In [1]:
```
# IMPORT PANDAS FIRAST

import pandas as pd
```

In [3]:
```
obj = pd.Series([3,5,2,0,8])   # Series ka S capital hona chahye
print(obj)
```

```
0    3
1    5
2    2
3    0
4    8
dtype: int64
```

pandas has two things in one object

In [4]:
```
# Value  and index
```

In [6]:
```
sales = pd.Series([100,200,300,400])
print(sales.values)
print(sales.index)
```

```
[100 200 300 400]
RangeIndex(start=0, stop=4, step=1)
```

In [7]:
```
#  index could be non numeric

#  index could be name,etc
```

In [9]:
```
sales = pd.Series([100,200,300,400] , index = ['a', 'b', 'c', 'd'])
print(sales)
```

```
a    100
b    200
c    300
d    400
dtype: int64
```

In [ ]:
```
#  we can find values and index seperately in seriese object
```

In [11]:
```
print(sales.values)   # print only values
```

```
[100 200 300 400]
```

In [12]:
```
print(sales.index)    #  print index only. if index are non numeric then dtype ob

###   In object we can store any data type or mixture
```

```
Index(['a', 'b', 'c', 'd'], dtype='object')
```

In [13]:
```
print(sales)  display values,index and data type also
```

```
a    100
b    200
c    300
d    400
dtype: int64
```

In [16]:
```
sales = pd.Series([100,200,300,400] , index = ['a', 'b', 'c', 'd'], name = " 4 mo
print(sales)                                                          # add new
```

```
a    100
b    200
c    300
d    400
Name:  4 month sales, dtype: int64
```

Create a panda series to store a conteen data to holds values of how many sanwiches are solds each day

In [19]:
```
sw = pd.Series([10,20,35,40,15,60,90],
               index = ['mon', 'tue', 'wed','thu', 'fri','sat', 'sun'])
print(sw)
```

```
mon    10
tue    20
wed    35
thu    40
fri    15
sat    60
sun    90
dtype: int64
```

In [20]:
```
# How we can find data
```

In [25]:
```
 # for single data or value

print(sw[1])   #  numpy index
```

```
20
```

In [26]:
```
print(sw['tue'])   # pandas index ,we can define
```

```
20
```

used of array notation

In [27]:
```
# for multiple data or vale
```

In [28]:
```
print(sw[[3,6]])
```

```
thu    40
sun    90
dtype: int64
```

In [30]:
```
print(sw[['thu', 'sun']])
```

```
thu    40
sun    90
dtype: int64
```

In [31]:
```
# we can use condition for access data
```

In [32]:
```
print(sw[sw > 30])
```

```
wed    35
thu    40
sat    60
sun    90
dtype: int64
```

In [33]:
```
sw *2   #  values multiply but not store in sw
```

Out[33]:
```
mon     20
tue     40
wed     70
thu     80
fri     30
sat    120
sun    180
dtype: int64
```

In [34]:
```
print(sw)    # not store
```

```
mon    10
tue    20
wed    35
thu    40
fri    15
sat    60
sun    90
dtype: int64
```

In [35]:
```
sw = sw*2
print(sw)
```

```
mon     20
tue     40
wed     70
thu     80
fri     30
sat    120
sun    180
dtype: int64
```

In [36]:
```
sw          # store in sw
```

Out[36]:
```
mon     20
tue     40
wed     70
thu     80
fri     30
sat    120
sun    180
dtype: int64
```

In [46]:
```
sw = pd.Series([10,20,35,40,15,60,90],
               index = ['mon', 'tue', 'wed','thu', 'fri','sat', 'sun'])
print(sw)
```

```
mon    10
tue    20
wed    35
thu    40
fri    15
sat    60
sun    90
dtype: int64
```

In [47]:
```
sw = sw*2
print(sw)
```

```
mon     20
tue     40
wed     70
thu     80
fri     30
sat    120
sun    180
dtype: int64
```

In [48]:
```
sw = sw / 2
print(sw)
```

```
mon    10.0
tue    20.0
wed    35.0
thu    40.0
fri    15.0
sat    60.0
sun    90.0
dtype: float64
```

In [49]:
```
# we can find any index
```

In [50]:
```
'fri' in sw
```

Out[50]:
```
True
```

In [ ]:
```
#  How we can use numpy array data in pandas
```

In [51]:
```
import numpy as np
```

In [62]:
```
arr = np.array([3,2,5,4,6])
ind = np.array(['a', 'b', 'c', 'd', 'e'])

object = pd.Series( arr, index = ind)
print(object)
```

```
a    3
b    2
c    5
d    4
e    6
dtype: int32
```

In [63]:
```
# we can also used dictionary in pandas
```

In [64]:
```
dic_data = {"punjab":4000, "sindh":3000, "kpk":2500, "balochistan":2000}
tex_by_state = pd.Series(dic_data)
print(tex_by_state)
```

```
punjab         4000
sindh          3000
kpk            2500
balochistan    2000
dtype: int64
```

In [65]:
```
print(tex_by_state.index)
```

```
Index(['punjab', 'sindh', 'kpk', 'balochistan'], dtype='object')
```

In [ ]:
```
# We can also change index and value atomaticaly shuffle
```

In [66]:
```
dic_data = {"punjab":4000, "sindh":3000, "kpk":2500, "balochistan":2000}
tex_by_state = pd.Series(dic_data, index = ["sindh", "punjab", "kpk", "balochista
print(tex_by_state)
```

```
sindh          3000
punjab         4000
kpk            2500
balochistan    2000
dtype: int64
```

** Source for series data**

- direct data in series method
- numpy array
- python list
- dictionary

In [67]:
```
# how we find null value
```

In [68]:
```
dic_data = {"punjab":4000, "sindh":3000, "kpk":2500, "balochistan":2000}
tex_by_state = pd.Series(dic_data, index = ["sindh", "punjab", "kpk", "balochista
print(tex_by_state)
```

```
sindh          3000.0
punjab         4000.0
kpk            2500.0
balochistan    2000.0
gb                NaN
dtype: float64
```

In [71]:
```
# NAN value mean, value does not exists in panda series
dic_data = {"punjab":4000, "sindh":3000, "kpk":2500, "balochistan":2000}
tex_by_state = pd.Series(dic_data, index = ["sindh", "punjab", "kpk", "balochista
print( pd.isnull (tex_by_state))
```

```
sindh          False
punjab         False
kpk            False
balochistan    False
gb              True
dtype: bool
```

In [73]:
```
tex_by_state.name = "state tex paying capacity"
tex_by_state.index.name = "state name"
print(tex_by_state)
print(tex_by_state.index)
```

```
state name
sindh          3000.0
punjab         4000.0
kpk            2500.0
balochistan    2000.0
gb                NaN
Name: state tex paying capacity, dtype: float64
Index(['sindh', 'punjab', 'kpk', 'balochistan', 'gb'], dtype='object', name='st
ate name')
```

In [75]:
```
 # we can also changed the index of panda series
```

DataFrame

In [1]:
```
#  we use dataframe in multiple dimension of data
```

In [2]:
```
import pandas as pd
```

In [3]:
```
data = {'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada', 'Nevada', 'Nevada'],
 'year': [2000, 2001, 2002, 2001, 2002, 2003],
 'pop': [1.5, 1.7, 3.6, 2.4, 2.9, 3.2]}
frame = pd.DataFrame(data)
```

In [4]:
```
frame
```

Out[4]:

	state	year	pop
0	Ohio	2000	1.5
1	Ohio	2001	1.7
2	Ohio	2002	3.6
3	Nevada	2001	2.4
4	Nevada	2002	2.9
5	Nevada	2003	3.2

In [5]:
```
# We makes a dictionary
```

In [6]:
```
data = {'state':['punjab','sindh','kpk','balochistan','gilgit'],'years':[2017,201
frame = pd.DataFrame(data)
print(frame)
```

```
         state  years  pop
0       punjab   2017  1.9
1        sindh   2018  2.4
2          kpk   2019  4.3
3  balochistan   2020  5.2
4       gilgit   2021  3.9
```

In [7]:
```
#  or other way
```

In [8]:
```
state=['punjab','sindh','khyberpakhtonkha','balochistan','gilgit']
years=[2017,2018,2019,2020,2021]
population = [3.9,3.3,2.4,2.9,3.7]
```

In [12]:
```
data = {'state':state,'yer':years,'pop': population}
frame = pd.DataFrame(data)
print(frame)
```

```
              state   yer  pop
0            punjab  2017  3.9
1             sindh  2018  3.3
2  khyberpakhtonkha  2019  2.4
3       balochistan  2020  2.9
4            gilgit  2021  3.7
```

In [13]:
```
#  Indexes optional practice

data = {'state':state,'yer':years,'pop': population}
frame = pd.DataFrame(data ,index=['1st','2nd','3rd','4th','5th'])
print(frame)
```

```
                 state   yer  pop
1st             punjab  2017  3.9
2nd              sindh  2018  3.3
3rd  khyberpakhtonkha  2019  2.4
4th        balochistan  2020  2.9
5th             gilgit  2021  3.7
```

In [17]:
```
#  Indexes optional practice

data = {'state':state,'yer':years,'pop': population}
frame = pd.DataFrame(data ,index=['1st','2nd','3rd','4th','5th'])
print(frame)
```

```
                 state   yer  pop
1st             punjab  2017  3.9
2nd              sindh  2018  3.3
3rd  khyberpakhtonkha  2019  2.4
4th        balochistan  2020  2.9
5th             gilgit  2021  3.7
```

In [18]:
```
# if you have lot of data then you found first five value with head()
```

In [19]:
```
frame.head()
```

Out[19]:

	state	yer	pop
1st	punjab	2017	3.9
2nd	sindh	2018	3.3
3rd	khyberpakhtonkha	2019	2.4
4th	balochistan	2020	2.9
5th	gilgit	2021	3.7

In [20]:
```
#  We can sweap / change column
```

In [22]:
```
data = {'state':['punjab','sindh','kpk','balochistan','gilgit'],'years':[2017,201
frame = pd.DataFrame(data, columns = ['years', 'state', 'pop'])
print(frame)
```

```
   years        state  pop
0   2017       punjab  1.9
1   2018        sindh  2.4
2   2019          kpk  4.3
3   2020  balochistan  5.2
4   2021       gilgit  3.9
```

In [23]:
```
frame2 = pd.DataFrame(data, columns = ['years', 'state','pop','debt'],
                      index = ['one','two','three', 'four','five'])

frame2.head()
```

Out[23]:

	years	state	pop	debt
one	2017	punjab	1.9	NaN
two	2018	sindh	2.4	NaN
three	2019	kpk	4.3	NaN
four	2020	balochistan	5.2	NaN
five	2021	gilgit	3.9	NaN

In [24]:
```
#  It is not index
#   these are colums
frame2.columns
```

Out[24]:
```
Index(['years', 'state', 'pop', 'debt'], dtype='object')
```

In [25]:
```
# these are index
frame2.index
```

Out[25]:
```
Index(['one', 'two', 'three', 'four', 'five'], dtype='object')
```

In [29]:
```
print(frame2.columns)
print(frame2.index)
```

```
Index(['years', 'state', 'pop', 'debt'], dtype='object')
Index(['one', 'two', 'three', 'four', 'five'], dtype='object')
```

In [11]:
```
data = {'state':['punjab','sindh','kpk','balochistan','gilgit'],'years':[2017,20
frame = pd.DataFrame(data, columns = ['years', 'state', 'pop'])
print(frame)
```

```
   years        state  pop
0   2017       punjab  1.9
1   2018        sindh  2.4
2   2019          kpk  4.3
3   2020  balochistan  5.2
4   2021       gilgit  3.9
```

In [12]:
```

frame2 = pd.DataFrame(data, columns = ['years', 'state','pop','debt'],
                      index = ['one','two','three', 'four','five'])
print(frame2)
```

```
       years        state  pop debt
one     2017       punjab  1.9  NaN
two     2018        sindh  2.4  NaN
three   2019          kpk  4.3  NaN
four    2020  balochistan  5.2  NaN
five    2021       gilgit  3.9  NaN
```

In [13]:
```
frame2.head()
```

Out[13]:

	years	state	pop	debt
one	2017	punjab	1.9	NaN
two	2018	sindh	2.4	NaN
three	2019	kpk	4.3	NaN
four	2020	balochistan	5.2	NaN
five	2021	gilgit	3.9	NaN

A column in data frame can be reterived as a series either by dictionary - likes notation or by Attributes

In [14]:
```
print(data)
```

```
{'state': ['punjab', 'sindh', 'kpk', 'balochistan', 'gilgit'], 'years': [2017,
2018, 2019, 2020, 2021], 'pop': [1.9, 2.4, 4.3, 5.2, 3.9]}
```

Dictionary like notation

In [16]:
```
print(frame)
                  #  these are two dictionaries
print(frame2)
```

```
   years        state  pop
0   2017       punjab  1.9
1   2018        sindh  2.4
2   2019          kpk  4.3
3   2020  balochistan  5.2
4   2021       gilgit  3.9
       years        state  pop debt
one     2017       punjab  1.9  NaN
two     2018        sindh  2.4  NaN
three   2019          kpk  4.3  NaN
four    2020  balochistan  5.2  NaN
five    2021       gilgit  3.9  NaN
```

In [17]:
```
frame['state']

#  This is a dictionary like notation to access or extract column in dataframe
```

Out[17]:
```
0         punjab
1          sindh
2            kpk
3    balochistan
4         gilgit
Name: state, dtype: object
```

In [19]:
```
#frame2['years']

frame2['pop']
```

Out[19]:
```
one      1.9
two      2.4
three    4.3
four     5.2
five     3.9
Name: pop, dtype: float64
```

In [20]:
```
#    There frame2 is an other method , attributes style accessing
#    dataframe data

frame2.state
```

Out[20]:
```
one           punjab
two            sindh
three            kpk
four     balochistan
five          gilgit
Name: state, dtype: object
```

In [22]:
```
print(frame2)
```

```
       years        state  pop debt
one     2017       punjab  1.9  NaN
two     2018        sindh  2.4  NaN
three   2019          kpk  4.3  NaN
four    2020  balochistan  5.2  NaN
five    2021       gilgit  3.9  NaN
```

In [21]:
```
#  if we find row ,we use these method

frame2.loc['one']
```

Out[21]:
```
years       2017
state     punjab
pop          1.9
debt         NaN
Name: one, dtype: object
```

In [23]:
```
frame2.loc['three']
```

Out[23]:
```
years     2019
state      kpk
pop        4.3
debt       NaN
Name: three, dtype: object
```

In [24]:
```
print(frame2.loc['five'])
```

```
years       2021
state     gilgit
pop          3.9
debt         NaN
Name: five, dtype: object
```

In [ ]:
```
#  there we can replaced NaN value
```

In [25]:
```
frame2
```

Out[25]:

	years	state	pop	debt
one	2017	punjab	1.9	NaN
two	2018	sindh	2.4	NaN
three	2019	kpk	4.3	NaN
four	2020	balochistan	5.2	NaN
five	2021	gilgit	3.9	NaN

In [27]:
```
data = {'state':['punjab','sindh','kpk','balochistan','gilgit'],'years':[2017,20
#frame = pd.DataFrame(data, columns = ['years', 'state', 'pop']

frame2 = pd.DataFrame(data, columns = ['years', 'state','pop','debt'],
                      index = ['one','two','three', 'four','five'])
print(frame2)
```

```
       years        state  pop debt
one     2017       punjab  1.9  NaN
two     2018        sindh  2.4  NaN
three   2019          kpk  4.3  NaN
four    2020  balochistan  5.2  NaN
five    2021       gilgit  3.9  NaN
```

In [33]:
```

data = {'state':['punjab','sindh','kpk','balochistan','gilgit'],'years':[2017,20
#frame = pd.DataFrame(data, columns = ['years', 'state', 'pop']

frame2 = pd.DataFrame(data, columns = ['years', 'state','pop','debt'],
                      index = ['one','two','three', 'four','five'])

ln = len(frame2)    #  Finding num of rows in dataframe
# print(frame2)
rnge = np.arange(ln)
print(rnge)
frame2['debt'] = rnge
print(frame2)
```

```
[0 1 2 3 4]
       years        state  pop  debt
one     2017       punjab  1.9     0
two     2018        sindh  2.4     1
three   2019          kpk  4.3     2
four    2020  balochistan  5.2     3
five    2021       gilgit  3.9     4
```

Remember that , inserting new values, neede to be match in length (number of elements)

In [3]:
```
import pandas as pd
data = {'state':['punjab','sindh','kpk','balochistan','gilgit'],'years':[2017,20

frame2 = pd.DataFrame(data, columns = ['years', 'state','pop','debt'],
                      index = ['one','two','three', 'four','five'])

frame2['debt'] = 37
print(frame2)
```

```
       years        state  pop  debt
one     2017       punjab  1.9    37
two     2018        sindh  2.4    37
three   2019          kpk  4.3    37
four    2020  balochistan  5.2    37
five    2021       gilgit  3.9    37
```

In [5]:
```
val =pd.Series ([1.2, 3.2, -2.2,-0.5,9.7],      #  column of debt , assigned a se
                index = ['one', 'two', 'three', 'four','five'])

frame2['debt'] = val
frame2.head()
```

Out[5]:

	years	state	pop	debt
one	2017	punjab	1.9	1.2
two	2018	sindh	2.4	3.2
three	2019	kpk	4.3	-2.2
four	2020	balochistan	5.2	-0.5
five	2021	gilgit	3.9	9.7

In [6]:
```
val =pd.Series ([1.2, 3.2, -2.2,-0.5,9.7,-4.7],      #  Length of passed values is
                                                     # Error occour
                index = ['one', 'two', 'three', 'four','five'])

frame2['debt'] = val
frame2.head()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-6-7701295db2bf> in <module>
----> 1 val =pd.Series ([1.2, 3.2, -2.2,-0.5,9.7,-4.7],      #  column of debt
 , assigned a series
      2                  index = ['one', 'two', 'three', 'four','five'])
      3 
      4 frame2['debt'] = val
      5 frame2.head()

~\anaconda3\lib\site-packages\pandas\core\series.py in __init__(self, data, ind
ex, dtype, name, copy, fastpath)
    311                 try:
    312                     if len(index) != len(data):
--> 313                         raise ValueError(
    314                             f"Length of passed values is {len(data)}, "
    315                             f"index implies {len(index)}."

ValueError: Length of passed values is 6, index implies 5.
```

In [8]:
```
val =pd.Series ([1.2, 3.2, -2.2,-0.5,9.7],
                index = ['one', 'pak', 'three', 'four','five'])
                            #  Series and DaraFrame indexes must be same, if inde
frame2['debt'] = val
frame2.head()
```

Out[8]:

	years	state	pop	debt
one	2017	punjab	1.9	1.2
two	2018	sindh	2.4	NaN
three	2019	kpk	4.3	-2.2
four	2020	balochistan	5.2	-0.5
five	2021	gilgit	3.9	9.7

Function Application and Mapping