VIT-AP UNIVERSITY, ANDHRA PRADESH

DATA WAREHOUSING AND DATA MINING

Academic year: 2022-2023 **Branch/ Class:** B.Tech

Semester: Fall **Date:10/9/22**

Faculty Name: Dr Aravapalli Rama Satish **School:** SCOPE

NAME: Majjiga Jaswanth REGNO:20BCD7171

---

1. Create a Pandas Data frame using

- 2D Lists / List of lists

  lst = [['Subbu', 'Kadi', 25], ['krish', 'Kevvu', 30], ['Raj', 'wilson', 26], ['Swetha', 'Kalimi', 22]]

```
import pandas as pd
data = [['Subbu', 'Kadi', 25],
        ['krish', 'Kevvu', 30],
        ['Raj', 'wilson', 26],
        ['Swetha','Kalimi', 22]]
df = pd.DataFrame(data)
print(df )
```

```
        0       1   2
0   Subbu    Kadi  25
1   krish   Kevvu  30
2     Raj  wilson  26
3  Swetha  Kalimi  22
```

- Dictionary

  data = {'Area':['Array', 'Stack', 'Queue'], ‘Student_1':[20, 21, 19], 'Student_2':[15, 20, 14]}

```
import pandas as pd
data = {
    'Area':['Array','Stack','Queue'],
    'Student_1':[20,21,19],
    'Student_2':[15,20,14]
}
df = pd.DataFrame(data)
print(df)
```

```
    Area  Student_1  Student_2
0  Array         20         15
1  Stack         21         20
2  Queue         19         14
```

- list of tuples

  data = [('Ravi', 18, 7), ('Raju', 15, 6), ('Jyotika', 17, 8), ('Minion', 18, 7), ('Swathi', 17, 5)]

```
import pandas as pd
data = [('Ravi',18,7),
        ('Raju',15,6),
        ('Jyothika',17,8),
        ('Minon',18,7),
        ('Swathi',17,5)]
df = pd.DataFrame(data,columns=['Name','Age','No'])
print(df)
```

```
       Name  Age  No
0      Ravi   18   7
1      Raju   15   6
2  Jyothika   17   8
3     Minon   18   7
4    Swathi   17   5
```

- list of dicts

  list = [{'Urbanpro': 'dataframe', 'Student': 'using', 'Site': 'list'}, {' Urbanpro ': 10, ' Student ': 20, ' Site ': 30}]

```
[2] import pandas as pd
    list = [{'Urbanpro': 'dataframe', 'Student': 'using', 'Site': 'list'},
            {' Urbanpro ': 10, ' Student ': 20, ' Site ': 30}]

    df = pd.DataFrame.from_records(list)
    print(df)
```

```
   Urbanpro Student  Site   Urbanpro    Student    Site
0  dataframe   using  list        NaN        NaN     NaN
1        NaN     NaN   NaN       10.0       20.0    30.0
```

- list of nested dictionaries

  list = [{ "Student": [{"Exam": 90, "Grade": "a"}, {"Exam": 99, "Grade": "b"}, {"Exam": 97, "Grade": "c"},], "Name": "Krish Naik" }, { "Student": [{"Exam": 89, "Grade": "a"}, {"Exam": 80, "Grade": "b"}], "Name": "Sudalai Raj" }]

```
import pandas as pd
list = [ {
"Student": [
{"Exam": 90, "Grade": "a"},
{"Exam": 99, "Grade": "b"},
{"Exam": 97, "Grade": "c"}, ],
"Name": "Krish Naik" },
{
"Student": [
{"Exam": 89, "Grade": "a"},
{"Exam": 80, "Grade": "b"} ],
"Name": "Sudalai Raj"
}]
rows=[]
for data in list:
  data_row = data['Student']
  n = data['Name']
  for row in data_row:
      row['Name'] = n
      rows.append(row)
      df = pd.DataFrame(rows)
print(df)
```

```
   Exam Grade         Name
0    90     a   Krish Naik
1    99     b   Krish Naik
2    97     c   Krish Naik
3    89     a  Sudalai Raj
4    80     b  Sudalai Raj
```

