create Pandas Series

In [1]:
```
import pandas as pd
```

In [2]:
```
a = pd.Series([1,2,3,4,5])
```

In [3]:
```
a
```

Out[3]:
```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

In [4]:
```
type(a)
```

Out[4]:
```
pandas.core.series.Series
```

In [5]:
```
a[2]
```

Out[5]:
```
3
```

In [6]:
```
a = pd.Series(['a','b','c'])
```

In [7]:
```
a
```

Out[7]:
```
0    a
1    b
2    c
dtype: object
```

In [8]:
```
a = pd.date_range(start = '01-01-2018', end = '23-5-2018')
```

In [9]:
```
a
```

Out[9]:
```
DatetimeIndex(['2018-01-01', '2018-01-02', '2018-01-03', '2018-01-04',
               '2018-01-05', '2018-01-06', '2018-01-07', '2018-01-08',
               '2018-01-09', '2018-01-10',
               ...
               '2018-05-14', '2018-05-15', '2018-05-16', '2018-05-17',
               '2018-05-18', '2018-05-19', '2018-05-20', '2018-05-21',
               '2018-05-22', '2018-05-23'],
              dtype='datetime64[ns]', length=143, freq='D')
```

In [10]:
```
type(a)
```

Out[10]:
```
pandas.core.indexes.datetimes.DatetimeIndex
```

Pandas dataframe

In [11]:
```
import numpy as np
```

In [13]:
```
temp = np.random.randint(low = 20, high =100, size = [20,])
name = np.random.choice(['Abhay','Teclov','Geekshub','Ankit'],20)
random = np.random.choice([10,11,13,12,14],20)
```

In [14]:
```
a = list(zip(temp, name, random))
```

In [15]:
```
df = pd.DataFrame(data = a, columns=['temp','name','random'])
```

In [16]:
```
df
```

Out[16]:

	temp	name	random
0	84	Teclov	10
1	86	Teclov	10
2	72	Geekshub	13
3	75	Teclov	13
4	25	Teclov	13
5	34	Abhay	10
6	47	Ankit	14
7	97	Abhay	12
8	33	Ankit	12
9	75	Geekshub	14
10	41	Geekshub	14
11	74	Geekshub	12
12	46	Teclov	10
13	77	Ankit	13
14	27	Abhay	13
15	39	Geekshub	13
16	72	Ankit	12
17	42	Teclov	12
18	51	Geekshub	13
19	26	Ankit	12

In [18]:
```
type(df)
```

Out[18]:
```
pandas.core.frame.DataFrame
```

In [19]:
```
temp = np.random.randint(low = 20, high =100, size = [20,])
name = np.random.choice(['Abhay','Teclov','Geekshub','Ankit'],20)
random = np.random.choice([10,11,13,12,14],20)
```

In [20]:
```
df = pd.DataFrame({'temp':temp, 'name':name, 'random':random})
```

In [21]:
```
type(df)
```

Out[21]:
```
pandas.core.frame.DataFrame
```

In [23]:
```
df.head()
```

Out[23]:

	name	random	temp
0	Teclov	13	39
1	Ankit	10	84
2	Teclov	12	71
3	Teclov	11	53
4	Geekshub	12	34

In [24]:
```
df.tail()
```

Out[24]:

	name	random	temp
15	Ankit	13	44
16	Abhay	11	98
17	Ankit	14	92
18	Teclov	10	82
19	Geekshub	10	70

In [25]:
```
df.shape
```

Out[25]:
```
(20, 3)
```

In [26]:
```
df.columns
```

Out[26]:
```
Index(['name', 'random', 'temp'], dtype='object')
```

In [27]:
```
df.name
```

Out[27]:
```
0       Teclov
1        Ankit
2       Teclov
3       Teclov
4     Geekshub
5     Geekshub
6        Abhay
7     Geekshub
8     Geekshub
9     Geekshub
10       Ankit
11       Abhay
12      Teclov
13       Ankit
14      Teclov
15       Ankit
16       Abhay
17       Ankit
```

