Ex.No.6

Data Wrangling

Aim:

To do Data Wrangling functions

Description:

Data wrangling is the task in data science and analysis which includes operations

like: Data Sorting: To rearrange values in ascending or descending order.

Data Filtration: To create a subset of available data.

Data Reduction: To eliminate or replace unwanted values.

Data Access: To read or write data files.

Data Processing: To perform aggregation, statistical, and similar operations on specific values.

- 1. Using join function to join two DataFrames.
- 2. Using combine function to combine two DataFrames.
- 3. Using merge function to merge two DataFrames.
- 4. Using replace function to replace the NaN values by average value.
- 5. Filtering and dropping the rows and rows and columns respectively.
- 6. Using concat function to concatenate two DataFrames.
- 7. Using melt function to reshape the DataFrame dimention.
- 8. Using groupby function to group the data set.
- 9. Using duplicated function to remove duplicated rows in the DataFram
- 10. Using merge function to merge two DataFrame data sets.

PROGRAM:

```
import pandas as pd
```

```
data1 = {'Name': ['Jai', 'Princi', 'Gaurav', 'Anuj', 'Ravi', 'Natasha', 'Tom', 'Rovana', 'Riya'], 'Roll No': [4,8,2,1,9,7,14,11,10], 'Age': [17, 17, 18, 17, 18, 17,19,16, 17], 'Gender': ['M', 'F', 'M', 'M', 'M', 'F','F',M', 'F']} data2 = {'Name': ['Kelly', 'Natasha', 'Jack', 'Stacy', 'Stark', 'Loki', 'Rovana', 'Tom'], 'Roll No': [5,7,3,12,13,6,11,14], 'Age': [19, 17,16,20, 17, 18, 16, 19], 'Gender': ['F','F','M', 'F', 'M', 'M', 'F', 'M'], 'Marks': [95,71, 76, 94, 'NaN', 80,83, 68]}
```

```
marks = {'Marks': [80, 76, 'NaN', 74, 66,71,68,83, 'NaN']}
df1= pd.DataFrame(data1)
df2= pd.DataFrame(data2)
marks = pd.DataFrame(marks)
print("\nOriginal DataFrame 1:\n",df1)
print("\nOriginal DataFrame 2:\n",df1)
print("\nMarks:\n",marks)
df1 = df1.join(marks) print("\nDataFrame
1:\n",df1)
# Compute average c
= avg=0
for ele in df1['Marks']: if
  str(ele).isnumeric():
     c += 1
     avg += ele
avg/= c
# Replace missing values
df1 = df1.replace(to_replace="NaN",value=avg) df2 =
df2.replace(to_replace="NaN",value=avg) # Display
data
print("\nReplacing NaN with Average marks:\nData Frame 1\n",df1)
print("\n\nData Frame 2\n",df2)
def myfunc(a, b):
  return a if a > b else b
df_combined = df1['Marks'].combine(df2['Marks'], myfunc)
```

```
# Print the result
print("\nCombining the above two DataFrames using combine function with some condition:\n", df_combined)
newdf = df1.merge(df2, how='right')
print("\nMerge operation:\n",newdf)
df3 = pd.concat([df1,df2])
print("\nConcatenated DataFrame using cancat function:\n",df3)
print("\nOriginal DataFrame:\n",df3)
#reshape DataFrame from wide format to long format
df = pd.melt(df3, id_vars='Roll No', value_vars=['Gender', 'Marks']) #view
updated DataFrame
print("\nReshaped Data Frame:\n",df)
# Filter top scoring students
df3=df3[df3['Marks'] >= 75] print("\nAfter
Filtering function:\n",df3) #Remove age
row
df3 = df3.drop(['Age'],axis=1)
# Display data
print("\nAfter Dropping function:\n",df3)
```

print("\nOriginal DataFrame:\n",df3)

