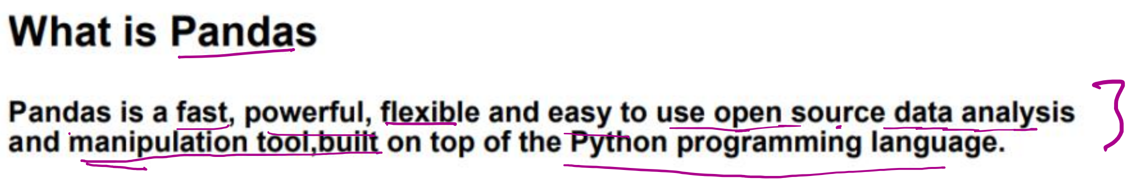
Session-8 Pandas

21 July 2024 01:00 PM

By Alok Ranjan
Linkedin- www.linkedin.com/in/alok-ranjandigu-coder
GitHub- https://github.com/alokyadav2020



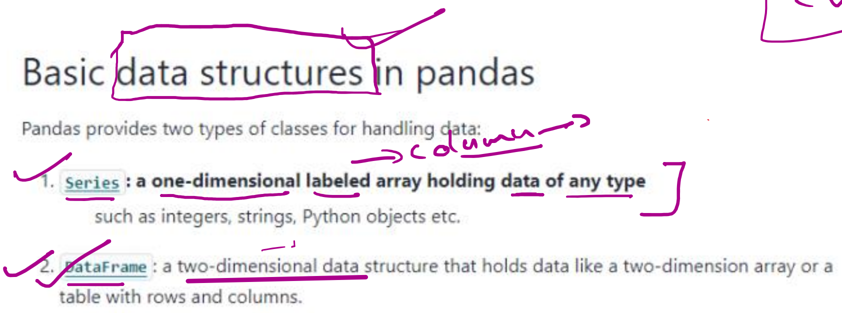
What is Pandas

Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool,built on top of the Python programming language.

CVS →

column

row

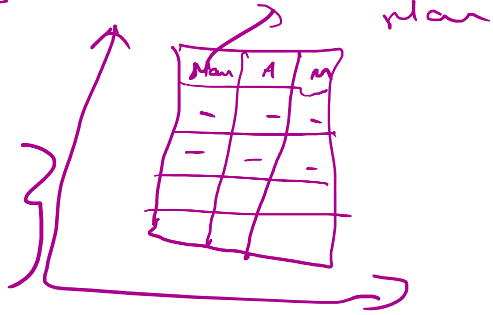


Basic data structures in pandas

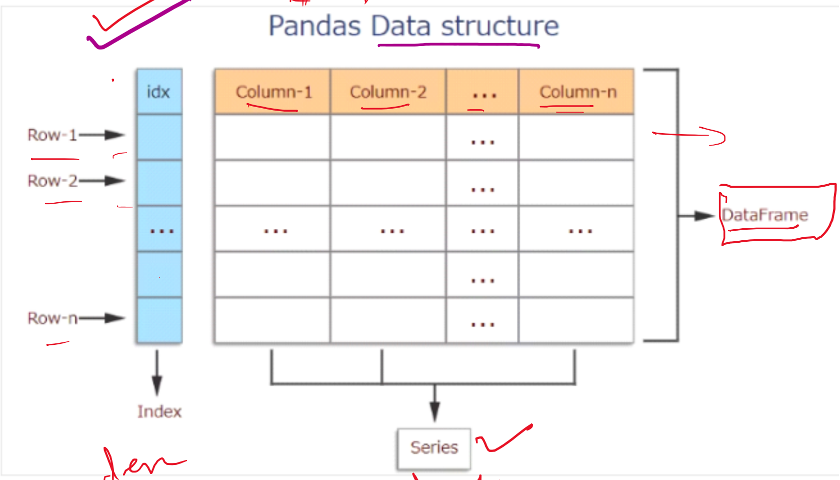
Pandas provides two types of classes for handling data:

→ column →

1. `Series` : **a one-dimensional labeled array holding data of any type** such as integers, strings, Python objects etc.
2. `DataFrame` : a two-dimensional data structure that holds data like a two-dimension array or a table with rows and columns.