```
18      Teclov
19    Geekshub
Name: name, dtype: object
```

In [28]:
```
df['name']
```

Out[28]:
```
0       Teclov
1        Ankit
2       Teclov
3       Teclov
4     Geekshub
5     Geekshub
6        Abhay
7     Geekshub
8     Geekshub
9     Geekshub
10       Ankit
11       Abhay
12      Teclov
13       Ankit
14      Teclov
15       Ankit
16       Abhay
17       Ankit
18      Teclov
19    Geekshub
Name: name, dtype: object
```

In [30]:
```
df['temp'].describe()
```

Out[30]:
```
count    20.0000
mean     63.3000
std      22.6718
min      23.0000
25%      43.7500
50%      70.5000
75%      82.2500
max      98.0000
Name: temp, dtype: float64
```

In [31]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 3 columns):
name      20 non-null object
random    20 non-null int64
temp      20 non-null int64
dtypes: int64(2), object(1)
memory usage: 560.0+ bytes
```

In [33]:
```
df.values
```

Out[33]:
```
array([['Teclov', 13, 39],
       ['Ankit', 10, 84],
       ['Teclov', 12, 71],
       ['Teclov', 11, 53],
       ['Geekshub', 12, 34],
       ['Geekshub', 11, 78],
       ['Abhay', 10, 43],
       ['Geekshub', 12, 87],
       ['Geekshub', 12, 34],
       ['Geekshub', 13, 71],
       ['Ankit', 12, 54],
```

```
       ['Abhay', 12, 83],
       ['Teclov', 12, 23],
       ['Ankit', 13, 82],
       ['Teclov', 10, 44],
       ['Ankit', 13, 44],
       ['Abhay', 11, 98],
       ['Ankit', 14, 92],
       ['Teclov', 10, 82],
       ['Geekshub', 10, 70]], dtype=object)
```

In [34]:
```
df
```

Out[34]:

	name	random	temp
0	Teclov	13	39
1	Ankit	10	84
2	Teclov	12	71
3	Teclov	11	53
4	Geekshub	12	34
5	Geekshub	11	78
6	Abhay	10	43
7	Geekshub	12	87
8	Geekshub	12	34
9	Geekshub	13	71
10	Ankit	12	54
11	Abhay	12	83
12	Teclov	12	23
13	Ankit	13	82
14	Teclov	10	44
15	Ankit	13	44
16	Abhay	11	98
17	Ankit	14	92
18	Teclov	10	82
19	Geekshub	10	70

In [38]:
```
df.set_index('temp', inplace = True)
```

In [39]:
```
df
```

Out[39]:

	name	random
temp		
39	Teclov	13
84	Ankit	10

temp	name	random
71	Teclov	12
53	Teclov	11
34	Geekshub	12
78	Geekshub	11
43	Abhay	10
87	Geekshub	12
34	Geekshub	12
71	Geekshub	13
54	Ankit	12
83	Abhay	12
23	Teclov	12
82	Ankit	13
44	Teclov	10
44	Ankit	13
98	Abhay	11
92	Ankit	14
82	Teclov	10
70	Geekshub	10

In [40]:
```
df.sort_index(axis =0, ascending=False)
```

Out[40]:

temp	name	random
98	Abhay	11
92	Ankit	14
87	Geekshub	12
84	Ankit	10
83	Abhay	12
82	Teclov	10
82	Ankit	13
78	Geekshub	11
71	Teclov	12
71	Geekshub	13

temp	name	random
70	Geekshub	10
54	Ankit	12
53	Teclov	11
44	Teclov	10
44	Ankit	13
43	Abhay	10
39	Teclov	13
34	Geekshub	12
34	Geekshub	12
23	Teclov	12