- panda's series

```
import pandas as pd
list = ['Jaswanth','jassu','chinnu','Mj']
list_Series = pd.Series(list)
print(list_Series)
```

```
0    Jaswanth
1       jassu
2      chinnu
3          Mj
dtype: object
```

2. Perform reindexing and reset of indexing of the data frame.

```
import pandas as pd
import numpy as np
column=['a','b','c','d','e']
index=['A','B','C','D','E']
df1 = pd.DataFrame(np.random.rand(5,5),
 columns=column, index=index)
print(df1)
print('\n\nDataframe after reindexing rows: \n',
df1.reindex(['B', 'D', 'A', 'C', 'E']))
```

```
          a         b         c         d         e
A  0.766666  0.359907  0.671286  0.981415  0.693702
B  0.163173  0.362906  0.042916  0.600710  0.570887
C  0.826144  0.176895  0.564887  0.966242  0.025717
D  0.876208  0.170144  0.140323  0.246849  0.085272
E  0.084982  0.188129  0.771549  0.796746  0.348090


Dataframe after reindexing rows:
          a         b         c         d         e
B  0.163173  0.362906  0.042916  0.600710  0.570887
D  0.876208  0.170144  0.140323  0.246849  0.085272
A  0.766666  0.359907  0.671286  0.981415  0.693702
C  0.826144  0.176895  0.564887  0.966242  0.025717
E  0.084982  0.188129  0.771549  0.796746  0.348090
```

3. Add a new column using map and dictionary into existing dataframe

```
dic={'continent':['Asia','Europe','Africa','Americas','Oceania'],
     'mean_lifeExp':[48.86,64.65,60.06,71.90,74.32]}
df=pd.DataFrame(dic)
pop_dict={'Europe': 24504794.99,
'Oceania': 8874672.33,
'Africa': 77038721.97,
'Asia': 9916003.14,
'Americas': 17169764.73}
print("Before adding column pop")
print(df)
print("After adding column pop")
df['pop']= df['continent'].map(pop_dict)
print(df)
```

```
Before adding column pop
  continent  mean_lifeExp
0      Asia         48.86
1    Europe         64.65
2    Africa         60.06
3  Americas         71.90
4   Oceania         74.32
After adding column pop
  continent  mean_lifeExp          pop
0      Asia         48.86   9916003.14
1    Europe         64.65  24504794.99
2    Africa         60.06  77038721.97
3  Americas         71.90  17169764.73
4   Oceania         74.32   8874672.33
```