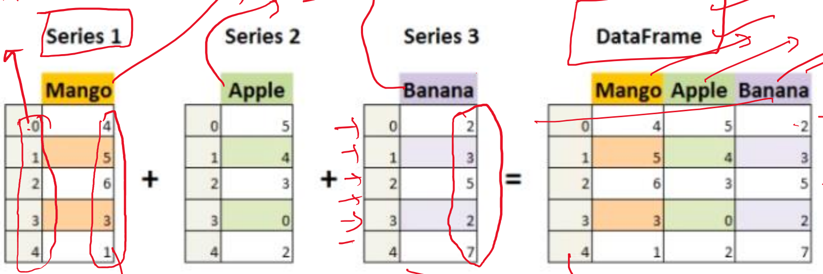


l = [1, 2, 3, 4]
0 2 2 3



series index

series Name



	Mango
0	4
1	5
2	6
3	3
4	1

	Apple
0	5
1	4
2	3
3	0
4	2

	Banana
0	2
1	3
2	5
3	2
4	7

	Mango	Apple	Banana
0	4	5	2
1	5	4	3
2	6	3	5
3	3	0	2
4	1	2	7

index

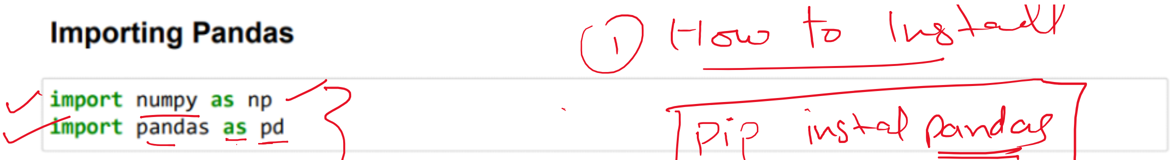
seriesvalue

③ (i) Name
(ii) Index
(iii) Value

Importing Pandas

① How to Install

```
import numpy as np
```



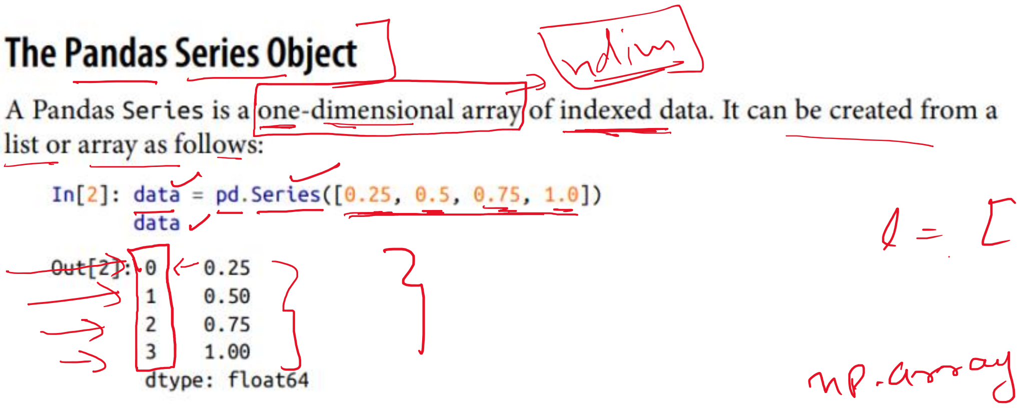
Importing Pandas

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

① How to Install

pip install pandas

pip3 install pandas



① The Pandas Series Object

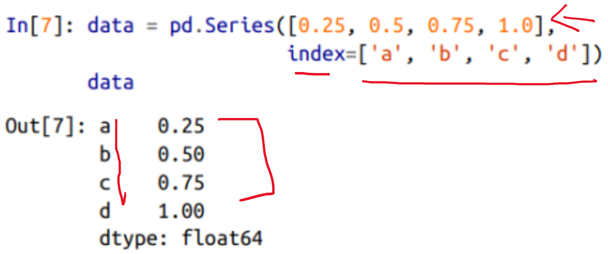
ndim

A Pandas Series is a one-dimensional array of indexed data. It can be created from a list or array as follows:

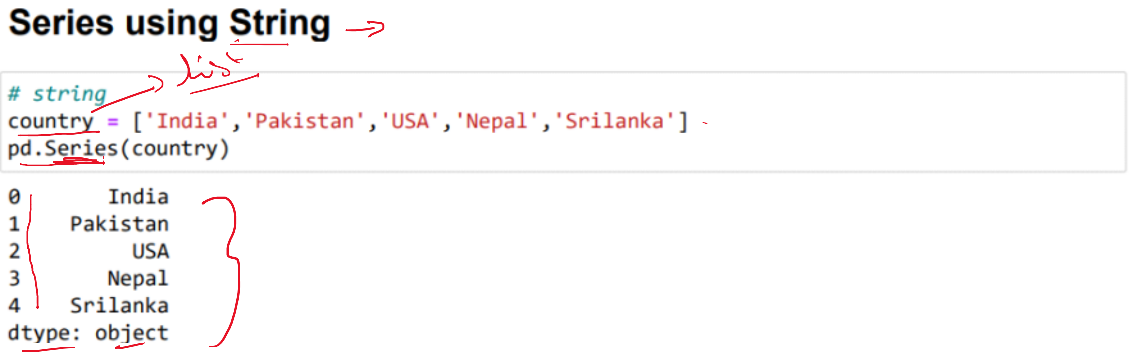
```
In[2]: data = pd.Series([0.25, 0.5, 0.75, 1.0])
       data

Out[2]: 0    0.25
        1    0.50
        2    0.75
        3    1.00
        dtype: float64
```

l = [1, 2, 3, 4]

np.array([1,2,3,4])



```
In[7]: data = pd.Series([0.25, 0.5, 0.75, 1.0],
                        index=['a', 'b', 'c', 'd'])
       data

Out[7]: a    0.25
        b    0.50
        c    0.75
        d    1.00
        dtype: float64
```

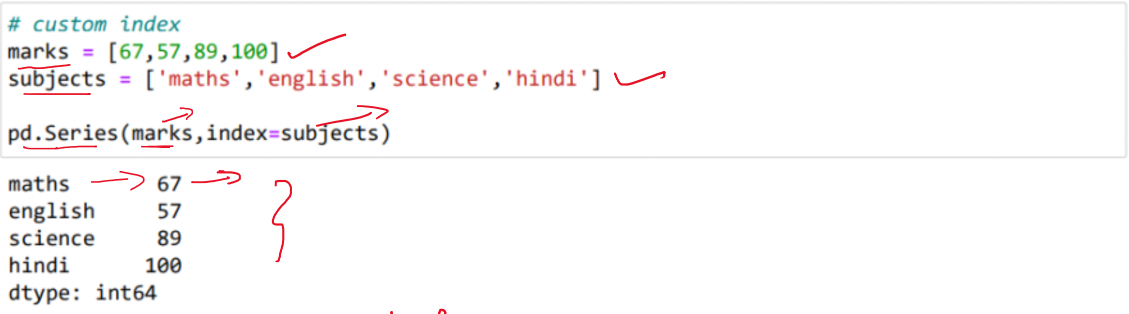


Series using String

list

```
# string
country = ['India','Pakistan','USA','Nepal','Srilanka']
pd.Series(country)
```

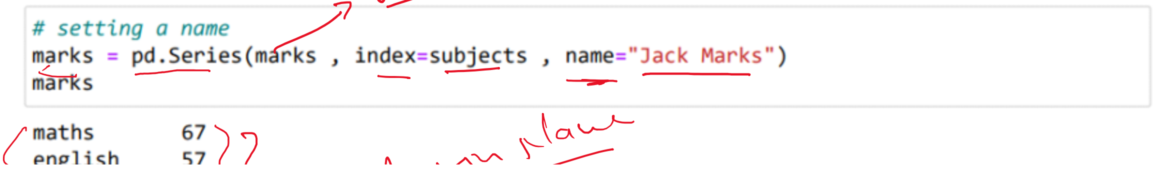
```
0       India
1    Pakistan
2         USA
3       Nepal
4    Srilanka
dtype: object
```



```
# custom index
marks = [67,57,89,100]
subjects = ['maths','english','science','hindi']

pd.Series(marks,index=subjects)
```

```
maths       67
english     57
science     89
hindi      100
dtype: int64
```

value



```
# setting a name
marks = pd.Series(marks , index=subjects , name="Jack Marks")
marks
```

```
maths       67
english     57
```

name

```
marks
```

```
maths        67
english      57
science      89
hindi       100
Name: Jack Marks, dtype: int64
```

column name

Series from dictionary

When a Pandas Series is converted to a dictionary using the to_dict() method, the resulting dictionary has the same keys and values as the Series. The index values of the Series become the keys in the dictionary, and the corresponding values become the values in the dictionary.