In [41]:
```
df.sort_values(by ='random', ascending = False)
```

Out[41]:

temp	name	random
92	Ankit	14
39	Teclov	13
44	Ankit	13
82	Ankit	13
71	Geekshub	13
34	Geekshub	12
23	Teclov	12
83	Abhay	12
54	Ankit	12
87	Geekshub	12
34	Geekshub	12
71	Teclov	12
78	Geekshub	11
53	Teclov	11
98	Abhay	11
84	Ankit	10
43	Abhay	10
44	Teclov	10

temp	name	random
82	Teclov	10
70	Geekshub	10

In [45]:
```
df.drop(['random'], axis =1)
```

Out[45]:

temp	name
39	Teclov
84	Ankit
71	Teclov
53	Teclov
34	Geekshub
78	Geekshub
43	Abhay
87	Geekshub
34	Geekshub
71	Geekshub
54	Ankit
83	Abhay
23	Teclov
82	Ankit
44	Teclov
44	Ankit
98	Abhay
92	Ankit
82	Teclov
70	Geekshub

In [46]:
```
df.head()
```

Out[46]:

temp	name	random
39	Teclov	13
84	Ankit	10

temp	name	random
71	Teclov	12
53	Teclov	11
34	Geekshub	12

In [47]:
```
df.iloc[[0,1]]
```

Out[47]:

temp	name	random
39	Teclov	13
84	Ankit	10

In [48]:
```
df.iloc[1:3,1]
```

Out[48]:
```
temp
84    10
71    12
Name: random, dtype: int64
```

In [49]:
```
df.iloc[[True,True,False,True]]
```

Out[49]:

temp	name	random
39	Teclov	13
84	Ankit	10
53	Teclov	11

In [50]:
```
df.head()
```

Out[50]:

temp	name	random
39	Teclov	13
84	Ankit	10
71	Teclov	12
53	Teclov	11
34	Geekshub	12

In [53]:
```
df.loc[39,:]
```

Out[53]:
```
name      Teclov
random        13
Name: 39, dtype: object
```

In [54]:
```
df.loc[[39,84,34]]
```

Out[54]:

temp	name	random
39	Teclov	13
84	Ankit	10
34	Geekshub	12
34	Geekshub	12

In [55]:
```
df.loc[[39,84],'name':'random']
```

Out[55]:

temp	name	random
39	Teclov	13
84	Ankit	10

In [56]:
```
df.loc[[True, True, False, True]]
```

Out[56]:

temp	name	random
39	Teclov	13
84	Ankit	10
53	Teclov	11

In [57]:
```
df.loc[df.random > 13]
```

Out[57]:

temp	name	random
92	Ankit	14

In [58]:
```
df.loc[(df.random > 13) | (df.random == 10),:]
```

Out[58]:

temp	name	random
84	Ankit	10
43	Abhay	10
44	Teclov	10
92	Ankit	14
82	Teclov	10

	name	random
temp		
70	Geekshub	10

In [59]:
```
# Merging & concat
d1 = pd.DataFrame([['a', 1], ['b', 2]],columns=['col1', 'number'])
d2 = pd.DataFrame([['c', 3, 'lion'], ['d', 4, 'tiger']],columns=['letter', 'number', 'a
```

In [60]:
```
d1
```

Out[60]:

	col1	number
0	a	1
1	b	2

In [61]:
```
d2
```

Out[61]:

	letter	number	animal
0	c	3	lion
1	d	4	tiger

In [62]:
```
pd.concat([d1,d2],axis =0)
```

Out[62]:

	animal	col1	letter	number
0	NaN	a	NaN	1
1	NaN	b	NaN	2
0	lion	NaN	c	3
1	tiger	NaN	d	4

In [63]:
```
pd.concat([d1,d2], axis =0, ignore_index=True)
```

Out[63]:

	animal	col1	letter	number
0	NaN	a	NaN	1
1	NaN	b	NaN	2
2	lion	NaN	c	3
3	tiger	NaN	d	4

