python

Course Id :INT 522

PART II-PANDAS

I INTRODUCTION

Pandas

- Pandas is probably the most powerful library in Data analysis.
- It provides high-performance tools for data manipulation and analysis.
- Furthermore, it is very effective at converting data formats and querying data out of databases.
- The two main data structures of Pandas are :

❑ **series**

❑ **data frame**

- To work with Pandas, we need to import the module.

import pandas as pd

II SERIES

Series

- A series in Pandas is a one-dimensional array which is labeled.
- You can imagine it to be the data science equivalent of an ordinary Python dictionary.
- In order to create a series, we use the constructor of the **Series** class. The first parameter that we pass is a list full of values (in this case numbers). The second parameter is the list of the indices or keys.

Series

```
import pandas as pd
#series of any thing int float or string [] {}


s=pd.Series([10, 'Namaste',23.5,'hello'])
print(s)
```

Changing the index of an element

- import pandas as pd
- s=pd.Series([1,2,3,4,5],
  ['a','b','c','d',’e’])
- print(s)

- import pandas as pd
- s=pd.Series([1,2,3,4,5],index=['a','b','c','d','e’])
- print(s)

Converting Dictionaries into Series

- Since series and dictionaries are quite similar, we can easily convert our Python dictionaries into Pandas series.

- import pandas as pd

  myDict = {'A':10, 'B':20, 'C':30}

  series=pd.Series(myDict)

  print(series)

  print(series['A'])

Changing the index of the element

- import pandas as pd

  myDict = {'A':10, 'B':20, 'C':30}

  series=pd.Series(myDict,index=['C','A','B’])

  print(series)

  print(series['B'])

- import pandas as pd

  myDict = {'A':10, 'B':20, 'C':30}

  series=pd.Series(myDict,index=['X','Y','Z’])

  print(series)

  print(series['X'])

Accessing a value

- import pandas as pd

  s =pd.Series([1,2,3,4,5])

  print(s)

  print(s[1])

- import pandas as pd

  s=pd.Series([1,2,3,4,5],['a','b','c','d','e’])

  print(s)

  print(s['c’])

  print(s[2])

What is the output of the code?

```
import pandas as pd
d={'jalandhar':800000,'Amritsar':1000000, 'Delhi': 200000}
cities=pd.Series(d)
print(cities)
print(cities['jalandhar'])
print(cities[cities>800000])
print(cities[cities>790000])
```

What is the output of the code?

```
import pandas as pd
d={'jalandhar':800,'Amritsar':1000, 'Delhi': 2000,
   'bombay':500,'ludhina':700}
cities=pd.Series(d)
print(cities)
cities['jalandhar']=900
print(cities)
```

What is the output of the code?

```
import pandas as pd
import numpy as np
d={'jalandhar':800,'Amritsar':1000, 'Delhi': 2000,
    'bombay':500,'ludhina':700}
cities=pd.Series(d)
print(cities)
print(np.square(cities))
print(cities.isnull())
```

III DATAFRAME

Dataframe

- In contrast to the series, a data frame is not one-dimensional but multi-dimensional and looks like a table.
- You can imagine it to be like an Excel table or a data base table.

Dataframe

- import pandas as pd

```
data = {'Name': ['Anna', 'Bob', 'Charles’],
         'Age': [24, 32, 35],
         'Height': [176, 187,    175]
        }
d=pd.DataFrame (data)
print(d)
```

Dataframes

```
import pandas as pd
data={
      'students':['ram', 'sham', 'tom', 'dom', 'tomy'],
      'maths':[98,50,23,72,87],
      'science':[96,45,76,54,1],
      'sports':['basketball','swimming','TT','Badminton','cricket']
      }

Student=pd.DataFrame(data)
print(Student)
```

```
import pandas as pd
data={
      'students':['ram', 'sham', 'tom', 'dom', 'tomy'],
      'maths':[98,50,23,72,87],
      'science':[96,45,76,54,1],
      'sports':['basketball','swimming','TT','Badminton','cricket']
      }

Student=pd.DataFrame(data, columns=['students','maths','science','sports'])
print(Student)
```

