In [1]:
```
from IPython.display import Image
Image(filename='logo.png', height=340, width=900)
```

Out[1]:

PANDAS

- Pandas is an open source library that is **built on top of Numpy**
- It allows for quick analysis and provides for **data cleaning and preparation**
- Pandas Library also has **built in data visualization** features essential for Data Analysis
- The name Pandas is derived from the word **"Panel Data"** – an Econometrics from Multidimensional data.
- Using Pandas, we can accomplish five typical steps in the **processing and analysis of data**:

  `load`, `prepare`, `manipulate`, `model`, and `analyze`

DATASTRUCTURES IN PANDAS

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Main Data Structures within the Python Pandas Framework

Data Structure	Dimension	Description
Series	1 DIMENSION	1D labeled homogeneous array, size immutable.
DataFrame	2 DIMENSION	General 2D labeled, size-mutable tabular structure with potentially heterogeneously typed columns

1.0 PANDAS - SERIES

- Series are very similar to ndarrays.
- the main difference between them is that with series you can provide custom index labels
- the operations on series automatically align the data based on the labels.

In [2]:
```
import numpy as np
import pandas as pd             #Note: It is common practice to import
 pandas with the shorthand "pd".
```

1.1 CREATION OF SERIES

1.1.1 CREATION OF SERIES USING LISTS

In [3]:
```
mylist = [10, 20, 30, 40]
```

In [4]:
```
# Series Objects have indexed labels
pd.Series(data=mylist)
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Out[4]:
```
0    10
1    20
2    30
3    40
dtype: int64
```

In [5]:
```
# In Series, you can als provide custom index labels
pd.Series(data=mylist, index=['a', 'b', 'c', 'd'])      # We could also
 have index = variablename where variable
                                                        # contains the
 labels
```

Out[5]:
```
a    10
b    20
c    30
d    40
dtype: int64
```

so, we can also have it as:

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

Out[6]:
```
a    10
b    20
c    30
d    40
dtype: int64
```

1.1.2 CREATION OF SERIES USING ARRAYS

In [7]:
```
myarray = np.array(mylist)
myarray
```

Out[7]:
```
array([10, 20, 30, 40])
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [8]:
```
pd.Series(myarray)
```

Out[8]:
```
0    10
1    20
2    30
3    40
dtype: int32
```

In [9]:
```
pd.Series(myarray, labels)
```

Out[9]:
```
a    10
b    20
c    30
d    40
dtype: int32
```

1.1.3 CREATION OF SERIES USING DICTIONARIES

In [10]:
```
mydict = {'a': 10, 'b': 20, 'c': 30, 'd': 40}
mydict
```

Out[10]:
```
{'a': 10, 'b': 20, 'c': 30, 'd': 40}
```

In [11]:
```
pd.Series(mydict)
```

Out[11]:
```
a    10
b    20
c    30
d    40
dtype: int64
```

1.2 ACCESSING SERIES VALUES

In [12]:
```
series1 = pd.Series([10,20,30,40], index=['DELHI', 'GURGAON', 'NOIDA',
'FARIDABAD'])
series1
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Out[12]:
```
DELHI        10
GURGAON      20
NOIDA        30
FARIDABAD    40
dtype: int64
```

In [13]:
```
series1['DELHI']
```

Out[13]:
```
10
```

In [14]:
```
series1['GURGAON']
```

Out[14]:
```
20
```

2.0 PANDAS - DATAFRAMES

- A DataFrame is a 2D table with labeled columns that can each hold different types of data.
- DataFrames are essentially a Python implementation of the types of tables you'd see in an Excel workbook or SQL database.
- DataFrames are the defacto standard data structure for working with tabular data in Python

2.1 CREATION OF DATAFRAME

In [15]:
```
# importing random library to help generate random numbers for DataFram
e
from numpy.random import randint
```

In [16]:
```
df=pd.DataFrame(randint(10,100,20).reshape(5,4), ['A', 'B', 'C', 'D',
'E'], ['GGN', 'DEL', 'NOI', 'FBD'])
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [17]:
```
df
```

Out[17]:

	GGN	DEL	NOI	FBD
A	79	42	90	88
B	87	76	19	90
C	75	61	38	86
D	84	41	13	39
E	36	90	66	49

- Each of the **above column** is actually a **pandas series**
- They all **share common indices** - A, B, C, D, E

