Developed by: Aman Singh || Register no.: 212224040020

```
import pandas as pd
import numpy as np
```

```
df= pd.read_csv("/content/SAMPLEIDS.csv")
df.head()
```

	SNO	REGNO	NAME	DOB	GENDER	ADDRESS	M1	M2	M3	M4	TOTAL	
0	1	1220121	ARUN	2000-02-10	MALE	THANDALAM	82.0	81.0	90.0	NaN	NaN	
1	2	1220122	BABU	1999-01-25	MALE	KANCHIPURAM	56.0	61.0	80.0	56.0	253.0	84.3
2	3	1220123	CHARAN	2000.09.21	MALE	THANDALAM	NaN	59.0	60.0	70.0	NaN	0.0

```
df.isnull()
```

	SNO	REGNO	NAME	DOB	GENDER	ADDRESS	M1	M2	M3	M4	TOTAL	AVG
0	False	False	False	False	False	False	False	False	False	True	True	True
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	True	False	False	False	True	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False	True	True	False	True	False
10	False	False	False	False	False	False	False	False	False	False	False	False
11	False	False	False	False	False	False	False	False	False	False	False	False
12	False	False	False	False	False	False	True	False	False	False	False	False
13	False	False	False	False	False	False	False	False	True	False	True	False
14	False	False	False	False	False	False	False	False	False	False	False	False
15	False	False	True	False	True	True	True	True	True	True	False	False
16	False	False	False	False	False	False	False	False	False	False	False	False
17	False	False	False	False	False	False	False	False	False	True	False	False
18	False	False	False	False	False	False	False	False	False	False	False	False
19	False	False	False	False	False	False	False	False	True	False	True	False
20	False	False	False	False	False	False	False	False	False	False	False	False

```python
df.fillna(method='ffill')
```

```
<ipython-input-47-5c0beae7dc1e>:1: FutureWarning: DataFrame.fillna with 'method' is depr
  df.fillna(method='ffill')
```

	SNO	REGNO	NAME	DOB	GENDER	ADDRESS	M1	M2	M3	M4	TOTAL	
0	1	1220121	ARUN	2000-02-10	MALE	THANDALAM	82.0	81.0	90.0	NaN	NaN	
1	2	1220122	BABU	1999-01-25	MALE	KANCHIPURAM	56.0	61.0	80.0	56.0	253.0	8
2	3	1220123	CHARAN	2000.09.21	MALE	THANDALAM	56.0	59.0	60.0	70.0	253.0	
3	4	1220124	DEVA	2000-11-09	MALE	POONAMALEE	74.0	79.0	80.0	74.0	307.0	10
4	5	1220125	ESTER	2000-11-21	FEMALE	CHITHUR	92.0	95.0	96.0	92.0	375.0	12
5	6	1220126	FARHANA	1999-03-05	FEMALE	THANDALAM	91.0	88.0	90.0	91.0	360.0	12
6	7	1220127	GANI	2000-10-02	MALE	KANCHIPURAM	49.0	51.0	70.0	49.0	219.0	7
7	7	1220127	GANI	2000-10-02	MALE	KANCHIPURAM	49.0	51.0	70.0	49.0	219.0	7
8	8	1220128	HEMA	1999-01-25	FEMALE	POONAMALEE	95.0	96.0	90.0	95.0	376.0	12
9	9	1220129	INDRA	2000.09.21	FEMALE	KANCHIPURAM	64.0	96.0	90.0	64.0	376.0	
10	10	1220130	JAHITH	2000-11-09	MALE	THANDALAM	34.0	45.0	50.0	34.0	163.0	5
11	11	1220131	KANI	2000-11-21	FEMALE	CHITHUR	96.0	95.0	96.0	96.0	383.0	12
12	12	1220132	LATHESSH	1999-03-05	MALE	THANDALAM	96.0	68.0	70.0	70.0	208.0	6
13	13	1220133	MANI	2000-10-02	MALE	KANCHIPURAM	71.0	76.0	70.0	71.0	208.0	
14	14	1220134	NANI	20001109	MALE	POONAMALEE	79.0	77.0	80.0	79.0	315.0	10

```python
df.describe()
```

	SNO	REGNO	M1	M2	M3	M4	TOTAL	AVG
count	21.000000	2.100000e+01	18.000000	19.000000	17.000000	18.000000	16.000000	20.000000
mean	10.333333	1.220130e+06	73.666667	74.315789	79.529412	73.166667	272.750000	72.733333
std	5.816643	5.816643e+00	17.580069	15.836149	13.010177	17.426315	102.048681	48.017127
min	1.000000	1.220121e+06	34.000000	45.000000	50.000000	34.000000	0.000000	0.000000
25%	6.000000	1.220126e+06	64.750000	62.500000	70.000000	65.500000	216.250000	40.750000
50%	10.000000	1.220130e+06	77.500000	77.000000	80.000000	75.000000	304.000000	78.666667
75%	15.000000	1.220135e+06	85.500000	86.500000	90.000000	85.500000	349.500000	113.333333
max	20.000000	1.220140e+06	96.000000	96.000000	96.000000	96.000000	383.000000	127.666667

SORTING by Label- pandas dataframe

```
df = pd.DataFrame({'col1':[1,2,3,4],'col2':[444,555,666,444],
            'col3':['abc','def','ghi','xyz']})
```

```
df.sort_values(by='col2')
```

	col1	col2	col3
0	1	444	abc
3	4	444	xyz
1	2	555	def
2	3	666	ghi

Sorting by label: - panada dataframe

```
df = pd.DataFrame(np.random.randn(10,2),
        index=[1,4,6,2,3,5,9,8,0,7],columns = ['col2','col1'])
df
```

	col2	col1
1	-1.505850	-1.421886
4	-0.212018	0.061502
6	-0.268462	-1.671703
2	-0.250026	-2.441124
3	-0.357959	0.019720
5	0.399652	-0.569949
9	-0.065985	-0.352390
8	1.321902	-1.071896
0	-1.146706	1.145248
7	-0.998230	-0.561377

```
df1=df.sort_index(ascending=False)
print(df1)
df2=df.sort_index()
print(df2)
```

```
       col2      col1
9 -0.065985 -0.352390
8  1.321902 -1.071896
7 -0.998230 -0.561377
6 -0.268462 -1.671703
5  0.399652 -0.569949
4 -0.212018  0.061502
3 -0.357959  0.019720
2 -0.250026 -2.441124
1 -1.505850 -1.421886
0 -1.146706  1.145248
```

```
       col2      col1
0 -1.146706  1.145248
1 -1.505850 -1.421886
2 -0.250026 -2.441124
3 -0.357959  0.019720
4 -0.212018  0.061502
5  0.399652 -0.569949
6 -0.268462 -1.671703
7 -0.998230 -0.561377
8  1.321902 -1.071896
9 -0.065985 -0.352390
```

GROUPING of DATAFRAME using python pandas

```
data = {'Company':['GOOG','GOOG','MSFT','MSFT','FB','FB'],
       'Person':['Sam','Charlie','Amy','Vanessa','Carl','Sarah'],
       'Sales':[200,120,340,124,243,350]}
df = pd.DataFrame(data)
df
```

	Company	Person	Sales
0	GOOG	Sam	200
1	GOOG	Charlie	120
2	MSFT	Amy	340
3	MSFT	Vanessa	124
4	FB	Carl	243
5	FB	Sarah	350

```
df.groupby('Company')['Sales'].mean()
```

	Sales
Company	
FB	296.5
GOOG	160.0
MSFT	232.0

dtype: float64

```
df.groupby('Company')['Sales'].std()
```

	Sales
Company	
FB	75.660426
GOOG	56.568542
MSFT	152.735065

dtype: float64

```
df.groupby('Company')['Sales'].min()
```

	Sales
Company	
FB	243
GOOG	120
MSFT	124

dtype: int64

```
import pandas as pd
data = {
    'Name': ['John', 'Sarah', 'Mike', 'Emily', 'David'],
    'Age': [25, 31, 29, 35, 27],
    'Gender': ['M', 'F', 'M', 'F', 'M'],
    'Salary': [50000, 70000, 60000, 80000, 55000]
}
df = pd.DataFrame(data)
print(df.head(3))
```

```
    Name  Age Gender  Salary
0   John   25      M   50000
1  Sarah   31      F   70000
2   Mike   29      M   60000
```

```
import pandas as pd
data = {
'Name': ['John','Sarah', 'Mike', 'Emily', 'David'],
'Age': [25, 31, 29, 35, 27],
'Gender': ['M', 'F', 'M','F','M'],
'Salary': [50000, 70000, 60000, 80000, 55000]
}
df = pd. DataFrame(data)
print(df.tail(3))
```

```
    Name  Age Gender  Salary
2   Mike   29      M   60000
3  Emily   35      F   80000
4  David   27      M   55000
```

```


import pandas as pd;
data = {
    'Name': ['John','Sarah','Mike', 'Emily', 'David'],
    'Age': [25, 31, 29, 35, 27],
    'Gender': ['M','F','M','F','M'],
    'Salary': [50000, 70000, 60000, 80000, 55000]
    }
df = pd.DataFrame(data)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Name    5 non-null      object
 1   Age     5 non-null      int64 
 2   Gender  5 non-null      object
 3   Salary  5 non-null      int64 
dtypes: int64(2), object(2)
memory usage: 292.0+ bytes
```

```
data = {
    'Name': ['John','Sarah','Mike', 'Emily', 'David'],
    'Age': [25, 31, 29, 35, 27],
    'Gender': ['M','F','M', 'F','M'],
    'Salary': [50000, 70000, 60000, 80000, 55000]
}
df = pd.DataFrame (data)
print(df.describe())
```

```
             Age        Salary
count   5.000000      5.000000
mean   29.400000  63000.000000
std     3.847077  12041.594579
min    25.000000  50000.000000
25%    27.000000  55000.000000
50%    29.000000  60000.000000
75%    31.000000  70000.000000
max    35.000000  80000.000000
```

```
data = {'name': ['Alice', 'Bob', 'Charlie', 'Dave', 'Emily', 'Frank'],
        'gender' : ['F', 'M', 'M', 'M', 'F', 'M'],
          'age': [25, 35, 40, 28, 30, 45],
        'salary': [50000, 70000, 60000, 80000, 65000, 90000]}
df = pd.DataFrame(data)
grouped = df.groupby('gender')['salary'].mean()
print(grouped)
```

```
gender
F    57500.0
M    75000.0
Name: salary, dtype: float64
```

Double-click (or enter) to edit

```
data = {'name': ['Alice', 'Bob', 'Charlie', 'Dave', 'Emily', 'Frank'],
      'gender': ['F','M', 'M', 'M', 'F', 'M'],
        'age': [25, 35, 40, 28, 30, 45],
        'salary': [50000, 70000, 60000, 80000, 65000, 900001]}
df = pd.DataFrame(data)
grouped = df.groupby('gender').count()
print (grouped)
```

```
        name  age  salary
gender
```

```
F          2   2       2
M          4   4       4
```

```
import pandas as pd
# Create a sample DataFrame with missing values
data = {'Name': ['Alice', 'Bob', 'Charlie', 'Dave', 'Eve'],
'Age': [25, 32, None, 41, 28],
'Salary': [50000, None, 70000, 90000, 60000]}
df = pd.DataFrame (data)
df
```

	Name	Age	Salary
0	Alice	25.0	50000.0
1	Bob	32.0	NaN
2	Charlie	NaN	70000.0
3	Dave	41.0	90000.0
4	Eve	28.0	60000.0

```
df_cleaned = df.dropna(subset=['Salary'])
print(df_cleaned)
```

```
      Name   Age   Salary
0    Alice  25.0  50000.0
2  Charlie   NaN  70000.0
3     Dave  41.0  90000.0
4      Eve  28.0  60000.0
```

```
# Remove rows with all missing values
df_cleaned_all = df.dropna(how='all')
print(df_cleaned_all)
```

```
      Name   Age   Salary
0    Alice  25.0  50000.0
1      Bob  32.0      NaN
2  Charlie   NaN  70000.0
3     Dave  41.0  90000.0
4      Eve  28.0  60000.0
```

```
df_cleaned_any = df.dropna (how='any')
print(df_cleaned_any)
```

```
    Name   Age   Salary
0  Alice  25.0  50000.0
3   Dave  41.0  90000.0
4    Eve  28.0  60000.0
```

```
data = {'Name': ['Alice', 'Bob', 'Charlie', 'Dave', 'Dave', 'Eve', 'Bob'],
        'Age': [25, np.nan, 35, 41, np.nan,np.nan, 85],
        'Salary': [50000, np.nan, 70000, np.nan, 60000, np.nan, 70000]}
df = pd.DataFrame(data)
df.duplicated()
```

	0
0	False
1	False
2	False
3	False
4	False

```
data = {'Name': ['Alice', 'Bob', 'Charlie', 'Dave', 'Eve'],
'Age': [25, np.nan, 35, 41, np.nan],
'Salary': [50000, np.nan, 70000, np.nan, 60000]}
df = pd.DataFrame(data)

df_filled = df.fillna(0)
print(df_filled)
```

```
      Name   Age   Salary
0    Alice  25.0  50000.0
1      Bob   0.0      0.0
2  Charlie  35.0  70000.0
3     Dave  41.0      0.0
4      Eve   0.0  60000.0
```

```
data = {'name': ['Alice', 'Bob', 'Charlie', 'Dave'],
        'age': [25, 32, 18, 471],
        'gender': ['F',' M',' M','M'],
        'height':[1.62, 1.78, 1.65, 1.831]}
df = pd. DataFrame (data)
df_filtered = df[(df ['gender']== 'M') & (df['height'] >1.7)]
df_filtered
```

	name	age	gender	height
3	Dave	471	M	1.831