------------Data Opration ----------------

1. Assignment 1 – DataFrame Create & Basic Info

In [3]:
```python
import pandas as pd

# Step 1: Create DataFrame from dictionary
data = {
    'Name': ['Ramiz', 'Aman', 'Neha'],
    'Age': [21, 22, 23],
    'Course': ['Python', 'AI', 'Data Science']
}

df = pd.DataFrame(data)

# Step 2: Print DataFrame
print("Full DataFrame:")
print(df)

# Step 3: Show first 2 rows
print("\nFirst 2 Records:")
print(df.head(2))

# Step 4: Show last 1 row
print("\nLast Record:")
print(df.tail(1))

# Step 5: Show DataFrame info
print("\nDataFrame Info:")
print(df.info())

# Step 6: Show statistics
print("\nStatistics Summary:")
print(df.describe())
```

```
Full DataFrame:
    Name  Age        Course
0  Ramiz   21        Python
1   Aman   22            AI
2   Neha   23  Data Science

First 2 Records:
    Name  Age  Course
0  Ramiz   21  Python
1   Aman   22      AI

Last Record:
   Name  Age        Course
2  Neha   23  Data Science

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Name    3 non-null      object
 1   Age     3 non-null      int64 
 2   Course  3 non-null      object
dtypes: int64(1), object(2)
memory usage: 204.0+ bytes
None

Statistics Summary:
        Age
count   3.0
mean   22.0
std     1.0
min    21.0
25%    21.5
50%    22.0
75%    22.5
max    23.0
```

2. Assignment 2 – Sorting & Filtering in DataFrame

In [4]:
```python
import pandas as pd

# Step 1: Create sample DataFrame
data = {
    'Name': ['Ramiz', 'Aman', 'Neha', 'Zara', 'Karan'],
    'Age': [21, 25, 22, 20, 23],
    'Marks': [88, 76, 95, 92, 85],
    'Course': ['Python', 'AI', 'Data Science', 'Web', 'Python']
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

# Step 2: Sort DataFrame by Marks (Descending)
sorted_marks = df.sort_values(by='Marks', ascending=False)
print("\nSorted by Marks (High to Low):\n", sorted_marks)

# Step 3: Filter students who scored more than 85
high_scorers = df[df['Marks'] > 85]
print("\nStudents with Marks > 85:\n", high_scorers)

# Step 4: Filter Python course students
python_students = df[df['Course'] == 'Python']
print("\nStudents in Python Course:\n", python_students)
```

```
Original DataFrame:
     Name  Age  Marks        Course
0  Ramiz   21     88        Python
1   Aman   25     76            AI
2   Neha   22     95  Data Science
3   Zara   20     92           Web
4  Karan   23     85        Python

Sorted by Marks (High to Low):
     Name  Age  Marks        Course
2   Neha   22     95  Data Science
3   Zara   20     92           Web
0  Ramiz   21     88        Python
4  Karan   23     85        Python
1   Aman   25     76            AI

Students with Marks > 85:
     Name  Age  Marks        Course
0  Ramiz   21     88        Python
2   Neha   22     95  Data Science
3   Zara   20     92           Web

Students in Python Course:
     Name  Age  Marks  Course
0  Ramiz   21     88  Python
4  Karan   23     85  Python
```

3. Assignment 3 – Conditional Selection & Index Manipulation

In [5]:
```python
import pandas as pd

# Step 1: Create DataFrame
data = {
    'Name': ['Ramiz', 'Aman', 'Neha', 'Zara', 'Karan'],
    'Age': [21, 25, 22, 20, 23],
    'Marks': [88, 76, 95, 92, 85],
    'Course': ['Python', 'AI', 'Data Science', 'Web', 'Python']
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

# Step 2: Conditional Selection - Students with Marks >= 90 OR Age < 21
cond_selection = df[(df['Marks'] >= 90) | (df['Age'] < 21)]
print("\nStudents with Marks >= 90 OR Age < 21:\n", cond_selection)

# Step 3: Conditional Selection - Python course students with Marks > 80
python_top = df[(df['Course'] == 'Python') & (df['Marks'] > 80)]
print("\nPython Course Students with Marks > 80:\n", python_top)

# Step 4: Set 'Name' as index
df_indexed = df.set_index('Name')
print("\nDataFrame with 'Name' as Index:\n", df_indexed)

# Step 5: Access row using index label
print("\nData for 'Neha':\n", df_indexed.loc['Neha'])
```

```
Original DataFrame:
     Name  Age  Marks        Course
0  Ramiz   21     88        Python
1   Aman   25     76            AI
2   Neha   22     95  Data Science
3   Zara   20     92           Web
4  Karan   23     85        Python

Students with Marks >= 90 OR Age < 21:
    Name  Age  Marks        Course
2  Neha   22     95  Data Science
3  Zara   20     92           Web

Python Course Students with Marks > 80:
     Name  Age  Marks  Course
0  Ramiz   21     88  Python
4  Karan   23     85  Python

DataFrame with 'Name' as Index:
        Age  Marks        Course
Name
Ramiz   21     88        Python
Aman    25     76            AI
Neha    22     95  Data Science
Zara    20     92           Web
Karan   23     85        Python

Data for 'Neha':
 Age                 22
Marks               95
Course    Data Science
Name: Neha, dtype: object
```

4. Assignment -4 Aggregation & GroupBy

In [6]:

```python
import pandas as pd

# Step 1: Create DataFrame
data = {
    'Name': ['Ramiz', 'Aman', 'Neha', 'Zara', 'Karan', 'Pooja', 'Vikas'],
    'Age': [21, 25, 22, 20, 23, 24, 22],
    'Marks': [88, 76, 95, 92, 85, 78, 90],
    'Course': ['Python', 'AI', 'Data Science', 'Web', 'Python', 'AI', 'Data Science']
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

# Step 2: Average marks per course
course_avg = df.groupby('Course')['Marks'].mean()
print("\nAverage Marks per Course:\n", course_avg)

# Step 3: Max marks per course
course_max = df.groupby('Course')['Marks'].max()
print("\nMax Marks per Course:\n", course_max)

# Step 4: Count of students per course
course_count = df.groupby('Course')['Name'].count()
print("\nNumber of Students per Course:\n", course_count)

# Step 5: Multiple aggregations in one line
course_stats = df.groupby('Course').agg({
    'Marks': ['mean', 'max', 'min'],
    'Age': 'mean'
})
print("\nCourse Stats (Mean, Max, Min Marks + Avg Age):\n", course_stats)
```

```
Original DataFrame:
     Name  Age  Marks        Course
0  Ramiz   21     88        Python
1   Aman   25     76            AI
2   Neha   22     95  Data Science
3   Zara   20     92           Web
4  Karan   23     85        Python
5  Pooja   24     78            AI
6  Vikas   22     90  Data Science

Average Marks per Course:
 Course
AI              77.0
Data Science    92.5
Python          86.5
Web             92.0
Name: Marks, dtype: float64

Max Marks per Course:
 Course
AI              78
Data Science    95
Python          88
Web             92
Name: Marks, dtype: int64

Number of Students per Course:
 Course
AI              2
Data Science    2
Python          2
Web             1
Name: Name, dtype: int64

Course Stats (Mean, Max, Min Marks + Avg Age):
              Marks           Age
              mean max min   mean
Course
AI            77.0  78  76   24.5
Data Science  92.5  95  90   22.0
Python        86.5  88  85   22.0
Web           92.0  92  92   20.0
```

5 Assignment 5 -Sorting & Merging DataFrames

In [7]:

```
import pandas as pd

# Step 1: Create first DataFrame (Student Info)
data1 = {
    'ID': [1, 2, 3, 4],
    'Name': ['Ramiz', 'Aman', 'Neha', 'Zara'],
    'Course': ['Python', 'AI', 'Data Science', 'Web']
}
df1 = pd.DataFrame(data1)
print("Student Info:\n", df1)

# Step 2: Create second DataFrame (Marks Info)
data2 = {
    'ID': [1, 2, 3, 4],
    'Marks': [88, 76, 95, 92],
    'Age': [21, 25, 22, 20]
}
df2 = pd.DataFrame(data2)
print("\nMarks Info:\n", df2)

# Step 3: Merge both DataFrames on 'ID'
merged_df = pd.merge(df1, df2, on='ID')
print("\nMerged DataFrame:\n", merged_df)

# Step 4: Sort by Marks (Descending)
sorted_df = merged_df.sort_values(by='Marks', ascending=False)
print("\nSorted by Marks (High to Low):\n", sorted_df)

# Step 5: Sort by Age (Ascending)
sorted_age = merged_df.sort_values(by='Age', ascending=True)
print("\nSorted by Age (Low to High):\n", sorted_age)
```

```
Student Info:
    ID   Name        Course
0   1  Ramiz        Python
1   2   Aman            AI
2   3   Neha  Data Science
3   4   Zara           Web

Marks Info:
    ID  Marks  Age
0   1     88   21
1   2     76   25
2   3     95   22
3   4     92   20

Merged DataFrame:
    ID   Name        Course  Marks  Age
0   1  Ramiz        Python     88   21
1   2   Aman            AI     76   25
2   3   Neha  Data Science     95   22
3   4   Zara           Web     92   20

Sorted by Marks (High to Low):
    ID   Name        Course  Marks  Age
2   3   Neha  Data Science     95   22
3   4   Zara           Web     92   20
0   1  Ramiz        Python     88   21
1   2   Aman            AI     76   25

Sorted by Age (Low to High):
    ID   Name        Course  Marks  Age
3   4   Zara           Web     92   20
0   1  Ramiz        Python     88   21
2   3   Neha  Data Science     95   22
1   2   Aman            AI     76   25
```

6. Assignment 6 - Filtering & Conditional Selection

In [8]:

```python
import pandas as pd

# Step 1: Create DataFrame
data = {
    'Name': ['Ramiz', 'Aman', 'Neha', 'Zara', 'John'],
    'Age': [21, 25, 22, 20, 23],
    'Course': ['Python', 'AI', 'Data Science', 'Web', 'Python'],
    'Marks': [88, 76, 95, 92, 60]
}
df = pd.DataFrame(data)
print("Original Data:\n", df)

# Step 2: Filter students with Marks >= 90
high_scorers = df[df['Marks'] >= 90]
print("\nStudents with Marks >= 90:\n", high_scorers)

# Step 3: Filter students in Python course
python_students = df[df['Course'] == 'Python']
print("\nStudents in Python Course:\n", python_students)

# Step 4: Filter students with Age between 21 and 23
age_range = df[(df['Age'] >= 21) & (df['Age'] <= 23)]
print("\nStudents with Age between 21 and 23:\n", age_range)

# Step 5: Multiple conditions (Python course & Marks > 80)
python_top = df[(df['Course'] == 'Python') & (df['Marks'] > 80)]
print("\nPython Students with Marks > 80:\n", python_top)
```

```
Original Data:
     Name  Age        Course  Marks
0  Ramiz   21        Python     88
1   Aman   25            AI     76
2   Neha   22  Data Science     95
3   Zara   20           Web     92
4   John   23        Python     60

Students with Marks >= 90:
    Name  Age        Course  Marks
2  Neha   22  Data Science     95
3  Zara   20           Web     92

Students in Python Course:
     Name  Age  Course  Marks
0  Ramiz   21  Python     88
4   John   23  Python     60

Students with Age between 21 and 23:
     Name  Age        Course  Marks
0  Ramiz   21        Python     88
2   Neha   22  Data Science     95
4   John   23        Python     60

Python Students with Marks > 80:
     Name  Age  Course  Marks
0  Ramiz   21  Python     88
```

7 Assignment -7 GroupBy & Aggregation

In [9]:
```python
import pandas as pd

# Sample Data
data = {
    'Name': ['Aman', 'Neha', 'Ramiz', 'Priya', 'John', 'Aman'],
    'Course': ['Python', 'AI', 'Python', 'AI', 'Python', 'AI'],
    'Marks': [85, 90, 78, 88, 95, 92]
}

df = pd.DataFrame(data)

# Group by 'Course' and find average marks
course_avg = df.groupby('Course')['Marks'].mean()
print("Average Marks per Course:\n", course_avg)

# Group by 'Name' and find total marks
name_total = df.groupby('Name')['Marks'].sum()
print("\nTotal Marks per Student:\n", name_total)

# Group by 'Course' and get multiple aggregations
multi_agg = df.groupby('Course')['Marks'].agg(['mean', 'max', 'min'])
print("\nMultiple Aggregations per Course:\n", multi_agg)
```

```
Average Marks per Course:
 Course
AI        90.0
Python    86.0
Name: Marks, dtype: float64

Total Marks per Student:
 Name
Aman     177
John      95
Neha      90
Priya     88
Ramiz     78
Name: Marks, dtype: int64

Multiple Aggregations per Course:
         mean  max  min
Course
AI      90.0   92   88
Python  86.0   95   78
```

Mini Project (Student Performance Analysis)

In [13]:
```python
import pandas as pd

# Step 1: Read CSV file
df = pd.read_csv("students.csv")
```

```python
# Step 2: Fill missing marks with mean
df['Marks'] = pd.to_numeric(df['Marks'], errors='coerce')  # convert to numeric
df['Marks'] = df['Marks'].fillna(df['Marks'].mean())

# Step 3: Group by Course and calculate average marks
course_avg = df.groupby('Course')['Marks'].mean()
print("\nAverage Marks per Course:\n", course_avg)

# Step 4: Filter top scorers (Marks >= 90)
top_students = df[df['Marks'] >= 90]
print("\nTop Scorers:\n", top_students[['Name', 'Marks']])

# Step 5: Save cleaned data
df.to_csv("cleaned_students.csv", index=False)
print("\nCleaned data saved to 'cleaned_students.csv'")
```

```
Average Marks per Course:
 Course
AI              89.111111
Data Science    85.000000
Python          86.333333
Web             90.000000
Name: Marks, dtype: float64

Top Scorers:
     Name  Marks
3  Priya   92.0
4   John   95.0
5   Zara   90.0

Cleaned data saved to 'cleaned_students.csv'
```

In [ ]:

```
```