key value

```
marks = {
    'maths':67,
    'english':57,
    'science':89,
    'hindi':100
}
marks_series = pd.Series(marks,name="jack Marks")
```

index

```
marks_series
```

```
maths        67
english      57
science      89
hindi       100
Name: jack Marks, dtype: int64
```

Series Attributes

size: Returns the number of elements in the Series.

```
# size
marks_series.size
```

```
4
```

```
# dtype
marks_series.dtype
```

```
dtype('int64')
```

```
# name
marks_series.name
```

```
'jack Marks'
```

unique is an attribute of a Pandas Series that returns an array of the unique values in the Series.

```
# is_unique
marks_series.is_unique
```

```
True
```

```
pd.Series([1,1,2,3,4,44,2]).is_unique #It gives false because of repetation
```

```
False
```

index: Returns the index labels of the Series.

```
# index
marks_series.index
```

```
Index(['maths', 'english', 'science', 'hindi'], dtype='object')
```

values: Returns the data contained in the Series as a NumPy array.

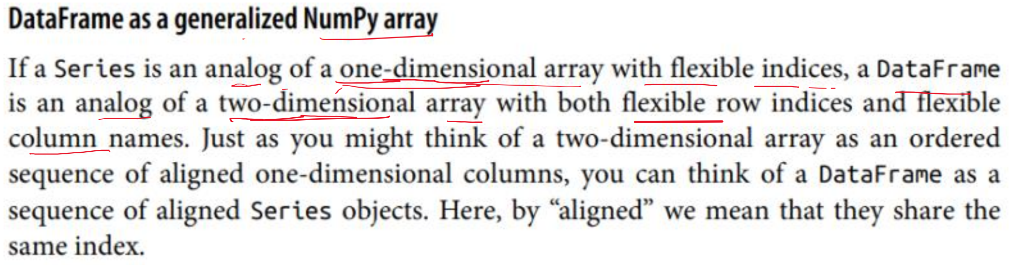
```
# values
marks_series.values
```

```
array([ 67,  57,  89, 100], dtype=int64)
```

```
type(marks_series.values)
```

```
numpy.ndarray
```

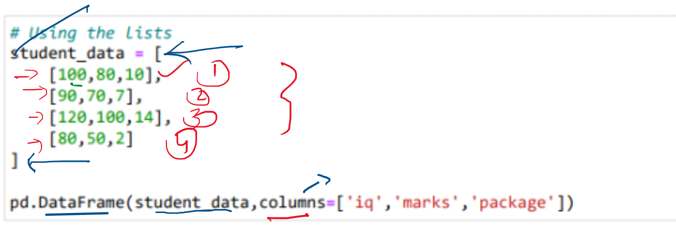
DataFrame



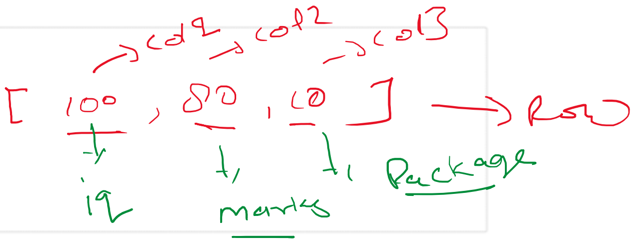
DataFrame as a generalized NumPy array

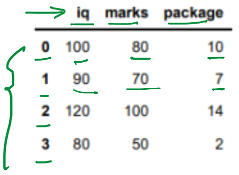
If a Series is an analog of a one-dimensional array with flexible indices, a DataFrame is an analog of a two-dimensional array with both flexible row indices and flexible column names. Just as you might think of a two-dimensional array as an ordered sequence of aligned one-dimensional columns, you can think of a DataFrame as a sequence of aligned Series objects. Here, by "aligned" we mean that they share the same index.

