Data Wrangling 101: Best Functions

1. drop() --Removes a column or row from the dataset
2. dropna() - Removes any rows with missing values.
3. fillna() - Fills missing values with a specified value or method.
4. drop_duplicates() - Removes duplicate rows from a DataFrame.
5. replace() - Replaces specific values with another value.
6. rename() - Renames columns or rows in a DataFrame.
7. str.replace() - Replaces a specific substring in a string column with another substring.
8. groupby() - Groups data based on a specified column and applies a function to each group.
9. pivot_table() - Creates a pivot table from a DataFrame.
10. merge() - Merges two DataFrames based on a common column.
11. where() -Use conditional logic to assign value.
12. transform() - Applies a function to each group in the DataFrame

In [1]:
```
#bringing in our data
import pandas as pd
import numpy as np
df = pd.read_csv('student_scores.csv')
df = df.drop('Unnamed: 0',axis=1)
df.head()
```

Out[1]:

	Name	Email	Age	Gender	City	Country	Math Score	Science Score
0	Joshua Pearson	ronaldlewis@example.com	44.0	Other	North Scottbury	Montserrat	10.0	22
1	Tommy Cole	swatson@example.com	53.0	Male	Lake Loganburgh	Equatorial Guinea	4.0	15
2	John Brock	georgesteven@example.org	50.0	Female	Ericchester	Sierra Leone	2.0	63
3	Steven Byrd	jessejenkins@example.net	34.0	Other	New Scotthaven	Sao Tome and Principe	57.0	86
4	Jose Anderson	vmcclain@example.net	55.0	Female	East Miafort	Germany	100.0	75

In [2]:
```
#Lets bring the English Scores
df2 = pd.read_csv('English_grades.csv')
df2 = df2.drop('Unnamed: 0',axis=1)
df2.head()
```

Out[2]:

	Name	English Score
0	Joshua Pearson	20
1	Tommy Cole	41
2	John Brock	31
3	Steven Byrd	18
4	Jose Anderson	59

In [3]:
```
df = df.merge(df2,on='Name')
df.head()
```

Out[3]:

	Name	Email	Age	Gender	City	Country	Math Score	Science Score	English Score
0	Joshua Pearson	ronaldlewis@example.com	44.0	Other	North Scottbury	Montserrat	10.0	22	20
1	Joshua Pearson	ronaldlewis@example.com	44.0	Other	North Scottbury	Montserrat	10.0	22	20
2	Tommy Cole	swatson@example.com	53.0	Male	Lake Loganburgh	Equatorial Guinea	4.0	15	41
3	Tommy Cole	swatson@example.com	53.0	Male	Lake Loganburgh	Equatorial Guinea	4.0	15	41
4	John Brock	georgesteven@example.org	50.0	Female	Ericchester	Sierra Leone	2.0	63	31

In [4]:

```
#checking the dataframe info

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 110 entries, 0 to 109
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Name           110 non-null    object 
 1   Email          110 non-null    object 
 2   Age            109 non-null    float64
 3   Gender         110 non-null    object 
 4   City           110 non-null    object 
 5   Country        110 non-null    object 
 6   Math Score     105 non-null    float64
 7   Science Score  110 non-null    int64  
 8   English Score  110 non-null    int64  
dtypes: float64(2), int64(2), object(5)
memory usage: 8.6+ KB
```

In [5]:

```
# filling null values
avg_age  = df['Age'].mean()
df['Age'] = df['Age'].fillna(avg_age)
```

```
df['Math Score'] =  df['Math Score'].fillna(0)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 110 entries, 0 to 109
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Name           110 non-null    object 
 1   Email          110 non-null    object 
 2   Age            110 non-null    float64
 3   Gender         110 non-null    object 
 4   City           110 non-null    object 
 5   Country        110 non-null    object 
 6   Math Score     110 non-null    float64
 7   Science Score  110 non-null    int64  
 8   English Score  110 non-null    int64  
dtypes: float64(2), int64(2), object(5)
memory usage: 8.6+ KB
```

In [6]:
```
#unique values
df['City'].unique()
```

```
Out[6]: array(['North Scottbury', 'Lake Loganburgh', 'Ericchester',
               'New Scotthaven', 'East Miafort', 'Morrismouth', 'Sandraburgh',
               'Lake Nicole', 'Port Joshua', 'North Brandonberg', 'North Cory',
               'East Nathanhaven', 'Pearsonchester', 'Michaelton',
               'Lake Benjaminfort', 'Dianeville', 'Calderonborough',
               'Palmerville', 'Castilloton', 'Chrismouth', 'Youngstad',
               'Campbellhaven', 'South Leroy', 'South Kimberly', 'Murphyside',
               'Lake Leroyfurt', 'New Michael', 'Romeromouth', 'Whitneyberg',
               'Port Matthewburgh', 'Lake Jenniferton', 'Lake Ashley',
               'New Thomas', 'Jacobchester', 'Waltersstad', 'Woodmouth',
               'Thomasbury', 'West Douglasmouth', 'Mccarthyfurt', 'New Julietown',
               'West Brian', 'Conleyland', 'Edwardshire', 'Munozchester',
               'New Gregory', 'Michaelland', 'Leefurt', 'Leeton', 'Salazarbury',
               'Port Josephchester', 'Courtneymouth', 'Port Ericport', 'Hillberg',
               'Karenside', 'East Richard', 'Laneland', 'East Jadebury',
               'Snowville', 'Coltonstad', 'Parkerland', 'Lake Gary',
               'Rebeccastad', 'Sueview', 'Port Stephen', 'Lake Travis',
               'Juanfort', 'South Marychester', 'New Jeremiahshire',
               'Taraborough', 'Davisview', 'Port Andrew', 'Mccarthybury',
               'Hillside', 'East Ann', 'West Jamesburgh', 'Port Angelashire',
               'Lake Cassandra', 'North Kellyfort', 'Jessicaberg', 'Mccartyberg',
               'South Kevin', 'South Anthonyside', 'North Crystalport',
               'East Timothyport', 'Amandaborough', 'Mcdonaldshire',
               'Port Vincentside', 'North Jeffreyborough', 'Joshuaport',
               'Tiffanyfurt', 'Pamelabury', 'East Megan', 'Lake Kathryn',
               'Anneport', 'New Kevinland', 'East Adam', 'Thomaschester',
               'Patelberg', 'Lake Dana', 'New Scottville'], dtype=object)
```

```
In [7]: #replacing values and strings
        df['City'] = df['City'].str.replace('Port','Pt.')
        df['City'].unique()
```

```
Out[7]: array(['North Scottbury', 'Lake Loganburgh', 'Ericchester',
               'New Scotthaven', 'East Miafort', 'Morrismouth', 'Sandraburgh',
               'Lake Nicole', 'Pt. Joshua', 'North Brandonberg', 'North Cory',
               'East Nathanhaven', 'Pearsonchester', 'Michaelton',
               'Lake Benjaminfort', 'Dianeville', 'Calderonborough',
               'Palmerville', 'Castilloton', 'Chrismouth', 'Youngstad',
               'Campbellhaven', 'South Leroy', 'South Kimberly', 'Murphyside',
               'Lake Leroyfurt', 'New Michael', 'Romeromouth', 'Whitneyberg',
               'Pt. Matthewburgh', 'Lake Jenniferton', 'Lake Ashley',
               'New Thomas', 'Jacobchester', 'Waltersstad', 'Woodmouth',
               'Thomasbury', 'West Douglasmouth', 'Mccarthyfurt', 'New Julietown',
               'West Brian', 'Conleyland', 'Edwardshire', 'Munozchester',
               'New Gregory', 'Michaelland', 'Leefurt', 'Leeton', 'Salazarbury',
               'Pt. Josephchester', 'Courtneymouth', 'Pt. Ericport', 'Hillberg',
               'Karenside', 'East Richard', 'Laneland', 'East Jadebury',
               'Snowville', 'Coltonstad', 'Parkerland', 'Lake Gary',
               'Rebeccastad', 'Sueview', 'Pt. Stephen', 'Lake Travis', 'Juanfort',
               'South Marychester', 'New Jeremiahshire', 'Taraborough',
               'Davisview', 'Pt. Andrew', 'Mccarthybury', 'Hillside', 'East Ann',
               'West Jamesburgh', 'Pt. Angelashire', 'Lake Cassandra',
               'North Kellyfort', 'Jessicaberg', 'Mccartyberg', 'South Kevin',
               'South Anthonyside', 'North Crystalport', 'East Timothyport',
               'Amandaborough', 'Mcdonaldshire', 'Pt. Vincentside',
               'North Jeffreyborough', 'Joshuaport', 'Tiffanyfurt', 'Pamelabury',
               'East Megan', 'Lake Kathryn', 'Anneport', 'New Kevinland',
               'East Adam', 'Thomaschester', 'Patelberg', 'Lake Dana',
               'New Scottville'], dtype=object)
```

```
In [8]: #replacing values and value count
        df['Gender'] = df['Gender'].replace({'Male':'M','Female':'F','Other':'O'})
        df['Gender'].value_counts(normalize=True)
```

```
Out[8]: F    0.372727
        O    0.345455
        M    0.281818
        Name: Gender, dtype: float64
```

```
In [9]: #grouping function
        #df.groupby('Gender')['Math Score'].sum()
        df.groupby('Gender')['Math Score'].agg(['sum','mean'])
```

Out[9]:

Gender	sum	mean
F	2250.0	54.878049
M	1164.0	37.548387
O	1762.0	46.368421

In [10]:

```
#pivot table
import numpy as np
pd.pivot_table(df, values = ['Math Score','Science Score','English Score'],
               index='Gender', aggfunc=np.mean, margins=True)
```

Out[10]:

Gender	English Score	Math Score	Science Score
F	41.560976	54.878049	57.463415
M	36.580645	37.548387	44.483871
O	41.263158	46.368421	51.447368
All	40.054545	47.054545	51.727273

In [11]:

```
#Transform
df['Total Score'] = df['Math Score'] + df['English Score'] + df['Science Score']
df['Average per Group'] = df.groupby('Gender')['Total Score'].transform('mean')
df.head()
```

Out[11]:

	Name	Email	Age	Gender	City	Country	Math Score	Science Score	English Score	Total Score	Average per Group
0	Joshua Pearson	ronaldlewis@example.com	44.0	O	North Scottbury	Montserrat	10.0	22	20	52.0	139.078947
1	Joshua Pearson	ronaldlewis@example.com	44.0	O	North Scottbury	Montserrat	10.0	22	20	52.0	139.078947
2	Tommy Cole	swatson@example.com	53.0	M	Lake Loganburgh	Equatorial Guinea	4.0	15	41	60.0	118.612903
3	Tommy Cole	swatson@example.com	53.0	M	Lake Loganburgh	Equatorial Guinea	4.0	15	41	60.0	118.612903
4	John Brock	georgesteven@example.org	50.0	F	Ericchester	Sierra Leone	2.0	63	31	96.0	153.902439

In [ ]:
```
# numpy conditional
```

In [15]:
```
avg_math  = df['Math Score'].mean()
df['Above Average Math Score' ] = np.where(df['Math Score'] >avg_math,'Yes','No')
df['Above Average Math Score' ].value_counts(normalize =True)
```

Out[15]:
```
No     0.509091
Yes    0.490909
Name: Above Average Math Score, dtype: float64
```

In [32]:
```
# function creation
def above_avg(df,col,new_column):
    col_mean = df[col].mean()
    df[new_column] = np.where(df[col] >col_mean,'Yes','No')
    return df
```

In [33]:
```
above_avg(df,'Science Score', 'Science Score Above Avg')
above_avg(df,'English Score', 'English Score Above Avg')
```

Out[33]:

	Name	Email	Age	Gender	City	Country	Math Score	Science Score	English Score	Total Score	Average per Group
0	Joshua Pearson	ronaldlewis@example.com	44.0	O	North Scottbury	Montserrat	10.0	22	20	52.0	139.078947
1	Joshua Pearson	ronaldlewis@example.com	44.0	O	North Scottbury	Montserrat	10.0	22	20	52.0	139.078947
2	Tommy Cole	swatson@example.com	53.0	M	Lake Loganburgh	Equatorial Guinea	4.0	15	41	60.0	118.612903
3	Tommy Cole	swatson@example.com	53.0	M	Lake Loganburgh	Equatorial Guinea	4.0	15	41	60.0	118.612903
4	John Brock	georgesteven@example.org	50.0	F	Ericchester	Sierra Leone	2.0	63	31	96.0	153.902439
...	...	...	...	...	...	...	...	...	...	...	...
105	Sheila Aguilar	rsmith@example.com	22.0	O	East Adam	Gabon	24.0	1	55	80.0	139.078947
106	Brittany Poole	joshuatorres@example.org	61.0	F	Thomaschester	Nepal	9.0	8	19	36.0	153.902439
107	Alicia Taylor	janice39@example.com	58.0	F	Patelberg	Saint Martin	83.0	100	32	215.0	153.902439
108	Ann Santos	janet15@example.org	18.0	F	Lake Dana	Mauritania	88.0	99	52	239.0	153.902439
109	Anthony Murphy	dustin05@example.net	24.0	M	New Scottville	Libyan Arab Jamahiriya	56.0	90	39	185.0	118.612903

110 rows × 14 columns

In [34]:
```
df.head()
```

Out[34]:

	Name	Email	Age	Gender	City	Country	Math Score	Science Score	English Score	Total Score	Average per Group	Abo Avera Ma Sco
0	Joshua Pearson	ronaldlewis@example.com	44.0	O	North Scottbury	Montserrat	10.0	22	20	52.0	139.078947	N
1	Joshua Pearson	ronaldlewis@example.com	44.0	O	North Scottbury	Montserrat	10.0	22	20	52.0	139.078947	N
2	Tommy Cole	swatson@example.com	53.0	M	Lake Loganburgh	Equatorial Guinea	4.0	15	41	60.0	118.612903	N
3	Tommy Cole	swatson@example.com	53.0	M	Lake Loganburgh	Equatorial Guinea	4.0	15	41	60.0	118.612903	N
4	John Brock	georgesteven@example.org	50.0	F	Ericchester	Sierra Leone	2.0	63	31	96.0	153.902439	N

In [ ]:

```
```