Pandas

Date: .....
Page: .....

Usage:

1. dict1 = { "name" : ['Ra', 'sha', 'Ga', 'Na'],
"marks" : [10, 20, 30, 40],
"City" : ['Rampur', 'cizer', 'Noway', 'Hetuda']
}

- We created a dictionary. Jupyter Notbook adds index itself.

2. df = pd.DataFrame(dict1)
df

- Turns into excel sheet

	Name	marks	City
0	Ra	10	Rampu
1	sha	20	cizer
2	Ga	30	Norway
3	Na	40	Hetuda

- pd.Dataframe: pd is typically the alias used for importing pandas library. Dataframe class is one of the fundamental data structure. used to store & manipulate 2-D labeled data.

- You can create dataframes from various data structure like lists, dictionaries, or Numpy arrays.

3. df.to_csv('std-record')

It creates csv file named 'std-record'

4. df.to_csv('std-record', index=False)
If you dont wannt index.

Date: .....

Page: .....

5. df = Pd.DataFrame read_csv ('...')

6. df.head() || first five

7. df.tail() || last five

8. df.describe()

- It computes the numerical data from the data frame.

	marks	
count	4	
mean	25	
std		
min	10	min score
25%		
50%		
75%		
max	40	max score

9. df['marks'] ⇒

	marks
0	10
1	20
2	30
3	40

- selecting a whole colmn

10. df['marks'][0] ⇒ 10

- select data from marks column index 0

11. df['marks'][1] = 30

edit value of index 1 to 30.

Date : .....
Page : .....

11. If you have to change index:

df.index = ['first', 'second', 'third', 'fourth']

Pandas has 2 types of data structures.

(i) Series - one dimensional array with indexes, stores single column or row of data in a Dataframe.

(ii) Dataframe - It is a tabular spreadsheet like structure representing rows each of which contains one or more columns.

Series - capable of holding any type of data.

Dataframe - potentially different data types

12. Ser = Pd.~~Dataframe~~ Series(np.random.rand(5), index = np.arange(5))

0	0.12359
1	0.23503
2	0.21571
3	0.53124
4	0.72596

- type(ser)
  ↳ Pandas.core.~~frame~~ series.~~Dataframe~~ Series
- Ser.describe()
  ↳ gives count, mean, std, min, max

13. new-df = Pd.Dataframe(np.random.rand(334,5), index = np.arange(334))

type(new-df) ⇒ pandas.core.frame.Dataframe

Date:
Page:

neo_df.describe()

14. neo_df.head()

	0	1	2	3	4
0	0.253	0.135	0.786	0.882	0.439
1	0.123	0.953	0.329	0.281	0.605
2	0.301	0.182	0.300	0.791	0.147
3	0.312	0.001	0.132	0.879	0.010
4	0.111	0.117	0.771	0.924	0.495

15. neo_df.index

RangeIndex([0, 1, 2 ... 300 .. 334])

16. neo_df.columns

RangeIndex(start=0, stop=5, step=1)

To convert floating point into numpy array.

17. neo_df.to_numpy()

```
array([[0.253, 0.135, 0.786, 0.882, 0.439
       [
       :
       334
       ])
```

Date :
Page :

18. new-df.T → rows & columns interchange

0 1 2 3 4 5 .. 334 columns

1

2

3

4

Rows

19. new_df.sort_index(axis=0, ascending=False)

Sorting according to index.

axis=0 ; it represent rows

by default ; ascending=True.

0 1 2 3 4

334 ..

333

:

:

20. If axis=1 ; it sorts columns.

4 3 2 1 0

0

1

2

:

21. type(new-df[0]) pandas.core.series.series

comb^n of series forms a dataframe.

Date:
Page:

>> new_df2 = new_df

>> new_df2[0][0] = 0.8783

>> new_df - it gets updated, but we had edited new_df2

so, new_df2 is a view of new_df. like a pointer, when you change view, original data gets updated

>> new_df3 = new_df.copy()

>> new_df3[0][0] = 0.329

col/m index

>> new df

newdf remains the same.

But, we shouldnot use chained indexing like df[0][0] = 2 in pandas because pandas might return a copy of the data instead of view of the data.

to ensure, modifications are made to the original Dataframe and avoid potential issues, it is recommended to use the '.loc' or '.iloc' ~~asse~~ accessors for setting values.

axis = 0 → row
axis = 1 → column

Page: .....

using loc/iloc instead of chained indexing

```
df.loc[0,0] = 2
df.iloc[0,0] = 2
```

```
new_df.loc[0,0] = 654
```

row column

To drop a column:

column name, column

```
new_df = new_df.drop(0, axis 1)
```

```
new_df.loc[[1,2], :]
```

row 1 & row 2 and all column

» `new_df.iloc[0,4]` || '0' row & 4th column value.

» `new_df.drop(['A', 'C'], axis=1)` || drop whole column of A & C.

» `new_df.drop(['B'], axis=1, inplace=True)` || change in orignal fom