Creating DataFrame

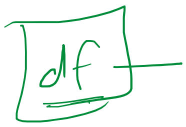


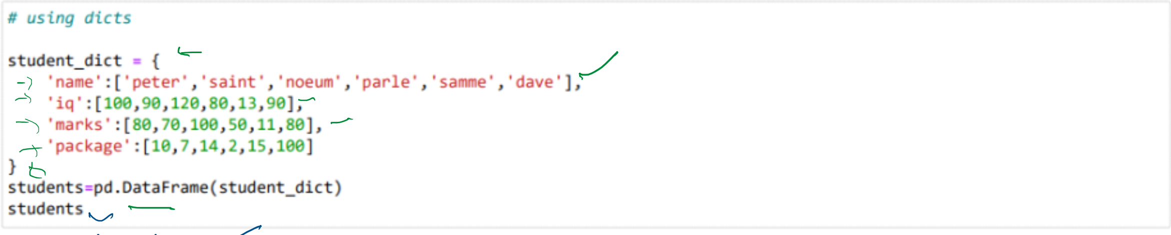
```
# Using the lists
student_data = [
    [100,80,10],
    [90,70,7],
    [120,100,14],
    [80,50,2]
]

pd.DataFrame(student_data,columns=['iq','marks','package'])
```



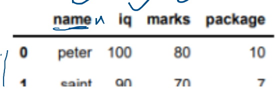


	iq	marks	package
0	100	80	10
1	90	70	7
2	120	100	14
3	80	50	2

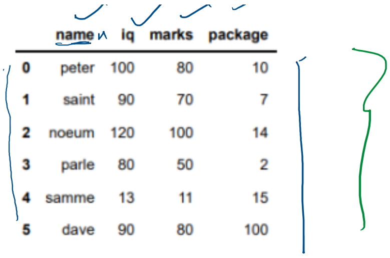




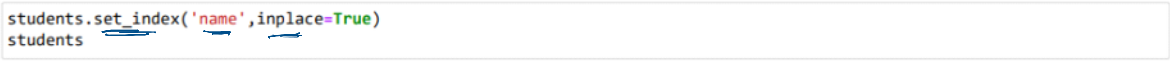
```
# using dicts

student_dict = {
    'name':['peter','saint','noeum','parle','samme','dave'],
    'iq':[100,90,120,80,13,90],
    'marks':[80,70,100,50,11,80],
    'package':[10,7,14,2,15,100]
}
students=pd.DataFrame(student_dict)
students
```



	name	iq	marks	package
0	peter	100	80	10
1	saint	90	70	7

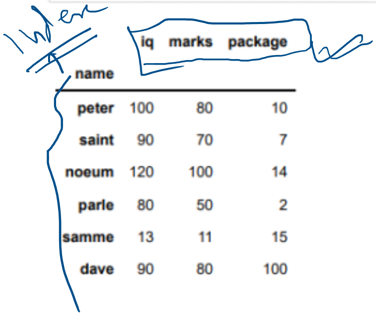


	name	iq	marks	package
0	peter	100	80	10
1	saint	90	70	7
2	noeum	120	100	14
3	parle	80	50	2
4	samme	13	11	15
5	dave	90	80	100

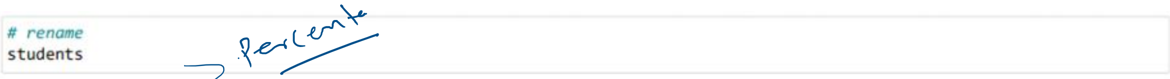


```
students.set_index('name',inplace=True)
students
```

index



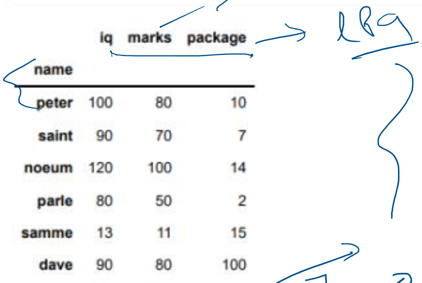
	iq	marks	package
name			
peter	100	80	10
saint	90	70	7
noeum	120	100	14
parle	80	50	2
samme	13	11	15
dave	90	80	100



```
# rename
students
```

Percente

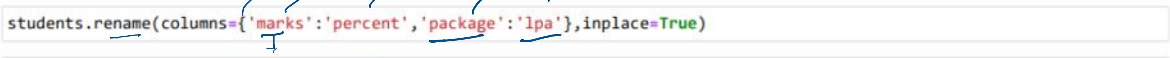
lpa



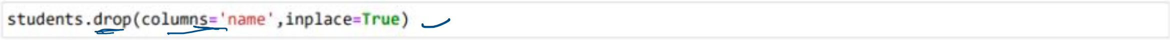
	iq	marks	package
name			
peter	100	80	10
saint	90	70	7
noeum	120	100	14
parle	80	50	2
samme	13	11	15
dave	90	80	100

key

value

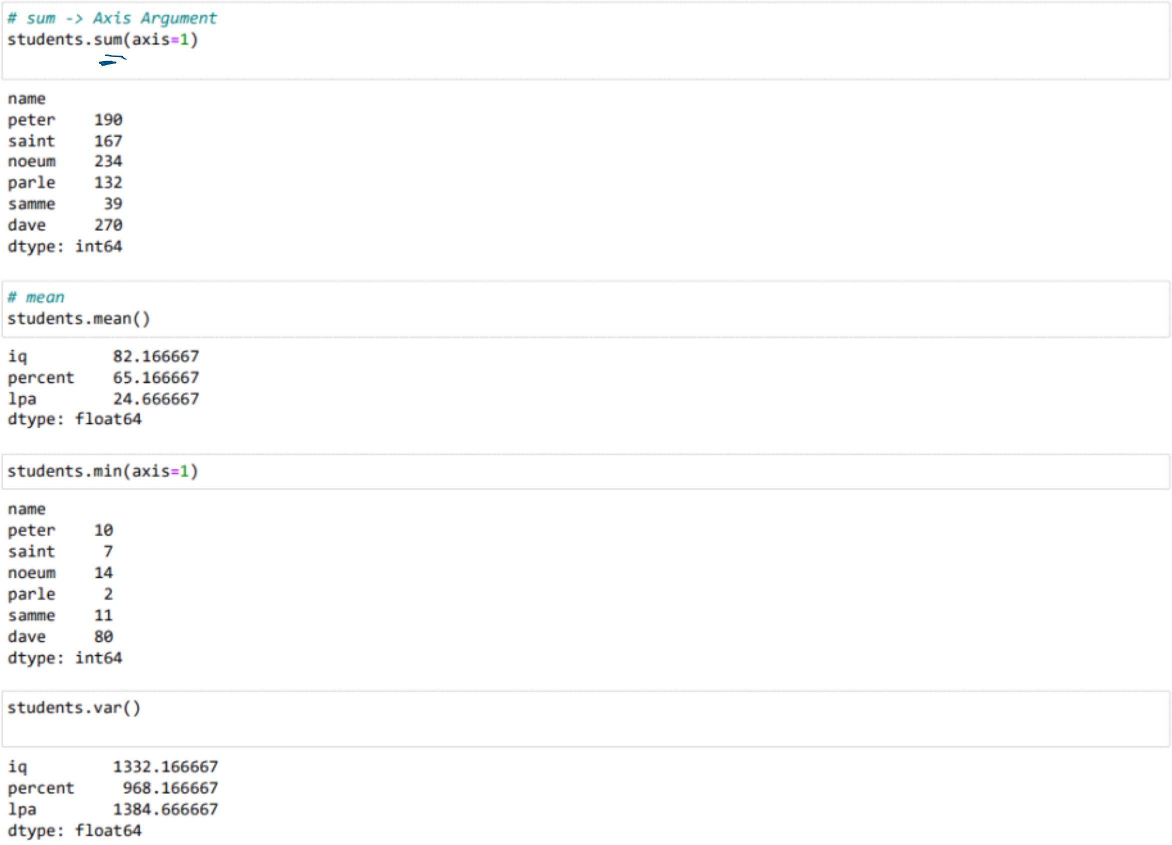


```
students.rename(columns={'marks':'percent','package':'lpa'},inplace=True)
```



```
students.drop(columns='name',inplace=True)
```

