Introduction to PANDAS

```
import pandas as pd
import numpy as np
PANDAS : Series data structure
s=pd.Series([3,9,-2,10,5])
s.sum()
25
s.min()
-2
s.max()
10
```

Creating a Data Frame

```
import pandas as pd
data = [['Dinesh',10],['Nithya',12],['Raji',13]]
df = pd.DataFrame(data,columns=['Name','Age'])
Name Age
0 Dinesh 10
1 Nithya 12
2 Raji 13
```

Indexed Data Frame

```
data = {'Name':['Kavitha', 'Sudha', 'Raju','Vignesh'],'Age':[28,34,29,42]}
df = pd.DataFrame(data, index=['rank1','rank2','rank3','rank4'])
Age Name
rank1 28 Kavitha
rank2 34 Sudha
rank3 29 Raju
rank4 42 Vignesh
```

Creating a DataFrame using Dictionary

```
df1=pd.DataFrame({'A':pd.Timestamp('20130102'),'B':np.array([3]*4,dtype='int32'),
'C':pd.Categorical(['Male','Female','Male','Female'])})
```

```
A         B      C
0 2013-01-02 3 Male
1 2013-01-02 3 Female
2 2013-01-02 3 Male
3 2013-01-02 3 Female
```

```
df1.shape
(4,3)
```

```
df1.dtypes
A datetime64
B int32
C category
dtype: object
```

```
df1.head()
#will display first 5 records
```

```
df1.tail()
#will display last 5 records
```

```
df.summary()
```

```
            A          B          C          D
count 6.000000  6.000000   6.000000   6.000000
mean -0.175649  -0.208576  0.019053   -0.145193
std 1.286893    1.446490   0.678454   0.722313
min -2.567050   -1.968611  -0.782803  -0.962028
25% -0.345430   -1.435541  -0.472918  -0.609707
50% 0.174139    0.179390   -0.058210  -0.306675
75% 0.447666    0.557708   0.519626   0.230624
max 1.144652    1.648411   0.912454   1.005213
```

```
df.T     # will transpose the data frame
```

```
import pandas as pd
import numpy as np
```

dates=pd.date_range('20130101', periods=100)

df = pd.DataFrame(np.random.randn(100,4), index=dates, columns=list('ABCD'))

To view first 5 records
>>> df.head()

To view last 5 records
>>> df.tail()

To view the index
>>> df.index

To view the column names
>>> df.columns

To transpose the df
>>> df.T

Sorting by Axis
>>> df.sort_index(axis=1, ascending=False)

Sorting by Values
>>> df.sort_values(by='B')

Slicing the rows
>>> df[0:3], which slice first 3 records (rows)

Slicing with index name
>>> df['20130105':'20130110']

Slicing with row and column index (like 2D Matrix)
>>> df.iloc[0], will fetch entire 1st row

>>> df.iloc[0,:2], will fetch 1st row, first 2 columns

>>> df.iloc[0,0], will fetch 1st row, 1st column element (single element)

Selecting a single column
>>> df['A'], which yields a Series

Selecting more than one column

>>> df[['A','B']], entire 2 columns

Selecting more than one column, with selected number of records
>>> df[['A','B']][:5], first 5 records

[OR]

>>> df.loc['20130101':'20130105',['A','B']][:5], first 5 records

Boolean Indexing

df[df.A>0], will fetch all positive values of A column

Include a 6th column (a categorical) character data

df['F']=['Male','Female','Female','Male','Female','Female']

Setting by assigning with a numpy array

df.loc[:,'D']=np.array([5]*len(df))

Which will replace the 'D', column with all 5

Deleting a row or column

df.drop ('col_name', axis =1, inplace=True)

will drop the column name specified in col_name

df.drop ('row_index', axis =0, inplace=True)

will drop the row specified in row_index

Concatenation of two Data Frames

Let df1 be of size 10 x 5 and df2 of size 10 x 3 if concatenated horizontally (as a new column insertion)

Df_new= pd.concat (df1, df2, axis=1)
Df_new.shape
10 x 8

Let A dataframe of size 10 x 5 and B dataframe of size 15 x 5 if concatenated Vertically (as a new row insertion)

D= pd.concat (A, B, axis=0)
D.shape

25 x 5

**** Sorting and Ordering a DataFrame:****

For the given DataFrame let us sort the age column

Let the dataframe be “A”

```
          Age     Name
rank1     28      Kavitha
rank2     34      Sudha
rank3     29      Raju
rank4     42      Vignesh

A.sort_values(by = ‘Age’)


          Age     Name
rank1     28      Kavitha
rank3     29      Sudha
rank2     34      Raju
rank4     42      Vignesh
```
