Pandas

What is pandas Pandas is a python library used for working data sets. It has functions for analyzing , cleaning , exploring and manipulating data. The name "Pandas" has a reference to both "Panel Data" , and "Python Data Analysis" and was created by Wes McKinney in 2008

Why use Pandas Pandas allows us to analyze big data and make conculsions based on statistical theories. Pandas can clean messy data sets , and make them readable and relevent . Relevent data is very important in data science.

What can Pandas Do Pandas gives you answers about the data . Like : Is there a correlation between two or more columns? What is the average value? Max value? Min value?

Pandas are also able to delete rows that are not relevent , or contains wrong values , like empty or NULL values.This is called cleaning the data.

Installation

In [1]:
```
pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\saich\anaconda3\lib\site-packages (2.1.4)
Requirement already satisfied: numpy<2,>=1.23.2 in c:\users\saich\anaconda3\lib\site-packages (from pandas) (1.2
6.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\saich\anaconda3\lib\site-packages (from pandas
) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\saich\anaconda3\lib\site-packages (from pandas) (2023.3.
post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\saich\anaconda3\lib\site-packages (from pandas) (2023.
3)
Requirement already satisfied: six>=1.5 in c:\users\saich\anaconda3\lib\site-packages (from python-dateutil>=2.8
.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

Key Data Structures Series : Defination : A one-dimensional labled array capble of holding any data type (e.g integer,string,float).

In [2]:
```
#Creating a Series
import pandas as pd
s = pd.Series([1,2,3,4],index = ['a','b','c','d'])
print(s)
```

```
a    1
b    2
c    3
d    4
dtype: int64
```

DataFrame : Defination : A two-dimensional labled data structure with columns of potentially different types , similar to spreadsheet(excel sheet) or SQL table.

In [3]:
```
#Creating a DataFrame
data = {
    'Name' : ['Upendra' , 'Ram' , 'Jaganadh'],
    'Age' : [30 , 35 , 27],
    'City' : ['Kunta' , 'Amaravathi' , 'Guntur']}
df =  pd.DataFrame(data)
print(df)
```

```
      Name  Age        City
0   Upendra   30       Kunta
1       Ram   35  Amaravathi
2  Jaganadh   27      Guntur
```

DataFrame Manipulations Accessing Data: By column : In column we get the total columns name form our data

In [4]:
```
print(df['Name'])
```

```
0     Upendra
1         Ram
2    Jaganadh
Name: Name, dtype: object
```

Accessing Data : By Row : In rows we get the specific location data on our demand

In [5]:
```
print(df.loc[0])
```

```
Name    Upendra
Age          30
City      Kunta
Name: 0, dtype: object
```

Filtering Data : Filtering data involves in getting data on our requirements.

In [6]:
```
fdf = df[df['Age']>27]
print(fdf)
```

```
     Name  Age        City
0  Upendra   30       Kunta
1      Ram   35  Amaravathi
```

Adding New Column : we can add any number of column on our requrirement .By just mentioning the column name with the varible name that we assign to the dataframe.

In [7]:
```
df['Movie'] = ['Raktha kaneru' , 'Raktha Charitra' , 'Nenu intha']
print(df)
```

```
      Name  Age        City            Movie
0   Upendra   30       Kunta    Raktha kaneru
1       Ram   35  Amaravathi  Raktha Charitra
2  Jaganadh   27      Guntur       Nenu intha
```

Handling Missing Data Identifying Missing Value : For correct data predction meachine wants correct data if is there any missing values is is difficult to meachine to predict a correct and accurate results.So we have to identify the missing values and rectifie them. Missing values can identify by using isnull() or isnull().sum()

In [8]:
```
df.loc[1, 'Age'] = None
print(df.isnull())
```

```
    Name    Age   City  Movie
0  False  False  False  False
1  False   True  False  False
2  False  False  False  False
```

In [9]:
```
df.isnull().sum()
```

Out[9]:
```
Name     0
Age      1
City     0
Movie    0
dtype: int64
```

Filling Missing Values :Missing value filling can be done in different ways like by using mode() , median() , mean()

In [10]:
```
df['Age'].fillna(df['Age'].mode(),inplace = True)
print(df)
```

```
      Name   Age        City            Movie
0   Upendra  30.0       Kunta    Raktha kaneru
1       Ram  30.0  Amaravathi  Raktha Charitra
2  Jaganadh  27.0      Guntur       Nenu intha
```

Dropping Missing Values :In some data sets we may have more missing values in that time it is good to drop the entire column if we drop the data by using dropna() we may miss important data. So we have to very careful on droping data

In [11]:
```
df.dropna(inplace = True)
print(df)
```

```
      Name   Age        City            Movie
0   Upendra  30.0       Kunta    Raktha kaneru
1       Ram  30.0  Amaravathi  Raktha Charitra
2  Jaganadh  27.0      Guntur       Nenu intha
```

Data Aggregation and Grouping : by using groupby('column') we can group by the data .And by using agg({'column1':'mean','column2':'sum',etc}) we can aggregate the data

In [12]:
```
ag = df.agg({'Age':'mean'})
print(ag)
```

```
Age    29.0
dtype: float64
```

Merging and Joining DataFrames : Concatenation :

In [13]:
```
df2 = pd.DataFrame({'Name':['Prasant'],'Age':[25],'City':['Markapuram'],
                    'Movie':['Kalki']})
con = pd.concat([df,df2] , ignore_index=True)
print(con)
```

```
      Name   Age        City           Movie
0   Upendra  30.0       Kunta    Raktha kaneru
1       Ram  30.0  Amaravathi  Raktha Charitra
2  Jaganadh  27.0      Guntur       Nenu intha
3   Prasant  25.0  Markapuram            Kalki
```

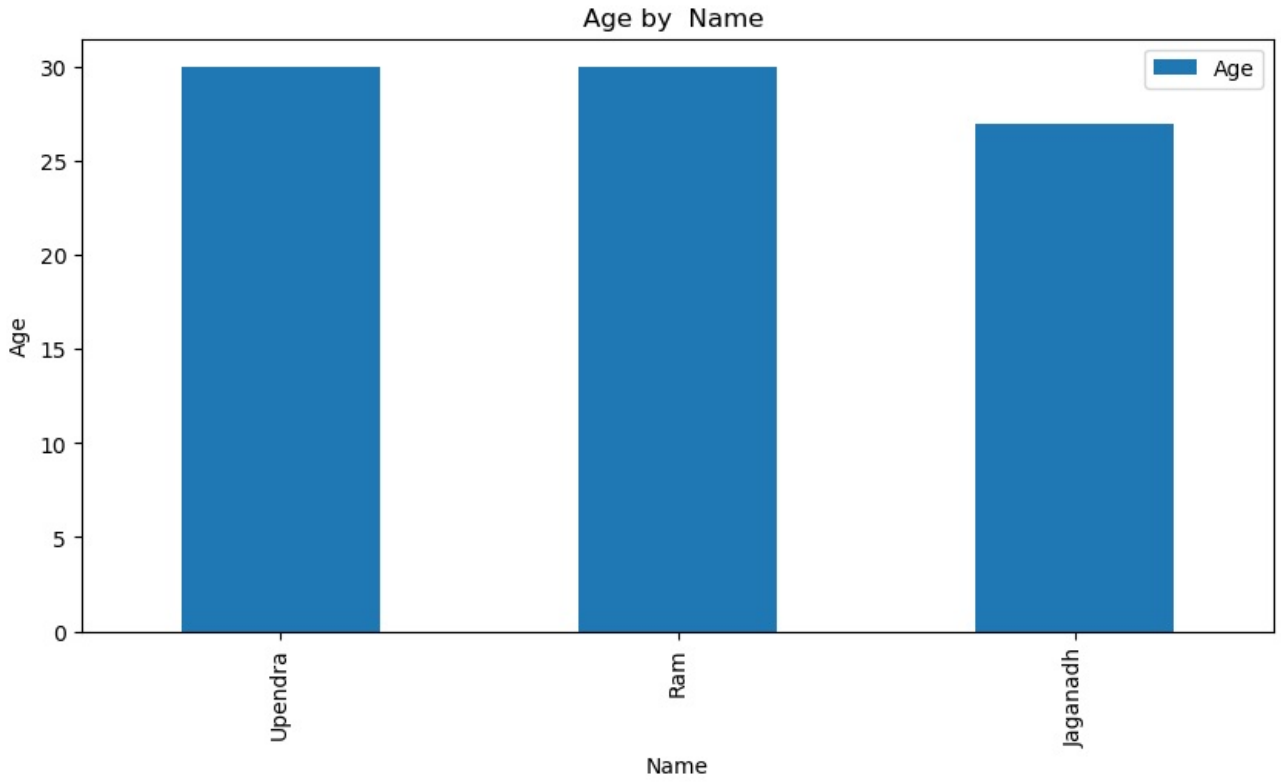
In [ ]:
```
Merging :
```

In [14]:
```
df3 = pd.DataFrame({'Movie':['Raktha kaneru','Raktha Charitra',' Kalki'],
                    'Rating':[4,5,4.7]})
mer = pd.merge(df,df3 , on ='Movie')
print(mer)
```

```
     Name   Age        City           Movie  Rating
0  Upendra  30.0       Kunta    Raktha kaneru     4.0
1      Ram  30.0  Amaravathi  Raktha Charitra     5.0
```

Data Visulization : Basic Plotting : Using Mathplotlib with Pandas for visulization.

In [21]:
```
import matplotlib.pyplot as plt
df.plot(x='Name' , y='Age',kind='bar' , figsize = (10,5))
plt.title('Age by  Name ')
plt.ylabel('Age')
plt.show()
```



In [ ]:
```
Time Series Data
Creating Time Series :
```

In [22]:
```
dates = pd.date_range('2024-10-27' , periods = 4)
time_series = pd.Series([100,200,300,400] , index = dates)
print(time_series)
```

```
2024-10-27    100
2024-10-28    200
2024-10-29    300
2024-10-30    400
Freq: D, dtype: int64
```

In [ ]:
```
Resampling Time Series:
```

In [23]:
```
resam = time_series.resample('2D').sum()
print(resam)
```

```
2024-10-27    300
2024-10-29    700
Freq: 2D, dtype: int64
```

File I/O with Pandas Reading CSV Files :

Writing to CSV Files :

In [25]:
```
df.to_csv('data.csv' , index = False)
```

```
df
```

Out[25]:

	Name	Age	City	Movie
0	Upendra	30.0	Kunta	Raktha kaneru
1	Ram	30.0	Amaravathi	Raktha Charitra
2	Jaganadh	27.0	Guntur	Nenu intha

In [27]:
```
df_csv = pd.read_csv('data.csv')
print(df_csv.head(2))
```

```
      Name   Age        City            Movie
0  Upendra  30.0       Kunta    Raktha kaneru
1      Ram  30.0  Amaravathi  Raktha Charitra
```

In [30]:
```
df_csv.tail(2)
```

Out[30]:

	Name	Age	City	Movie
1	Ram	30.0	Amaravathi	Raktha Charitra
2	Jaganadh	27.0	Guntur	Nenu intha

Applying Functions Using apply()

In [31]:
```
df['Age Before One Movie'] = df['Age'].apply(lambda x : x*0.8)
print(df)
```

```
      Name   Age        City            Movie  Age Before One Movie
0  Upendra  30.0       Kunta    Raktha kaneru                  24.0
1      Ram  30.0  Amaravathi  Raktha Charitra                  24.0
2  Jaganadh  27.0      Guntur       Nenu intha                  21.6
```

String Manipualtion String Operation :

In [34]:
```
df['Name upper'] = df['Name'].str.upper()
df
```

Out[34]:

	Name	Age	City	Movie	Age Before One Movie	Name upper
0	Upendra	30.0	Kunta	Raktha kaneru	24.0	UPENDRA
1	Ram	30.0	Amaravathi	Raktha Charitra	24.0	RAM
2	Jaganadh	27.0	Guntur	Nenu intha	21.6	JAGANADH

String Replacement :

In [35]:
```
df['City'] = df['City'].str.replace('Amaravathi' , 'AMVT')
df
```

Out[35]:

	Name	Age	City	Movie	Age Before One Movie	Name upper
0	Upendra	30.0	Kunta	Raktha kaneru	24.0	UPENDRA
1	Ram	30.0	AMVT	Raktha Charitra	24.0	RAM
2	Jaganadh	27.0	Guntur	Nenu intha	21.6	JAGANADH

Categorical Data Creating Categorical Data :

In [36]:
```
df['City'] = df['City'].astype('category')
print(df['City'].cat.codes)
```

```
0    2
1    0
2    1
dtype: int8
```

Advanced Indexing and Selection MultiIndex :

In [37]:
```
arr = [['A', 'A' , 'B' , 'B'],['one','two','one','two']]
index = pd.MultiIndex.from_arrays(arr , names = ('first','second'))
df_mult = pd.DataFrame({'data':[1,2,3,4]},index=index)
print(df_mult)
```

```
              data
first second
A     one        1
      two        2
B     one        3
      two        4
```

Selecting Data with MultiIndex :

In [38]:
```
print(df_mult.loc['A'])
```

```
       data
second
one       1
two       2
```

Reshaping Data Using melt() :

In [40]:
```
df_melt = df.melt(id_vars = ['Name'] , value_vars = ['Name' , 'Age'],
                  var_name = 'Varible' , value_name = 'value')
df_melt
```

Out[40]:

	Name	Varible	value
0	Upendra	Age	30.0
1	Ram	Age	30.0
2	Jaganadh	Age	27.0

Common Methods and Functions Basic Descriptive Statistics :

In [41]:
```
df.describe()
```

Out[41]:

	Age	Age Before One Movie
count	3.000000	3.000000
mean	29.000000	23.200000
std	1.732051	1.385641
min	27.000000	21.600000
25%	28.500000	22.800000
50%	30.000000	24.000000
75%	30.000000	24.000000
max	30.000000	24.000000

Getting Unique Values :

In [42]:
```
uq_ci = df['City'].unique()
print(uq_ci)
```

```
['Kunta', 'AMVT', 'Guntur']
Categories (3, object): ['AMVT', 'Guntur', 'Kunta']
```

Counting Values :

In [45]:
```
ci_co = df.value_counts()
print(ci_co)
```

```
Name      Age   City    Movie            Age Before One Movie  Name upper
Jaganadh  27.0  Guntur  Nenu intha       21.6                  JAGANADH      1
Ram       30.0  AMVT    Raktha Charitra  24.0                  RAM           1
Upendra   30.0  Kunta   Raktha kaneru    24.0                  UPENDRA       1
Name: count, dtype: int64
```

Advanced Grouping Techinues Using groupby() with Multiple Columns :

In [ ]:
```
#df['Actor'] = ['Upendra' , 'Vivek Obreai' , 'Ravi Teja']
#gr_mu = df.groupby(['Movie' , 'Actor']).mean()
#print(gr_mu)
```

DataFrame Indexing Techniques using .iloc[] for position-based indexing :

In [46]:
```
print(df.iloc[0])
```

```
Name                          Upendra
Age                              30.0
City                            Kunta
Movie                   Raktha kaneru
Age Before One Movie             24.0
Name upper                    UPENDRA
Name: 0, dtype: object
```

using.loc[] for label-based-indexing

In [47]:
```
print(df.loc[0 , 'Name'])
```

```
Upendra
```