In [64]:
```
pd.concat([d1,d2], axis = 1)
```

Out[64]:

	col1	number	letter	number	animal
0	a	1	c	3	lion
1	b	2	d	4	tiger

In [65]:
```
d1 = pd.DataFrame({
    "city" : ["lucknow","kanpur","agra","delhi"],
    "temperature" : [32,45,30,40]
})
```

In [66]:
```
d1
```

Out[66]:

	city	temperature
0	lucknow	32
1	kanpur	45
2	agra	30
3	delhi	40

In [67]:
```
d2 = pd.DataFrame({
    "city" : ["delhi","lucknow","kanpur"],
    "humidity" : [68,65,75]
})
```

In [68]:
```
d2
```

Out[68]:

	city	humidity
0	delhi	68
1	lucknow	65
2	kanpur	75

In [69]:
```
df = pd.merge(d1,d2, on='city')
```

In [70]:
```
df
```

Out[70]:

	city	temperature	humidity
0	lucknow	32	65
1	kanpur	45	75
2	delhi	40	68

In [71]:
```
pd.merge(d1,d2, on=['city'], how ='outer')
```

Out[71]:

	city	temperature	humidity
0	lucknow	32	65.0
1	kanpur	45	75.0
2	agra	30	NaN
3	delhi	40	68.0

```
pd.merge(d1, d2, on =['city'], how='left')
```

In [72]:

Out[72]:

	city	temperature	humidity
0	lucknow	32	65.0
1	kanpur	45	75.0
2	agra	30	NaN
3	delhi	40	68.0

In [73]:
```
# dataset from https://github.com/codebasics/py/blob/master/pandas/6_handling_missing_d
```

In [74]:
```
df1 = pd.read_csv("weather_data.csv")
```

In [75]:
```
df1
```

Out[75]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [76]:
```
# pip3 install openpyxl
df1.to_excel('df_xl.xlsx', sheet_name = 'weather_data')
```

In [77]:
```
# pip3 install xlrd
df2 = pd.read_excel('df_xl.xlsx')
```

In [78]:
```
df2
```

Out[78]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [79]:
```
df2.to_csv('file.csv')
```

In [80]:
```
df2.to_csv('file_noindex.csv', index = False)
```

In [82]:
```
df_group = df2.groupby("event")
```

In [83]:
```
df_group
```

Out[83]:
```
<pandas.core.groupby.DataFrameGroupBy object at 0x10cf31f98>
```

In [84]:
```
for temperature in df_group:
    print(temperature)
```

```
('Rain',         day  temperature  windspeed event
0  1/1/2017           32          6  Rain
4  1/5/2017           32          4  Rain)
('Snow',         day  temperature  windspeed event
2  1/3/2017           28          2  Snow
3  1/4/2017           24          7  Snow)
('Sunny',         day  temperature  windspeed  event
1  1/2/2017           35          7  Sunny
5  1/6/2017           31          2  Sunny)
```

In [85]:
```
df_group.get_group('Rain')
```

Out[85]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
4	1/5/2017	32	4	Rain

In [86]:
```
df_group.describe()
```

Out[86]:

	temperature													
	count	mean	std	min	25%	50%	75%	max	count	mean	std	min	25%	50%
event														
Rain	2.0	32.0	0.000000	32.0	32.0	32.0	32.0	32.0	2.0	5.0	1.414214	4.0	4.50	5.0
Snow	2.0	26.0	2.828427	24.0	25.0	26.0	27.0	28.0	2.0	4.5	3.535534	2.0	3.25	4.5
Sunny	2.0	33.0	2.828427	31.0	32.0	33.0	34.0	35.0	2.0	4.5	3.535534	2.0	3.25	4.5

