Name : Tejaswini Sunil Mahale
Practical No.5
Roll No.26 Sub:-DV

Program/Notebook related to any dataset using pandas DataFrame methods: Count() Method, describe() Method, drop_duplicated() Method, empty property, filter() Method, equals() Method

1) Groupby: - It is used to group the data accroding to categories and apply a functionto the categories. It is also used to aggregate the data efficiently

```
#Importing Required Libraries
import pandas as pd

#Reding excel file
df = pd.read_csv('result.csv')
df

#Grouping by role og player
RoleGroup = df.groupby('Role')
#first()  mtehod is used to print first entry from each group
RoleGroup.first()

#count() method is used to count total number of groups
RoleGroup.count()

#Splitting Data intomultiple groups
Role_Filter = df['Role'] =='Batsman'
df[Role_Filter]

Role_Filter = df['Role'] =='Bowler'
df[Role_Filter]

Role_Filter = df['Role'] =='All-Rounder'
df[Role_Filter]

   #Splitting the data and running an aggregation function
Role_Filter = df['Role']=='Batsman'
BatsmanCost = df[Role_Filter]['Cost']
print(BatsmanCost)
print(BatsmanCost.sum())
```

```
0    9.5
1   10.0
2    9.0
8    8.5
Name: Cost, dtype: float64
37.0
```

```
Role_Filter = df['Role']=='Batsman'
BatsmanCost = df[Role_Filter]['Cost'].sum()

Role_Filter = df['Role']=='Bowler'
BowlerCost = df[Role_Filter]['Cost'].sum()

Role_Filter = df['Role']=='All-Rounder'
```

```
All_RounderCost = df[Role_Filter]['Cost'].sum()

Role_Filter = df['Role']=='Wkt-Kepper'
Wkt_KepperCost = df[Role_Filter]['Cost'].sum()

print(BatsmanCost , BowlerCost, All_RounderCost, Wkt_KepperCost )
```

```
37.0 25.5 26.0 8.5
```

```
RoleGroup = df.groupby('Role')['Cost'].sum().sort_values(ascending=False)
RoleGroup
```

```
Role
Batsman        37.0
All-Rounder    26.0
Bowler         25.5
Wkt-Kepper      8.5
Name: Cost, dtype: float64
```

```
RoleTeamGroup = df.groupby(['Role', 'Team'])
RoleTeamGroup.first()
```

Correlations : - It is way to determine if two varibales in dataset are related to each other in any way.It is used find the pairwise correlation of all numerical columns in the DataFrame. Here any non-numeric data type columns in DataFrame is also ignored (-1 to 1)

```
d = pd.read_excel('cricket2.xlsx')
d.head()
```

```
d.corr()
```

```
#Showing the corelation between  two columns
d['Sixes'].corr(d['Fours'])
```

```
0.7663582481705323
```

2) count() method : - prints the number of values in the specified column

```
print(d['Sixes'].count())
```

```
11
```

3) describe() method : - Provides a discription summery for the DataFrame column

```
d.describe()
```

4) drop_dupucates() method :- It is used to drop duplicate values from the DataFrame or from specific columnof the DataFrame

```
dr = pd.read_excel('Fees_Data.xlsx')
dr
```

```
#Droping repeated valuese from Branch column
DropDuplicates = dr['Branch'].drop_duplicates()
print(DropDuplicates)
```

```
0         CIVIL
1            IT
2           NaN
3      Computer
9    Electrical
11        Civil
24   Mechanical
38           EE
Name: Branch, dtype: object
```

- 5) empty Property :- used to determine if DataFrame is empty or not

```
#Checking if the Fees_Data.xlsx is empty or not
print(dr.empty)
```

```
False
```

- 6) filter() method :- Filter the DataFrame and return only the rows/columns that are specified in the filter

```
#Reading Excel File
Data = pd.read_excel('Fees_Data.xlsx')
Data.head()
```

```
#applying filter of give file
filtered = Data.filter(items=['Sr. No','EN', 'Branch', 'Time'])
print(filtered)
```

```
    Sr. No          EN      Branch      Time
0        1  EN23204195       CIVIL   2.45 pm
1        2  EN23146043          IT  11.40 am
2        3  EN23135942         NaN  11.50 am
3        4  EN23135942    Computer  12:00:00
4        5  EN23119584    Computer  12:10:00
..     ...         ...         ...       ...
66      67  EN23273933       Civil  17:00:00
67      68  EN23200937       Civil  11.58 am
68      69  EN23117836  Electrical  11.40 am
69      70  EN23247685       Civil  11.46 am
70      71  EN23237346       Civil  12.00 pm

[71 rows x 4 columns]
```

- 7) equals() method : - Compare two DataFrames to determine if they are equal or not

```
#Copying above file and storing to NewData
NewData = Data.copy()
NewData.head()
```

```
#Checking if  Data and NewData are equal or not
print(Data.equals(NewData))
```

```
True
```

```
Data1 = pd.read_excel('Book1.xlsx')
Data1.head()
```

```
#Cheking if Data And Data1are equal or not
print(Data.equals(Data1))
```

```
False
```