importing pandas

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

In [4]:
```
df=pd.DataFrame()
```

In [5]:
```
df
```

Out[5]:

In [8]:
```
import pandas as pd
mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pd.DataFrame(mydataset)

print(myvar)
```

```
    cars  passings
0    BMW         3
1  Volvo         7
2   Ford         2
```

Most important function

pd.read_csv(filename) : It read the data from CSV file. pd.read_table(filename) : It is used to read the data from delimited text file. pd.read_excel(filename) : It read the data from an Excel file. pd.read_sql(query,connection _object) : It read the data from a SQL table/database. df.to_csv(filename): It writes to a CSV file. df.to_excel(filename): It writes to an Excel file. df.to_sql(table_name, connection_object): It writes to a SQL table. function for sorting df[df[col] > 0.5]: Returns the rows where column col is greater than 0.5 df[(df[col] > 0.5) & (df[col] < 0.7)] : Returns the rows where 0.7 > col > 0.5 df.sort_values(col1) :It sorts the values by col1 in ascending order. df.sort_values(col2,ascending=False) :It sorts the values by col2 in descending order.

creating series from array

In [10]:
```
import pandas as pd
import numpy as np
info = np.array(['P','a','n','d','a','s'])
a = pd.Series(info)
print(a)
```

```
0    P
1    a
2    n
3    d
4    a
5    s
dtype: object
```

Creating series from dictionary

In [12]:
```
import pandas as pd
import numpy as np
info = {'x' : 0., 'y' : 1., 'z' : 2.}
a = pd.Series(info)
print (a)
```

```
x    0.0
y    1.0
z    2.0
dtype: float64
```

Retrieving Index array and data array of a series object

In [20]:
```
import numpy as np
import pandas as pd
x=pd.Series(data=[2,4,6,8])
y=pd.Series(data=[11.2,18.6,22.5], index=['a','b','c'])
print(x.index) #returns (start,stop,step)
print(x.values) #returns values in x
print(y.index)   #returns ('a','b','c'  , dyype='object')
print(y.values)  #returns values  stored in y
```

```
RangeIndex(start=0, stop=4, step=1)
[2 4 6 8]
Index(['a', 'b', 'c'], dtype='object')
[11.2 18.6 22.5]
```

Retrieving Dimension, Size and Number of bytes:

In [24]:
```
import numpy as np
import pandas as pd
a=pd.Series(data=[1,2,3,4])   #creation of dataseries
b=pd.Series(data=[4.9,8.2,5.6])  #creation of dataseries
index=(['x','y','z']) #customizing index
print("type:",a.ndim, b.ndim)   #returns the type of array i.e either 1D , 2D , #3D
print("size:",a.size, b.size)   #returns the size of the array i.e. 4 and 3
print("total size of array:",a.nbytes, b.nbytes) #returns the byte size of the array
```

```
type: 1 1
size: 4 3
total size of array: 32 24
```

Convert Pandas DataFrame to CSV using 'framename'.to_csv

The Pandas to_csv() function is used to convert the DataFrame into CSV data. To write the CSV data into a file, we can simply pass a file object to the function. Otherwise, the CSV data is returned in a string format.

In [26]:
```
import pandas as pd
data = {'Name': ['Smith', 'Parker'], 'ID': [101, 102], 'Language': ['Python', 'JavaS
info = pd.DataFrame(data)  #storing data as a dataframe info
print('DataFrame Values:\n', info)  #printing info dataframe
```

```
csv_data = info.to_csv()  #converting to csv type using "to_csv()"
print('\nCSV String Values:\n', csv_data) #printing csv data
```

```
DataFrame Values:
      Name   ID    Language
0   Smith  101      Python
1  Parker  102  JavaScript

CSV String Values:
 ,Name,ID,Language
0,Smith,101,Python
1,Parker,102,JavaScript
```

A csv stands for Comma Separated Values, which is defined as a simple file format that uses specific structuring to arrange tabular data. It stores tabular data such as spreadsheet or database in plain text and has a common format for data interchange. The csv file is opened into the excel file, and the rows and columns data define the standard format.

Pandas DataFrame.count()

In [4]:

```
import pandas as pd
import numpy as np
info = pd.DataFrame({"Person":["Parker", "Smith", "William", "John"],
"Age": [27., 29, np.nan, 32]
info.count()
```

```
  File "<ipython-input-4-6a6d477d652d>", line 5
    pd.info.count()
    ^
SyntaxError: invalid syntax
```

converting dataframe to numpy using "dataframe".to_numpy()

In [7]:

```
import pandas as pd
info = pd.DataFrame([[17, 62, 35],[25, 36, 54],[42, 20, 15],[48, 62, 76]],
columns=['x', 'y', 'z'])
print('DataFrame\n----------\n', info)
#convert the dataframe to a numpy array
arr = info.to_numpy()
print('\nNumpy Array\n----------\n', arr)
```

```
DataFrame
----------
     x   y   z
0  17  62  35
1  25  36  54
2  42  20  15
3  48  62  76

Numpy Array
----------
 [[17 62 35]
 [25 36 54]
 [42 20 15]
 [48 62 76]]
```

changing the sepration of comma with other

symbols

In [13]:
```
import pandas as pd
data = {'Name': ['Smith', 'Parker'], 'ID': [101, pd.NaT], 'Language': ['Python', 'Ja
info = pd.DataFrame(data)
print('DataFrame:\n', info)
print('\n')
csv_data = info.to_csv(sep='|')
print(csv_data)
```

```
DataFrame:
      Name   ID    Language
0   Smith  101      Python
1  Parker  NaT  JavaScript


|Name|ID|Language
0|Smith|101|Python
1|Parker||JavaScript
```

Python Pandas Reading Files

For reading the Pandas files, firstly we have to load data from file formats into a DataFrame. You need only a single line to load your data in code.made usning df=pd.read_csv('"filename.csv') then we can print df to get the output

genrating a excel sheet

In [15]:
```
import pandas as pd
# create dataframe using dicrtionary
info_marks = pd.DataFrame({'name': ['Parker', 'Smith', 'William', 'Terry'],
     'Maths': [78, 84, 67, 72],
     'Science': [89, 92, 61, 77],
     'English': [72, 75, 64, 82]})

# render dataframe as html
writer = pd.ExcelWriter('output.xlsx')
info_marks.to_excel(writer)
writer.save()
print('DataFrame is written successfully to the Excel File.')
```

```
DataFrame is written successfully to the Excel File.
```

In [ ]: