Pandas

- Open source data analysis and manipulation tool, built on top of the python programming language.
- It provides high-performance, easy to use data structures and data analysis tools for working with structured and time series data.

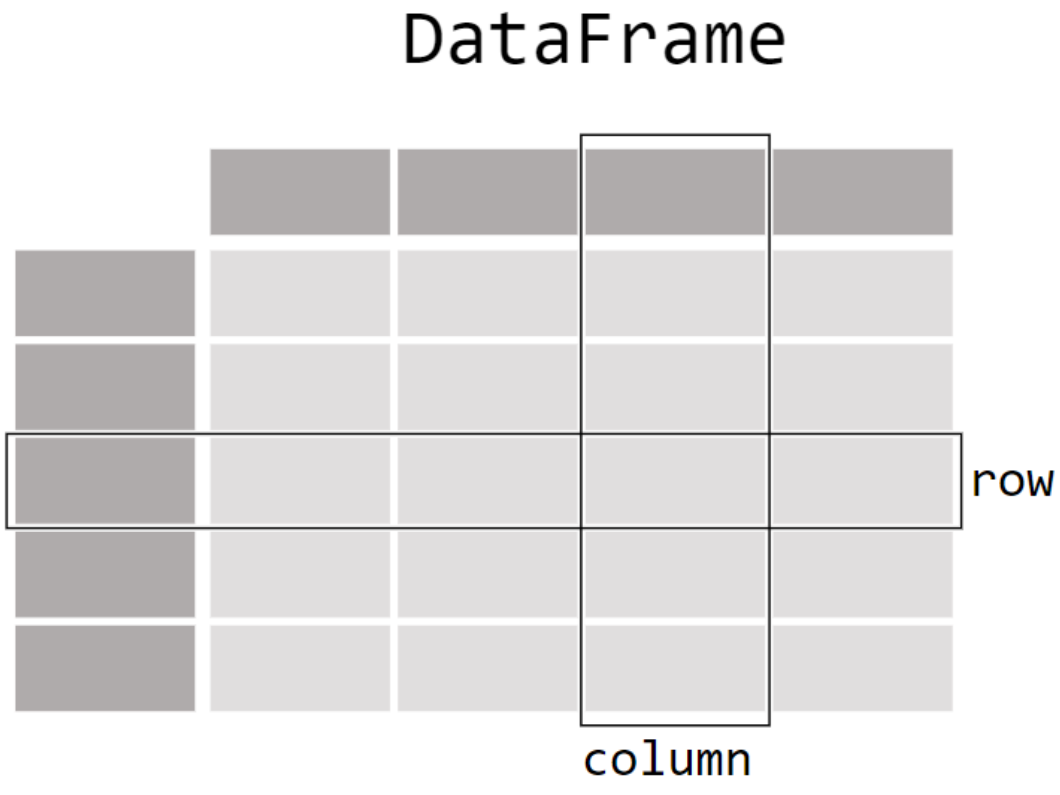
Numpy

Matplotlib

What kind of data does pandas handle?

Straight to tutorial...

When working with tabular data, such as data stored in spreadsheets or databases, pandas is the right tool for you. pandas will help you to explore, clean, and process your data. In pandas, a data table is called a `DataFrame`.



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

In [3]:
```
print(pd.__version__)
```

```
1.5.3
```

In [4]:
```
data={
    "names":['a','b','c'],
    "ages":[25,28,21]
}
```

In [5]:
```
df=pd.DataFrame(data)
df
```

Out[5]:

	names	ages
0	a	25
1	b	28
2	c	21

In [7]:
```
df.head(2)
```

Out[7]:

	names	ages
0	a	25
1	b	28

In [8]:
```
df.tail(1)
```

Out[8]:

	names	ages
2	c	21

In [9]:
```
df.describe()
```

Out[9]:

	ages
count	3.000000
mean	24.666667
std	3.511885
min	21.000000
25%	23.000000
50%	25.000000
75%	26.500000
max	28.000000