4. Read a CSV file into a data frame and print the first few rows. Re-shape the data frame using stack, unstack, and melt methods. (file is enclosed – nba.csv)

```
df = pd.read_csv("nba.csv")
print("Head\n\n",df.head())
df_stacked = df.stack()
print("\n\nStacked\n\n",df_stacked.head(26))
df_unstacked = df_stacked.unstack()
print("\n\nUnsatcked\n\n",df_unstacked.head(10))
df_melt = df.melt(id_vars =['Name', 'Team'])
print("\n\nMelted\n\n",df_melt.head(10))
```

```
Head

              Name           Team  Number Position   Age Height  Weight  \
0  Avery Bradley  Boston Celtics     0.0       PG  25.0    6-2   180.0
1    Jae Crowder  Boston Celtics    99.0       SF  25.0    6-6   235.0
2   John Holland  Boston Celtics    30.0       SG  27.0    6-5   205.0
3    R.J. Hunter  Boston Celtics    28.0       SG  22.0    6-5   185.0
4  Jonas Jerebko  Boston Celtics     8.0       PF  29.0   6-10   231.0

              College     Salary
0               Texas  7730337.0
1           Marquette  6796117.0
2  Boston University        NaN
3      Georgia State  1148640.0
4                 NaN  5000000.0
```

```
Stacked

0  Name            Avery Bradley
   Team            Boston Celtics
   Number                     0.0
   Position                    PG
   Age                       25.0
   Height                     6-2
   Weight                   180.0
   College                 Texas
   Salary              7730337.0
1  Name              Jae Crowder
   Team            Boston Celtics
   Number                    99.0
   Position                    SF
   Age                       25.0
   Height                     6-6
   Weight                   235.0
   College              Marquette
   Salary              6796117.0
2  Name             John Holland
   Team            Boston Celtics
   Number                    30.0
   Position                    SG
   Age                       27.0
   Height                     6-5
   Weight                   205.0
   College     Boston University
dtype: object
```

```
Unsatcked

             Name           Team Number Position   Age Height Weight  \
0  Avery Bradley  Boston Celtics    0.0       PG  25.0    6-2  180.0
1    Jae Crowder  Boston Celtics   99.0       SF  25.0    6-6  235.0
2   John Holland  Boston Celtics   30.0       SG  27.0    6-5  205.0
3    R.J. Hunter  Boston Celtics   28.0       SG  22.0    6-5  185.0
4  Jonas Jerebko  Boston Celtics    8.0       PF  29.0   6-10  231.0
5   Amir Johnson  Boston Celtics   90.0       PF  29.0    6-9  240.0
6  Jordan Mickey  Boston Celtics   55.0       PF  21.0    6-8  235.0
7   Kelly Olynyk  Boston Celtics   41.0        C  25.0    7-0  238.0
8   Terry Rozier  Boston Celtics   12.0       PG  22.0    6-2  190.0
9   Marcus Smart  Boston Celtics   36.0       PG  22.0    6-4  220.0

             College      Salary
0              Texas   7730337.0
1          Marquette   6796117.0
2  Boston University         NaN
3      Georgia State   1148640.0
4                NaN   5000000.0
5                NaN  12000000.0
6                LSU   1170960.0
7            Gonzaga   2165160.0
8         Louisville   1824360.0
9     Oklahoma State   3431040.0
```

```
Melted

             Name           Team variable value
0  Avery Bradley  Boston Celtics   Number   0.0
1    Jae Crowder  Boston Celtics   Number  99.0
2   John Holland  Boston Celtics   Number  30.0
3    R.J. Hunter  Boston Celtics   Number  28.0
4  Jonas Jerebko  Boston Celtics   Number   8.0
5   Amir Johnson  Boston Celtics   Number  90.0
6  Jordan Mickey  Boston Celtics   Number  55.0
7   Kelly Olynyk  Boston Celtics   Number  41.0
8   Terry Rozier  Boston Celtics   Number  12.0
9   Marcus Smart  Boston Celtics   Number  36.0
```

5. Make a column of data frame as an index, then do reset of indexing without losing the column data

```python
import pandas as pd
data = {'Name':['Jai', 'Princi', 'Gaurav', 'Anuj', 'Geeku'],
        'Age':[27, 24, 22, 32, 15],
 'Address':['Delhi', 'Kanpur', 'Allahabad', 'Kannauj', 'Noida'],
 'Qualification':['Msc', 'MA', 'MCA', 'Phd', '10th'] }
index = {'a', 'b', 'c', 'd', 'e'}
df = pd.DataFrame(data, index)
df.set_index(['Age'], inplace = True)
print("Making a column of data frame as an index:\n\n\n",df)
df.reset_index(level =['Age'], inplace = True)
print("\n\nIndexing without losing data:\n\n\n",df)
```

```
Making a column of data frame as an index:


         Name    Address Qualification
Age
27        Jai      Delhi           Msc
24     Princi     Kanpur            MA
22     Gaurav  Allahabad           MCA
32       Anuj    Kannauj           Phd
15      Geeku      Noida          10th


Indexing without losing data:


    Age    Name    Address Qualification
0    27     Jai      Delhi           Msc
1    24  Princi     Kanpur            MA
2    22  Gaurav  Allahabad           MCA
3    32    Anuj    Kannauj           Phd
4    15   Geeku      Noida          10th
```

6. Change the column names and row indexes in multiple ways of a data frame.

```
import pandas as pd
df=pd.DataFrame({"Name":['Tom','Nick','John','Peter'],
 "Age":[15,26,17,28]})
print("Change1:\n\n",df)
df.columns =['Col_1', 'Col_2']
df.index = ['Row_1', 'Row_2', 'Row_3', 'Row_4']
print("\n\nchange2:\n\n",df)
df = df.rename(columns = {"Col_1":"Mod_col"})
print("\n\nchange3:\n\n",df)
df = df.rename(columns=lambda x: x+'x')
print("\n\nchange4:\n\n",df)
```

```
Change1:

     Name  Age
0    Tom   15
1   Nick   26
2   John   17
3  Peter   28


change2:

       Col_1  Col_2
Row_1    Tom     15
Row_2   Nick     26
Row_3   John     17
Row_4  Peter     28


change3:

       Mod_col  Col_2
Row_1      Tom     15
Row_2     Nick     26
Row_3     John     17
Row_4    Peter     28


change4:

       Mod_colx  Col_2x
Row_1       Tom      15
Row_2      Nick      26
Row_3      John      17
Row_4     Peter      28
```

7. Iterate over the rows of the data frame and print data of required columns.

```
import pandas as pd


dict = {'name':["aparna", "pankaj", "sudhir", "Geeku"],
        'degree': ["MBA", "BCA", "M.Tech", "MBA"],
 'score':[90, 40, 80, 98]}
df = pd.DataFrame(dict)
for i, j in df.iterrows():
 print(i, j)
print()
```

```
0 name      aparna
degree       MBA
score         90
Name: 0, dtype: object
1 name      pankaj
degree       BCA
score         40
Name: 1, dtype: object
2 name      sudhir
degree    M.Tech
score         80
Name: 2, dtype: object
3 name      Geeku
degree      MBA
score        98
Name: 3, dtype: object
```

```
import pandas as pd
data = pd.read_csv("nba.csv")
for i, j in data.iterrows():
 print(i, j)
print()
```

```
Position           PF
Age              20.0
Height           6-10
Weight          234.0
College       Kentucky
Salary       2239800.0
Name: 452, dtype: object
453 Name        Shelvin Mack
Team           Utah Jazz
Number               8.0
Position              PG
Age                 26.0
Height               6-3
Weight             203.0
College           Butler
Salary         2433333.0
Name: 453, dtype: object
454 Name        Raul Neto
Team        Utah Jazz
Number           25.0
Position           PG
Age              24.0
Height            6-1
Weight          179.0
College           NaN
Salary       900000.0
Name: 454, dtype: object
455 Name        Tibor Pleiss
Team           Utah Jazz
Number              21.0
Position               C
```

8. Select the rows of a data frame based on conditions:

Data: record = { 'Name': ['Ankit', 'Amit', 'Aishwarya', 'Priyanka', 'Priya', 'Shaurya'], 'Age': [21, 19, 20, 18, 17, 21], 'Stream': ['Math', 'Commerce', 'Science', 'Math', 'Math', 'Science'], 'Percentage': [88, 92, 95, 70, 65, 78] }

```
record = { 'Name': ['Ankit', 'Amit', 'Aishwarya', 'Priyanka', 'Priya', 'Shaurya' ],
'Age': [21, 19, 20, 18, 17, 21],
'Stream': ['Math', 'Commerce', 'Science', 'Math', 'Math', 'Science'],
 'Percentage': [88, 92, 95, 70, 65, 78] }
df=pd.DataFrame(record)
df
```

	Name	Age	Stream	Percentage
0	Ankit	21	Math	88
1	Amit	19	Commerce	92
2	Aishwarya	20	Science	95
3	Priyanka	18	Math	70
4	Priya	17	Math	65
5	Shaurya	21	Science	78

- Percentage > 80

```
[35] df.loc[df['Percentage']>80]
```

	Name	Age	Stream	Percentage
0	Ankit	21	Math	88
1	Amit	19	Commerce	92
2	Aishwarya	20	Science	95

- Percentage != 95

```
df.loc[df['Percentage']!=95]
```

	Name	Age	Stream	Percentage
0	Ankit	21	Math	88
1	Amit	19	Commerce	92
3	Priyanka	18	Math	70
4	Priya	17	Math	65
5	Shaurya	21	Science	78