In [18]:
```
df['GGN']  #The Output below looks like a SERIES
```

Out[18]:
```
A    79
B    87
C    75
D    84
E    36
Name: GGN, dtype: int32
```

In [19]:
```
type(df['GGN'])
```

Out[19]:
```
pandas.core.series.Series
```

In [20]:
```
# We can access multiple columns by passing a list of columns as below
df[['GGN', 'DEL']]
```

Out[20]:

	GGN	DEL
A	79	42

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	GGN	DEL
B	87	76
C	75	61
D	84	41
E	36	90

In [21]:
```
type(df[['GGN', 'DEL']])
```

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

2.2 CREATION OF NEW COLUMNS IN A DATAFRAME

2.2.1 ADDING A COLUMN USING EXISTING COLUMNS

In [22]:
```
df['SML'] = df['DEL'] + df['GGN']
```

In [23]:
```
df
```

Out[23]:

	GGN	DEL	NOI	FBD	SML
A	79	42	90	88	121
B	87	76	19	90	163
C	75	61	38	86	136
D	84	41	13	39	125
E	36	90	66	49	126

2.2.2 ADDING A COLUMN USING SINGLE VALUE

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [24]:
```
df['HYD'] = 75
```

In [25]:
```
df
```

Out[25]:

	GGN	DEL	NOI	FBD	SML	HYD
A	79	42	90	88	121	75
B	87	76	19	90	163	75
C	75	61	38	86	136	75
D	84	41	13	39	125	75
E	36	90	66	49	126	75

2.3 REMOVING COLUMNS

In [26]:
```
df.drop('HYD', axis = 1)    # Axis = 1, means dropping the columns
```

Out[26]:

	GGN	DEL	NOI	FBD	SML
A	79	42	90	88	121
B	87	76	19	90	163
C	75	61	38	86	136
D	84	41	13	39	125
E	36	90	66	49	126

In [27]:
```
df                    # Original dataframe does not change
```

Out[27]:

	GGN	DEL	NOI	FBD	SML	HYD

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	GGN	DEL	NOI	FBD	SML	HYD
A	79	42	90	88	121	75
B	87	76	19	90	163	75
C	75	61	38	86	136	75
D	84	41	13	39	125	75
E	36	90	66	49	126	75

In [28]:
```
df.drop('HYD', axis = 1, inplace=True)    # inplace = True, updates the
 value of the variable storing the dataframe
```

In [29]:
```
df
```

Out[29]:

	GGN	DEL	NOI	FBD	SML
A	79	42	90	88	121
B	87	76	19	90	163
C	75	61	38	86	136
D	84	41	13	39	125
E	36	90	66	49	126

In [30]:
```
df.drop(['SML', 'NOI'], axis = 1, inplace=True)   # To drop multiple co
lumns use a list of columns.
```

In [31]:
```
df
```

Out[31]:

	GGN	DEL	FBD
A	79	42	88

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	GGN	DEL	FBD
B	87	76	90
C	75	61	86
D	84	41	39
E	36	90	49

In [32]:
```
df.drop('E', axis = 0)    # To drop the rows. Note inplace is not used
 so original df have E as column
```

Out[32]:

	GGN	DEL	FBD
A	79	42	88
B	87	76	90
C	75	61	86
D	84	41	39

In [33]:
```
df
```

Out[33]:

	GGN	DEL	FBD
A	79	42	88
B	87	76	90
C	75	61	86
D	84	41	39
E	36	90	49

2.4 ACCESSING ROWS

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

2.4.1 Using .loc Method

In [34]:
```
df.loc['A']   # Returns a Series Object; ALL Rows are also Series Objec
ts
```

Out[34]:
```
GGN    79
DEL    42
FBD    88
Name: A, dtype: int32
```

2.4.2 Using .iloc Method

In [35]:
```
df.iloc[0]
```

Out[35]:
```
GGN    79
DEL    42
FBD    88
Name: A, dtype: int32
```

2.4.3 More than one rows & columns

In [36]:
```
df.loc[['A','B'], ['GGN', 'FBD']]
```

Out[36]:

	GGN	FBD
A	79	88
B	87	90

In [37]:
```
df.iloc[0:2,[0,2]]
```

Out[37]:

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	GGN	FBD
A	79	88
B	87	90

In [38]:
```
df.loc[['A', 'C', 'E'], ['GGN', 'FBD']]
```

Out[38]:

	GGN	FBD
A	79	88
C	75	86
E	36	49

In [39]:
```
df.iloc[[0,2,4], [0,2]]
```

Out[39]:

	GGN	FBD
A	79	88
C	75	86
E	36	49

2.4.4 CONDITIONAL SELECTION

In [40]:
```
newdf = df > 50
```

In [41]:
```
newdf
```

Out[41]:

	GGN	DEL	FBD

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	GGN	DEL	FBD
A	True	False	True
B	True	True	True
C	True	True	True
D	True	False	False
E	False	True	False

In [42]:
```
df[newdf]
```

Out[42]:

	GGN	DEL	FBD
A	79.0	NaN	88.0
B	87.0	76.0	90.0
C	75.0	61.0	86.0
D	84.0	NaN	NaN
E	NaN	90.0	NaN

In [43]:
```
df[df > 35]            # Directly passing the conditional selection cri
teria
```

Out[43]:

	GGN	DEL	FBD
A	79	42	88
B	87	76	90
C	75	61	86
D	84	41	39
E	36	90	49

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [44]:
```
df[df['GGN'] > 35]    # Most Often used
```

Out[44]:

	GGN	DEL	FBD
A	79	42	88
B	87	76	90
C	75	61	86
D	84	41	39
E	36	90	49

In [45]:
```
df
```

Out[45]:

	GGN	DEL	FBD
A	79	42	88
B	87	76	90
C	75	61	86
D	84	41	39
E	36	90	49

In [46]:
```
# All rows where FBD < 65
df[df['FBD'] < 65]
```

Out[46]:

	GGN	DEL	FBD
D	84	41	39
E	36	90	49

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [47]:
```
# We can also apply slicing on this
df[df['FBD'] < 65]['GGN']
```

Out[47]:
```
D    84
E    36
Name: GGN, dtype: int32
```

In [48]:
```
# We can also apply slicing on this
df[df['FBD'] < 65][['GGN', 'DEL']]
```

Out[48]:

	GGN	DEL
D	84	41
E	36	90

2.4.5 CONDITIONAL SELECTION (MULTIPLE CONDITIONS)

In [49]:
```
# Pulling all rows with FBD < 65 and GGN > 40
df[(df['FBD'] < 65) & (df['GGN']>40)]
```

Out[49]:

	GGN	DEL	FBD
D	84	41	39

In [50]:
```
# Pulling all rows with FBD < 65 OR GGN > 40
df[(df['FBD'] < 65) | (df['GGN']<40)]
```

Out[50]:

	GGN	DEL	FBD
D	84	41	39
E	36	90	49

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

2.5 RESETTING INDEX

In [51]:
```
df.reset_index()  # inplace = False, you can use inplace= True
```

Out[51]:

	index	GGN	DEL	FBD
0	A	79	42	88
1	B	87	76	90
2	C	75	61	86
3	D	84	41	39
4	E	36	90	49

2.6 SETTING A NEW INDEX

In [52]:
```
names = ['Alpha', 'Beta', 'Gamma', 'Delta', 'Epsilon']
df['NewCol'] = names
df
```

Out[52]:

	GGN	DEL	FBD	NewCol
A	79	42	88	Alpha
B	87	76	90	Beta
C	75	61	86	Gamma
D	84	41	39	Delta
E	36	90	49	Epsilon

In [53]:
```
df.set_index('NewCol')
```

Out[53]:

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	GGN	DEL	FBD
NewCol			
Alpha	79	42	88
Beta	87	76	90
Gamma	75	61	86
Delta	84	41	39
Epsilon	36	90	49

2.7 TREATMENT OF MISSING VALUES

In [54]:
```
df=pd.DataFrame({'A':[1,2,np.nan,4,7],'B':[np.nan,3.5,4,5,8], 'C':[3,np
.nan,2.5,6,9],
                 'D':[4,np.nan,np.nan,np.nan,9], 'E':[9,8,6,6,4]})
```

In [55]:
```
df
```

Out[55]:

	A	B	C	D	E
0	1.0	NaN	3.0	4.0	9
1	2.0	3.5	NaN	NaN	8
2	NaN	4.0	2.5	NaN	6
3	4.0	5.0	6.0	NaN	6
4	7.0	8.0	9.0	9.0	4

2.7.1 DROPNA METHOD

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [56]:
```
df.dropna()                    # It will drop ROWS with one or more missin
g values
```

Out[56]:

	A	B	C	D	E
4	7.0	8.0	9.0	9.0	4

In [57]:
```
df.dropna(axis=1)              # It will drop COLUMNS with one or more mis
sing values
```

Out[57]:

	E
0	9
1	8
2	6
3	6
4	4

In [58]:
```
df.dropna(thresh=4)            # Out of the 5 Columns, it should have valu
es in atleast 4 of them for it to be retained
```

Out[58]:

	A	B	C	D	E
0	1.0	NaN	3.0	4.0	9
3	4.0	5.0	6.0	NaN	6
4	7.0	8.0	9.0	9.0	4

2.7.2 FILLNA METHOD

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

2.7.2A FILLNA USING ANY VALUE

In [59]:
```
df.fillna(value = 'MY VALUE')
```

Out[59]:

	A	B	C	D	E
0	1	MY VALUE	3	4	9
1	2	3.5	MY VALUE	MY VALUE	8
2	MY VALUE	4	2.5	MY VALUE	6
3	4	5	6	MY VALUE	6
4	7	8	9	9	4

In [60]:
```
df.fillna(value = 100)
```

Out[60]:

	A	B	C	D	E
0	1.0	100.0	3.0	4.0	9
1	2.0	3.5	100.0	100.0	8
2	100.0	4.0	2.5	100.0	6
3	4.0	5.0	6.0	100.0	6
4	7.0	8.0	9.0	9.0	4

2.7.2B FILLNA USING CENTRAL VALUES

In [61]:
```
df['A']
```

Out[61]:
```
0    1.0
1    2.0
2    NaN
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
3    4.0
4    7.0
Name: A, dtype: float64
```

In [62]:
```
df['A'].fillna(value= df['A'].mean())
```

Out[62]:
```
0    1.0
1    2.0
2    3.5
3    4.0
4    7.0
Name: A, dtype: float64
```

In [63]:
```
df['A'].fillna(value= df['A'].median())
```

Out[63]:
```
0    1.0
1    2.0
2    3.0
3    4.0
4    7.0
Name: A, dtype: float64
```

2.7.3 FILLNA USING FOR LOOP

In [64]:
```
#for replacing the Nan Values with Mean of Columns for all the columns
 in a DataFrame
for x in df.columns:
    df[x]=df[x].fillna(value=df[x].mean())
```

In [65]:
```
df
```

Out[65]:

	A	B	C	D	E
0	1.0	5.125	3.000	4.0	9
1	2.0	3.500	5.125	6.5	8

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	A	B	C	D	E
2	3.5	4.000	2.500	6.5	6
3	4.0	5.000	6.000	6.5	6
4	7.0	8.000	9.000	9.0	4

2.8 GROUPBY METHOD

In [66]:
```
groupdata = {'Company': ['Flipkart', 'Flipkart', 'SnapDeal', 'SnapDeal'
, 'Myntra', 'Myntra', 'Jabong', 'Jabong'],
        'Employee': ['Bhupesh', 'Vaibhav', 'Sharad', 'Vipul', 'Sachin',
 'Akshay', 'Dishant', 'Vikram'],
        'Revenue': [300, 400, 250, 150, 550, 375, 700, 680]}
```

In [67]:
```
df = pd.DataFrame(groupdata)
df
```

Out[67]:

	Company	Employee	Revenue
0	Flipkart	Bhupesh	300
1	Flipkart	Vaibhav	400
2	SnapDeal	Sharad	250
3	SnapDeal	Vipul	150
4	Myntra	Sachin	550
5	Myntra	Akshay	375
6	Jabong	Dishant	700
7	Jabong	Vikram	680

In [68]:
```
df.groupby('Company').sum()
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Out[68]:

	Revenue
Company	
Flipkart	700
Jabong	1380
Myntra	925
SnapDeal	400

In [69]:
```
df.groupby('Company').mean()
```

Out[69]:

	Revenue
Company	
Flipkart	350.0
Jabong	690.0
Myntra	462.5
SnapDeal	200.0

In [70]:
```
df.groupby('Company').count()
```

Out[70]:

	Employee	Revenue
Company		
Flipkart	2	2
Jabong	2	2
Myntra	2	2

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	Employee	Revenue
Company		
SnapDeal	2	2

In [71]:
```
df.groupby('Company').mean().loc['Flipkart']
```

Out[71]:
```
Revenue    350.0
Name: Flipkart, dtype: float64
```

In [72]:
```
df.groupby('Company').describe()
```

Out[72]:

	Revenue							
	count	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
Company								
Flipkart	2.0	350.0	70.710678	300.0	325.00	350.0	375.00	400.0
Jabong	2.0	690.0	14.142136	680.0	685.00	690.0	695.00	700.0
Myntra	2.0	462.5	123.743687	375.0	418.75	462.5	506.25	550.0
SnapDeal	2.0	200.0	70.710678	150.0	175.00	200.0	225.00	250.0

In [73]:
```
df.groupby('Company').describe().transpose()
```

Out[73]:

	Company	Flipkart	Jabong	Myntra	SnapDeal
Revenue	count	2.000000	2.000000	2.000000	2.000000
	mean	350.000000	690.000000	462.500000	200.000000
	std	70.710678	14.142136	123.743687	70.710678
	min	300.000000	680.000000	375.000000	150.000000

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	Company	Flipkart	Jabong	Myntra	SnapDeal
	25%	325.000000	685.000000	418.750000	175.000000
	50%	350.000000	690.000000	462.500000	200.000000
	75%	375.000000	695.000000	506.250000	225.000000
	max	400.000000	700.000000	550.000000	250.000000

2.9 COMBINING DATAFRAMES

In [74]:
```
df1=pd.DataFrame({'StudentName':["Akash", "Vivek", "Sondeep", "Pranav",
 "Purnima", "Divya" "Saroj",
                                  "Nimisha", "ajay", "Manju", "Mr. X"],
                  'Class':['I', 'III', 'IV', 'I', 'V', 'IV', 'III', 'IV'
, 'IV', 'III',]})
```

In [75]:
```
df2=pd.DataFrame({'City':["Shimla", "Gurgaon", "Delhi", "Amritsar", "Ja
landhar", "Gandhinagar",
                           "Almora", "Ooty", "Mumbai", "Chennai", "Gurga
on", "Delhi"],
                  'Age': [6,8,9,6,10,9,8,9,9,8,10, 11]})
```

In [76]:
```
df3=pd.DataFrame({'RollNo.':[1,3,5,7,9,11,13,15,17,19],
                  'PinCode':["171001", "122002", "110005", "183005", "1
81001", "168754", "987654",
                             "547645", "654789", "123456"]})
```

In [77]:
```
df1
```

Out[77]:

	StudentName	Class
0	Akash	I
1	Vivek	III

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	StudentName	Class
2	Sondeep	IV
3	Pranav	I
4	Purnima	V
5	DivyaSaroj	IV
6	Nimisha	III
7	ajay	IV
8	Manju	IV
9	Mr. X	III

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

Out[78]:

	City	Age
0	Shimla	6
1	Gurgaon	8
2	Delhi	9
3	Amritsar	6
4	Jalandhar	10
5	Gandhinagar	9
6	Almora	8
7	Ooty	9
8	Mumbai	9
9	Chennai	8
10	Gurgaon	10

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	City	Age
11	Delhi	11

In [79]:
```
df3
```

Out[79]:

	RollNo.	PinCode
0	1	171001
1	3	122002
2	5	110005
3	7	183005
4	9	181001
5	11	168754
6	13	987654
7	15	547645
8	17	654789
9	19	123456

2.9.1 CONCATENATION

Incase the user wants to concatenate the data horizontally, use "axis=1", default value is "axis=0"

In [80]:
```
pd.concat([df1, df2], axis= 1)  # CONCATENATING AT COLUMNS
```

Out[80]:

	StudentName	Class	City	Age

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	StudentName	Class	City	Age
0	Akash	I	Shimla	6
1	Vivek	III	Gurgaon	8
2	Sondeep	IV	Delhi	9
3	Pranav	I	Amritsar	6
4	Purnima	V	Jalandhar	10
5	DivyaSaroj	IV	Gandhinagar	9
6	Nimisha	III	Almora	8
7	ajay	IV	Ooty	9
8	Manju	IV	Mumbai	9
9	Mr. X	III	Chennai	8
10	NaN	NaN	Gurgaon	10
11	NaN	NaN	Delhi	11

In [81]:
```
pd.concat([df1, df2], axis= 0)
```

```
C:\Users\Admin\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: Fut
ureWarning: Sorting because non-concatenation axis is not aligned. A fu
ture version
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=Tru
e'.

