Pandas

- is a python library which is used for creating and manipulating dataframes.

In [123]:
```
import pandas as pd

import warnings
warnings.filterwarnings('ignore')
```

Series

In [5]:
```
seri = pd.Series([12,34,45])
seri
```

Out[5]:
```
0    12
1    34
2    45
dtype: int64
```

In [6]:
```
seri[0]
```

Out[6]:
```
12
```

In [7]:
```
seri = pd.Series([12,34,45],index = [23,14,18])
seri
```

Out[7]:
```
23    12
14    34
18    45
dtype: int64
```

In [9]:
```
seri[23]
```

Out[9]:
```
12
```

In [12]:
```
seri = pd.Series([12,34,45],index = ['A','B','C'])
seri
```

Out[12]:
```
A    12
B    34
C    45
dtype: int64
```

In [13]:
```
seri['A']
```

Out[13]:
```
12
```

loc

In [15]:
```
seri.loc['A']     # it takes labeled index
```

Out[15]:
```
12
```

iloc

In [17]:
```
seri.iloc[0]     # it takes default index location
```

Out[17]:
```
12
```

updating a series object

In [19]:
```
seri['A'] = "Updated"
```

In [20]:
```
seri
```

Out[20]:
```
A    Updated
B         34
C         45
dtype: object
```

DataFrames

In [21]:
```
info = {'fname':['Rahul','Shravan','Mohit','Vikram'],
       'lname':['Singh','Kumar','Verma','Vedha'],
       'age':[27,45,33,49]}

info       # dictionary
```

Out[21]:
```
{'fname': ['Rahul', 'Shravan', 'Mohit', 'Vikram'],
 'lname': ['Singh', 'Kumar', 'Verma', 'Vedha'],
 'age': [27, 45, 33, 49]}
```

In [23]:
```
df = pd.DataFrame(info)
df
```

Out[23]:

	fname	lname	age
0	Rahul	Singh	27
1	Shravan	Kumar	45
2	Mohit	Verma	33
3	Vikram	Vedha	49

In [26]:
```
df['fname']
```

Out[26]:
```
0      Rahul
1    Shravan
2      Mohit
3     Vikram
Name: fname, dtype: object
```

loc on dataframe

In [28]:
```
df.loc[1]
```

Out[28]:
```
fname    Shravan
lname      Kumar
age           45
Name: 1, dtype: object
```

iloc in dataframe

In [29]:
```
df.iloc[1]
```

Out[29]:
```
fname    Shravan
lname      Kumar
age           45
Name: 1, dtype: object
```

index

In [30]:
```
df.index
```

Out[30]:
```
RangeIndex(start=0, stop=4, step=1)
```

In [31]:
```
df.index = ['A','B','C','D']     # updating the numberic indexes with charact
```

In [32]:
```
df
```

Out[32]:

	fname	lname	age
A	Rahul	Singh	27
B	Shravan	Kumar	45
C	Mohit	Verma	33
D	Vikram	Vedha	49

In [33]:
```
df.iloc[1]
```

Out[33]:
```
fname    Shravan
lname      Kumar
age           45
Name: B, dtype: object
```

In [35]:
```
df.loc['B']
```

Out[35]:
```
fname    Shravan
lname      Kumar
age           45
Name: B, dtype: object
```

In [38]:
```
df = pd.DataFrame({'A':[2,3,4],'B':[10,11,12]}, index = [10,11,12])
df
```

Out[38]:

	A	B
10	2	10
11	3	11
12	4	12

In [45]:
```
df.iloc[-1]
```

Out[45]:
```
A     4
B    12
Name: 12, dtype: int64
```

read_csv(),read_excel()

In [46]:
```python
pd.read_excel(r"C:\Users\Bhupendra\Downloads\titanic3.xls")
```

Out[46]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	emba
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	
3	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	
...	...	...	...	...	...	...	...	...	...	...	
1304	3	0	Zabour, Miss. Hileni	female	14.5000	1	0	2665	14.4542	NaN	
1305	3	0	Zabour, Miss. Thamine	female	NaN	1	0	2665	14.4542	NaN	
1306	3	0	Zakarian, Mr. Mapriededer	male	26.5000	0	0	2656	7.2250	NaN	
1307	3	0	Zakarian, Mr. Ortin	male	27.0000	0	0	2670	7.2250	NaN	
1308	3	0	Zimmerman, Mr. Leo	male	29.0000	0	0	315082	7.8750	NaN	

1309 rows × 14 columns

In [42]:
```python
print('nest\net')
```

```
nest
et
```

In [43]:
```python
print(r'nest\net')
```

```
nest\net
```

In [51]:
```
df = pd.read_excel(r"C:\Users\Bhupendra\Downloads\titanic3 (1).xls")
```

shape

In [52]:
```
df.shape
```

Out[52]:
```
(1309, 14)
```

columns

In [53]:
```
df.columns
```

Out[53]:
```
Index(['pclass', 'survived', 'name', 'sex', 'age', 'sibsp', 'parch', 'ticket',
       'fare', 'cabin', 'embarked', 'boat', 'body', 'home.dest'],
      dtype='object')
```

head() : top 5 rows by default

In [54]:
```
df.head()
```

Out[54]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S
3	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	S
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	S

