Pandas- Series

INTERNSHIPSTUDIO

- Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects).The axis labels are collectively called index.

A pandas Series can be created using the following constructor –

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

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

Creating Series

- ndArray
- Dict
- Scalar value or constant

Create a Series from ndarray

- If data is an ndarray, then index passed must be of the same length.
- Creating a simple series

```
data = np.array(['a','b','c','d'])
s = pd.Series(data)
print s
```

Its **output** is as follows –

```
0   a
1   b
2   c
3   d
dtype: object
```

- We did not pass any index, so by default, it assigned the indexes ranging from 0 to len(data)-1, i.e., 0 to 3.

Example 2

Lets pass the index values here. Now we can see the customized indexed values in the output.

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

s = pd.Series(data,index=[100,101,102,103])
print s
```

Its **output** is as follows –

```
100  a
101  b
102  c
103  d
dtype: object
```

Create a Series from dict

- A dictionery can be passed as input and if no index is specified, then the dictionary keys are taken in a sorted order to construct index. If index is passed, the values in data corresponding to the labels in the index will be pulled out.

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

Its **output** is as follows –

```
a 0.0
b 1.0
c 2.0
dtype: float64
```

Observe – Dictionary keys are used to construct index.

Example 2

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

Its **output** is as follows –

```
b 1.0
c 2.0
d NaN
a 0.0
dtype: float64
```

Observe – Index order is persisted and the missing element is filled with NaN (Not a Number).

Create a Series from Scalar

- If data is a scalar value, an index must be provided. The value will be repeated to match the length of index.

```
s = pd.Series(5, index=[0, 1, 2, 3])
 print s
```

Its **output** is as follows –

```
0  5
1  5
2  5
3  5
dtype: int64
```

INTERNSHIPSTUDIO

Accessing Data from Series with Position

- Data in the series can be accessed similar to that in an **ndarray.**

Example 1

- Retrieve the first element. As we already know, the counting starts from zero for the array, which means the first element is stored at zeroth position and so on.

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

Its **output** is as follows – 1

Retrieve Data Using Label (Index)

- A Series is like a fixed-size **dict** in that you can get and set values by index label.

- Retrieve multiple elements using a list of index label values.

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

Its **output** is as follows −

a 1

c 3

d 4

dtype: int64

Pandas

INTERNSHIPSTUDIO

- How to create series from Numpy array?
- How to create series using dict?
- How to create series using scaler?
- How to Accessing Data from Series with Position?
- How to Retrieve Data Using Label ?