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

Out[81]:

	Age	City	Class	StudentName

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	Age	City	Class	StudentName
0	NaN	NaN	I	Akash
1	NaN	NaN	III	Vivek
2	NaN	NaN	IV	Sondeep
3	NaN	NaN	I	Pranav
4	NaN	NaN	V	Purnima
5	NaN	NaN	IV	DivyaSaroj
6	NaN	NaN	III	Nimisha
7	NaN	NaN	IV	ajay
8	NaN	NaN	IV	Manju
9	NaN	NaN	III	Mr. X
0	6.0	Shimla	NaN	NaN
1	8.0	Gurgaon	NaN	NaN
2	9.0	Delhi	NaN	NaN
3	6.0	Amritsar	NaN	NaN
4	10.0	Jalandhar	NaN	NaN
5	9.0	Gandhinagar	NaN	NaN
6	8.0	Almora	NaN	NaN
7	9.0	Ooty	NaN	NaN
8	9.0	Mumbai	NaN	NaN
9	8.0	Chennai	NaN	NaN
10	10.0	Gurgaon	NaN	NaN
11	11.0	Delhi	NaN	NaN

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [82]:
```
pd.concat([df1, df2, df3], axis=1)
```

Out[82]:

	StudentName	Class	City	Age	RollNo.	PinCode
0	Akash	I	Shimla	6	1.0	171001
1	Vivek	III	Gurgaon	8	3.0	122002
2	Sondeep	IV	Delhi	9	5.0	110005
3	Pranav	I	Amritsar	6	7.0	183005
4	Purnima	V	Jalandhar	10	9.0	181001
5	DivyaSaroj	IV	Gandhinagar	9	11.0	168754
6	Nimisha	III	Almora	8	13.0	987654
7	ajay	IV	Ooty	9	15.0	547645
8	Manju	IV	Mumbai	9	17.0	654789
9	Mr. X	III	Chennai	8	19.0	123456
10	NaN	NaN	Gurgaon	10	NaN	NaN
11	NaN	NaN	Delhi	11	NaN	NaN

2.9.2 MERGING DATAFRAMES

In [83]:
```
leftdf=pd.DataFrame({'RollNo.':[1,2,3,4,5,6,7,8,9,10,11],
                     'StudentName':["Akash", "Vivek", "Sondeep", "Pranav",
"Purnima",
                                    "Divya","Saroj", "Nimisha", "ajay", "Ma
nju", "Mr. X"]})
```

In [84]:
```
rightdf=pd.DataFrame({'RollNo.':[1,2,3,4,5,6,7,8,9,10],
                      'City':["Shimla", "Gurgaon", "Delhi", "Amritsar",
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
 "Jalandhar", "Gandhinagar",
                                "Almora", "Ooty", "Mumbai", "Chennai"]})
```

In [85]:
```
onemoredf=pd.DataFrame({'RollNo.':[1,3,5,7,9,11,13,15,17,19],
                        'PinCode':["171001", "122002", "110005", "183005", "181001", "168754", "987654",
                                   "547645", "654789", "123456"]})
```

In [86]:
```
pd.merge(leftdf,rightdf, how='inner', on='RollNo.')
```

Out[86]:

	RollNo.	StudentName	City
0	1	Akash	Shimla
1	2	Vivek	Gurgaon
2	3	Sondeep	Delhi
3	4	Pranav	Amritsar
4	5	Purnima	Jalandhar
5	6	Divya	Gandhinagar
6	7	Saroj	Almora
7	8	Nimisha	Ooty
8	9	ajay	Mumbai
9	10	Manju	Chennai

In [87]:
```
pd.merge(leftdf,rightdf, how='left', on='RollNo.')
```

Out[87]:

	RollNo.	StudentName	City
0	1	Akash	Shimla
1	2	Vivek	Gurgaon

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	RollNo.	StudentName	City
2	3	Sondeep	Delhi
3	4	Pranav	Amritsar
4	5	Purnima	Jalandhar
5	6	Divya	Gandhinagar
6	7	Saroj	Almora
7	8	Nimisha	Ooty
8	9	ajay	Mumbai
9	10	Manju	Chennai
10	11	Mr. X	NaN

In [88]:
```
pd.merge(leftdf,onemoredf, how='right', on='RollNo.')
```

Out[88]:

	RollNo.	StudentName	PinCode
0	1	Akash	171001
1	3	Sondeep	122002
2	5	Purnima	110005
3	7	Saroj	183005
4	9	ajay	181001
5	11	Mr. X	168754
6	13	NaN	987654
7	15	NaN	547645
8	17	NaN	654789
9	19	NaN	123456

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	RollNo.	StudentName	PinCode

In [89]:
```
pd.merge(leftdf,onemoredf, how='outer', on='RollNo.')
```

Out[89]:

	RollNo.	StudentName	PinCode
0	1	Akash	171001
1	2	Vivek	NaN
2	3	Sondeep	122002
3	4	Pranav	NaN
4	5	Purnima	110005
5	6	Divya	NaN
6	7	Saroj	183005
7	8	Nimisha	NaN
8	9	ajay	181001
9	10	Manju	NaN
10	11	Mr. X	168754
11	13	NaN	987654
12	15	NaN	547645
13	17	NaN	654789
14	19	NaN	123456

In [90]:
```
#Executing the same functions using the JOIN Prompt
#This can be done after setting index to the common column in both the
 tables. for this we use the "set_index" prompt to
#set indices
leftdf.set_index('RollNo.').join(onemoredf.set_index('RollNo.'), how='i
nner')
```

Out[90]:

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Out[90]:

	StudentName	PinCode
RollNo.		
1	Akash	171001
3	Sondeep	122002
5	Purnima	110005
7	Saroj	183005
9	ajay	181001
11	Mr. X	168754

2.10 OPERATIONS

In [91]:
```
df= pd.DataFrame({'Column A': [1,2,3,4,5,6,7,8],
                  'Column B': [54321, 12345, 67890,98765, 98765, 67890,
12345,98765],
                  'Column C': ['Ankur', 'Amit', 'Abhishek', 'Arpit', 'Am
an', 'Arpan', 'Bhupesh', 'Bharat']})
```

In [92]:
```
df.head()
```

Out[92]:

	Column A	Column B	Column C
0	1	54321	Ankur
1	2	12345	Amit
2	3	67890	Abhishek
3	4	98765	Arpit
4	5	98765	Aman

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

2.10.1 UNIQUE VALUES IN ANY COLUMN

In [93]:
```
df['Column B'].unique()
```

Out[93]:
```
array([54321, 12345, 67890, 98765], dtype=int64)
```

2.10.1B NO. OF UNIQUE VALUES IN ANY COLUMN

In [94]:
```
# No. of Unique Values
len(df['Column B'].unique())
```

Out[94]:
```
4
```

In [95]:
```
# No. of Unique Values
df['Column B'].nunique()
```

Out[95]:
```
4
```

2.10.1C NO. OF UNIQUE VALUES AND THEIR COUNTS

In [96]:
```
df['Column B'].value_counts()
```

Out[96]:
```
98765    3
67890    2
12345    2
54321    1
Name: Column B, dtype: int64
```

2.10.2 APPLY METHOD

In [97]:
```
df['Column C'].apply(len)
```

Out[97]:
```
0    5
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
1    4
2    8
3    5
4    4
5    5
6    7
7    6
Name: Column C, dtype: int64
```

In [98]:
```
df['Column B'].apply(lambda x: x/2)
```

Out[98]:
```
0    27160.5
1     6172.5
2    33945.0
3    49382.5
4    49382.5
5    33945.0
6     6172.5
7    49382.5
Name: Column B, dtype: float64
```

2.10.3 GET THE COLUMNS NAMES OR INDEX NAMES

In [99]:
```
df.columns
```

Out[99]:
```
Index(['Column A', 'Column B', 'Column C'], dtype='object')
```

In [100]:
```
df.index
```

Out[100]:
```
RangeIndex(start=0, stop=8, step=1)
```

2.10.4 SORTING DATAFRAMES

In [101]:
```
df.sort_values(by='Column B')
```

Out[101]:

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	Column A	Column B	Column C
1	2	12345	Amit
6	7	12345	Bhupesh
0	1	54321	Ankur
2	3	67890	Abhishek
5	6	67890	Arpan
3	4	98765	Arpit
4	5	98765	Aman
7	8	98765	Bharat

In [102]:
```
# SORTING BY DESCENDING ORDER
df.sort_values(by='Column B', ascending = False)
```

Out[102]:

	Column A	Column B	Column C
3	4	98765	Arpit
4	5	98765	Aman
7	8	98765	Bharat
2	3	67890	Abhishek
5	6	67890	Arpan
0	1	54321	Ankur
1	2	12345	Amit
6	7	12345	Bhupesh