- Stream is either Math or Commerce

```
[39] df.loc[(df['Stream']=='Math') | (df['Stream']=='Commerce')]
```

	Name	Age	Stream	Percentage
0	Ankit	21	Math	88
1	Amit	19	Commerce	92
3	Priyanka	18	Math	70
4	Priya	17	Math	65

- Stream is neither Math nor Commerce

```
[40] df.loc[(df['Stream']!='Math') & (df['Stream']!='Commerce')]
```

	Name	Age	Stream	Percentage
2	Aishwarya	20	Science	95
5	Shaurya	21	Science	78

- Age = 21 and Stream is either Math or Commerce

```
df.loc[(df['Age']==21) & ((df['Stream']=='Math') | (df['Stream']=='Commerce'))]
```

	Name	Age	Stream	Percentage
0	Ankit	21	Math	88

- Apply the above using loc[].

9. Convert a data frame into a list of lists using iloc[] and for loop.

```
import pandas as pd

new_dictionary = {
    'Country_name': ['Germany', 'Australia', 'France','England'],
    'Values': [843,290,179,926],
    'City':['paris','Japna','Kenya','Newzealand']

}
z = pd.DataFrame(new_dictionary)
print(z)
m = z.iloc[:, 0].tolist()
u = z.iloc[:, 1].tolist()
w = z.iloc[:, 2].tolist()
print("list of val:",m)
print("list of val:",u)
print("list of val:",w)
```

```
  Country_name  Values        City
0      Germany     843       paris
1    Australia     290       Japna
2       France     179       Kenya
3      England     926  Newzealand
list of val: ['Germany', 'Australia', 'France', 'England']
list of val: [843, 290, 179, 926]
list of val: ['paris', 'Japna', 'Kenya', 'Newzealand']
```

10. Drop the rows from the data frame which does not satisfy specific condition, such as age < 25 (Work on nba.csv file).

```
import pandas as pd

df = pd.read_csv('nba.csv')
df = df.dropna(how = 'all')
df.drop(df[df['Age'] < 25].index, inplace = True)

print(df.head(15))
print(df.shape)
```

```
                Name           Team  Number Position   Age Height  Weight  \
0      Avery Bradley  Boston Celtics     0.0       PG  25.0    6-2   180.0
1        Jae Crowder  Boston Celtics    99.0       SF  25.0    6-6   235.0
2       John Holland  Boston Celtics    30.0       SG  27.0    6-5   205.0
4      Jonas Jerebko  Boston Celtics     8.0       PF  29.0   6-10   231.0
5       Amir Johnson  Boston Celtics    90.0       PF  29.0    6-9   240.0
7       Kelly Olynyk  Boston Celtics    41.0        C  25.0    7-0   238.0
11     Isaiah Thomas  Boston Celtics     4.0       PG  27.0    5-9   185.0
12       Evan Turner  Boston Celtics    11.0       SG  27.0    6-7   220.0
14      Tyler Zeller  Boston Celtics    44.0        C  26.0    7-0   253.0
15  Bojan Bogdanovic   Brooklyn Nets    44.0       SG  27.0    6-8   216.0
17   Wayne Ellington   Brooklyn Nets    21.0       SG  28.0    6-4   200.0
19      Jarrett Jack   Brooklyn Nets     2.0       PG  32.0    6-3   200.0
21   Sean Kilpatrick   Brooklyn Nets     6.0       SG  26.0    6-4   219.0
23       Brook Lopez   Brooklyn Nets    11.0        C  28.0    7-0   275.0
25       Willie Reed   Brooklyn Nets    33.0       PF  26.0   6-10   220.0

              College      Salary
0               Texas   7730337.0
1           Marquette   6796117.0
2   Boston University         NaN
4                 NaN   5000000.0
5                 NaN  12000000.0
7             Gonzaga   2165160.0
11         Washington   6912869.0
12         Ohio State   3425510.0
14     North Carolina   2616975.0
15                NaN   3425510.0
17     North Carolina   1500000.0
19       Georgia Tech   6300000.0
21         Cincinnati    134215.0
23           Stanford  19689000.0
25       Saint Louis    947276.0
(303, 9)
```