In [12]:
```
df.dtypes
```

Out[12]:
```
names    object
ages      int64
dtype: object
```

In [13]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   names   3 non-null      object
 1   ages    3 non-null      int64 
dtypes: int64(1), object(1)
memory usage: 180.0+ bytes
```

In [14]:
```
df.columns=["emp_name","age"]
df
```

Out[14]:

	emp_name	age
0	a	25
1	b	28
2	c	21

In [15]:
```
df2=df.rename(columns={"emp_name":"emp_fname","age":"emp_age"})
df2
```

Out[15]:

	emp_fname	emp_age
0	a	25
1	b	28
2	c	21

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

Out[16]:

	emp_name	age
0	a	25
1	b	28
2	c	21

In [21]:
```
dept=["Sales","Marketing","IT"]
df2=df.assign(company=dept)
df2
```

Out[21]:

	emp_name	age	company
0	a	25	Sales
1	b	28	Marketing
2	c	21	IT

In [22]:
```
df2.dtypes
```

Out[22]:
```
emp_name    object
age          int64
company     object
dtype: object
```

In [23]:
```
df3=df2.convert_dtypes()
df3.dtypes
```

Out[23]:
```
emp_name    string
age          Int64
company     string
dtype: object
```

In [24]:
```
df4=df2.astype({"age":float})
df4.dtypes
```

Out[24]:
```
emp_name     object
age         float64
company      object
dtype: object
```

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

Out[26]:
```
(3, 2)
```

In [28]:
```
len(df.index)
```

Out[28]:
```
3
```

In [29]:
```
len(df.columns)
```

Out[29]:
```
2
```

In [30]:
```
technologies = {
    'Courses':["Spark","PySpark","Hadoop","Python","pandas"],
    'Fee' :[20000,25000,26000,22000,24000],
    'Duration':['30day','40days','35days','40days','60days'],
    'Discount':[1000,2300,1200,2500,2000]
              }
```

In [31]:
```
index_label=['r1','r2','r3','r4','r5']
```

In [32]:
```
df=pd.DataFrame(technologies)
df
```

Out[32]:

	Courses	Fee	Duration	Discount
0	Spark	20000	30day	1000
1	PySpark	25000	40days	2300
2	Hadoop	26000	35days	1200
3	Python	22000	40days	2500
4	pandas	24000	60days	2000

In [33]:
```
df=pd.DataFrame(technologies,index=index_label)
df
```

Out[33]:

	Courses	Fee	Duration	Discount
r1	Spark	20000	30day	1000
r2	PySpark	25000	40days	2300
r3	Hadoop	26000	35days	1200
r4	Python	22000	40days	2500
r5	pandas	24000	60days	2000

In [34]:
```
df.loc['r1']
```

Out[34]:
```
Courses     Spark
Fee         20000
Duration    30day
Discount     1000
Name: r1, dtype: object
```

In [35]:
```
df.loc[['r1','r2']]
```

Out[35]:

	Courses	Fee	Duration	Discount
r1	Spark	20000	30day	1000
r2	PySpark	25000	40days	2300

In [36]:
```
df.loc[:,"Courses"]
```

Out[36]:
```
r1      Spark
r2    PySpark
r3     Hadoop
r4     Python
r5     pandas
Name: Courses, dtype: object
```

In [38]:
```
df.loc[:,["Courses","Fee"]]
```

Out[38]:

	Courses	Fee
r1	Spark	20000
r2	PySpark	25000
r3	Hadoop	26000

In [39]:
```
df.loc['r1':'r4']
```

Out[39]:

	Courses	Fee	Duration	Discount
r1	Spark	20000	30day	1000
r2	PySpark	25000	40days	2300
r3	Hadoop	26000	35days	1200
r4	Python	22000	40days	2500

In [41]:
```
df.loc[df["Discount"]>=2000]
```

Out[41]:

	Courses	Fee	Duration	Discount
r2	PySpark	25000	40days	2300
r4	Python	22000	40days	2500
r5	pandas	24000	60days	2000

In [42]:
```
df.query("Courses=='PySpark'")
```

Out[42]:

	Courses	Fee	Duration	Discount
r2	PySpark	25000	40days	2300

In [43]:
```
df.query("Courses!='PySpark'")
```

Out[43]:

	Courses	Fee	Duration	Discount
r1	Spark	20000	30day	1000
r3	Hadoop	26000	35days	1200
r4	Python	22000	40days	2500
r5	pandas	24000	60days	2000

In [46]:
```
df2=df.drop("Discount",axis='columns')  #df.drop(columns="Discount")
df2
```

Out[46]:

	Courses	Fee	Duration
r1	Spark	20000	30day
r2	PySpark	25000	40days
r3	Hadoop	26000	35days
r4	Python	22000	40days