- # Here df.duplicated() list duplicate Entries in Rollno.
- # So that ~(NOT) is placed in order to get non duplicate values. non_duplicate

=df3[~df3.duplicated('Roll No')]

#printing non-duplicate values

print("\nRemoved duplicated rows:\n",non_duplicate)

OUTPUT:

Original DataFrame 1:

	Name	Roll No	Age	e Gender	
0	Jai	4	4	17	M
1	Princi		8	17	F
2	Gaurav		2	18	\mathbf{M}
3	Anuj		1	17	M
4	Ravi		9	18	M
5	Natasha		7	17	F
6	Tom	14	4	19	F
7	Rovana	1	1	16	M
8	Riya	10	0	17	F

Original DataFrame 2:

	Name	Roll No Ag	e Gender	Marks	
0	Kelly	5	19	F	95
1	Natasha	7	17	F	71
2	Jack	3	16	M	76
3	Stacy	12	20	F	94
4	Stark	13	17	M	NaN
5	Loki	6	18	M	80
6	Rovana	11	16	F	83
7	Tom	14	19	M	68

Marks:

	er recor
	Marks
0	80
1	76
2	NaN
3	74
4	66
5	71
6	68
7	83
8	NaN

DataFrame 1:

	Name Re	oll No Age	e Gender	Marks	
0	Jai	4	17	M	80
1	Princi	8	17	F	76
2	Gaurav	2	18	M	NaN
3	Anuj	1	17	M	74
4	Ravi	9	18	M	66
5	Natasha	7	17	F	71
6	Tom	14	19	F	68
7	Rovana	11	16	M	83
8	Riva	10	17	F	NaN

Replacing NaN with Average marks:

Data Frame 1

	Name	Roll No	Ag	e Gender	Marks	
0	Jai		4	17	M	80.0
1	Princi		8	17	F	76.0
2	Gaurav		2	18	M	74.0
3	Anuj		1	17	M	74.0
4	Ravi		9	18	M	66.0
5	Natasha		7	17	F	71.0
6	Tom		14	19	F	68.0
7	Rovana		11	16	M	83.0
8	Riya		10	17	F	74.0

Data Frame 2

	Name	Roll No Ag	ge Gender	Marks	
0	Kelly	5	19	F	95.0
1	Natasha	7	17	F	71.0
2	Jack	3	16	M	76.0
3	Stacy	12	20	F	94.0
4	Stark	13	17	M	74.0
5	Loki	6	18	M	80.0
6	Rovana	11	16	F	83.0
7	Tom	14	19	M	68.0

Combining the above two DataFrames using combine function with some condition:

0 95.0 1 76.0 2 76.0 3 94.0 4 74.0 5 80.0 6 83.0 7 83.0 8 NaN

Name: Marks, dtype: float64

Me	roe	0	peration	٦.
1110	150	v	peranoi	1.

	Name	Roll No	Ag	e Gender	Marks	
0	Kelly		5	19	F	95.0
1	Natasha		7	17	F	71.0
2	Jack		3	16	M	76.0
3	Stacy	1	12	20	F	94.0
4	Stark	1	13	17	M	74.0
5	Loki		6	18	M	80.0
6	Rovana	1	11	16	F	83.0
7	Tom]	14	19	M	68.0

Concatenated DataFrame using cancat function: Name Roll No Age Gender Marks

	Ochuci	Marks			
0	Jai	4	17	M	80.0
1	Princi	8	17	F	76.0
2	Gaurav	2	18	M	74.0
3	Anuj	1	17	M	74.0
4	Ravi	9	18	M	66.0
5	Natasha	7	17	F	71.0
6	Tom	14	19	F	68.0
7	Rovana	11	16	M	83.0
8	Riya	10	17	F	74.0
0	Kelly	5	19	F	95.0
1	Natasha	7	17	F	71.0
2	Jack	3	16	M	76.0
3	Stacy	12	20	F	94.0
4	Stark	13	17	M	74.0
5	Loki	6	18	M	80.0
6	Rovana	11	16	F	83.0
7	Tom	14	19	M	68.0

Gro	up by age	17:				
	Name	Roll	No	Age	Gender	Marks
0	Jai		4	17	M	80.0
1	Princi		8	17	F	76.0
3	Anuj		1	17	M	74.0
5	Natasha		7	17	F	71.0
8	Riya		10	17	F	74.0
1	Natasha		7	17	F	71.0
4	Stark		13	17	M	74.0

Ong	inal DataFra		Α	C 1	M - 1 -	
		Roll No		e Gender		
0	Jai		4	17	M	80.0
1	Princi		8	17	F	76.0
2	Gaurav		2	18	M	74.0
3	Anuj		1	17	M	74.0
4	Ravi		9	18	M	66.0
5	Natasha		7	17	F	71.0
6	Tom		14	19	F	68.0
7	Rovana		11	16	M	83.0
8	Riya		10	17	F	74.0
0	Kelly		5	19	F	95.0
1	Natasha		7	17	F	71.0
2	Jack		3	16	M	76.0
3	Stacy		12	20	F	94.0
4	Stark		13	17	M	74.0
5	Loki		6	18	M	80.0
6	Rovana		11	16	F	83.0
7	Tom		14	19	M	68.0

Reshaped Data Frame:

Resnaped Data Frame.							
Roll No variable value							
0	4	Gender	M				
1	8	Gender	F				
2	2	Gender	M				
3	1	Gender	M				
4	9	Gender	M				
5	7	Gender	F				
6	14	Gender	F				
7	11	Gender	M				
8	10	Gender	F				
9	5 7	Gender	F				
10	7	Gender	F				
11	3	Gender	M				
12	12	Gender	F				
13	13	Gender	M				
14	6	Gender	M				
15	11	Gender	F				
16	14	Gender	M				
17	4	Marks	80.0				
18	8	Marks	76.0				
19	2	Marks	74.0				
20	1	Marks	74.0				
21	9	Marks	66.0				
21 22	7	Marks	71.0				
23	14	Marks	68.0				
24	11	Marks	83.0				
25	10	Marks	74.0				
26	5	Marks	95.0				
27	7	Marks	71.0				
28	3	Marks	76.0				
29	12	Marks	94.0				
30	13	Marks	74.0				
31	6	Marks	80.0				
32	11	Marks	83.0				
33	14	Marks	68.0				

After Filtering function:

	Name	Roll No A	Age Gender	Marks	
0	Jai	4	17	M	80.0
1	Princi	8	17	F	76.0
7	Rovana	11	16	M	83.0
0	Kelly	5	19	F	95.0
2	Jack	3	16	M	76.0
3	Stacy	12	20	F	94.0
5	Loki	6	18	M	80.0
6	Rovana	11	16	F	83.0

After Dropping function:

	Name	Roll No Gender		Marks
0	Jai	4	M	80.0
1	Princi	8	F	76.0
7	Rovana	11	M	83.0
0	Kelly	5	F	95.0
2	Jack	3	M	76.0
3	Stacy	12	F	94.0
5	Loki	6	M	80.0
6	Rovana	11	F	83.0

Original DataFrame:

	Name	Roll No Gender		Marks
0	Jai	4	M	80.0
1	Princi	8	F	76.0
7	Rovana	11	M	83.0
0	Kelly	5	F	95.0
2	Jack	3	M	76.0
3	Stacy	12	F	94.0
5	Loki	6	M	80.0
6	Rovana	11	F	83.0

Removed duplicated rows:

	Name	Roll No Gender	Marks	
0	Jai	4	M	80.0
1	Princi	8	F	76.0
7	Rovana	11	M	83.0
0	Kelly	5	F	95.0
2	Jack	3	M	76.0
3	Stacy	12	F	94.0
5	Loki	6	M	80.0

Result:

The programs were run successfully