In [60]:
```
df.head(10)      # top 2 rows
```

Out[60]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarke
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	
3	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	
5	1	1	Anderson, Mr. Harry	male	48.0000	0	0	19952	26.5500	E12	
6	1	1	Andrews, Miss. Kornelia Theodosia	female	63.0000	1	0	13502	77.9583	D7	
7	1	0	Andrews, Mr. Thomas Jr	male	39.0000	0	0	112050	0.0000	A36	
8	1	1	Appleton, Mrs. Edward Dale (Charlotte Lamson)	female	53.0000	2	0	11769	51.4792	C101	
9	1	0	Artagaveytia, Mr. Ramon	male	71.0000	0	0	PC 17609	49.5042	NaN	

tail() : bottom 5

In [61]:
```
df.tail()    # bottom 5 rows
```

Out[61]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarkec
1304	3	0	Zabour, Miss. Hileni	female	14.5	1	0	2665	14.4542	NaN	C
1305	3	0	Zabour, Miss. Thamine	female	NaN	1	0	2665	14.4542	NaN	C
1306	3	0	Zakarian, Mr. Mapriededer	male	26.5	0	0	2656	7.2250	NaN	C
1307	3	0	Zakarian, Mr. Ortin	male	27.0	0	0	2670	7.2250	NaN	C
1308	3	0	Zimmerman, Mr. Leo	male	29.0	0	0	315082	7.8750	NaN	S

In [62]:
```
df.tail(2)    # bottom 2 rows
```

Out[62]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	b
1307	3	0	Zakarian, Mr. Ortin	male	27.0	0	0	2670	7.225	NaN	C	N
1308	3	0	Zimmerman, Mr. Leo	male	29.0	0	0	315082	7.875	NaN	S	N

Slicing the columns

In [63]:
```python
df[['pclass','survived','name']]
```

Out[63]:

	pclass	survived	name
0	1	1	Allen, Miss. Elisabeth Walton
1	1	1	Allison, Master. Hudson Trevor
2	1	0	Allison, Miss. Helen Loraine
3	1	0	Allison, Mr. Hudson Joshua Creighton
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)
...	...	...	...
1304	3	0	Zabour, Miss. Hileni
1305	3	0	Zabour, Miss. Thamine
1306	3	0	Zakarian, Mr. Mapriededer
1307	3	0	Zakarian, Mr. Ortin
1308	3	0	Zimmerman, Mr. Leo

1309 rows × 3 columns

In [67]:
```python
df.iloc[:,0:3]
```

Out[67]:

	pclass	survived	name
0	1	1	Allen, Miss. Elisabeth Walton
1	1	1	Allison, Master. Hudson Trevor
2	1	0	Allison, Miss. Helen Loraine
3	1	0	Allison, Mr. Hudson Joshua Creighton
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)
...	...	...	...
1304	3	0	Zabour, Miss. Hileni
1305	3	0	Zabour, Miss. Thamine
1306	3	0	Zakarian, Mr. Mapriededer
1307	3	0	Zakarian, Mr. Ortin
1308	3	0	Zimmerman, Mr. Leo

1309 rows × 3 columns

renaming column names

In [68]:
```python
df.columns
```

Out[68]:
```
Index(['pclass', 'survived', 'name', 'sex', 'age', 'sibsp', 'parch', 'ticket',
       'fare', 'cabin', 'embarked', 'boat', 'body', 'home.dest'],
      dtype='object')
```

In [71]:
```
df.rename(columns={'home.dest':'home.destination'}, inplace = True)
```

In [73]:
```
df.columns
```

Out[73]:
```
Index(['pclass', 'survived', 'name', 'sex', 'age', 'sibsp', 'parch', 'ticket',
       'fare', 'cabin', 'embarked', 'boat', 'body', 'home.destination'],
      dtype='object')
```

info()

In [75]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   pclass            1309 non-null   int64  
 1   survived          1309 non-null   int64  
 2   name              1309 non-null   object 
 3   sex               1309 non-null   object 
 4   age               1046 non-null   float64
 5   sibsp             1309 non-null   int64  
 6   parch             1309 non-null   int64  
 7   ticket            1309 non-null   object 
 8   fare              1308 non-null   float64
 9   cabin             295 non-null    object 
 10  embarked          1307 non-null   object 
 11  boat              486 non-null    object 
 12  body              121 non-null    float64
 13  home.destination  745 non-null    object 
dtypes: float64(3), int64(4), object(7)
memory usage: 143.3+ KB
```

In [79]:
```
df.ticket
```

Out[79]:
```
0        24160
1       113781
2       113781
3       113781
4       113781
         ...  
1304      2665
1305      2665
1306      2656
1307      2670
1308    315082
Name: ticket, Length: 1309, dtype: object
```

value_counts()

In [80]:
```
df.ticket.value_counts()
```

Out[80]:
```
CA. 2343        11
CA 2144          8
1601             8
S.O.C. 14879     7
3101295          7
                ..
C 7076           1
341826           1
7546             1
3474             1
315082           1
Name: ticket, Length: 939, dtype: int64
```

In [81]:
```
df.pclass.value_counts()
```

Out[81]:
```
3    709
1    323
2    277
Name: pclass, dtype: int64
```

In [82]:
```
df.pclass.value_counts(ascending = True)
```

Out[82]:
```
2    277
1    323
3    709
Name: pclass, dtype: int64
```

describe()

- considers only numeric columns

In [83]:
```
df.describe()
```

Out[83]:

	pclass	survived	age	sibsp	parch	fare	body
count	1309.000000	1309.000000	1046.000000	1309.000000	1309.000000	1308.000000	121.000000
mean	2.294882	0.381971	29.881135	0.498854	0.385027	33.295479	160.809917
std	0.837836	0.486055	14.413500	1.041658	0.865560	51.758668	97.696922
min	1.000000	0.000000	0.166700	0.000000	0.000000	0.000000	1.000000
25%	2.000000	0.000000	21.000000	0.000000	0.000000	7.895800	72.000000
50%	3.000000	0.000000	28.000000	0.000000	0.000000	14.454200	155.000000
75%	3.000000	1.000000	39.000000	1.000000	0.000000	31.275000	256.000000
max	3.000000	1.000000	80.000000	8.000000	9.000000	512.329200	328.000000

select_dtypes()

- select the columns with specified data type

In [86]:
```
df.select_dtypes('int')
```

Out[86]:

	pclass	survived	sibsp	parch
0	1	1	0	0
1	1	1	1	2
2	1	0	1	2
3	1	0	1	2
4	1	0	1	2
...	...	...	...	...
1304	3	0	1	0
1305	3	0	1	0
1306	3	0	0	0
1307	3	0	0	0
1308	3	0	0	0

1309 rows × 4 columns

In [87]:
```
df.select_dtypes('int').columns    # get the column names only
```

Out[87]:
```
Index(['pclass', 'survived', 'sibsp', 'parch'], dtype='object')
```

In [88]:
```
df.select_dtypes('float').columns
```

Out[88]:
```
Index(['age', 'fare', 'body'], dtype='object')
```

In [89]:
```
df.select_dtypes('object').columns
```

Out[89]:
```
Index(['name', 'sex', 'ticket', 'cabin', 'embarked', 'boat',
       'home.destination'],
      dtype='object')
```

check the null values

isnull()

In [96]:
```
# checking the columns having null values in it

df.isnull().sum()
```

Out[96]:
```
pclass                 0
survived               0
name                   0
sex                    0
age                  263
sibsp                  0
parch                  0
ticket                 0
fare                   1
cabin               1014
embarked               2
boat                 823
body                1188
home.destination     564
dtype: int64
```

In [95]:
```
sum([True, False, True])
```

Out[95]:
```
2
```

In [97]:
```
df.isnull().any()    # return True if it has any null value in the column
```

Out[97]:
```
pclass              False
survived            False
name                False
sex                 False
age                  True
sibsp               False
parch               False
ticket              False
fare                 True
cabin                True
embarked             True
boat                 True
body                 True
home.destination     True
dtype: bool
```

getting the columns having null values in it

In [109]:
```
df.loc[:,df.isnull().any()]     # columns having null values
```

Out[109]:

	age	fare	cabin	embarked	boat	body	home.destination
0	29.0000	211.3375	B5	S	2	NaN	St Louis, MO
1	0.9167	151.5500	C22 C26	S	11	NaN	Montreal, PQ / Chesterville, ON
2	2.0000	151.5500	C22 C26	S	NaN	NaN	Montreal, PQ / Chesterville, ON
3	30.0000	151.5500	C22 C26	S	NaN	135.0	Montreal, PQ / Chesterville, ON
4	25.0000	151.5500	C22 C26	S	NaN	NaN	Montreal, PQ / Chesterville, ON
...	...	...	...	...	...	...	...
1304	14.5000	14.4542	NaN	C	NaN	328.0	NaN
1305	NaN	14.4542	NaN	C	NaN	NaN	NaN
1306	26.5000	7.2250	NaN	C	NaN	304.0	NaN
1307	27.0000	7.2250	NaN	C	NaN	NaN	NaN
1308	29.0000	7.8750	NaN	S	NaN	NaN	NaN

1309 rows × 7 columns

Second Method to get the columns having null values

In [112]:
```
col_names = df.isnull().any().index     # get the column names
```

In [113]:
```
bool_values = df.isnull().any().values     # store the result for each column
```

In [115]:
```
# do boolean indexing

null_cols = col_names[bool_values]      # get the columns where we have True(
```

In [116]:
```
print(null_cols)
```

```
Index(['age', 'fare', 'cabin', 'embarked', 'boat', 'body', 'home.destination'],
dtype='object')
```

In [117]:
```
df[null_cols]        # finally print the dataframe having only null values
```

Out[117]:

	age	fare	cabin	embarked	boat	body	home.destination
0	29.0000	211.3375	B5	S	2	NaN	St Louis, MO
1	0.9167	151.5500	C22 C26	S	11	NaN	Montreal, PQ / Chesterville, ON
2	2.0000	151.5500	C22 C26	S	NaN	NaN	Montreal, PQ / Chesterville, ON
3	30.0000	151.5500	C22 C26	S	NaN	135.0	Montreal, PQ / Chesterville, ON
4	25.0000	151.5500	C22 C26	S	NaN	NaN	Montreal, PQ / Chesterville, ON
...	...	...	...	...	...	...	...
1304	14.5000	14.4542	NaN	C	NaN	328.0	NaN
1305	NaN	14.4542	NaN	C	NaN	NaN	NaN
1306	26.5000	7.2250	NaN	C	NaN	304.0	NaN
1307	27.0000	7.2250	NaN	C	NaN	NaN	NaN
1308	29.0000	7.8750	NaN	S	NaN	NaN	NaN

1309 rows × 7 columns

Handling Missing Values

mean()

In [124]:
```
df.mean()
```

Out[124]:
```
pclass        2.294882
survived      0.381971
age          29.881135
sibsp         0.498854
parch         0.385027
fare         33.295479
body        160.809917
dtype: float64
```

In [127]:
```
df['age'].mean()
```

Out[127]:
```
29.8811345124283
```

median()

In [128]:
```
df.median()
```

Out[128]:
```
pclass        3.0000
survived      0.0000
age          28.0000
sibsp         0.0000
parch         0.0000
fare         14.4542
body        155.0000
dtype: float64
```

In [129]:
```
df['pclass'].median()
```

Out[129]:
```
3.0
```

mode()

In [134]:
```
df['home.destination'].value_counts()
```

Out[134]:
```
New York, NY                                   64
London                                         14
Montreal, PQ                                   10
Paris, France                                   9
Cornwall / Akron, OH                            9
                                               ..
Chelsea, London                                 1
Harrow-on-the-Hill, Middlesex                   1
Copenhagen, Denmark                             1
Guernsey / Montclair, NJ and/or Toledo, Ohio    1
Antwerp, Belgium / Stanton, OH                  1
Name: home.destination, Length: 369, dtype: int64
```

In [135]:
```
df['home.destination'].mode()
```

Out[135]:
```
0    New York, NY
dtype: object
```

In [136]:
```
df.name.value_counts()
```

Out[136]:
```
Connolly, Miss. Kate             2
Kelly, Mr. James                 2
Allen, Miss. Elisabeth Walton    1
Ilmakangas, Miss. Ida Livija     1
Ilieff, Mr. Ylio                 1
                                ..
Hart, Miss. Eva Miriam           1
Harris, Mr. Walter               1
Harris, Mr. George               1
Harper, Rev. John                1
Zimmerman, Mr. Leo               1
Name: name, Length: 1307, dtype: int64
```

In [141]:
```
df.name.mode()
```

Out[141]:
```
0    Connolly, Miss. Kate
1        Kelly, Mr. James
dtype: object
```

In [142]:
```
df.name.mode()[0]
```

Out[142]:
```
'Connolly, Miss. Kate'
```

Dropping missing values

In [143]:
```
df.isnull().sum()
```

Out[143]:
```
pclass                 0
survived               0
name                   0
sex                    0
age                  263
sibsp                  0
parch                  0
ticket                 0
fare                   1
cabin               1014
embarked               2
boat                 823
body                1188
home.destination     564
dtype: int64
```

In [147]:
```
# droping columns 'cabin' & 'body' as they have higher missing values

df.drop(['cabin','body'], axis = 1, inplace = True)    # or you can write, ax
```

In [150]:
```
def check_nulls():
    return df.isnull().sum()
```

In [151]:
```
check_nulls()
```

Out[151]:
```
pclass               0
survived             0
name                 0
sex                  0
age                263
sibsp                0
parch                0
ticket               0
fare                 1
embarked             2
boat               823
home.destination   564
dtype: int64
```

In [154]:
```
# check 'fare' column

df[df.fare.isnull()]       # boolean indexing
```

Out[154]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked	boat	home.des
1225	3	0	Storey, Mr. Thomas	male	60.5	0	0	3701	NaN	S	NaN	

In [157]:
```
df.drop(1225, inplace = True)     # drop the row at index 1225; by default ax
```

In [163]:
```
df[df.embarked.isnull()]
```

Out[163]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked	boat	home.de
168	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	NaN	6	
284	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	NaN	6	Cin

In [164]:
```
df.drop([168,284], inplace = True)
```

In [166]:
```
df.shape
```

Out[166]:
```
(1306, 12)
```

In [168]:
```
check_nulls()
```

Out[168]:
```
pclass                0
survived              0
name                  0
sex                   0
age                 263
sibsp                 0
parch                 0
ticket                0
fare                  0
embarked              0
boat                822
home.destination    562
dtype: int64
```

Imputing the missing values

fillna()

In [173]:
```
df.age = df.age.fillna(df.age.mean())
```

In [174]:
```
check_nulls()
```

Out[174]:
```
pclass                0
survived              0
name                  0
sex                   0
age                   0
sibsp                 0
parch                 0
ticket                0
fare                  0
embarked              0
boat                822
home.destination    562
dtype: int64
```

In [179]:
```
df.drop(['boat','home.destination'], axis = 1, inplace = True)
```

In [180]:
```
df.shape
```

Out[180]:
```
(1306, 10)
```

Changing data types of column

In [182]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1306 entries, 0 to 1308
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   pclass    1306 non-null   int64  
 1   survived  1306 non-null   int64  
 2   name      1306 non-null   object 
 3   sex       1306 non-null   object 
 4   age       1306 non-null   float64
 5   sibsp     1306 non-null   int64  
 6   parch     1306 non-null   int64  
 7   ticket    1306 non-null   object 
 8   fare      1306 non-null   float64
 9   embarked  1306 non-null   object 
dtypes: float64(2), int64(4), object(4)
memory usage: 112.2+ KB
```

In [215]:
```
import re
int_ticket = [                     ]

for i in df.ticket:

    try:
        match = int("".join(re.findall("[0-9]",str(i))))

    except:
        match = 0

    int_ticket.append(match)
```

In [217]:
```
df.ticket = int_ticket
```

In [218]:
```
df.ticket
```

Out[218]:
```
0        24160
1       113781
2       113781
3       113781
4       113781
         ...  
1304      2665
1305      2665
1306      2656
1307      2670
1308    315082
Name: ticket, Length: 1306, dtype: int64
```

astype()

In [221]:
```
df.survived.astype('float')
```

Out[221]:
```
0       1.0
1       1.0
2       0.0
3       0.0
4       0.0
       ...
1304    0.0
1305    0.0
1306    0.0
1307    0.0
1308    0.0
Name: survived, Length: 1306, dtype: float64
```

update some value directly through index

In [225]:
```
df.head()
```

Out[225]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	S
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	S
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	S
3	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	S
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	S

In [226]:
```
df.iloc[1,6] = 'abcd'
```

In [227]:
```
df.head()
```

Out[227]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	S
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	abcd	113781	151.5500	S
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	S
3	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	S
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	S

pd.to_numeric()

In [234]:
```
pd.to_numeric(df.parch, errors = 'coerce')
```

Out[234]:
```
0       0.0
1       NaN
2       2.0
3       2.0
4       2.0
       ...
1304    0.0
1305    0.0
1306    0.0
1307    0.0
1308    0.0
Name: parch, Length: 1306, dtype: float64
```

map

In [237]:
```
df.pclass = df.pclass.map({1:'Upper',2:"Middle",3:"Lower"})
```

In [238]:
```
df
```

Out[238]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
0	Upper	1	Allen, Miss. Elisabeth Walton	female	29.000000	0	0	24160	211.3375	S
1	Upper	1	Allison, Master. Hudson Trevor	male	0.916700	1	abcd	113781	151.5500	S
2	Upper	0	Allison, Miss. Helen Loraine	female	2.000000	1	2	113781	151.5500	S
3	Upper	0	Allison, Mr. Hudson Joshua Creighton	male	30.000000	1	2	113781	151.5500	S
4	Upper	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.000000	1	2	113781	151.5500	S
...	...	...	...	...	...	...	...	...	...	...
1304	Lower	0	Zabour, Miss. Hileni	female	14.500000	1	0	2665	14.4542	C
1305	Lower	0	Zabour, Miss. Thamine	female	29.813199	1	0	2665	14.4542	C
1306	Lower	0	Zakarian, Mr. Mapriededer	male	26.500000	0	0	2656	7.2250	C
1307	Lower	0	Zakarian, Mr. Ortin	male	27.000000	0	0	2670	7.2250	C
1308	Lower	0	Zimmerman, Mr. Leo	male	29.000000	0	0	315082	7.8750	S

1306 rows × 10 columns

sample()

In [240]:
```
df.sample(5)   # in each run it will select any random 5 rows from the dataf
```

Out[240]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
399	Middle	0	Drew, Mr. James Vivian	male	42.000000	1	1	28220	32.5000	S
440	Middle	1	Herman, Mrs. Samuel (Jane Laver)	female	48.000000	1	2	220845	65.0000	S
812	Lower	0	Fox, Mr. Patrick	male	29.813199	0	0	368573	7.7500	Q
650	Lower	0	Attalah, Miss. Malake	female	17.000000	0	0	2627	14.4583	C
1044	Lower	1	Murphy, Miss. Nora	female	29.813199	0	0	36568	15.5000	Q

In [244]:
```
df.sample(5, random_state = 10)    # fixing the randomness; in each run it w
```

Out[244]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
444	Middle	0	Hickman, Mr. Stanley George	male	21.000000	2	0	14879	73.5000	S
376	Middle	1	Collett, Mr. Sidney C Stuart	male	24.000000	0	0	28034	10.5000	S
1299	Lower	0	Yasbeck, Mr. Antoni	male	27.000000	1	0	2659	14.4542	C
350	Middle	1	Brown, Miss. Edith Eileen	female	15.000000	0	2	29750	39.0000	S
1193	Lower	0	Scanlan, Mr. James	male	29.813199	0	0	36209	7.7250	Q

nlargest()

In [246]:
```
df.nlargest(5,'ticket')
```

Out[246]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
1117	Lower	0	Pekoniemi, Mr. Edvard	male	21.0	0	0	23101294	7.925	S
1248	Lower	0	Tikkanen, Mr. Juho	male	32.0	0	0	23101293	7.925	S
959	Lower	0	Leinonen, Mr. Antti Gustaf	male	32.0	0	0	23101292	7.925	S
1118	Lower	0	Peltomaki, Mr. Nikolai Johannes	male	25.0	0	0	23101291	7.925	S
861	Lower	0	Heininen, Miss. Wendla Maria	female	23.0	0	0	23101290	7.925	S

In [248]:
```
df.nlargest(5,['age','ticket'])
```

Out[248]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
14	Upper	1	Barkworth, Mr. Algernon Henry Wilson	male	80.0	0	0	27042	30.0000	S
61	Upper	1	Cavendish, Mrs. Tyrell William (Julia Florence...	female	76.0	1	0	19877	78.8500	S
1235	Lower	0	Svensson, Mr. Johan	male	74.0	0	0	347060	7.7750	S
135	Upper	0	Goldschmidt, Mr. George B	male	71.0	0	0	17754	34.6542	C
9	Upper	0	Artagaveytia, Mr. Ramon	male	71.0	0	0	17609	49.5042	C

nsmallest()

In [249]:
```
df.nsmallest(5,['age','ticket'])
```

Out[249]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
763	Lower	1	Dean, Miss. Elizabeth Gladys "Millvina"	female	0.1667	1	2	2315	20.5750	S
747	Lower	0	Danbom, Master. Gilbert Sigvard Emanuel	male	0.3333	0	2	347080	14.4000	S
1240	Lower	1	Thomas, Master. Assad Alexander	male	0.4167	0	1	2625	8.5167	C
427	Middle	1	Hamalainen, Master. Viljo	male	0.6667	1	1	250649	14.5000	S
657	Lower	1	Baclini, Miss. Eugenie	female	0.7500	2	1	2666	19.2583	C

sort_values()

In [252]:
```
df.sort_values(by = ['age']).head()
```

Out[252]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
763	Lower	1	Dean, Miss. Elizabeth Gladys "Millvina"	female	0.1667	1	2	2315	20.5750	S
747	Lower	0	Danbom, Master. Gilbert Sigvard Emanuel	male	0.3333	0	2	347080	14.4000	S
1240	Lower	1	Thomas, Master. Assad Alexander	male	0.4167	0	1	2625	8.5167	C
427	Middle	1	Hamalainen, Master. Viljo	male	0.6667	1	1	250649	14.5000	S
657	Lower	1	Baclini, Miss. Eugenie	female	0.7500	2	1	2666	19.2583	C

In [254]:
```
df.sort_values(by = ['age'], ascending = False).head()
```

Out[254]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
14	Upper	1	Barkworth, Mr. Algernon Henry Wilson	male	80.0	0	0	27042	30.0000	S
61	Upper	1	Cavendish, Mrs. Tyrell William (Julia Florence...	female	76.0	1	0	19877	78.8500	S
1235	Lower	0	Svensson, Mr. Johan	male	74.0	0	0	347060	7.7750	S
135	Upper	0	Goldschmidt, Mr. George B	male	71.0	0	0	17754	34.6542	C
9	Upper	0	Artagaveytia, Mr. Ramon	male	71.0	0	0	17609	49.5042	C

duplicated()

In [257]:
```
df.duplicated().sum()    # there is no duplicated rows in dataframe
```

Out[257]:
```
0
```

drop_duplicates()

In [259]:
```
df.drop_duplicates(keep = 'first')    # it will drop all the duplicate rows
```

Out[259]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
0	Upper	1	Allen, Miss. Elisabeth Walton	female	29.000000	0	0	24160	211.3375	S
1	Upper	1	Allison, Master. Hudson Trevor	male	0.916700	1	abcd	113781	151.5500	S
2	Upper	0	Allison, Miss. Helen Loraine	female	2.000000	1	2	113781	151.5500	S
3	Upper	0	Allison, Mr. Hudson Joshua Creighton	male	30.000000	1	2	113781	151.5500	S
4	Upper	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.000000	1	2	113781	151.5500	S
...	...	...	...	...	...	...	...	...	...	...
1304	Lower	0	Zabour, Miss. Hileni	female	14.500000	1	0	2665	14.4542	C
1305	Lower	0	Zabour, Miss. Thamine	female	29.813199	1	0	2665	14.4542	C
1306	Lower	0	Zakarian, Mr. Mapriededer	male	26.500000	0	0	2656	7.2250	C
1307	Lower	0	Zakarian, Mr. Ortin	male	27.000000	0	0	2670	7.2250	C
1308	Lower	0	Zimmerman, Mr. Leo	male	29.000000	0	0	315082	7.8750	S

1306 rows × 10 columns

In [261]:
```
df
```

Out[261]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	embarked
0	Upper	1	Allen, Miss. Elisabeth Walton	female	29.000000	0	0	24160	211.3375	S
1	Upper	1	Allison, Master. Hudson Trevor	male	0.916700	1	abcd	113781	151.5500	S
2	Upper	0	Allison, Miss. Helen Loraine	female	2.000000	1	2	113781	151.5500	S
3	Upper	0	Allison, Mr. Hudson Joshua Creighton	male	30.000000	1	2	113781	151.5500	S
4	Upper	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.000000	1	2	113781	151.5500	S
...	...	...	...	...	...	...	...	...	...	...
1304	Lower	0	Zabour, Miss. Hileni	female	14.500000	1	0	2665	14.4542	C
1305	Lower	0	Zabour, Miss. Thamine	female	29.813199	1	0	2665	14.4542	C
1306	Lower	0	Zakarian, Mr. Mapriededer	male	26.500000	0	0	2656	7.2250	C
1307	Lower	0	Zakarian, Mr. Ortin	male	27.000000	0	0	2670	7.2250	C
1308	Lower	0	Zimmerman, Mr. Leo	male	29.000000	0	0	315082	7.8750	S

1306 rows × 10 columns

Exporting the dataframe as excel or csv file

In [ ]:
```
!pip install xlwt    # run this cell incase you get error saying xlwt not fo
```

In [263]:
```
df.to_excel("new_titanic.xls")    # exporting the dataframe in excel format
```

In [272]:
```
df1 = pd.DataFrame({"A":[1,2,3,5],
                    "B":[11,12,13,14]})

df2 = pd.DataFrame({"C":[1,2,3,6],
                    "D":[11,12,13,14]})

df1
```

Out[272]:

	A	B
0	1	11
1	2	12
2	3	13
3	5	14

In [273]:
```
df2
```

Out[273]:

	C	D
0	1	11
1	2	12
2	3	13
3	6	14

Merging two dataframes

In [279]:
```
# inner
df1.merge(df2,how ='inner',left_on='A',right_on='C')     # common column is A
```

Out[279]:

	A	B	C	D
0	1	11	1	11
1	2	12	2	12
2	3	13	3	13

In [276]:
```
df1.merge(df2,left_on='A',right_on='D')     # A & D are not common hence no
```

Out[276]:

A	B	C	D

In [278]:
```
# left join
df1.merge(df2, how = "left", left_on='A',right_on='C')
```

Out[278]:

	A	B	C	D
0	1	11	1.0	11.0
1	2	12	2.0	12.0
2	3	13	3.0	13.0
3	5	14	NaN	NaN

In [280]:
```
# right join

df1.merge(df2,how = "right", left_on='A',right_on='C')
```

Out[280]:

	A	B	C	D
0	1.0	11.0	1	11
1	2.0	12.0	2	12
2	3.0	13.0	3	13
3	NaN	NaN	6	14

In [283]:
```
# cross join

df1.merge(df2,how = "cross")
```

Out[283]:

	A	B	C	D
0	1	11	1	11
1	1	11	2	12
2	1	11	3	13
3	1	11	6	14
4	2	12	1	11
5	2	12	2	12
6	2	12	3	13
7	2	12	6	14
8	3	13	1	11
9	3	13	2	12
10	3	13	3	13
11	3	13	6	14
12	5	14	1	11
13	5	14	2	12
14	5	14	3	13
15	5	14	6	14

concat

In [287]:

```
df1 = pd.DataFrame({"A":[1,2,3,5],
                    "B":[11,12,13,14]})

df2 = pd.DataFrame({"C":[1,2,3,6],
                    "D":[11,12,13,14]})


pd.concat([df1,df2], axis = 0)   # by default axis = 0
```

Out[287]:

	A	B	C	D
0	1.0	11.0	NaN	NaN
1	2.0	12.0	NaN	NaN
2	3.0	13.0	NaN	NaN
3	5.0	14.0	NaN	NaN
0	NaN	NaN	1.0	11.0
1	NaN	NaN	2.0	12.0
2	NaN	NaN	3.0	13.0
3	NaN	NaN	6.0	14.0

In [286]:

```
df1 = pd.DataFrame({"A":[1,2,3,5],
                    "B":[11,12,13,14]})

df2 = pd.DataFrame({"C":[1,2,3,6],
                    "D":[11,12,13,14]})


pd.concat([df1,df2], axis = 1)
```

Out[286]:

	A	B	C	D
0	1	11	1	11
1	2	12	2	12
2	3	13	3	13
3	5	14	6	14

In [285]:
```python
df1 = pd.DataFrame({"A":[1,2,3,5],
                    "B":[11,12,13,14]})

df2 = pd.DataFrame({"A":[1,2,3,6],
                    "B":[11,12,13,14]})


pd.concat([df1,df2])
```

Out[285]:

	A	B
0	1	11
1	2	12
2	3	13
3	5	14
0	1	11
1	2	12
2	3	13
3	6	14

In [288]:
```python
df1 = pd.DataFrame({"A":[1,2,3,5],
                    "B":[11,12,13,14]})

df2 = pd.DataFrame({"A":[1,2,3,6],
                    "C":[11,12,13,14]})


pd.concat([df1,df2])
```

Out[288]:

	A	B	C
0	1	11.0	NaN
1	2	12.0	NaN
2	3	13.0	NaN
3	5	14.0	NaN
0	1	NaN	11.0
1	2	NaN	12.0
2	3	NaN	13.0
3	6	NaN	14.0

groupby()

- group the dataframe based on certain columns

In [289]:
```
df = pd.DataFrame({'name':['A','B','C','D'],
                   'subject':['w','x','x','z'],
                   'marks':[56,78,24,67]})

df
```

Out[289]:

	name	subject	marks
0	A	w	56
1	B	x	78
2	C	x	24
3	D	z	67

In [292]:
```
df.groupby(by = 'subject')['marks'].max()    # maximum marks in each group
```

Out[292]:
```
subject
w    56
x    78
z    67
Name: marks, dtype: int64
```

In [293]:
```
df.groupby(by = 'subject')['marks'].min()    # min marks in each group
```

Out[293]:
```
subject
w    56
x    24
z    67
Name: marks, dtype: int64
```

In [296]:
```
df.groupby(by = 'subject')['name','marks'].min()
```

Out[296]:

	name	marks
subject		
w	A	56
x	B	24
z	D	67

In [297]:
```
df.groupby(by = 'subject').min()
```

Out[297]:

	name	marks
subject		
w	A	56
x	B	24
z	D	67

insert a column in dataframe

In [298]:
```
df = pd.DataFrame({'name':['A','B','C','D'],
                   'subject':['w','x','x','z'],
                   'marks':[56,78,24,67]})

df
```

Out[298]:

	name	subject	marks
0	A	w	56
1	B	x	78
2	C	x	24
3	D	z	67

In [307]:
```
age = [26,45,23,18]
df.insert(1,"age",age)
```

In [308]:
```
df
```

Out[308]:

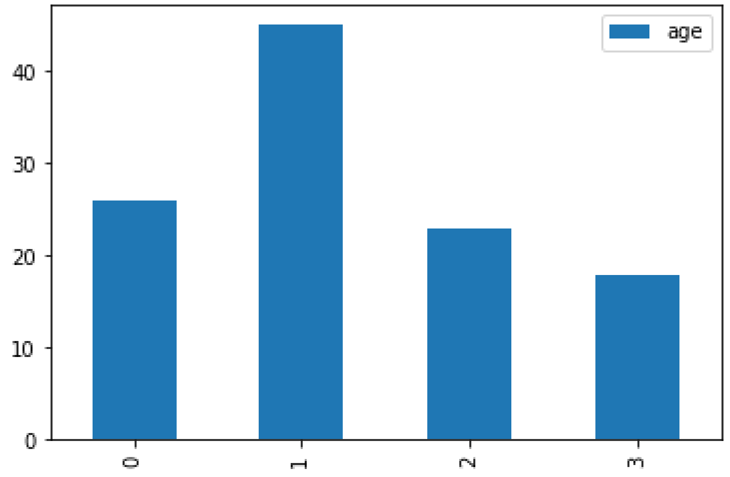
	name	age	subject	marks
0	A	26	w	56
1	B	45	x	78
2	C	23	x	24
3	D	18	z	67

Pandas Plots

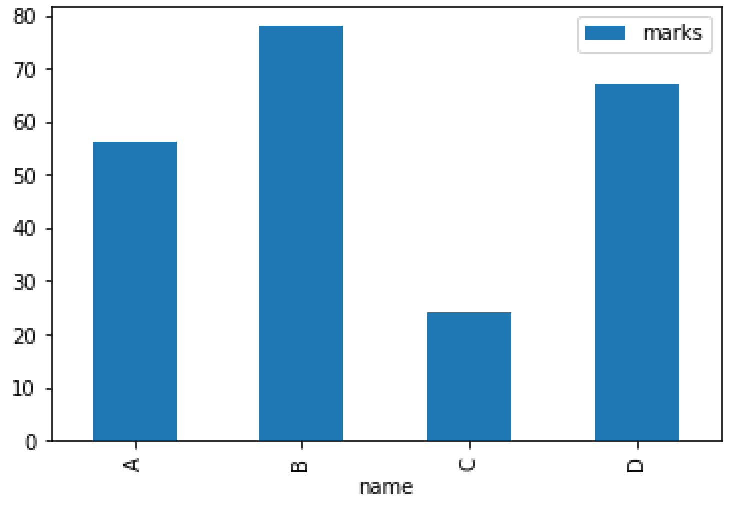
```
- 'line' : line plot (default)
- 'bar' : vertical bar plot
- 'barh' : horizontal bar plot
- 'hist' : histogram
- 'box' : boxplot
- 'kde' : Kernel Density Estimation plot
- 'density' : same as 'kde'
- 'area' : area plot
- 'pie' : pie plot
- 'scatter' : scatter plot (DataFrame only)
- 'hexbin' : hexbin plot (DataFrame only)
```

Vertical Bar

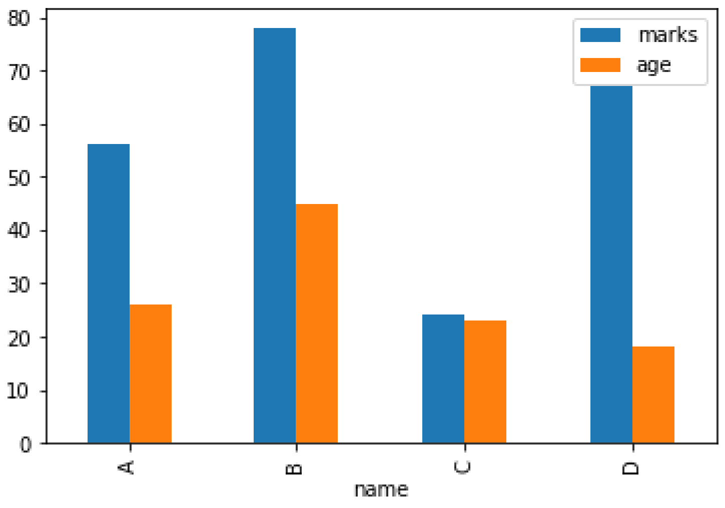
In [322]:
```
df[["name","age"]].plot(kind='bar');
```



In [327]:
```
df.plot(kind='bar',x = 'name',y='marks');
```

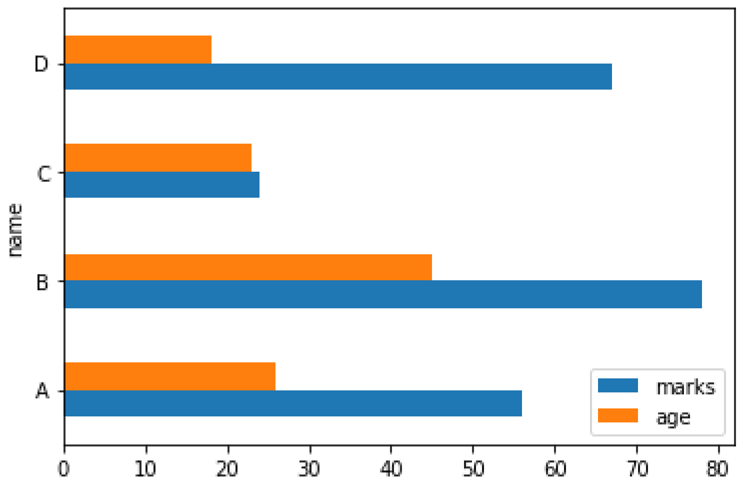


In [331]:
```
df.plot(kind='bar',x = 'name',y=['marks','age']);
```



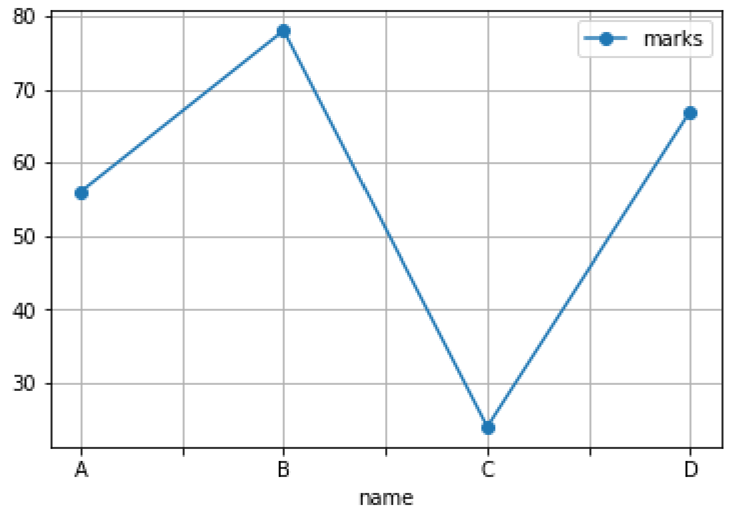
Horizontal Bar

In [336]:
```
df.plot(kind='barh',x = 'name',y=['marks','age']);
```



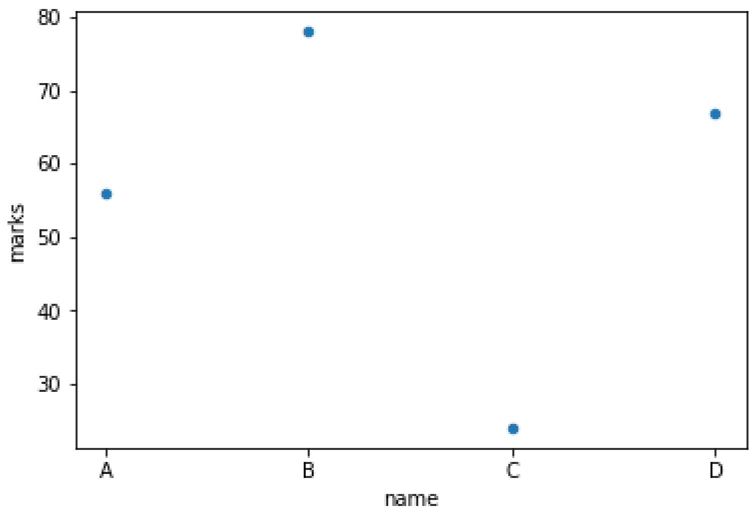
Line

In [335]:
```
df.plot(kind='line',x = 'name',y='marks',grid = True,marker='o');
```



Scatter Plot

In [341]:
```
df.plot(kind='scatter',x = 'name',y='marks');
```



GEOPANDAS : next good thing to explore!

Do some Goooggling!

END

https://geopandas.org/en/stable/getting_started/introduction.html
(https://geopandas.org/en/stable/getting_started/introduction.html)

In [ ]:
```

```