r5	pandas	24000	60days

Read and Write

In [3]:
```
import pandas as pd
```

In [5]:
```
emp_df=pd.read_csv("employee.csv")
emp_df.head(5)
```

Out[5]:

	First Name	Gender	Start Date	Last Login Time	Salary	Bonus %	Senior Management	Team
0	Douglas	Male	8/6/1993	12:42 PM	97308	6.945	True	Marketing
1	Thomas	Male	3/31/1996	6:53 AM	61933	4.170	True	NaN
2	Maria	Female	4/23/1993	11:17 AM	130590	11.858	False	Finance
3	Jerry	Male	3/4/2005	1:00 PM	138705	9.340	True	Finance
4	Larry	Male	1/24/1998	4:47 PM	101004	1.389	True	Client Services

In [6]:
```
emp_df.shape
```

Out[6]:
```
(1000, 8)
```

In [7]:
```
emp_df.describe()
```

Out[7]:

	Salary	Bonus %
count	1000.000000	1000.000000

In [9]:
```
emp_df.dtypes
```

Out[9]:
```
First Name            object
Gender                object
Start Date            object
Last Login Time       object
Salary                 int64
Bonus %              float64
Senior Management     object
Team                  object
dtype: object
```

In [12]:
```
df=emp_df.convert_dtypes()
df.dtypes
```

Out[12]:
```
First Name            string
Gender                string
Start Date            string
Last Login Time       string
Salary                 Int64
Bonus %              Float64
Senior Management    boolean
Team                  string
dtype: object
```

In [14]:
```
df["Start Date"]=pd.to_datetime(df["Start Date"])
df.dtypes
```

Out[14]:
```
First Name                  string
Gender                      string
Start Date          datetime64[ns]
Last Login Time             string
Salary                       Int64
Bonus %                    Float64
Senior Management          boolean
Team                        string
dtype: object
```

In [15]:
```
df.nunique()
```

Out[15]:
```
First Name           200
Gender                 2
Start Date           972
Last Login Time      720
Salary               995
Bonus %              971
Senior Management      2
Team                  10
dtype: int64
```

In [17]:
```
df.isnull().sum()
```

Out[17]:
```
First Name            67
Gender               145
Start Date             0
Last Login Time        0
Salary                 0
Bonus %                0
Senior Management     67
Team                  43
dtype: int64
```

In [19]:
```
df["Gender"].fillna("No Gender", inplace = True)
df.isnull().sum()
```

Out[19]:
```
First Name           67
Gender                0
Start Date            0
Last Login Time       0
Salary                0
Bonus %               0
Senior Management    67
Team                 43
dtype: int64
```

In [21]:
```
new = df.groupby(['Gender']).size()
new
```

Out[21]:
```
Gender
Female       431
Male         424
No Gender    145
dtype: int64
```

In [22]:
```
df['Senior Management'] = df['Senior Management'].fillna(df['Senior Management'].mode()[0])
df.isnull().sum()
```

Out[22]:
```
First Name           67
Gender                0
Start Date            0
Last Login Time       0
Salary                0
Bonus %               0
Senior Management     0
Team                 43
dtype: int64
```

In [26]:
```
df.to_csv("Final_emp_df")
```

Numpy

In [1]:
```
import numpy as np
```

In [2]:
```
arr=np.array([1,2,3])
```

In [3]:
```
arr
```

Out[3]:
```
array([1, 2, 3])
```

In [5]:
```
arr2=np.array([[1,2,3],[4,5,6]])
```

In [6]:
```
arr2
```

Out[6]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [7]:
```
arr.ndim
```

Out[7]:
```
1
```

In [8]:
```
arr2.ndim
```

Out[8]:
```
2
```

In [9]:
```
arr3=np.array([[[1,23,33],[2,22,54],[5,6,8]],[[10,20,30],[25,22,55],[50,60,80]]])
```

In [10]:
```
arr3
```

Out[10]:
```
array([[[ 1, 23, 33],
        [ 2, 22, 54],
        [ 5,  6,  8]],

       [[10, 20, 30],
        [25, 22, 55],
        [50, 60, 80]]])
```

In [11]:
```
arr3.ndim
```

Out[11]:
```
3
```

In [12]:
```
arr
```

Out[12]:
```
array([1, 2, 3])
```

In [13]:
```
arr[1]
```

Out[13]:
```
2
```

In [14]:
```
arr[0:2]
```

Out[14]:
```
array([1, 2])
```

In [15]:
```
arr2
```

Out[15]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [17]:
```
arr2[0,2]
```

Out[17]:
```
3
```

In [18]:
```
arr2[1,0:2]
```

Out[18]:
```
array([4, 5])
```

In [19]:
```
arr3[0][1][2]
```

Out[19]:
```
54
```

In [21]:
```
arr3[1,1,1:3]
```

Out[21]:
```
array([22, 55])
```

In [19]:
```
arr1 = [[1,2],[3,4],[5,6]]
arr2 = [[6,7],[8,9],[10,11]]
res = np.concatenate((arr1, arr2))
print(res)
```

```
[[ 1  2]
 [ 3  4]
 [ 5  6]
 [ 6  7]
 [ 8  9]
 [10 11]]
```

In [20]:
```
arr1 = [1,2,3,4,5]
arr2 = [6,7,8,9,10]
res = np.concatenate((arr1, arr2))
print(res)
```

```
[ 1  2  3  4  5  6  7  8  9 10]
```

In [21]:
```
temperature_data = [25.3, 26.1, 24.8, 23.5, 27.2]
pressure_data = [101.2, 100.8, 101.5, 100.2, 101.0]
humidity_data = [55.2, 54.8, 56.5, 53.7, 55.9]
temperature_array=np.array(temperature_data)
pressure_array=np.array(pressure_data)
humidity_array=np.array(humidity_data)
```

In [22]:
```
print("Temperature Mean:", np.mean(temperature_array))
print("Pressure Mean:", np.mean(pressure_array))
print("Humidity Mean:", np.mean(humidity_array))
```

```
Temperature Mean: 25.380000000000003
Pressure Mean: 100.94
Humidity Mean: 55.21999999999999
```

In [23]:
```
print("Temperature Standard Deviation::", np.std(temperature_array))
print("Pressure Standard Deviation::", np.std(pressure_array))
print("Humidity Standard Deviation::", np.std(humidity_array))
```

```
Temperature Standard Deviation:: 1.2416118556135003
Pressure Standard Deviation:: 0.43634848458542813
Humidity Standard Deviation:: 0.9579144011862429
```

In [24]:
```
temperature_min=np.min(temperature_array)
temperature_min
```

Out[24]:
```
23.5
```

In [25]:
```
temperature_max=np.max(temperature_array)
temperature_max
```

Out[25]:
```
27.2
```

Matplotlib

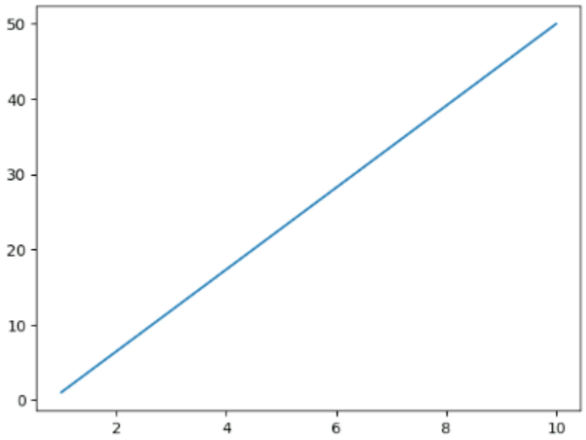
In [1]:
```
import matplotlib.pyplot as plt
```
```
Matplotlib is building the font cache; this may take a moment.
```

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

In [3]:
```
x_axis=np.array([1,10])
y_axis=np.array([1,50])
```

In [4]:
```
plt.plot(x_axis,y_axis)
```

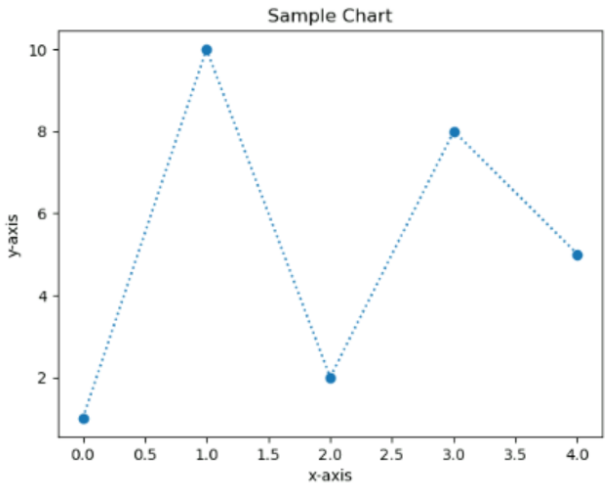
Out[4]:
```
[<matplotlib.lines.Line2D at 0x7f4fccac3b90>]
```



In [15]:
```
y_points=np.array([1,10,2,8,5])
plt.plot(y_points,marker="o",linestyle="dotted")

plt.title("Sample Chart")
plt.xlabel("x-axis")
plt.ylabel("y-axis")
```

Out[15]:
```
Text(0, 0.5, 'y-axis')
```



In [16]:
```
months=['January', 'February', 'March', 'April', 'May']
sales=[2500, 3200, 2800, 4100, 3700]
```

In [17]:
```
months=np.array(months)
sales=np.array(sales)
```

In [18]:
```
plt.bar(months,sales)
```

Out[18]:
```
<BarContainer object of 5 artists>
```