Pandas

Data Structure :

Data Structure	Dimensions	Description
Series	1	1D labeled homogeneous array, sizeimmutable.
Data frames	2	General 2D labeled, size-mutable tabular structure with potentially heterogeneously typed columns.
Panel	3	General 3D labeled, size-mutable array.

pandas.Series

A pandas Series can be created using the following constructor −

pandas.Series(data, index, dtype, copy)

Parameter	Description
Data	data takes various forms like ndarray, list, constants
Index	Index values must be unique and hashable, same length as data. Default **np.arrange(n)** if no index is passed.
Dtype	dtype is for data type. If None, data type will be inferred
Copy	Copy data. Default False

Create an Series :

Create An Empty Series:

```
import pandas as pd
s = pd.Series()
Print(s)
```

Create a Series from ndarray

If data is an ndarray, then index passed must be of the same length. If no index is passed, then by default index will be **range(n)** where **n** is array length, i.e., [0,1,2,3…. **range(len(array))-1].**

```
#import the pandas library and aliasing as pd as well as numpy
np data = np.array(['a','b','c','d'])
s = pd.Series(data)
Print(s)
```

Create a Series from dict

```
import pandas as pd
import numpy as np
data = {'a' : 0., 'b' : 1., 'c' : 2.}
s = pd.Series(data)
print (s)
```

Observe − Dictionary keys are used to construct index.

```
import pandas as pd
import numpy as np
data = {'a' : 0., 'b' : 1., 'c' : 2.}
s = pd.Series(data,index=['b','c','d','a']) print(s)
```

Create a Series from Scalar :

```
import pandas as pd
import numpy as np
s = pd.Series(5, index=[0, 1, 2, 3])
Print(s)
```

Accessing Data from Series with Position (similar to that in an **ndarray.)**

```
import pandas as pd
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
#retrieve the first element
print (s[0] )
#retrieve the first three element
print (s[:3])
#retrieve the last three element
print (s[-3:])
```

Retrieve Data Using Label (Index)

Retrieve a single element using index label value.

```
import pandas as pd
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
#retrieve a single element
print (s['a'])
```

Retrieve multiple elements using a list of index label values.

```
import pandas as pd
s = pd.Series([1,2,3,4,5],index = ['a','b','c','d','e'])
#retrieve multiple elements
print (s[['a','c','d']] )
```

If a label is not contained, an exception is raised.

```
print s['f']
```

DataFrame:

- A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns.
- pandas.DataFrame

  A pandas DataFrame can be created using the following constructor −

  SYNTAX: pandas.DataFrame(data, index, columns, dtype, copy)

Parameter	Description
Data	data takes various forms like ndarray, series, map, lists, dict, constants.
Index	For the row labels, the Index to be used for the resulting frame.
Columns	For column labels
Dtype	Data type of each column.
Copy	This command (or whatever it is) is used for copying of data, if the default is False.

Create DataFrame :

- Create an Empty DataFrame
- import pandas as pd
  df = pd.DataFrame()
  print(df)
- Create a DataFrame from Lists
- import pandas as pd
  data = [1,2,3,4,5] df = pd.DataFrame(data)
  print df
- import pandas as pd
  data = [['Alex',10],['Bob',12],['Clarke',13]]
  df = pd.DataFrame(data,columns=['Name','Age']) #dtype=float
  print df

Create a DataFrame from List of Dicts:

EXAMPLE 1:

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
df = pd.DataFrame(data)
Print(df)
```

EXAMPLE 2:

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
df = pd.DataFrame(data, index=['first', 'second'])
Print(df)
```

EXAMPLE 3:

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
df = pd.DataFrame(d)
Print(df )
```

- Column Selection

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
df = pd.DataFrame(d)
print (df ['one'])
```

- Column Addition

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']), ‘
two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
df = pd.DataFrame(d)
# Adding a new column to an existing DataFrame
print ("Adding a new column by passing as Series:")
df['three']=pd.Series([10,20,30],index=['a','b','c'])
Print(df)
print ("Adding a new column using the existing columns in DataFrame:")
df['four']=df['one']+df['three']
Print(df )
```

- Column Deletion

```
# Using the previous DataFrame, we will delete a column
# using del function
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd']),
'three' : pd.Series([10,20,30], index=['a','b','c'])}
df = pd.DataFrame(d) print ("Our dataframe is:")
Print(df)
# using del function
print("Deleting the first column using DEL function:")
del df['one']
Print(df )
# using pop function
print ("Deleting another column using POP function:")
df.pop('two')
Print(df )
```

Panel

pandas.Panel()
A Panel can be created using the following constructor −
pandas.Panel(data, items, major_axis, minor_axis, dtype, copy)

Parameter	Description
Data	Data takes various forms like ndarray, series, map
Items	axis=0
Major_axis	axis=1
Minor_axis	axis=2
Dtype	Data type of each column
Copy	Copy data. Default, **false**

Create Panel

- From 3D ndarray
- import pandas as pd
- import numpy as np
- data = np.random.rand(2,4,5)
- p = pd.Panel(data) print p
- From dict of DataFrame Objects
- import pandas as pd
- import numpy as np
- data = {'Item1' : pd.DataFrame(np.random.randn(4, 3)), 'Item2' : pd.DataFrame(np.random.randn(4, 2))}
- p = pd.Panel(data)
- print p

Selecting the Data from Panel

- Using Items
- import pandas as pd
- import numpy as np
- data = {'Item1' : pd.DataFrame(np.random.randn(4, 3)), 'Item2' : pd.DataFrame(np.random.randn(4, 2))}
- p = pd.Panel(data)
- print p['Item1']
- Using major_axis
- data = {'Item1' : pd.DataFrame(np.random.randn(4, 3)), 'Item2' : pd.DataFrame(np.random.randn(4, 2))}
- p = pd.Panel(data)
- print p.major_xs(1)

Using minor_axis :

```
import pandas as pd
import numpy as np
data = {'Item1' : pd.DataFrame(np.random.randn(4, 3)),
'Item2' : pd.DataFrame(np.random.randn(4, 2))}
p = pd.Panel(data)
Print(p.minor_xs(1))
```

Series Basic Functionality :

Method	Description
Axes	Returns a list of the row axis labels
Dtype	Returns the dtype of the object.
Empty	Returns True if series is empty.
Ndim	Returns the number of dimensions of the underlying data, by definition 1.
Size	Returns the number of elements in the underlying data.
Value	Returns the Series as ndarray.
Head()	Returns the first n rows.

Axes :

- Returns the list of the labels of the series.
- import pandas as pd
- import numpy as np
- #Create a series with 100 random numbers
- s = pd.Series(np.random.randn(4))
- print("The axes are:")
- Print(s.axes）

Empty :

Returns the Boolean value saying whether the Object is empty or not.
True indicates that the object is empty.

```
import pandas as pd
import numpy as np
 #Create a series with 100 random numbers
s = pd.Series(np.random.randn(4))
print ("Is the Object empty?")
Print(s.empty)
```

Ndim :

- Returns the number of dimensions of the object. By definition, a Series is a 1D data structure, so it returns

```
import pandas as pd
import numpy as np
#Create a series with 4 random numbers
s = pd.Series(np.random.randn(4))
print(s)
print ("The dimensions of the object:")
print (s.ndim)
```

Size :

- Returns the size(length) of the series.
- import pandas as pd
- import numpy as np
- #Create a series with 4 random numbers
- s = pd.Series(np.random.randn(2))
- Print(s)
- print("The size of the object:")
- Print(s.size)

Values :

- Returns the actual data in the series as an array.
- import pandas as pd
- import numpy as np
- #Create a series with 4 random numbers
- s = pd.Series(np.random.randn(4))
- Print(s)
- print("The actual data series is:")
- Print(s.values)

Head & Tail:

HEAD: returns the first **n** rows(observe the index values)

```
import pandas as pd
import numpy as np
#Create a series with 4 random numbers
s = pd.Series(np.random.randn(4))
print ("The original series is:")
Print(s)
print ("The first two rows of the data series:")
Print(s.head(2))
```

TAIL:**tail()** returns the last **n** rows(observe the index values)

```
print s.tail(2)
```

DataFrame Basic Functionality:

Method	Parameter
T	Transposes rows and columns.
axes	Returns a list with the row axis labels and column axis labels as the only members.
dtypes	Returns the dtypes in this object.
empty	True if NDFrame is entirely empty [no items]; if any of the axes are of length 0.
ndim	Number of axes / array dimensions.
shape	Returns a tuple representing the dimensionality of the DataFrame.
size	Number of elements in the NDFrame.
values	Numpy representation of NDFrame.
head()	Returns the first n rows.
tail()	Returns last n rows.

Transpose :

```
import pandas as pd
import numpy as np
# Create a Dictionary of series
d= {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
'Age':pd.Series([25,26,25,23,30,29,23]),
'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
# Create a DataFrame
df = pd.DataFrame(d)
print ("The transpose of the data series is:")
print df.T
```

Axes :

```
import pandas as pd
import numpy as np
#Create a Dictionary of series
d={'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
'Age':pd.Series([25,26,25,23,30,29,23]),
'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
#Create a DataFrame
df = pd.DataFrame(d)
print("Row axis labels and column axis labels are:")
Print(df.axes)
```

Dtypes :

```
import pandas as pd
import numpy as np
#Create a Dictionary of series
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
'Age':pd.Series([25,26,25,23,30,29,23]),
'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
#Create a DataFrame
df = pd.DataFrame(d)
print ("The data types of each column are:")
Print(df.dtypes )
```

Shape :

```
import pandas as pd
import numpy as np
#Create a Dictionary of series
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
'Age':pd.Series([25,26,25,23,30,29,23]),
'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
#Create a DataFrame
df = pd.DataFrame(d)
print ("Our object is:")
Print(df)
print ("The shape of the object is:")
Print(df.shape)
```

Size :

```
import pandas as pd
import numpy as np
#Create a Dictionary of series
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
'Age':pd.Series([25,26,25,23,30,29,23]),
'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
#Create a DataFrame
df = pd.DataFrame(d)
print ("Our object is:")
Print(df)
print ("The total number of elements in our object is:")
Print(df.size)
```

Values :

```
import pandas as pd
import numpy as np
#Create a Dictionary of series
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
'Age':pd.Series([25,26,25,23,30,29,23]),
'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
#Create a DataFrame
df = pd.DataFrame(d)
print ("Our object is:")
Print(df)
print ("The actual data in our data frame is:")
Print(df.values)
```

Head And Tail :

```
import pandas as pd
import numpy as np
#Create a Dictionary of series
d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack']),
'Age':pd.Series([25,26,25,23,30,29,23]),
'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8])}
#Create a DataFrame
df = pd.DataFrame(d)
print ("Our data frame is:")
print (df)
print ("The first two rows of the data frame is:")
Print(df.head(2))
print df.tail(2)
```