In [87]:
```
def hot_temp(x):
    return x > 30
```

In [88]:
```
df2['hot_temp'] = df2['temperature'].apply(hot_temp)
```

In [89]:
```
df2
```

Out[89]:

	day	temperature	windspeed	event	hot_temp
0	1/1/2017	32	6	Rain	True
1	1/2/2017	35	7	Sunny	True
2	1/3/2017	28	2	Snow	False
3	1/4/2017	24	7	Snow	False
4	1/5/2017	32	4	Rain	True

	day	temperature	windspeed	event	hot_temp
5	1/6/2017	31	2	Sunny	True

In [90]:
```
df2['hot_temp'] = df2['temperature'].apply(lambda x: x > 30)
```

In [91]:
```
df2
```

Out[91]:

	day	temperature	windspeed	event	hot_temp
0	1/1/2017	32	6	Rain	True
1	1/2/2017	35	7	Sunny	True
2	1/3/2017	28	2	Snow	False
3	1/4/2017	24	7	Snow	False
4	1/5/2017	32	4	Rain	True
5	1/6/2017	31	2	Sunny	True

In [92]:
```
#pivot table
```

In [93]:
```
df2.pivot_table(values = 'temperature', index = 'event', aggfunc = 'mean')
```

Out[93]:

	temperature
event	
Rain	32
Snow	26
Sunny	33

In [95]:
```
df2.pivot_table(columns = 'temperature')
```

Out[95]:

temperature	24	28	31	32	35
hot_temp	False	False	True	True	True
windspeed	7	2	2	5	7

In [96]:
```
help(pd.DataFrame.pivot_table)
```

```
Help on function pivot_table in module pandas.core.frame:

pivot_table(self, values=None, index=None, columns=None, aggfunc='mean', fill_value=Non
e, margins=False, dropna=True, margins_name='All')
    Create a spreadsheet-style pivot table as a DataFrame. The levels in
    the pivot table will be stored in MultiIndex objects (hierarchical
    indexes) on the index and columns of the result DataFrame

    Parameters
    ----------
    values : column to aggregate, optional
    index : column, Grouper, array, or list of the previous
        If an array is passed, it must be the same length as the data. The
```

```
    list can contain any of the other types (except list).
    Keys to group by on the pivot table index.  If an array is passed,
    it is being used as the same manner as column values.
columns : column, Grouper, array, or list of the previous
    If an array is passed, it must be the same length as the data. The
    list can contain any of the other types (except list).
    Keys to group by on the pivot table column.  If an array is passed,
    it is being used as the same manner as column values.
aggfunc : function or list of functions, default numpy.mean
    If list of functions passed, the resulting pivot table will have
    hierarchical columns whose top level are the function names
    (inferred from the function objects themselves)
fill_value : scalar, default None
    Value to replace missing values with
margins : boolean, default False
    Add all row / columns (e.g. for subtotal / grand totals)
dropna : boolean, default True
    Do not include columns whose entries are all NaN
margins_name : string, default 'All'
    Name of the row / column that will contain the totals
    when margins is True.

Examples
--------
>>> df = pd.DataFrame({"A": ["foo", "foo", "foo", "foo", "foo",
...                          "bar", "bar", "bar", "bar"],
...                    "B": ["one", "one", "one", "two", "two",
...                          "one", "one", "two", "two"],
...                    "C": ["small", "large", "large", "small",
...                          "small", "large", "small", "small",
...                          "large"],
...                    "D": [1, 2, 2, 3, 3, 4, 5, 6, 7]})
>>> df
     A    B      C  D
0  foo  one  small  1
1  foo  one  large  2
2  foo  one  large  2
3  foo  two  small  3
4  foo  two  small  3
5  bar  one  large  4
6  bar  one  small  5
7  bar  two  small  6
8  bar  two  large  7

>>> table = pivot_table(df, values='D', index=['A', 'B'],
...                     columns=['C'], aggfunc=np.sum)
>>> table
... # doctest: +NORMALIZE_WHITESPACE
C        large  small
A   B
bar one    4.0    5.0
    two    7.0    6.0
foo one    4.0    1.0
    two    NaN    6.0

Returns
-------
table : DataFrame

See also
--------
DataFrame.pivot : pivot without aggregation that can handle
    non-numeric data
```

In [ ]: