In [1]:

```
# import lib
import pandas as pd
import numpy as np
```

In [ ]:

```
# Concat Function
# TO combine series or Data Frame and panel object we use Concat function

# Syntax pandas.concat()
```

In [ ]:

```

```

In [2]:

```
# create series

data1 = pd.Series([10,20,30])
data1
```

Out[2]:

```
0    10
1    20
2    30
dtype: int64
```

In [3]:

```
data2 = pd.Series([50,60,70])
data2
```

Out[3]:

```
0    50
1    60
2    70
dtype: int64
```

In [ ]:

```
'''
pd.concat()

Signature:
pd.concat(
    objs,
    axis=0,
    join='outer',
    join_axes=None,
    ignore_index=False,
    keys=None,
    levels=None,
    names=None,
    verify_integrity=False,
    sort=None,
    copy=True,
)

'''
```

In [5]:

```
pd.concat([data1,data2])
```

Out[5]:

```
0    10
1    20
2    30
0    50
1    60
2    70
dtype: int64
```

In [ ]:

```

```

In [ ]:

```

```

In [8]:

```
# IF series data values are diff

data3 = pd.Series([10,20,30,40,50])
data3
```

Out[8]:

```
0    10
1    20
2    30
3    40
4    50
dtype: int64
```

In [9]:

```
data4 = pd.Series([100,200,300,400,500,600,700,800])
data4
```

Out[9]:

```
0    100
1    200
2    300
3    400
4    500
5    600
6    700
7    800
dtype: int64
```

In [10]:

```
pd.concat([data3,data4])
```

Out[10]:

```
0     10
1     20
2     30
3     40
4     50
0    100
1    200
2    300
3    400
4    500
5    600
6    700
7    800
dtype: int64
```

In [ ]:

```

```

In [ ]:

```
```

In [11]:

```
# Data Frame
data5 = pd.DataFrame({
                      'Sno':[101,102,103,104,105],
                       'City':['Hyd','Bang','Mumbai','Ambala','Delhi'],
                       'Salary':[8000,12000,15000,18000,22000]
                      })
data5
```

Out[11]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [12]:

```
data6 = pd.DataFrame({
                      'Sno':[201,202,203,204,205],
                       'City':['Up','Mp','Puna','Sec','Gujrat'],
                       'Salary':[38000,32000,45000,48000,52000]
                      })
data6
```

Out[12]:

	Sno	City	Salary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [13]:

```
pd.concat([data5,data6])
```

Out[13]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [14]:

```
pd.concat([data6,data5])
```

Out[14]:

	Sno	City	Salary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [ ]:

```

```

In [15]:

```
# Default axis=0
pd.concat([data6,data5],axis=0)
```

Out[15]:

	Sno	City	Salary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [16]:

```
# axis = 0 => index or rows
# axis = 1 => Columns
pd.concat([data6,data5],axis=1)
```

Out[16]:

	Sno	City	Salary	Sno	City	Salary
0	201	Up	38000	101	Hyd	8000
1	202	Mp	32000	102	Bang	12000
2	203	Puna	45000	103	Mumbai	15000
3	204	Sec	48000	104	Ambala	18000
4	205	Gujrat	52000	105	Delhi	22000

In [ ]:

```

```

In [ ]:

```

```

In [17]:

```
# Data Frame
data7 = pd.DataFrame({
                        'Sno':[101,102,103,104,105],
                         'City':['Hyd','Bang','Mumbai','Ambala','Delhi'],
                         'Salary':[8000,12000,15000,18000,22000]
                        })
data7
```

Out[17]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [18]:

```
data8 = pd.DataFrame({
                        'Eno':[201,202,203,204,205],
                         'ECity':['Up','Mp','Puna','Sec','Gujrat'],
                         'ESalary':[38000,32000,45000,48000,52000]
                        })
data8
```

Out[18]:

	Eno	ECity	ESalary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [19]:

```
pd.concat([data7,data8],axis=1)
```

Out[19]:

	Sno	City	Salary	Eno	ECity	ESalary
0	101	Hyd	8000	201	Up	38000
1	102	Bang	12000	202	Mp	32000
2	103	Mumbai	15000	203	Puna	45000
3	104	Ambala	18000	204	Sec	48000
4	105	Delhi	22000	205	Gujrat	52000

In [20]:

```
pd.concat([data7,data8],axis=0)
```

```
C:\Users\Mithun\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: FutureW
arning: Sorting because non-concatenation axis is not aligned. A future vers
ion
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=False'.

To retain the current behavior and silence the warning, pass 'sort=True'.

  """Entry point for launching an IPython kernel.
```

Out[20]:

	City	ECity	ESalary	Eno	Salary	Sno
0	Hyd	NaN	NaN	NaN	8000.0	101.0
1	Bang	NaN	NaN	NaN	12000.0	102.0
2	Mumbai	NaN	NaN	NaN	15000.0	103.0
3	Ambala	NaN	NaN	NaN	18000.0	104.0
4	Delhi	NaN	NaN	NaN	22000.0	105.0
0	NaN	Up	38000.0	201.0	NaN	NaN
1	NaN	Mp	32000.0	202.0	NaN	NaN
2	NaN	Puna	45000.0	203.0	NaN	NaN
3	NaN	Sec	48000.0	204.0	NaN	NaN
4	NaN	Gujrat	52000.0	205.0	NaN	NaN

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
```

In [22]:

```
data10 = pd.DataFrame({
                       'Sno':[101,102,103,104,105],
                        'City':['Hyd','Bang','Mumbai','Ambala','Delhi'],
                        'Salary':[8000,12000,15000,18000,22000]
                       })
data10
```

Out[22]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [23]:

```
data11 = pd.DataFrame({
                       'Sno':[201,202,203,204,205],
                        'City':['Up','Mp','Puna','Sec','Gujrat'],
                        'Salary':[38000,32000,45000,48000,52000]
                       })
data11
```

Out[23]:

	Sno	City	Salary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [24]:

```
# Deafult ignore_index=False,
pd.concat([data10,data11],axis=0,ignore_index=False)
```

Out[24]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [25]:

```
pd.concat([data10,data11],axis=0,ignore_index=True)
```

Out[25]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000
5	201	Up	38000
6	202	Mp	32000
7	203	Puna	45000
8	204	Sec	48000
9	205	Gujrat	52000

In [ ]:

```

```

In [ ]:

```
```

In [ ]:

```
# Data Frame if we have data set of diff size of rows ? solution
```

In [26]:

```
data12 = pd.DataFrame({
                       'Sno':[101,102,103,104,105],
                        'City':['Hyd','Bang','Mumbai','Ambala','Delhi'],
                        'Salary':[8000,12000,15000,18000,22000]
                       })
data12
```

Out[26]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [27]:

```
data13 = pd.DataFrame({
                       'Sno':[201,202,203],
                        'City':['Up','Mp','Puna'],
                        'Salary':[38000,32000,45000]
                       })
data13
```

Out[27]:

	Sno	City	Salary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000

In [28]:

```
pd.concat([data12,data13],axis=0)
```

Out[28]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000

In [29]:

```
pd.concat([data12,data13],axis=1)
```

Out[29]:

	Sno	City	Salary	Sno	City	Salary
0	101	Hyd	8000	201.0	Up	38000.0
1	102	Bang	12000	202.0	Mp	32000.0
2	103	Mumbai	15000	203.0	Puna	45000.0
3	104	Ambala	18000	NaN	NaN	NaN
4	105	Delhi	22000	NaN	NaN	NaN

In [ ]:

```

```

In [ ]:

```

```

In [30]:

```
#join : {'inner', 'outer'}, default 'outer'
#    How to handle indexes on other axis(es)


pd.concat([data12,data13],axis=1,join='outer')
```

Out[30]:

	Sno	City	Salary	Sno	City	Salary
0	101	Hyd	8000	201.0	Up	38000.0
1	102	Bang	12000	202.0	Mp	32000.0
2	103	Mumbai	15000	203.0	Puna	45000.0
3	104	Ambala	18000	NaN	NaN	NaN
4	105	Delhi	22000	NaN	NaN	NaN

In [31]:

```
pd.concat([data12,data13],axis=1,join='inner')
```

Out[31]:

	Sno	City	Salary	Sno	City	Salary
0	101	Hyd	8000	201	Up	38000
1	102	Bang	12000	202	Mp	32000
2	103	Mumbai	15000	203	Puna	45000

In [32]:

```
pd.concat([data13,data12],axis=1,join='inner')
```

Out[32]:

	Sno	City	Salary	Sno	City	Salary
0	201	Up	38000	101	Hyd	8000
1	202	Mp	32000	102	Bang	12000
2	203	Puna	45000	103	Mumbai	15000

In [ ]:

```

```

In [ ]:

```

```

In [34]:

```
# join_axes=None,
# join_axes : list of Index objects
#    Specific indexes to use for the other n - 1 axes instead of performing
#    inner/outer set logic
data14 = pd.DataFrame({
                       'Sno':[101,102,103,104,105],
                        'City':['Hyd','Bang','Mumbai','Ambala','Delhi'],
                        'Salary':[8000,12000,15000,18000,22000]
                       })
data14
```

Out[34]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [35]:

```
data15 = pd.DataFrame({
                       'Sno':[201,202,203],
                        'City':['Up','Mp','Puna'],
                        'Salary':[38000,32000,45000]
                       })
data15
```

Out[35]:

	Sno	City	Salary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000

In [36]:

```
pd.concat([data14,data15],axis=1,join_axes=None)
```

Out[36]:

	Sno	City	Salary	Sno	City	Salary
0	101	Hyd	8000	201.0	Up	38000.0
1	102	Bang	12000	202.0	Mp	32000.0
2	103	Mumbai	15000	203.0	Puna	45000.0
3	104	Ambala	18000	NaN	NaN	NaN
4	105	Delhi	22000	NaN	NaN	NaN

In [37]:

```
pd.concat([data14,data15],axis=1,join_axes=[data14.index])
```

Out[37]:

	Sno	City	Salary	Sno	City	Salary
0	101	Hyd	8000	201.0	Up	38000.0
1	102	Bang	12000	202.0	Mp	32000.0
2	103	Mumbai	15000	203.0	Puna	45000.0
3	104	Ambala	18000	NaN	NaN	NaN
4	105	Delhi	22000	NaN	NaN	NaN

In [38]:

```
pd.concat([data14,data15],axis=1,join_axes=[data15.index])
```

Out[38]:

	Sno	City	Salary	Sno	City	Salary
0	101	Hyd	8000	201	Up	38000
1	102	Bang	12000	202	Mp	32000
2	103	Mumbai	15000	203	Puna	45000

In [ ]:

```

```

In [ ]:

```
# keys=None
# keys : sequence, default None
#    If multiple levels passed, should contain tuples. Construct
#    hierarchical index using the passed keys as the outermost level
```

In [ ]:

```
```

In [39]:

```
data16 = pd.DataFrame({
                       'Sno':[101,102,103,104,105],
                        'City':['Hyd','Bang','Mumbai','Ambala','Delhi'],
                        'Salary':[8000,12000,15000,18000,22000]
                       })
data16
```

Out[39]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [42]:

```
data17 = pd.DataFrame({
                       'Sno':[201,202,203,204,205],
                        'City':['Up','Mp','Puna','Sec','Gujrat'],
                        'Salary':[38000,32000,45000,48000,52000]
                       })
data17
```

Out[42]:

	Sno	City	Salary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [43]:

```
pd.concat([data16,data17],keys=None)
```

Out[43]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [44]:

```
pd.concat([data16,data17],keys=['First Data Set','Sec Data Set'])
```

Out[44]:

		Sno	City	Salary
First Data Set	**0**	101	Hyd	8000
	1	102	Bang	12000
	2	103	Mumbai	15000
	3	104	Ambala	18000
	4	105	Delhi	22000
Sec Data Set	**0**	201	Up	38000
	1	202	Mp	32000
	2	203	Puna	45000
	3	204	Sec	48000
	4	205	Gujrat	52000

In [ ]:

```

```

In [45]:

```
pd.concat([data16,data17],axis=1,keys=['First Data Set','Sec Data Set'])
```

Out[45]:

	First Data Set			Sec Data Set		
	Sno	City	Salary	Sno	City	Salary
0	101	Hyd	8000	201	Up	38000
1	102	Bang	12000	202	Mp	32000
2	103	Mumbai	15000	203	Puna	45000
3	104	Ambala	18000	204	Sec	48000
4	105	Delhi	22000	205	Gujrat	52000

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [46]:

```
data18 = pd.DataFrame({
                       'Sno':[101,102,103,104,105],
                        'City':['Hyd','Bang','Mumbai','Ambala','Delhi'],
                        'Salary':[8000,12000,15000,18000,22000]
                       })
data18
```

Out[46]:

	Sno	City	Salary
0	101	Hyd	8000
1	102	Bang	12000
2	103	Mumbai	15000
3	104	Ambala	18000
4	105	Delhi	22000

In [47]:

```
data19 = pd.DataFrame({
                       'Eno':[201,202,203,204,205],
                        'ECity':['Up','Mp','Puna','Sec','Gujrat'],
                        'ESalary':[38000,32000,45000,48000,52000]
                       })
data19
```

Out[47]:

	Eno	ECity	ESalary
0	201	Up	38000
1	202	Mp	32000
2	203	Puna	45000
3	204	Sec	48000
4	205	Gujrat	52000

In [48]:

```
pd.concat([data7,data8],axis=1)
```

Out[48]:

	Sno	City	Salary	Eno	ECity	ESalary
0	101	Hyd	8000	201	Up	38000
1	102	Bang	12000	202	Mp	32000
2	103	Mumbai	15000	203	Puna	45000
3	104	Ambala	18000	204	Sec	48000
4	105	Delhi	22000	205	Gujrat	52000

In [49]:

```
pd.concat([data7,data8],axis=0)
```

```
C:\Users\Mithun\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: FutureW
arning: Sorting because non-concatenation axis is not aligned. A future vers
ion
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=False'.

To retain the current behavior and silence the warning, pass 'sort=True'.

  """Entry point for launching an IPython kernel.
```

Out[49]:

	City	ECity	ESalary	Eno	Salary	Sno
0	Hyd	NaN	NaN	NaN	8000.0	101.0
1	Bang	NaN	NaN	NaN	12000.0	102.0
2	Mumbai	NaN	NaN	NaN	15000.0	103.0
3	Ambala	NaN	NaN	NaN	18000.0	104.0
4	Delhi	NaN	NaN	NaN	22000.0	105.0
0	NaN	Up	38000.0	201.0	NaN	NaN
1	NaN	Mp	32000.0	202.0	NaN	NaN
2	NaN	Puna	45000.0	203.0	NaN	NaN
3	NaN	Sec	48000.0	204.0	NaN	NaN
4	NaN	Gujrat	52000.0	205.0	NaN	NaN

In [50]:

```
pd.concat([data7,data8],axis=0,sort=False)
```

Out[50]:

	Sno	City	Salary	Eno	ECity	ESalary
0	101.0	Hyd	8000.0	NaN	NaN	NaN
1	102.0	Bang	12000.0	NaN	NaN	NaN
2	103.0	Mumbai	15000.0	NaN	NaN	NaN
3	104.0	Ambala	18000.0	NaN	NaN	NaN
4	105.0	Delhi	22000.0	NaN	NaN	NaN
0	NaN	NaN	NaN	201.0	Up	38000.0
1	NaN	NaN	NaN	202.0	Mp	32000.0
2	NaN	NaN	NaN	203.0	Puna	45000.0
3	NaN	NaN	NaN	204.0	Sec	48000.0
4	NaN	NaN	NaN	205.0	Gujrat	52000.0

In [ ]:

```

```

In [ ]:

```

```