2.10.5 CHECKING FOR NULL VALUES

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [103]:
```
df.isnull()
```

Out[103]:

	Column A	Column B	Column C
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
5	False	False	False
6	False	False	False
7	False	False	False

In [104]:
```
df.isnull().sum()
```

Out[104]:
```
Column A    0
Column B    0
Column C    0
dtype: int64
```

2.11 DATA INPUT & OUTPUT

- CSV
- EXCEL
- HTML
- SQL

- conda install sqlalchemy
- conda install lxml

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

- conda install html5lib
- conda install BeautifulSoup4

2.11.1 CSV FILES

In [105]:
```
data = pd.read_csv('Example1.csv')
```

In [106]:
```
data.head()
```

Out[106]:

	Series_reference	Period	Data_value	STATUS	UNITS
0	ECTA.S19A1	2001.03	2462.5	F	Dollars
1	ECTA.S19A1	2002.03	17177.2	F	Dollars
2	ECTA.S19A1	2003.03	22530.5	F	Dollars
3	ECTA.S19A1	2004.03	28005.1	F	Dollars
4	ECTA.S19A1	2005.03	30629.6	F	Dollars

2.11.2 SAVING IN OTHER FORMATS

In [107]:
```
data.to_excel('Example2.xlsx', index = False)
```

2.11.3 EXCEL FILES

In [108]:
```
data1 = pd.read_excel('Example2.xlsx')
```

In [109]:
```
data1.head()
```

Out[109]:

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	Series_reference	Period	Data_value	STATUS	UNITS
0	ECTA.S19A1	2001.03	2462.5	F	Dollars
1	ECTA.S19A1	2002.03	17177.2	F	Dollars
2	ECTA.S19A1	2003.03	22530.5	F	Dollars
3	ECTA.S19A1	2004.03	28005.1	F	Dollars
4	ECTA.S19A1	2005.03	30629.6	F	Dollars

2.11.3 HTML PAGES

In [110]:
```
data2 = pd.read_html('https://www.fdic.gov/bank/individual/failed/banklist.html')
```

In [111]:
```
data3=data2[0]
```

In [112]:
```
data3
```

Out[112]:

	Bank Name	City	ST	CERT	Acquiring Institution	Closing Date
0	The First State Bank	Barboursville	WV	14361	MVB Bank, Inc.	April 3, 2020
1	Ericson State Bank	Ericson	NE	18265	Farmers and Merchants Bank	February 14, 2020
2	City National Bank of New Jersey	Newark	NJ	21111	Industrial Bank	November 1, 2019
3	Resolute Bank	Maumee	OH	58317	Buckeye State Bank	October 25, 2019

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

	Bank Name	City	ST	CERT	Acquiring Institution	Closing Date
4	Louisa Community Bank	Louisa	KY	58112	Kentucky Farmers Bank Corporation	October 25, 2019
...	...	...	...	...	...	...
556	Superior Bank, FSB	Hinsdale	IL	32646	Superior Federal, FSB	July 27, 2001
557	Malta National Bank	Malta	OH	6629	North Valley Bank	May 3, 2001
558	First Alliance Bank & Trust Co.	Manchester	NH	34264	Southern New Hampshire Bank & Trust	February 2, 2001
559	National State Bank of Metropolis	Metropolis	IL	3815	Banterra Bank of Marion	December 14, 2000
560	Bank of Honolulu	Honolulu	HI	21029	Bank of the Orient	October 13, 2000

561 rows × 6 columns

2.11.4 SQL DATA

In [113]:
```
from sqlalchemy import create_engine
```

In [114]:
```
engine = create_engine('sqlite:///:memory:')
```

In [115]:
```
data.to_sql('mysqltable', engine)
```

In [116]:
```
sqldf = pd.read_sql('mysqltable', con=engine)
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

In [117]:
```
sqldf.head()
```

Out[117]:

	index	Series_reference	Period	Data_value	STATUS	UNITS
0	0	ECTA.S19A1	2001.03	2462.5	F	Dollars
1	1	ECTA.S19A1	2002.03	17177.2	F	Dollars
2	2	ECTA.S19A1	2003.03	22530.5	F	Dollars
3	3	ECTA.S19A1	2004.03	28005.1	F	Dollars
4	4	ECTA.S19A1	2005.03	30629.6	F	Dollars

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD