LABS

Data Mining

Eng: Malek Almosanif

Lab 4 => Data Integration by Malek

Pandas Type of Merge

Types of Join Operations In merge()

Explaine:

1-Inner Join (افتراضي) – يحتفظ بالسجلات المشتركة بين الجدولين.(Default) 2-Left Join – يحتفظ بجميع البيانات من الجدول الأيسر ويضيف المطابقة من الجدول الأيمن. 3-Right Join – يحتفظ بجميع البيانات من الجدول الأيمن ويضيف المطابقة من الجدول الأيسر. 4-Outer Join – يحتفظ بجميع البيانات من كلا الجدولين، مع ملء القيم غير الموجودة بـ NaN.

join_Types

Left Join

```
import pandas as pd

# create dataframes from the dictionaries
data1 = {
    'EmployeeID': ['E001', 'E002', 'E003', 'E004', 'E005'],
    'Name': ['John Doe', 'Jane Smith', 'Peter Brown', 'Tom Johnson',
'Rita Patel'],
    'DeptID': ['D001', 'D003', 'D001', 'D002', 'D006'],
}
employees = pd.DataFrame(data1)
print("employees:\n")
print(employees)
print("*******************************")
data2 = {
    'DeptID': ['D001', 'D002', 'D003', 'D004'],
    'DeptName': ['Sales', 'HR', 'Admin', 'Marketing']
}
departments = pd.DataFrame(data2)
print("departments:\n")
print(departments)
print("*******************************")
# left merge the dataframes
df_merge = pd.merge(employees, departments, on = 'DeptID', how =
'left', sort = True)
print("print(df_merge):\n")
print(df_merge)
```

```
employees:

  EmployeeID         Name DeptID
```

```
0       E001     John Doe   D001
1       E002   Jane Smith   D003
2       E003  Peter Brown   D001
3       E004  Tom Johnson   D002
4       E005   Rita Patel   D006
*******************************
departments:

  DeptID   DeptName
0   D001      Sales
1   D002         HR
2   D003      Admin
3   D004  Marketing
*******************************
print(df_merge):

  EmployeeID         Name DeptID DeptName
0       E001     John Doe   D001    Sales
1       E003  Peter Brown   D001    Sales
2       E004  Tom Johnson   D002       HR
3       E002   Jane Smith   D003    Admin
4       E005   Rita Patel   D006      NaN
```

Right Join

```
import pandas as pd

# create dataframes from the dictionaries
data1 = {
    'EmployeeID': ['E001', 'E002', 'E003', 'E004', 'E005'],
    'Name': ['John Doe', 'Jane Smith', 'Peter Brown', 'Tom Johnson', 
'Rita Patel'],
    'DeptID': ['D001', 'D003', 'D001', 'D002', 'D006'],
}
employees = pd.DataFrame(data1)
print("employees:\n")
print(employees)
print("*******************************")
data2 = {
    'DeptID': ['D001', 'D002', 'D003', 'D004'],
    'DeptName': ['Sales', 'HR', 'Admin', 'Marketing']
}
departments = pd.DataFrame(data2)
print("departments:\n")
print(departments)
print("*******************************")
# left merge the dataframes
df_merge = pd.merge(employees, departments, on = 'DeptID', how = 
'right', sort = True)
print("print(df_merge):\n")
print(df_merge)
print("*******************************")

employees:

  EmployeeID         Name DeptID
0       E001     John Doe   D001
1       E002   Jane Smith   D003
2       E003  Peter Brown   D001
3       E004  Tom Johnson   D002
4       E005   Rita Patel   D006
*******************************
departments:

  DeptID   DeptName
0   D001      Sales
1   D002         HR
2   D003      Admin
3   D004  Marketing
*******************************
print(df_merge):

  EmployeeID         Name DeptID   DeptName
0       E001     John Doe   D001      Sales
1       E003  Peter Brown   D001      Sales
2       E004  Tom Johnson   D002         HR
3       E002   Jane Smith   D003      Admin
4        NaN          NaN   D004  Marketing
*******************************
```

Inner Join

```
import pandas as pd

# create dataframes from the dictionaries
data1 = {
    'EmployeeID': ['E001', 'E002', 'E003', 'E004', 'E005'],
    'Name': ['John Doe', 'Jane Smith', 'Peter Brown', 'Tom Johnson',
'Rita Patel'],
    'DeptID': ['D001', 'D003', 'D001', 'D002', 'D006'],
}
employees = pd.DataFrame(data1)
print("employees:\n")
print(employees)
print("*******************************")
data2 = {
    'DeptID': ['D001', 'D002', 'D003', 'D004'],
    'DeptName': ['Sales', 'HR', 'Admin', 'Marketing']
}
departments = pd.DataFrame(data2)
```

```
print("departments:\n")
print(departments)
print("*******************************")
# left merge the dataframes
df_merge = pd.merge(employees, departments, on = 'DeptID', how =
'inner')
print("df_merge:\n")
print(df_merge)
print("*******************************")
```

```
employees:

  EmployeeID         Name DeptID
0       E001     John Doe   D001
1       E002   Jane Smith   D003
2       E003  Peter Brown   D001
3       E004  Tom Johnson   D002
4       E005   Rita Patel   D006
*******************************
departments:

  DeptID   DeptName
0   D001      Sales
1   D002         HR
2   D003      Admin
3   D004  Marketing
*******************************
df_merge:

  EmployeeID         Name DeptID DeptName
0       E001     John Doe   D001    Sales
1       E002   Jane Smith   D003    Admin
2       E003  Peter Brown   D001    Sales
3       E004  Tom Johnson   D002       HR
*******************************
```

Outer Join

```
import pandas as pd

# create dataframes from the dictionaries
data1 = {
    'EmployeeID': ['E001', 'E002', 'E003', 'E004', 'E005'],
    'Name': ['John Doe', 'Jane Smith', 'Peter Brown', 'Tom Johnson',
'Rita Patel'],
    'DeptID': ['D001', 'D003', 'D001', 'D002', 'D006'],
}
employees = pd.DataFrame(data1)
print("employees:\n")
print(employees)
print("*******************************")
data2 = {
    'DeptID': ['D001', 'D002', 'D003', 'D004'],
    'DeptName': ['Sales', 'HR', 'Admin', 'Marketing']
}
departments = pd.DataFrame(data2)
print("departments:\n")
print(departments)
print("*******************************")
# left merge the dataframes
df_merge = pd.merge(employees, departments, on = 'DeptID', how =
'outer',sort=True)
print("df_merge:\n")
print(df_merge)
print("*******************************")
```

```
employees:

  EmployeeID         Name DeptID
0       E001     John Doe   D001
1       E002   Jane Smith   D003
2       E003  Peter Brown   D001
3       E004  Tom Johnson   D002
4       E005   Rita Patel   D006
*******************************
departments:

  DeptID   DeptName
0   D001      Sales
1   D002         HR
2   D003      Admin
3   D004  Marketing
*******************************
df_merge:

  EmployeeID         Name DeptID   DeptName
0       E001     John Doe   D001      Sales
1       E003  Peter Brown   D001      Sales
2       E004  Tom Johnson   D002         HR
3       E002   Jane Smith   D003      Admin
4        NaN          NaN   D004  Marketing
5       E005   Rita Patel   D006        NaN
*******************************
```

Methods

✔ إذا كنت تعمل مع مفاتيح مشتركة، استخدم merge() أو join(). ✔ إذا كنت تريد فقط تكديس البيانات، استخدم concat(). ✔ إذا كنت تضيف صفوفًا جديدة، استخدم concat()

Merge Method:

 merge() → إذا كنت تحتاج إلى دمج البيانات بناءً على عمود مشترك.

```
import pandas as pd

# create dataframes from the dictionaries
data1 = {
    'EmployeeID' : ['E001', 'E002', 'E003', 'E004', 'E005'],
    'Name' : ['John Doe', 'Jane Smith', 'Peter Brown', 'Tom Johnson', 
'Rita Patel'],
    'DeptID': ['D001', 'D003', 'D001', 'D002', 'D003'],
}
employees = pd.DataFrame(data1)
print("Employees:")
print(employees)
print('***********************************************')
data2 = {
    'DeptID': ['D001', 'D002', 'D003'],
    'DeptName': ['Sales', 'HR', 'Admin']
}
departments = pd.DataFrame(data2)
print("Departments:")
print(departments)
print('***********************************************')
# merge dataframes employees and departments
merged_df = pd.merge(employees, departments)

# display DataFrames
print("Merged DataFrame:")
print(merged_df)
```

```
Employees:
  EmployeeID         Name DeptID
0       E001     John Doe   D001
1       E002   Jane Smith   D003
2       E003  Peter Brown   D001
3       E004  Tom Johnson   D002
4       E005   Rita Patel   D003
***********************************************
Departments:
  DeptID DeptName
0   D001    Sales
1   D002       HR
2   D003    Admin
***********************************************
Merged DataFrame:
  EmployeeID         Name DeptID DeptName
0       E001     John Doe   D001    Sales
1       E002   Jane Smith   D003    Admin
```

```
2       E003   Peter Brown   D001     Sales
3       E004   Tom Johnson   D002        HR
4       E005    Rita Patel   D003     Admin
```

Merge DataFrames Based on Keys

```
import pandas as pd

# create dataframes from the dictionaries
data1 = {
    'EmployeeID': ['E001', 'E002', 'E003', 'E004', 'E005'],
    'Name': ['John Doe', 'Jane Smith', 'Peter Brown', 'Tom Johnson',
'Rita Patel'],
    'DeptID1': ['D001', 'D003', 'D001', 'D002', 'D006'],
}
employees = pd.DataFrame(data1)

data2 = {
    'DeptID2': ['D001', 'D002', 'D003', 'D004'],
    'DeptName': ['Sales', 'HR', 'Admin', 'Marketing']
}
departments = pd.DataFrame(data2)

# merge the dataframes
df_merge = pd.merge(employees, departments, left_on='DeptID1',
right_on = 'DeptID2', sort = True)

print(df_merge)

  EmployeeID         Name DeptID1 DeptID2 DeptName
0       E001     John Doe    D001    D001    Sales
1       E003  Peter Brown    D001    D001    Sales
2       E004  Tom Johnson    D002    D002       HR
3       E002   Jane Smith    D003    D003    Admin
```

Join Method:

 join() → شعار .إذا كنت تحتاج إلى دمج البيانات بناءً على الفهرس Markdown

```
import pandas as pd

# إنشاء أول DataFrame
df1 = pd.DataFrame({'Name': ['Ali', 'Sara', 'Omar']}, index=[1, 2, 4])
df2 = pd.DataFrame({'Score': [85, 90, 75]}, index=[1, 2, 3])

merged_df = df1.join(df2,how='inner')
merged_df = df1.join(df2,how='left')
merged_df = df1.join(df2,how='right')
merged_df = df1.join(df2,how='outer')
```

```
print(merged_df)
```

```
   Name  Score
1   Ali   85.0
2  Sara   90.0
3   NaN   75.0
4  Omar    NaN
```

Concat Method:

 concat() → شعار .إذا كنت تريد دمج البيانات عموديًا أو أفقيًا بدون شرط مشترك Markdown

```
import pandas as pd

# إنشاء DataFrame الأول
data1 = {
    'C1': ['A', 'B', 'C'],
    'C2': [2.1, 4.3, -6.5],
    'C3': [23, 14, 64]
}
df1 = pd.DataFrame(data1)
print(df1)
print('***********************')
# إنشاء DataFrame الثاني
data2 = {
    'C1': ['E', 'F', 'G'],
    'C2': [5.2, 0.5, 7.6],
    'C3': [1, 144, 39]
}
df2 = pd.DataFrame(data2)
print(df2)
print('***********************')
# دمج الجدولين باستخدام concat()
df_concat = pd.concat([df1, df2])

# طباعة النتيجة
print(df_concat)
```

```
  C1   C2  C3
0  A  2.1  23
1  B  4.3  14
2  C -6.5  64
***********************
  C1   C2   C3
0  E  5.2    1
1  F  0.5  144
2  G  7.6   39
***********************
```

```
  C1   C2   C3
0  A  2.1   23
1  B  4.3   14
2  C -6.5   64
0  E  5.2    1
1  F  0.5  144
2  G  7.6   39
```

```
import pandas as pd

# إنشاء DataFrame الأول
data1 = {
    'C1': ['A', 'B', 'C'],
    'C2': [2.1, 4.3, -6.5],
    'C3': [23, 14, 64]
}
df1 = pd.DataFrame(data1)
print(df1)
print('***********************')
# إنشاء DataFrame الثاني
data2 = {
    'C1': ['E', 'F', 'G'],
    'C2': [5.2, 0.5, 7.6],
    'C3': [1, 144, 39]
}
df2 = pd.DataFrame(data2)
print(df2)
print('***********************')
# دمج الجدولين باستخدام concat()
df_concat = pd.concat([df1, df2],ignore_index=True)

# طباعة النتيجة
print(df_concat)
```

```
  C1   C2  C3
0  A  2.1  23
1  B  4.3  14
2  C -6.5  64
***********************
  C1   C2   C3
0  E  5.2    1
1  F  0.5  144
2  G  7.6   39
***********************
  C1   C2   C3
0  A  2.1   23
1  B  4.3   14
2  C -6.5   64
3  E  5.2    1
```

```
4  F  0.5  144
5  G  7.6   39
```

```
import pandas as pd

# إنشاء DataFrame الأول
data1 = {
    'C1': ['A', 'B', 'C'],
    'C2': [2.1, 4.3, -6.5],
    'C3': [23, 14, 64]
}
df1 = pd.DataFrame(data1)
print(df1)
print('***********************')
# إنشاء DataFrame الثاني
data2 = {
    'C4': ['E', 'F', 'G'],
    'C5': [5.2, 0.5, 7.6],
    'C3': [1, 144, 39]
}
df2 = pd.DataFrame(data2)
print(df2)
print('***********************')
# دمج الجدولين باستخدام concat()
df_concat = pd.concat([df1, df2],axis=1)

# طباعة النتيجة
print(df_concat)
```

```
  C1   C2  C3
0  A  2.1  23
1  B  4.3  14
2  C -6.5  64
***********************
  C4   C5   C3
0  E  5.2    1
1  F  0.5  144
2  G  7.6   39
***********************
  C1   C2  C3 C4   C5   C3
0  A  2.1  23  E  5.2    1
1  B  4.3  14  F  0.5  144
2  C -6.5  64  G  7.6   39
```

```
import pandas as pd

# إنشاء DataFrame الأول
data1 = {
    'C1': ['A', 'B', 'C'],
    'C2': [2.1, 4.3, -6.5],
    'C3': [23, 14, 64]
}
df1 = pd.DataFrame(data1)
print(df1)
print('***********************')
# إنشاء DataFrame الثاني
data2 = {
    'C4': ['E', 'F', 'G'],
    'C5': [5.2, 0.5, 7.6],
    'C3': [1, 144, 39]
}
df2 = pd.DataFrame(data2)
print(df2)
print('***********************')
# دمج الجدولين باستخدام concat()
df_concat = pd.concat([df1, df2],axis=0)

# طباعة النتيجة
print(df_concat)
```

```
  C1   C2  C3
0  A  2.1  23
1  B  4.3  14
2  C -6.5  64
***********************
  C4   C5   C3
0  E  5.2    1
1  F  0.5  144
2  G  7.6   39
***********************
    C1   C2   C3   C4   C5
0    A  2.1   23  NaN  NaN
1    B  4.3   14  NaN  NaN
2    C -6.5   64  NaN  NaN
0  NaN  NaN    1    E  5.2
1  NaN  NaN  144    F  0.5
2  NaN  NaN   39    G  7.6
```

Work With DataSet

```
import pandas as pd

dataset1=pd.read_csv('student_data2.csv')
dataset2=pd.read_json('student_data2.json')

dataset1.head()
```

```
   Unnamed: 0  StudentID  gender student_race  parental_education  \
0           0        663  female      Class C         high school
1           1        287  female      Class B    some high school
```

```
2           2        626    male      Class B  associate's degree
3           3        686    male      Class E        some college
4           4        773  female      Class C   bachelor's degree

          lunch test_preparation_course math_Score  reading_Score  \
0      standard                    none        mid           69.0
1      standard                    none        mid           89.0
2  free/reduced               completed        mid           70.0
3      standard               completed       high           75.0
4  free/reduced                    none        mid           78.0

   writing_score  Studed_Hour
0             67           11
1             82            1
2             63           10
3             68            2
4             79            7
```

```
dataset2.head()
```

```
   std_ID  Sex race_ethnicity parental_level_of_education
lunch  \
0     158    2        group B          associate's degree
standard
1   20932    1        group C                some college
free/reduced
2     291    1        group D            some high school
standard
3     538    1        group E           bachelor's degree
standard
4     367    1        group C           bachelor's degree
free/reduced

  test_preparation_course  math_digree  reading_digree  writing_score
\
0               completed           61              86             87

1               completed           67              64             70

2                    none           86              73             70

3               completed           85              66             71

4                    none           61              66             61


   Sumation     Avarge
0       234  78.000000
1       201  67.000000
2       229  76.333333
```

```
3        222  74.000000
4        188  62.666667
```

```
dataset1.isna().sum()
```

```
Unnamed: 0                  0
StudentID                   0
gender                      0
student_race                0
parental_education          0
lunch                       0
test_preparation_course     0
math_Score                  0
reading_Score               0
writing_score               0
Studed_Hour                 0
dtype: int64
```

```
dataset2.isna().sum()
```

```
std_ID                          0
Sex                             0
race_ethnicity                  0
parental_level_of_education     0
lunch                           0
test_preparation_course         0
math_digree                     0
reading_digree                  0
writing_score                   0
Sumation                        0
Avarge                          0
dtype: int64
```

```
dataset1.shape
```

```
(381, 11)
```

```
dataset2.shape
```

```
(381, 11)
```

```
dataset1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 11 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Unnamed: 0               381 non-null    int64
 1   StudentID                381 non-null    int64
 2   gender                   381 non-null    object
 3   student_race                381 non-null    object
 4   parental_education          381 non-null    object
 5   lunch                       381 non-null    object
 6   test_preparation_course     381 non-null    object
 7   math_Score                  381 non-null    object
 8   reading_Score               381 non-null    float64
 9   writing_score               381 non-null    int64
 10  Studed_Hour                 381 non-null    int64
dtypes: float64(1), int64(4), object(6)
memory usage: 32.9+ KB
```

```
dataset2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 381 entries, 0 to 380
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   std_ID                       381 non-null    int64
 1   Sex                          381 non-null    int64
 2   race_ethnicity               381 non-null    object
 3   parental_level_of_education  381 non-null    object
 4   lunch                        381 non-null    object
 5   test_preparation_course      381 non-null    object
 6   math_digree                  381 non-null    int64
 7   reading_digree               381 non-null    int64
 8   writing_score                381 non-null    int64
 9   Sumation                     381 non-null    int64
 10  Avarge                       381 non-null    float64
dtypes: float64(1), int64(6), object(4)
memory usage: 35.7+ KB
```

Drop The Column Unnamed: 0 in dataset1

```
dataset1.drop(columns=['Unnamed: 0'], inplace=True)

dataset1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   StudentID                381 non-null    int64
 1   gender                   381 non-null    object
 2   student_race             381 non-null    object
 3   parental_education       381 non-null    object
 4   lunch                    381 non-null    object
 5   test_preparation_course  381 non-null    object
 6   math_Score               381 non-null    object
 7   reading_Score              381 non-null    float64
 8   writing_score              381 non-null    int64
 9   Studed_Hour                381 non-null    int64
dtypes: float64(1), int64(3), object(6)
memory usage: 29.9+ KB
```

```
dataset1.shape
```

```
(381, 10)
```

Rename The Columns Name

```
dataset1.rename(columns={
    'StudentID':'std_ID ',
    'gender':'Sex',
    'student_race':'race_ethnicity',
    'parental_education':'parental_level_of_education',
    'math_Score':'math_digree',
    'reading_Score':'reading_digree'},inplace=True)
```

```
dataset1.columns
```

```
Index(['std_ID ', 'Sex', 'race_ethnicity',
'parental_level_of_education',
       'lunch', 'test_preparation_course', 'math_digree',
'reading_digree',
       'writing_score', 'Studed_Hour'],
      dtype='object')
```

```
dataset2.columns
```

```
Index(['std_ID', 'Sex', 'race_ethnicity',
'parental_level_of_education',
       'lunch', 'test_preparation_course', 'math_digree',
'reading_digree',
       'writing_score', 'Sumation', 'Avarge'],
      dtype='object')
```

DataType Solve

```
dataset1.Sex.unique()
```

```
array(['female', 'male'], dtype=object)
```

```
dataset2.Sex.unique()
```

```
array([2, 1], dtype=int64)
```

```
for i in range(len(dataset1['Sex'])):
    if dataset1.loc[i,'Sex']=='male':
        dataset1.loc[i,'Sex']=1
    else:
        dataset1.loc[i,'Sex']=2

dataset1['Sex']=dataset2['Sex'].astype('int64')

dataset1.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 10 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   std_ID                       381 non-null    int64  
 1   Sex                          381 non-null    int64  
 2   race_ethnicity               381 non-null    object 
 3   parental_level_of_education  381 non-null    object 
 4   lunch                        381 non-null    object 
 5   test_preparation_course      381 non-null    object 
 6   math_digree                  381 non-null    object 
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64  
 9   Studed_Hour                  381 non-null    int64  
dtypes: float64(1), int64(4), object(5)
memory usage: 29.9+ KB

dataset2['reading_digree']=dataset2['reading_digree'].astype('float64'
)

dataset2.info()

<class 'pandas.core.frame.DataFrame'>
Index: 381 entries, 0 to 380
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   std_ID                       381 non-null    int64  
 1   Sex                          381 non-null    int64  
 2   race_ethnicity               381 non-null    object 
 3   parental_level_of_education  381 non-null    object 
 4   lunch                        381 non-null    object 
 5   test_preparation_course      381 non-null    object 
 6   math_digree                  381 non-null    int64  
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64  
 9   Sumation                     381 non-null    int64  
 10  Avarge                       381 non-null    float64
dtypes: float64(2), int64(5), object(4)
memory usage: 35.7+ KB
```

Data Type And Values Solve

```
dataset1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 10 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   std_ID                       381 non-null    int64  
 1   Sex                          381 non-null    int64  
 2   race_ethnicity               381 non-null    object 
 3   parental_level_of_education  381 non-null    object 
 4   lunch                        381 non-null    object 
 5   test_preparation_course      381 non-null    object 
 6   math_digree                  381 non-null    object 
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64  
 9   Studed_Hour                  381 non-null    int64  
dtypes: float64(1), int64(4), object(5)
memory usage: 29.9+ KB
```

```
dataset2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 381 entries, 0 to 380
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   std_ID                       381 non-null    int64  
 1   Sex                          381 non-null    int64  
 2   race_ethnicity               381 non-null    object 
 3   parental_level_of_education  381 non-null    object 
 4   lunch                        381 non-null    object 
 5   test_preparation_course      381 non-null    object 
 6   math_digree                  381 non-null    int64  
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64  
 9   Sumation                     381 non-null    int64  
 10  Avarge                       381 non-null    float64
dtypes: float64(2), int64(5), object(4)
memory usage: 35.7+ KB
```

```
dataset1.math_digree.unique()
```

```
array(['mid', 'high', 'low'], dtype=object)
```

```
dataset2.math_digree.unique()
```

```
array([ 61,  67,  86,  85,  42,  82,  47,  49,  72,  69,  59,  91,
35,
       100,  65,  76,  32,  68,  50,  63,  87,  75,  53,  52,  73,
77,
        39,  57,  70,  40,  45,  78,  54,  64,  94,  58,  81,  92,
62,
        74,  66,  98,  55,  90,  29,  84,  89,  51,  43,  79,  56,
46,
        99,  44,  93,   0,  83,  80,  71,  30,  95,   8,  48,  88,
27,
        60,  97,  36,  37,  41,  33,  28,  23,  96], dtype=int64)

import numpy as np
def generate_random_score(category):
    if category == 'high':
        return np.random.randint(85, 101)
    elif category == 'mid':
        return np.random.randint(60, 85)
    elif category == 'low':
        return np.random.randint(0, 60)
    else:
        return np.nan

dataset1['math_digree'] =
dataset1['math_digree'].apply(generate_random_score)

dataset1.math_digree.unique()

array([ 74,  60,  66,  86,  77,  65,  40,   2,  67,  82,   4,  55,
78,
        73,   1,  87,  63,  29,  85,   0,  71,  10,  98,  70,  79,
83,
        39,  80,  88,  97,  72,  22,  64,  95,  81,  84,  33,  31,
75,
        62,  24,  23,  26,   9,  25,  58,  76,  46,  69,  32,  49,
27,
        61,  90,  48,  21,  52,  59,  93,  99,  94,  38,  47,  13,
30,
        56,  96,  14,  50,  89,  68,  16,  34, 100,   5,   6,  35,
43,
        36,  42,  44,  37,   3,  92,  17,  12,  18,  53,   8,  19],
      dtype=int64)

dataset1.math_digree=dataset1.math_digree.astype('int64')

dataset1.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 10 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   std_ID                       381 non-null    int64
 1   Sex                          381 non-null    int64
 2   race_ethnicity               381 non-null    object
 3   parental_level_of_education  381 non-null    object
 4   lunch                        381 non-null    object
 5   test_preparation_course      381 non-null    object
 6   math_digree                  381 non-null    int64
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64
 9   Studed_Hour                  381 non-null    int64
dtypes: float64(1), int64(5), object(4)
memory usage: 29.9+ KB
```

```
dataset2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 381 entries, 0 to 380
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   std_ID                       381 non-null    int64
 1   Sex                          381 non-null    int64
 2   race_ethnicity               381 non-null    object
 3   parental_level_of_education  381 non-null    object
 4   lunch                        381 non-null    object
 5   test_preparation_course      381 non-null    object
 6   math_digree                  381 non-null    int64
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64
 9   Sumation                     381 non-null    int64
 10  Avarge                       381 non-null    float64
dtypes: float64(2), int64(5), object(4)
memory usage: 35.7+ KB
```

Solve The Column Sumation and Avarge on DataSet1

```
dataset1['Sumation']=dataset1['math_digree']
+dataset1['reading_digree']+dataset1['writing_score']
```

```
dataset1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   std_ID                       381 non-null    int64
 1   Sex                          381 non-null    int64
 2   race_ethnicity               381 non-null    object
 3   parental_level_of_education  381 non-null    object
 4   lunch                        381 non-null    object
 5   test_preparation_course      381 non-null    object
 6   math_digree                  381 non-null    int64
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64
 9   Studed_Hour                  381 non-null    int64
 10  Sumation                     381 non-null    float64
dtypes: float64(2), int64(5), object(4)
memory usage: 32.9+ KB
```

```
dataset1['Avarge']=dataset1['Sumation']/3
```

```
dataset1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 12 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   std_ID                       381 non-null    int64
 1   Sex                          381 non-null    int64
 2   race_ethnicity               381 non-null    object
 3   parental_level_of_education  381 non-null    object
 4   lunch                        381 non-null    object
 5   test_preparation_course      381 non-null    object
 6   math_digree                  381 non-null    int64
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64
 9   Studed_Hour                  381 non-null    int64
 10  Sumation                     381 non-null    float64
 11  Avarge                       381 non-null    float64
dtypes: float64(3), int64(5), object(4)
memory usage: 35.8+ KB
```

```
dataset1['Sumation']=dataset1['Sumation'].astype('int64')
```

```
dataset1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 12 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   std_ID                       381 non-null    int64
 1   Sex                          381 non-null    int64
 2   race_ethnicity               381 non-null    object
 3   parental_level_of_education  381 non-null    object
 4   lunch                        381 non-null    object
 5   test_preparation_course      381 non-null    object
 6   math_digree                  381 non-null    int64
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64
 9   Studed_Hour                    381 non-null    int64  
 10  Sumation                       381 non-null    int64  
 11  Avarge                         381 non-null    float64
dtypes: float64(2), int64(6), object(4)
memory usage: 35.8+ KB
```

Solve The Studed_Hour Column on DataSet1

```
dataset1.drop(columns='Studed_Hour',inplace=True)

dataset1.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381 entries, 0 to 380
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   std_ID                       381 non-null    int64  
 1   Sex                          381 non-null    int64  
 2   race_ethnicity               381 non-null    object 
 3   parental_level_of_education  381 non-null    object 
 4   lunch                        381 non-null    object 
 5   test_preparation_course      381 non-null    object 
 6   math_digree                  381 non-null    int64  
 7   reading_digree               381 non-null    float64
 8   writing_score                381 non-null    int64  
 9   Sumation                     381 non-null    int64  
 10  Avarge                       381 non-null    float64
dtypes: float64(2), int64(5), object(4)
memory usage: 32.9+ KB
```

ConCatnet The Two DataSet

```
newDataSet=pd.concat([dataset1,dataset1],ignore_index=True)

dataset1.shape

(381, 11)

dataset2.shape

(381, 11)

newDataSet.shape

(762, 11)

newDataSet.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 762 entries, 0 to 761
```

```
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   std_ID                       762 non-null    int64  
 1   Sex                          762 non-null    int64  
 2   race_ethnicity               762 non-null    object 
 3   parental_level_of_education  762 non-null    object 
 4   lunch                        762 non-null    object 
 5   test_preparation_course      762 non-null    object 
 6   math_digree                  762 non-null    int64  
 7   reading_digree               762 non-null    float64
 8   writing_score                762 non-null    int64  
 9   Sumation                     762 non-null    int64  
 10  Avarge                       762 non-null    float64
dtypes: float64(2), int64(5), object(4)
memory usage: 65.6+ KB
```

```
newDataSet.isna().sum()
```

```
std_ID                         0
Sex                            0
race_ethnicity                 0
parental_level_of_education    0
lunch                          0
test_preparation_course        0
math_digree                    0
reading_digree                 0
writing_score                  0
Sumation                       0
Avarge                         0
dtype: int64
```

```
newDataSet.columns
```

```
Index(['std_ID ', 'Sex', 'race_ethnicity',
'parental_level_of_education',
       'lunch', 'test_preparation_course', 'math_digree',
'reading_digree',
       'writing_score', 'Sumation', 'Avarge'],
      dtype='object')
```

```
newDataSet.head()
```

```
   std_ID   Sex race_ethnicity parental_level_of_education
lunch  \
0      663    2        Class C                 high school
standard
1      287    1        Class B            some high school
standard
2      626    1        Class B          associate's degree
free/reduced
```

```
3      686    1        Class E                some college
standard
4      773    1        Class C           bachelor's degree
free/reduced

  test_preparation_course  math_digree  reading_digree  writing_score
\
0                    none           63            69.0             67

1                    none           62            89.0             82

2               completed           83            70.0             63

3               completed           85            75.0             68

4                    none           75            78.0             79


   Sumation     Avarge
0       199  66.333333
1       233  77.666667
2       216  72.000000
3       228  76.000000
4       232  77.333333
```

```
newDataSet.tail()
```

```
     std_ID   Sex race_ethnicity parental_level_of_education
lunch  \
757    20894    1        Class A             master's degree
free/reduced
758    20946    2        Class B                 high school
standard
759    20989    1        Class A                 high school
standard
760      861    1        Class E             master's degree
free/reduced
761    20980    2        Class D                 high school
standard

    test_preparation_course  math_digree  reading_digree
writing_score  \
757                    none           45            57.0
73
758                    none            1            68.0
66
759                    none           18            51.0
57
760                    none           63            86.0
87
```

```
761                  completed           41              41.0
47

     Sumation     Avarge
757       175  58.333333
758       135  45.000000
759       126  42.000000
760       236  78.666667
761       129  43.000000

newDataSet.to_csv('Intgreted_data_set.csv')
```