Accessing a value

- import pandas as pd

  data = {'Name': ['Anna', 'Bob', 'Charles'], 'Age': [24, 32, 35],'Height': [176, 187, 175]}

  d=pd.DataFrame (data)

  print(d)

  print(d['Name'][1])

Extracting selected columns

- import pandas as pd
- data = {'Name': ['Anna', 'Bob', 'Charles'], 'Age': [24, 32, 35],'Height': [176, 187, 175]}
- d=pd.DataFrame (data)
- print(d)
- print(d[['Name','Height']])

IV DATAFRAME FUNCTIONS

(A) Basic Functions

BASIC FUNCTIONS AND ATTRIBUTES	
FUNCTION	**DESCRIPTION**
df.T	Transposes the rows and columns of the data frame
df.dtypes	Returns data types of the data frame
df.ndim	Returns the number of dimensions of the data frame
df.shape	Returns the shape of the data frame
df.size	Returns the number of elements in the data frame
df.head(n)	Returns the first *n* rows of the data frame (default is five)
df.tail(n)	Returns the last *n* rows of the data frame (default is five)

How to use basic functions?

```
import pandas as pd
data = {'Name': ['Anna', 'Bob', 'Charles','Daniel', 'Evan', 'Fiona','Gerald', 'Henry', 'India'],
          'Age': [24,32,35,45,22,54,55,43,25],
       'Height': [176,187,175,182,176, 189,165,187,167]
        }
df=pd.DataFrame(data)
print(df.T)
print()
print(df.ndim)
print()
print(df.shape)
print()
print(df.size)
print()
print(df.head())
print()
print(df.tail())
```

(B) Statistical Functions

FUNCTION	DESCRIPTION
count()	Count the number of non-null elements
sum()	Returns the sum of values of the selected columns
mean()	Returns the arithmetic mean of values of the selected columns
median()	Returns the median of values of the selected columns
mode()	Returns the value that occurs most often in the columns selected
std()	Returns standard deviation of the values
min()	Returns the minimum value
max()	Returns the maximum value
abs()	Returns the absolute values of the elements
prod()	Returns the product of the selected elements
describe()	Returns data frame with all statistical values summarized

How to use statistical functions?

```
import pandas as pd
data = {'Name': ['Anna', 'Bob', 'Charles','Daniel', 'Evan', 'Fiona','Gerald', 'Henry', 'India'],
          'Age': [24,32,35,45,22,54,55,43,25],
       'Height': [176,187,175,182,176, 189,165,187,167]
        }
df=pd.DataFrame(data)
print(df['Age'].mean())
print()
print(df['Height'].median())
print()
print(df.sum())
print()
print(df.mean())
print()
print(df['Height'].mode())
print()
print(df.count())
```

(C) Numpy Functions

- Instead of using the built-in Pandas functions, we can also use the methods we already know.
- For this, we just use the **apply** function of the data frame and then pass our desired method.

How to use Numpy function in pandas?

```
import pandas as pd
import numpy as np
data = {'Name': ['Anna', 'Bob', 'Charles','Daniel', 'Evan', 'Fiona','Gerald', 'Henry', 'India'],
         'Age': [24,32,35,45,22,54,55,43,25],
      'Height': [176,187,175,182,176, 189,165,187,167]
        }
df=pd.DataFrame(data)
print(df['Age'].apply(np.sin)
```

V ITERATING

Iterating

- Iterating over data frames is quite easy with Pandas. We can either do it in the classic way or use specific functions for it.

```
for x in df['Age']:
    print(x)
```

FUNCTION	DESCRIPTION
iteritems()	Iterator for key-value pairs
iterrows()	Iterator for the rows (index, series)
itertuples()	Iterator for the rows as named tuples

How to iterate in pandas?

```
import pandas as pd
data = {'Name': ['Anna', 'Bob', 'Charles','Daniel', 'Evan', 'Fiona','Gerald', 'Henry', 'India'],
          'Age': [24,32,35,45,22,54,55,43,25],
       'Height': [176,187,175,182,176, 189,165,187,167]
        }
df=pd.DataFrame(data)

for x in df['Age']:
 print(x)

 print()

for key,value in df.iteritems():
    print("{}: {}".format(key, value))

for index,value in df.iterrows():
    print(index,value)
```

VI SORTING

(A) Sorting by Index

```
import pandas as pd
import numpy as np
df = pd.DataFrame(np.random.rand(10,2),index=[1,5,3,6,7,2,8,9,0,4],columns=['A','B'])
print(df)
print()
print(df.sort_index())
```

Inplace parameter

- When we use functions that manipulate our data frame, we don't actually change it but we return a manipulated copy. If we wanted to apply the changes on the actual data frame, we would need to do it like this:

  df = df.sort_index()

- But Pandas offers us another alternative as well. This alternative is the parameter inplace. When this parameter is set to True, the changes get applied to our actual data frame.
- df.sort_index(inplace=True)

Inplace parameter

```
import pandas as pd
import numpy as np
df = pd.DataFrame(np.random.rand(10,2),index=[1,5,3,6,7,2,8,9,0,4],columns=['A','B'])
print(df)
print()
print(df.sort_index(inplace=True))
```

(B) Sort by Column

```
import pandas as pd
data = {'Name': ['Anna', 'Bob', 'Charles','Daniel', 'Evan', 'Fiona','Gerald', 'Henry', 'India'],
        'Age': [24,24,35,45,22,54,54,43,25],
        'Height': [176,187,175,182,176,189,165,187,167]
        }
df = pd.DataFrame(data)
print(df)
print(df.sort_values(by=['Age','Height'],inplace=True))
```

VII CSV FILE

Reading data from csv file

```
import pandas as pd
df = pd.read_csv('Book1.csv')
df.set_index('id', inplace=True)
print(df)
```

Writing data into csv files

```
import pandas as pd
data = {'Name': ['Anna', 'Bob', 'Charles','Daniel', 'Evan', 'Fiona','Gerald', 'Henry', 'India'],
        'Age': [24,24,35,45,22,54,54,43,25],
        'Height': [176,187,175,182,176,189,165,187,167]
        }
df = pd.DataFrame(data)
df.to_csv('Book1.csv')
```

VIII JOINING AND MERGING

Merging

```
import pandas as pd
names = pd.DataFrame({'id': [1,2,3,4,5],'name': ['Anna', 'Bob', 'Charles', 'Daniel', 'Evan'],})
ages = pd.DataFrame({'id': [1,2,3,4,5],'age': [20,30,40,50,60]})
df = pd.merge(names,ages,on='id')
print(df)
print()
print(df.set_index('id', inplace=True))
```

Joining

JOIN MERGE TYPES	
JOIN	**DESCRIPTION**
left	Uses all keys from left object and merges with right
right	Uses all keys from right object and merges with left
outer	Uses all keys from both objects and merges them
inner	Uses only the keys which both objects have and merges them (default)

Inner join

```
import pandas as pd
names = pd.DataFrame({'id': [1,2,3,4,5],'name': ['Anna', 'Bob', 'Charles', 'Daniel', 'Evan'],})
ages = pd.DataFrame({'id': [1,2,3,4,5],'age': [20,30,40,50,60]})
df = pd.merge(names,ages,on='id',how='inner')
print(df)
```

```
import pandas as pd
names = pd.DataFrame({'id': [1,2,3,4,5,7],'name': ['Anna', 'Bob', 'Charles', 'Daniel', 'Evan','Fiona'],})
ages = pd.DataFrame({'id': [1,2,3,4,5,7],'age': [20,30,40,50,60,70]})
df = pd.merge(names,ages,on='id',how='inner')
print(df)
print()
print(df.set_index('id', inplace=True))
```

Left join

```
import pandas as pd
names = pd.DataFrame({'id': [1,2,3,4,5,6],'name': ['Anna', 'Bob', 'Charles', 'Daniel', 'Evan','Fiona'],})
ages = pd.DataFrame({'id': [1,2,3,4,5,7],'age': [20,30,40,50,60,70]})
df = pd.merge(names,ages,on='id',how='left')
print(df)
print()
print(df.set_index('id', inplace=True))
```

Right join

```
import pandas as pd
names = pd.DataFrame({'id': [1,2,3,4,5,6],'name': ['Anna', 'Bob', 'Charles', 'Daniel', 'Evan','Fiona'],})
ages = pd.DataFrame({'id': [1,2,3,4,5,7],'age': [20,30,40,50,60,70]})
df = pd.merge(names,ages,on='id',how='right')
print(df)
print()
print(df.set_index('id', inplace=True))
```

Outer join

```
import pandas as pd
names = pd.DataFrame({'id': [1,2,3,4,5,6],'name': ['Anna', 'Bob', 'Charles', 'Daniel', 'Evan','Fiona'],})
ages = pd.DataFrame({'id': [1,2,3,4,5,7],'age': [20,30,40,50,60,70]})
df = pd.merge(names,ages,on='id',how='outer')
print(df)
print()
print(df.set_index('id', inplace=True))
```

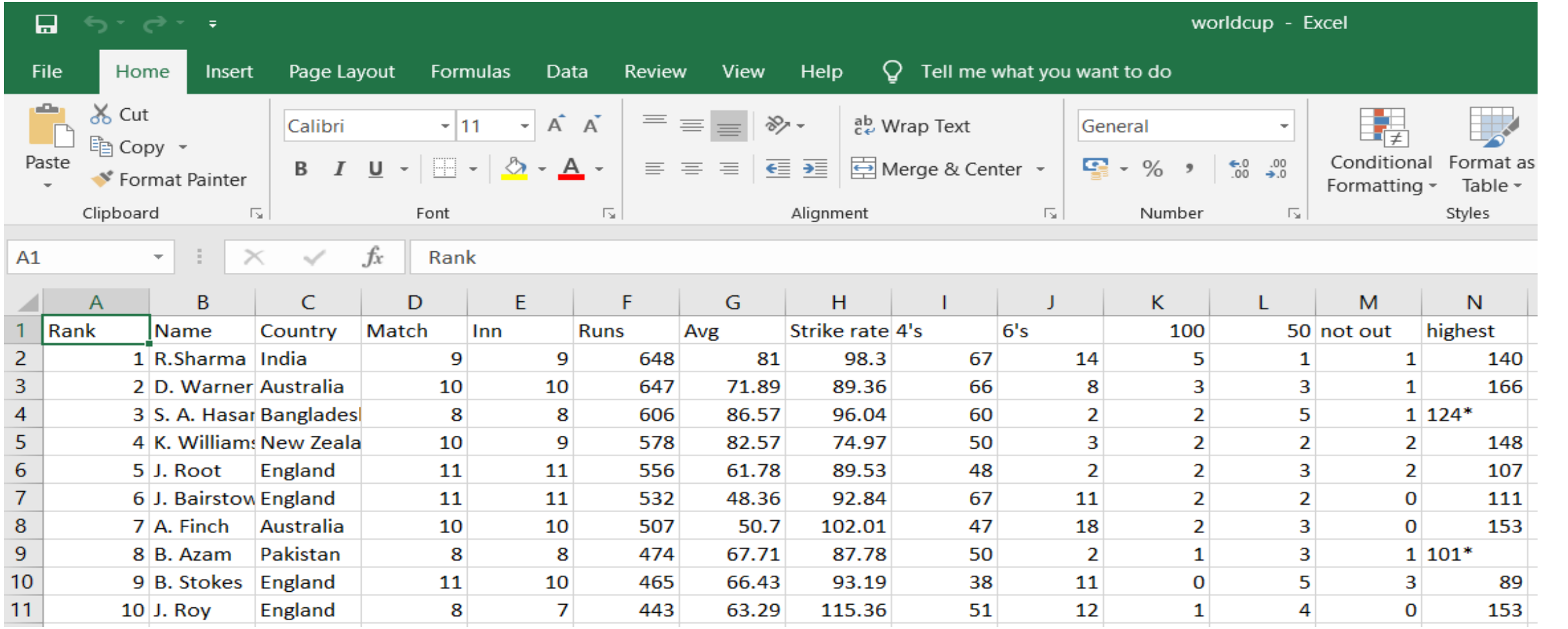
IX QUERYING DATA

Extracting Selected Data

```
import pandas as pd
names = pd.DataFrame({'id': [1,2,3,4,5,6],'name': ['Anna', 'Bob', 'Charles', 'Daniel', 'Evan','Fiona'],})
ages = pd.DataFrame({'id': [1,2,3,4,5,6],'age': [20,30,40,50,60,70]})
df = pd.merge(names,ages,on='id')
print(df)
print()
print(df.loc[df['age'] > 20])
print(df.loc[df['age'] == 20])
print(df.loc[df['age'] > 30]['name'])
```

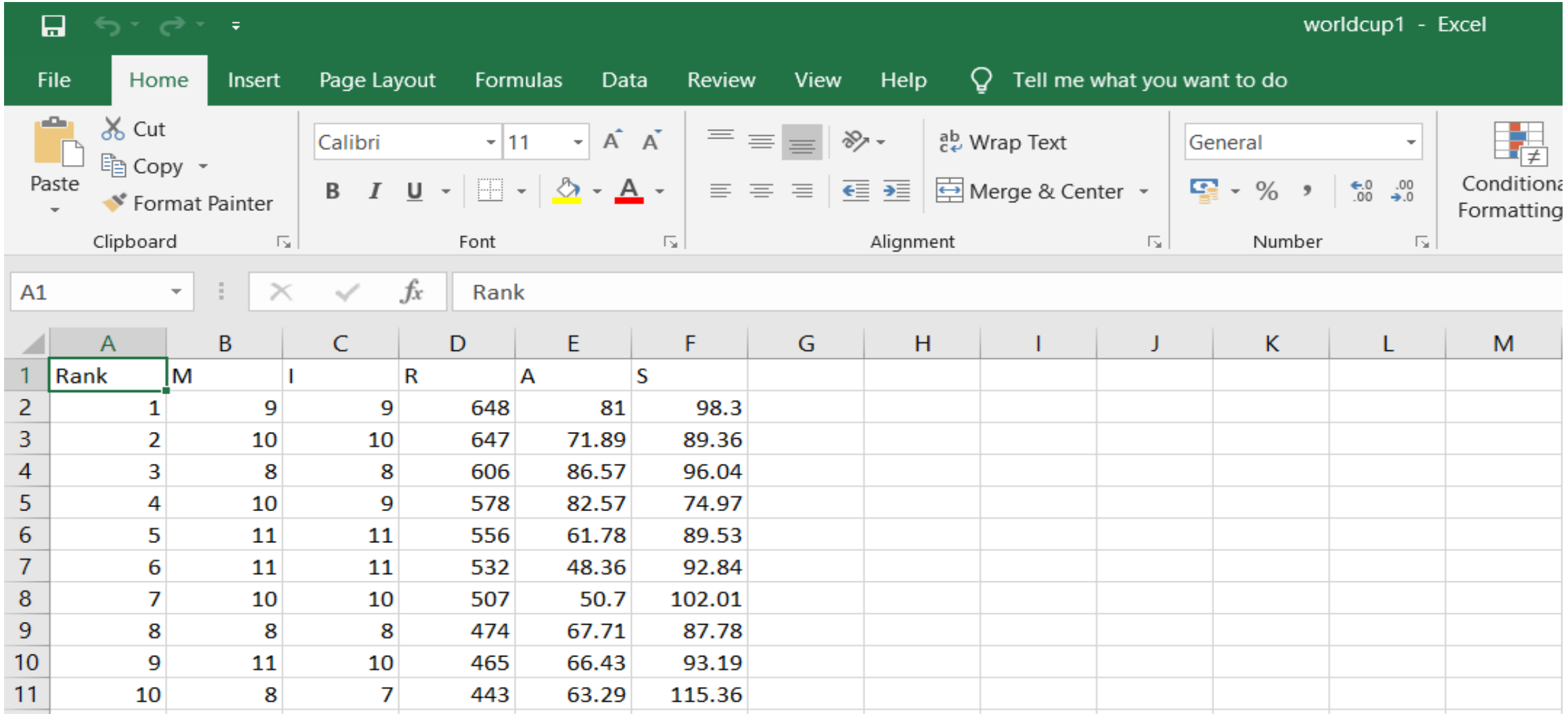
X Working on CSV files

CSV file 1: ’ worldcup.csv’



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Rank	Name	Country	Match	Inn	Runs	Avg	Strike rate	4's	6's	100	50	not out	highest
2	1	R.Sharma	India	9	9	648	81	98.3	67	14	5	1	1	140
3	2	D. Warner	Australia	10	10	647	71.89	89.36	66	8	3	3	1	166
4	3	S. A. Hasar	Banglades	8	8	606	86.57	96.04	60	2	2	5	1	124*
5	4	K. Williams	New Zeala	10	9	578	82.57	74.97	50	3	2	2	2	148
6	5	J. Root	England	11	11	556	61.78	89.53	48	2	2	3	2	107
7	6	J. Bairstow	England	11	11	532	48.36	92.84	67	11	2	2	0	111
8	7	A. Finch	Australia	10	10	507	50.7	102.01	47	18	2	3	0	153
9	8	B. Azam	Pakistan	8	8	474	67.71	87.78	50	2	1	3	1	101*
10	9	B. Stokes	England	11	10	465	66.43	93.19	38	11	0	5	3	89
11	10	J. Roy	England	8	7	443	63.29	115.36	51	12	1	4	0	153

CSV file 2: ’ worldcup1.csv’



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Rank	M	I	R	A	S							
2	1	9	9	648	81	98.3							
3	2	10	10	647	71.89	89.36							
4	3	8	8	606	86.57	96.04							
5	4	10	9	578	82.57	74.97							
6	5	11	11	556	61.78	89.53							
7	6	11	11	532	48.36	92.84							
8	7	10	10	507	50.7	102.01							
9	8	8	8	474	67.71	87.78							
10	9	11	10	465	66.43	93.19							
11	10	8	7	443	63.29	115.36							

Extracting selected columns

```
import pandas as pd
filenames=['worldcup.csv','worldcup1.csv']
newl=[]
for f in filenames:
    newl.append(pd.read_csv(f))
print(newl[0]['Name'])                    #only Name column
print(newl[0][['Country','Match']])       # Country and Match
```

Extracting selected rows

```
import pandas as pd
filenames=['worldcup.csv','worldcup1.csv']
newl=[]
for f in filenames:
    newl.append(pd.read_csv(f))
print(newl[0][0:2])
```

Merging the files

```
import pandas as pd
filenames=['worldcup.csv','worldcup1.csv']
newl=[]
for f in filenames:
    newl.append(pd.read_csv(f))
newl1 = pd.merge(newl[0], newl[1], on='Rank', how='inner')
print(newl)
```