11. Develop a function to insert a row at a specific position in the data frame.

```
import pandas as pd

df = pd.DataFrame({'Date':['10/2/2011', '12/2/2011', '13/2/2011', '14/2/2011'],
 'Event':['Music', 'Poetry', 'Theatre', 'Comedy'],  'Cost':[10000, 5000, 15000, 2000]})
print(df)
```

```
        Date    Event   Cost
0  10/2/2011    Music  10000
1  12/2/2011   Poetry   5000
2  13/2/2011  Theatre  15000
3  14/2/2011   Comedy   2000
```

```python
def Insert_row(row_number, df, row_value):
 start_upper = 0
 end_upper = row_number
 start_lower = row_number
 end_lower = df.shape[0]
 upper_half = [*range(start_upper, end_upper, 1)]
 lower_half = [*range(start_lower, end_lower, 1)]
 lower_half = [x.__add__(1) for x in lower_half]
 index_ = upper_half + lower_half
 df.index = index_
 df.loc[row_number] = row_value
 df = df.sort_index()
 return df
row_number = 2
row_value = ['11/2/2011', 'Wrestling', 12000]

if row_number > df.index.max()+1:
 print("Invalid row_number")
else:
 df = Insert_row(row_number, df, row_value)
 print(df)
```

```
         Date      Event   Cost
0  10/2/2011      Music  10000
1  12/2/2011     Poetry   5000
2  11/2/2011  Wrestling  12000
3  13/2/2011    Theatre  15000
4  14/2/2011     Comedy   2000
```

12. Rank the rows of a data frame based on a single attribute (Marks) and arrange the rows based on rank. Data: student_details = {'Name':['Raj', 'Raj', 'Raj', 'Aravind', 'Aravind', 'Aravind', 'John', 'John', 'John', 'Arjun', 'Arjun', 'Arjun'], 'Subject':['Maths', 'Physics', 'Chemistry', 'Maths', 'Physics', 'Chemistry', 'Maths', 'Physics', 'Chemistry', 'Maths','Physics', 'Chemistry'], 'Marks':[80, 90, 75, 60, 40, 60, 80, 55, 100, 90, 75, 70] }

```
import pandas as pd
student_details = {
'Name':['Raj', 'Raj', 'Raj', 'Aravind', 'Aravind', 'Aravind', 'John', 'John','John', 'Arjun', 'Arjun', 'Arjun'],
'Subject':['Maths', 'Physics', 'Chemistry', 'Maths', 'Physics',
'Chemistry', 'Maths', 'Physics', 'Chemistry', 'Maths','Physics', 'Chemistry'],
'Marks':[80, 90, 75,60, 40, 60, 80, 55, 100, 90, 75, 70] }
df = pd.DataFrame(student_details)
print(df)
```

```
       Name    Subject  Marks
0       Raj      Maths     80
1       Raj    Physics     90
2       Raj  Chemistry     75
3   Aravind      Maths     60
4   Aravind    Physics     40
5   Aravind  Chemistry     60
6      John      Maths     80
7      John    Physics     55
8      John  Chemistry    100
9     Arjun      Maths     90
10    Arjun    Physics     75
11    Arjun  Chemistry     70
```

```
df['Mark_Rank'] = df['Marks'].rank(ascending = 0)
df = df.set_index('Mark_Rank')
print(df)
```

```
                Name    Subject  Marks
Mark_Rank
4.5              Raj      Maths     80
2.5              Raj    Physics     90
6.5              Raj  Chemistry     75
9.5          Aravind      Maths     60
12.0         Aravind    Physics     40
9.5          Aravind  Chemistry     60
4.5             John      Maths     80
11.0            John    Physics     55
1.0             John  Chemistry    100
2.5            Arjun      Maths     90
6.5            Arjun    Physics     75
8.0            Arjun  Chemistry     70
```

```
df = df.sort_index()
print(df)
```

```
                Name    Subject  Marks
Mark_Rank
1.0             John  Chemistry    100
2.5              Raj    Physics     90
2.5            Arjun      Maths     90
4.5              Raj      Maths     80
4.5             John      Maths     80
6.5              Raj  Chemistry     75
6.5            Arjun    Physics     75
8.0            Arjun  Chemistry     70
9.5          Aravind      Maths     60
9.5          Aravind  Chemistry     60
11.0            John    Physics     55
12.0         Aravind    Physics     40
```

13. Sort the rows of a data frame using single column (Maths) or multiple columns (Maths then

Science).

data = {'name': ['Raj', 'Aravind', 'John', 'Arjun', 'Williams'], 'Maths': [8, 5, 6, 9, 7], 'Science': [7, 9, 5, 4, 7], 'English': [7, 4, 7, 6, 8]}.

```
import pandas as pd
data= {'name': ['Raj', 'Aravind', 'John', 'Arjun', 'Williams'],
       'Maths': [8, 5, 6, 9, 7], 'Science': [7, 9, 5, 4, 7],
       'English': [7, 4, 7, 6, 8]}
df = pd.DataFrame(data, index = ['r1','r2','r3','r4','r0'])
df.sort_values(by='Maths')
```

	name	Maths	Science	English
r2	Aravind	5	9	4
r3	John	6	5	7
r0	Williams	7	7	8
r1	Raj	8	7	7
r4	Arjun	9	4	6

```
df.sort_values(by = ['Maths','Science'])
```

	name	Maths	Science	English
r2	Aravind	5	9	4
r3	John	6	5	7
r0	Williams	7	7	8
r1	Raj	8	7	7
r4	Arjun	9	4	6

14. Identify which driver scored max points and having max age. Similarly min points and min age.

dict1 = {'Driver':['Raj', 'Aravind', 'John', 'Arjun', 'Williams', 'Senthil', 'Krishna', 'Kalyani', 'Kavitha', 'Kiran', 'Kumar', 'Rani', 'Pavitra', 'Sophiya', 'Sujal', 'Sundari', 'Karishma', 'Kunjal', 'Kushal', 'Rishi'],

'Points':[408, 320, 251, 249, 247, 170, 69, 62, 56, 53, 50, 49, 39, 37, 29, 12, 9, 6, 4, 1],

'Age':[33, 31, 39, 21, 29, 29, 31, 28, 26, 24, 37, 22, 21, 32, 22, 26, 28, 20, 29, 23]}

```
dict1={'Driver':['Raj', 'Aravind', 'John','Arjun', 'Williams', 'Senthil','Krishna', 'Kalyani',
                 'Kavitha', 'Kiran', 'Kumar', 'Rani', 'Pavitra','Sophiya', 'Sujal', 'Sundari',
                  'Karishma', 'Kunjal', 'Kushal', 'Rishi'],
       'Points':[408, 320, 251, 249, 247, 170, 69, 62, 56,
                  53, 50, 49, 39, 37, 29, 12, 9, 6, 4, 1],
       'Age':[33, 31, 39, 21, 29, 29, 31, 28, 26, 24, 37, 22, 21, 32, 22, 26, 28, 20, 29, 23]}
df=pd.DataFrame(dict1)
df
max_points = df['Points'].max()
print("max points ",max_points)
max_age=df['Age'].max()
print("max age ",max_age)
min_points = df['Points'].min()
print("min points ",min_points)
min_age=df['Age'].min()
print("min age ",min_age)
```

```
max points  408
max age  39
min points  1
min age  20
```

15. Filter rows checking Position contains PG and College must contains like UC. (Work on nba.csv file)

```
df = pd.read_csv('nba.csv')
df1 = df[df['Position'].str.contains("PG") & df['College'].str.contains('UC')]
df1
```

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
142	Darren Collison	Sacramento Kings	7.0	PG	28.0	6-0	175.0	UCLA	5013559.0
264	Jordan Farmar	Memphis Grizzlies	4.0	PG	29.0	6-2	180.0	UCLA	NaN
290	Jrue Holiday	New Orleans Pelicans	11.0	PG	25.0	6-4	205.0	UCLA	10595507.0
402	Zach LaVine	Minnesota Timberwolves	8.0	PG	21.0	6-5	189.0	UCLA	2148360.0
426	Russell Westbrook	Oklahoma City Thunder	0.0	PG	27.0	6-3	200.0	UCLA	16744218.0

16. Select ‘n’ random rows from the data frame or n% rows.

```
df = pd.read_csv('nba.csv')
df.sample(n=6)
```

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
290	Jrue Holiday	New Orleans Pelicans	11.0	PG	25.0	6-4	205.0	UCLA	10595507.0
106	Brandon Bass	Los Angeles Lakers	2.0	PF	31.0	6-8	250.0	LSU	3000000.0
163	Bobby Portis	Chicago Bulls	5.0	PF	21.0	6-11	230.0	Arkansas	1391160.0
233	Wesley Matthews	Dallas Mavericks	23.0	SG	29.0	6-5	220.0	Marquette	16407500.0
362	Andrew Nicholson	Orlando Magic	44.0	PF	26.0	6-9	250.0	St. Bonaventure	2380593.0
185	Reggie Bullock	Detroit Pistons	25.0	SF	25.0	6-7	205.0	North Carolina	1252440.0

17. List out the column names of a data frame.

```
df = pd.read_csv('nba.csv')
df.columns
```

```
Index(['Name', 'Team', 'Number', 'Position', 'Age', 'Height', 'Weight',
       'College', 'Salary'],
      dtype='object')
```

18. Rename a column name(s) of a data frame.

```
rankings = {'test': ['India', 'South Africa', 'England',  'New Zealand',
                     'Australia'],  'odi': ['England', 'India', 'New Zealand',
 'South Africa', 'Pakistan'],  't20': ['Pakistan', 'India', 'Australia',
 'England', 'New Zealand']}
rankings_pd = pd.DataFrame(rankings)
print("Before modifying first column")
print(rankings_pd.columns)
print(rankings_pd)
rankings_pd.rename(columns = {'test':'TEST'}, inplace = True)
print("\nAfter modifying first column:\n", rankings_pd.columns)
```

```
Before modifying first column
Index(['test', 'odi', 't20'], dtype='object')
          test           odi          t20
0        India       England     Pakistan
1 South Africa         India        India
2      England   New Zealand    Australia
3  New Zealand  South Africa      England
4    Australia      Pakistan  New Zealand

After modifying first column:
 Index(['TEST', 'odi', 't20'], dtype='object')
```

19. Print unique values of a column.

```
rankings = {'test': ['India', 'South Africa', 'England',  'New Zealand', 'Australia','India'],
            'odi': ['England', 'India', 'New Zealand','South Africa', 'Pakistan','Pakistan'],
            't20': ['Pakistan', 'India', 'Australia','England', 'New Zealand','Australia']}
df = pd.DataFrame(rankings)
print(df['test'].unique())
```

```
['India' 'South Africa' 'England' 'New Zealand' 'Australia']
```

20. Display the list of indices whose rows satisfy specific conditions on a column.

```
import pandas as pd
df = pd.DataFrame({
 'Date':['10/2/2011', '11/2/2011', '12/2/2011', '13/2/2011'],
 'Product':['Umbrella', 'Matress', 'Badminton', 'Shuttle'],
 'Last_Price':[1200, 1500, 1600, 352],  'Updated_Price':[1250, 1450, 1550, 400],
 'Discount':[10, 10, 10, 10]
 })
df.index =['Item 1', 'Item 2', 'Item 3', 'Item 4']
print(df)
```

	Date	Product	Last_Price	Updated_Price	Discount
Item 1	10/2/2011	Umbrella	1200	1250	10
Item 2	11/2/2011	Matress	1500	1450	10
Item 3	12/2/2011	Badminton	1600	1550	10
Item 4	13/2/2011	Shuttle	352	400	10

```
[61] Index_label = df[df['Updated_Price']>1000].index.tolist()
     print(Index_label)
```

```
['Item 1', 'Item 2', 'Item 3']
```