Maths Method



```
# sum -> Axis Argument
students.sum(axis=1)
```

```
name
peter    190
saint    167
noeum    234
parle    132
samme     39
dave     270
dtype: int64
```

```
# mean
students.mean()
```

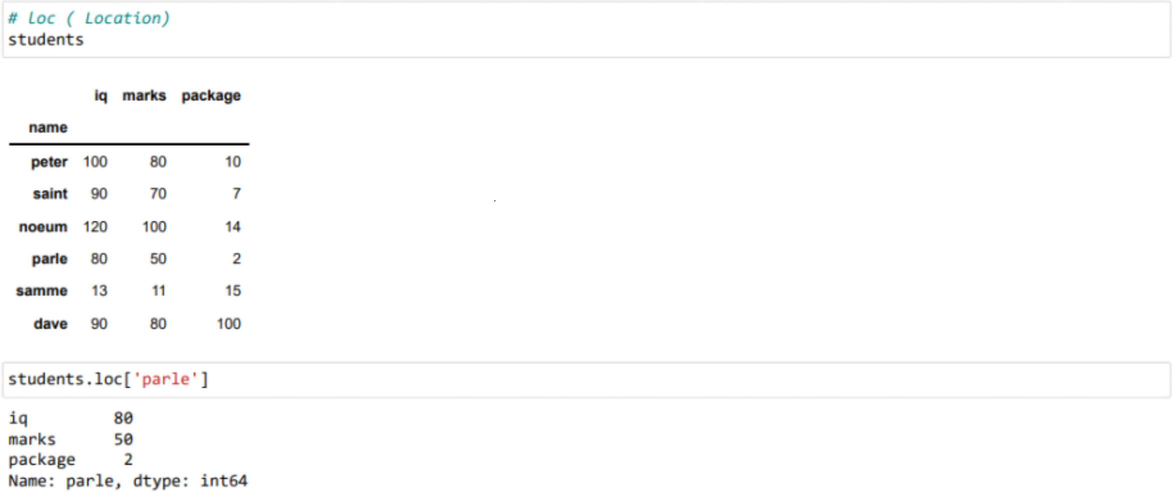
```
iq         82.166667
percent    65.166667
lpa        24.666667
dtype: float64
```

```
students.min(axis=1)
```

```
name
peter    10
saint     7
noeum    14
parle     2
samme    11
dave     80
dtype: int64
```

```
students.var()
```

```
iq         1332.166667
percent     968.166667
lpa        1384.666667
dtype: float64
```

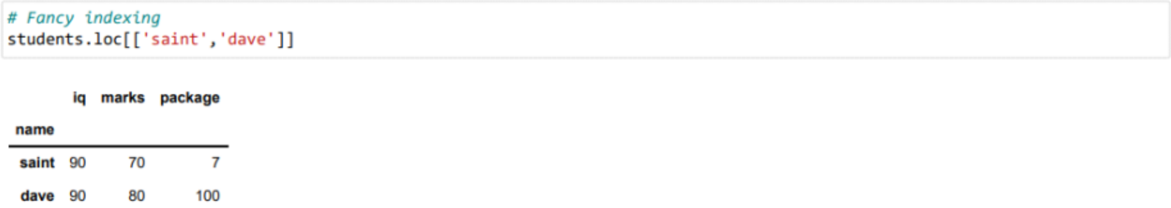


```
# Loc ( Location)
students
```

	iq	marks	package
name			
peter	100	80	10
saint	90	70	7
noeum	120	100	14
parle	80	50	2
samme	13	11	15
dave	90	80	100

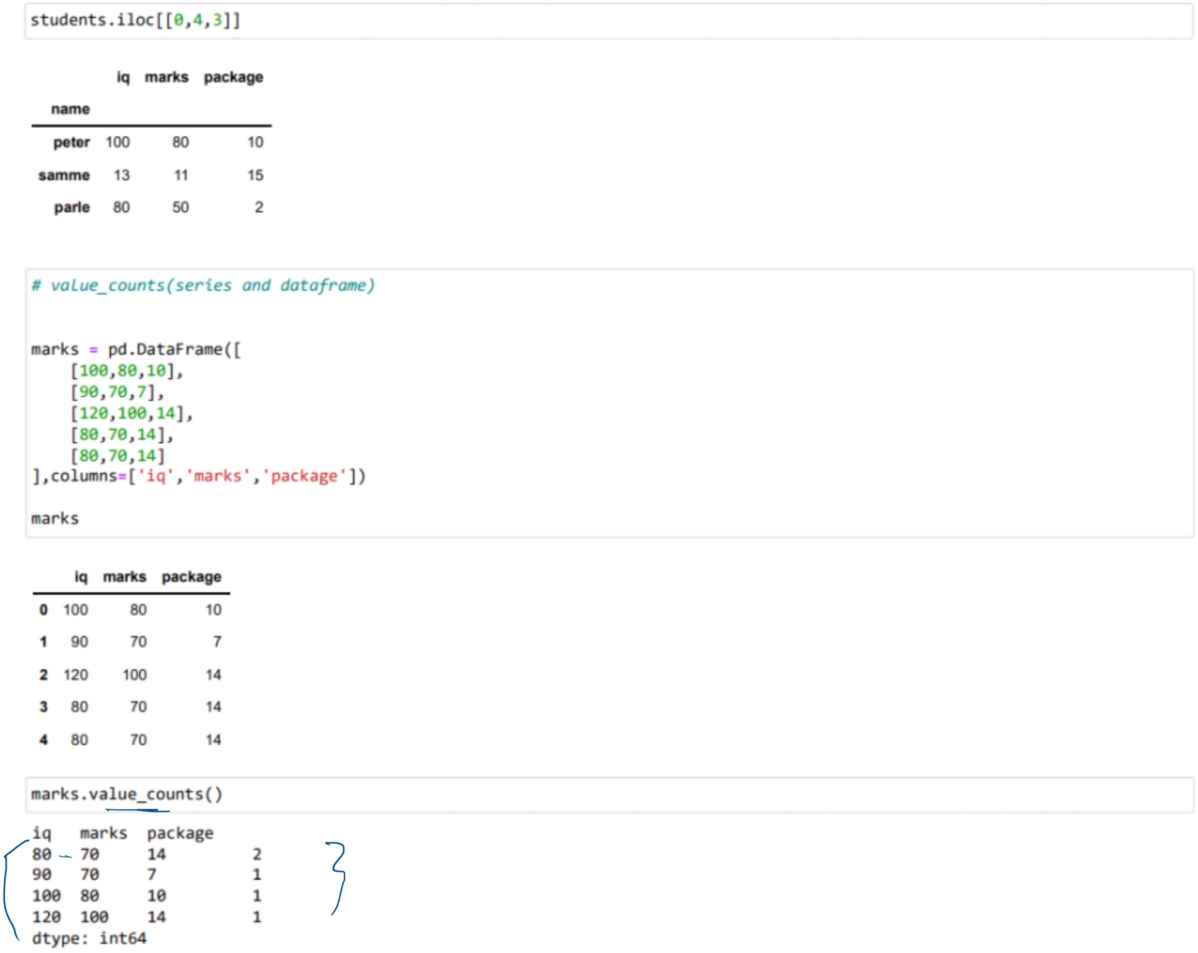
```
students.loc['parle']
```

```
iq         80
marks      50
package     2
Name: parle, dtype: int64
```



```
# Fancy indexing
students.loc[['saint','dave']]
```

	iq	marks	package
name			
saint	90	70	7
dave	90	80	100



```
students.iloc[[0,4,3]]
```

	iq	marks	package
name			
peter	100	80	10
samme	13	11	15
parle	80	50	2

```
# value_counts(series and dataframe)


marks = pd.DataFrame([
    [100,80,10],
    [90,70,7],
    [120,100,14],
    [80,70,14],
    [80,70,14]
],columns=['iq','marks','package'])

marks
```

	iq	marks	package
0	100	80	10
1	90	70	7
2	120	100	14
3	80	70	14
4	80	70	14

```
marks.value_counts()
```

```
iq   marks  package
80   70     14         2
90   70     7          1
100  80     10         1
120  100    14         1
dtype: int64
```