
```

Out[33]:

	Id	Company	Bean Origin	Review Date	Cocoa Percent	Company Location	Rating
0	1	A. Morin	Agua Grande	2016.0	63%	France	3.75
1	2	A. Morin	Kpime	2015.0	70%	France	2.75
2	3	A. Morin	Atsane	2015.0	70%	France	3.00
3	4	A. Morin	Akata	2015.0	70%	France	3.50
4	5	A. Morin	Quilla	2015.0	70%	France	3.50
...	...	...	...	...	...	...	...
1790	1791	Zotter	Peru	2011.0	70%	Austria	3.75
1791	1792	Zotter	Congo	2011.0	65%	Austria	3.00
1792	1793	Zotter	Kerala State	2011.0	65%	Austria	3.50
1793	1794	Zotter	Kerala State	2011.0	62%	Austria	3.25
1794	1795	Zotter	Brazil, Mitzi Blue	2010.0	65%	Austria	3.00

1795 rows × 7 columns

In [34]:

```
data.tail()
```

Out[34]:

	Id	Company	Bean Origin	Review Date	Cocoa Percent	Company Location	Rating
1790	1791	Zotter	Peru	2011.0	70%	Austria	3.75
1791	1792	Zotter	Congo	2011.0	65%	Austria	3.00
1792	1793	Zotter	Kerala State	2011.0	65%	Austria	3.50
1793	1794	Zotter	Kerala State	2011.0	62%	Austria	3.25
1794	1795	Zotter	Brazil, Mitzi Blue	2010.0	65%	Austria	3.00

In [41]:

```
data.head(20)
```

Out[41]:

	Id	Company	Bean Origin	Review Date	Cocoa Percent	Company Location	Rating
0	1	A. Morin	Agua Grande	2016.0	63%	France	3.75
1	2	A. Morin	Kpime	2015.0	70%	France	2.75
2	3	A. Morin	Atsane	2015.0	70%	France	3.00
3	4	A. Morin	Akata	2015.0	70%	France	3.50

	Id	Company	Bean Origin	Review Date	Cocoa Percent	Company Location	Rating
4	5	A. Morin	Quila	2015.0	70%	France	3.50
5	6	A. Morin	Carenero	NaN	70%	France	2.75
6	7	A. Morin	Cuba	2014.0	70%	France	3.50
7	8	A. Morin	Sur del Lago	2014.0	70%	France	3.50
8	9	A. Morin	Puerto Cabello	2014.0	70%	France	3.75
9	10	A. Morin	Pablino	2014.0	70%	France	4.00
10	11	A. Morin	Panama	2013.0	70%	France	2.75
11	12	A. Morin	Madagascar	2013.0	70%	France	3.00
12	13	A. Morin	Brazil	2013.0	70%	France	3.25
13	14	A. Morin	Equateur	2013.0	70%	France	3.75
14	15	A. Morin	Colombie	2013.0	70%	France	2.75
15	16	A. Morin	Birmanie	2013.0	70%	France	3.00
16	17	A. Morin	Papua New Guinea	NaN	70%	France	3.25
17	18	A. Morin	Chuao	2013.0	70%	France	4.00
18	19	A. Morin	Piura	2013.0	70%	France	3.25
19	20	A. Morin	Chanchamayo Province	2013.0	70%	France	3.50

In [2]:

```
data.describe()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-2bb0b18689d4> in <module>
----> 1 data.describe()

NameError: name 'data' is not defined
```

In [43]:

```
import pandas as pd
```

In [14]:

```
df=pd.read_csv("iris.csv")
```

In [15]:

```
df
```

Out[15]:

	sepal_length	sepal_width	petal_length	petal_width	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

In [16]:

```
df.head()
```

Out[16]:

	sepal_length	sepal_width	petal_length	petal_width	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

In [17]:

```
df.tail()
```

Out[17]:

	sepal_length	sepal_width	petal_length	petal_width	target
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

In [18]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   target        150 non-null    int64  
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
```

In [19]:

```
df.describe()
```

Out[19]:

	sepal_length	sepal_width	petal_length	petal_width	target
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667	1.000000
std	0.828066	0.433594	1.764420	0.763161	0.819232
min	4.300000	2.000000	1.000000	0.100000	0.000000
25%	5.100000	2.800000	1.600000	0.300000	0.000000
50%	5.800000	3.000000	4.350000	1.300000	1.000000
75%	6.400000	3.300000	5.100000	1.800000	2.000000

	sepal_length	sepal_width	petal_length	petal_width	target
max	7.900000	4.400000	6.900000	2.500000	2.000000

In [27]:

```
df.isnull().sum()
```

Out[27]:

```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
target          0
dtype: int64
```

In [29]:

```
df.shape
```

Out[29]:

```
(150, 5)
```

In [31]:

```
df.columns
```

Out[31]:

```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'target'], dtype='ob
ject')
```

In [ ]:

```

```