1. Compact Assignment -> Use(Concat + Basic Merge (inner, left , or right)

In [36]:
```
import pandas as pd

df1 =pd.DataFrame({

     "ID":[1,2,3],
     "Name":['Ramiz', 'Aman', 'Neha']
})
print("show the first data  rows:\n")

print(df1)


df2 =pd.DataFrame({

    "ID": [4,5],

     "Name":['sakil', 'kaka']

})
print("show in the Second data rows : \n ")
print(df2)

print("1,The Add Two Data Frame : \n ")

df =pd.concat([df1, df2], ignore_index=True)

print("The  Result Show : \n", df)


# part 2. Merge in and the join in :

print("_____________________________This is the Part 2 Mergein the Data and jointin the Data: ! _____________________")


Student =pd.DataFrame({

        "ID": [1, 2, 3, 4],

        "Name": ["Ramiz", "BoB", "Neha", "Zara"],
        "CourseID": [101, 102, 101, 103]
})

print("Show in the First Data :  \n", Student)

Course =pd.DataFrame ({


       "CourseID": [101, 102, 104],

       "CourseName": ['Python', 'ai', 'web dev']

})

print("Show in the  Second  Data Rows result : \n", Course)

print(" First  the Joint and Inner")

joint_inner =pd.merge(Student ,Course, on='CourseID', how='inner')

print("Show in the Inner Result ", joint_inner)

print("Second The  joint  and left " )


joint_left =pd.merge(Student, Course, on='CourseID', how='left')


print("show in the left Reuslt ", joint_left)


print('Third  joint and Right ')


joint_right =pd.merge(Student, Course , on='CourseID', how='right')
```

```
print("Showin the Right Result: \n" ,joint_right)
```

```
show the first data  rows:

   ID   Name
0   1  Ramiz
1   2   Aman
2   3   Neha
show in the Second data rows :

   ID   Name
0   4  sakil
1   5   kaka
1,The Add Two Data Frame :

The  Result Show :
    ID   Name
0   1  Ramiz
1   2   Aman
2   3   Neha
3   4  sakil
4   5   kaka
_____________________________This is the Part 2 Mergein the Data and jointin the Data: ! ____________________________
__________
Show in the First Data :
    ID   Name  CourseID
0   1  Ramiz       101
1   2    BoB       102
2   3   Neha       101
3   4   Zara       103
Show in the  Second  Data Rows result :
    CourseID CourseName
0       101     Python
1       102         ai
2       104    web dev
 First  the Joint and Inner
Show in the Inner Result     ID   Name  CourseID CourseName
0   1  Ramiz       101     Python
1   2    BoB       102         ai
2   3   Neha       101     Python
Second The  joint  and left
show in the left Reuslt     ID   Name  CourseID CourseName
0   1  Ramiz       101     Python
1   2    BoB       102         ai
2   3   Neha       101     Python
3   4   Zara       103        NaN
Third  joint and Right
Showin the Right Result:
     ID   Name  CourseID CourseName
0  1.0  Ramiz       101     Python
1  3.0   Neha       101     Python
2  2.0    BoB       102         ai
3  NaN    NaN       104    web dev
```

2. Compact Assignment (Outer Join + Multiple keys + df. join())

In [ ]:
```
import pandas as pd

Student =pd.DataFrame({


     "ID": [1, 2, 3, 4 ],

      "Name": ['Ramiz', 'Aman ', 'naha', 'zara'],

      "CourseID": [101, 102, 101, 103]

})

courses =pd.DataFrame({

  "CourseID": [101, 102, 104],

  "CourseName":['python', 'ai', 'web dev']

})
```

```
df =pd.merge(Student, courses, on="CourseID", how="outer")


print(df)
# The Part 2  Multip0le Keys ::


print("_____This is  the  part 2 Multip0le Keys_____ ")


df1 =pd.DataFrame({

    "Id": [1, 2, 3],

    "City": ["Delhi", "Mumbai", "Kolkata"],

    "Marks": [88, 76, 95]



})

print("Show in the First Data: ", df1)




df2 =pd.DataFrame({

  "Id": [1, 2, 3],

    "City": ["Delhi", "Mumbai", "Kolkata"],

    "Course": ["python",'Ai', "Web Dev" ]

})

print("Show in the Second Data: \n", df2)


print("Merging and Multiple Keys(ID + City)\n")

df_mearge =pd.merge(df1, df2, on=['Id', 'City'],  how='inner')


print(df_mearge)

#This is the Part 3 df. join():


print("______ This is the part 3 df. join _____")

# ----- Part 3: df.join() -----
left = pd.DataFrame({

         "Name": ["Ramiz", "Aman", "Neha"],
          'index':[1, 2, 3]
} )
right = pd.DataFrame({
        "Marks": [88, 76, 95],

        'index':[1, 2, 3]

})

print("\nJoin using Index:")
print(pd.merge(left, right, on="index"))
```

_____ Mini Project Student Report Data Analysis_____ :

In [92]:
```
import pandas as pd
import numpy as np

# Students Data
students = pd.DataFrame({
    "ID": [1, 2, 3, 4, 5, 5],   # Duplicate ID intentionally
    "Name": ["Ramiz", "Aman", "Neha", "Zara", "Karan", "Karan"],
```

```
    "CourseID": [101, 102, 101, 103, 104, 104]
})

# Courses Data
courses = pd.DataFrame({
    "CourseID": [101, 102, 103, 104],
    "CourseName": ["Python", "AI", "Data Science", "Web Development"]
})

# Results Data (with missing values)
results = pd.DataFrame({
    "ID": [1, 2, 3, 4, 5],
    "Marks": [88, np.nan, 95, 76, 90]   # Aman has missing marks
})

# ---------------- Step 1: Merge Students + Courses ----------------
merged = pd.merge(students, courses, on="CourseID", how="inner")

# ---------------- Step 2: Merge with Results ----------------
final = pd.merge(merged, results, on="ID", how="inner")

print("Raw Merged Data:")
print(final)

# ---------------- Step 3: Data Cleaning ----------------
# Remove duplicates
final = final.drop_duplicates()

# Handle missing marks → fill with average
final["Marks"] = final["Marks"].fillna(final["Marks"].mean())

print("\nCleaned Data:")
print(final)

# ---------------- Step 4: GroupBy & Aggregation ----------------
course_stats = final.groupby("CourseName").agg({
    "Marks": ["mean", "max", "min", "count"]
})

print("\nCourse-wise Statistics:")
print(course_stats)

# ---------------- Step 5: Filter Top Scorers ----------------
top_scorers = final[final["Marks"] >= 90]

print("\nTop Scorers (Marks >= 90):")
print(top_scorers[["Name", "CourseName", "Marks"]])
```

```
Raw Merged Data:
   ID   Name  CourseID       CourseName  Marks
0   1  Ramiz       101           Python   88.0
1   2   Aman       102               AI    NaN
2   3   Neha       101           Python   95.0
3   4   Zara       103     Data Science   76.0
4   5  Karan       104  Web Development   90.0
5   5  Karan       104  Web Development   90.0

Cleaned Data:
   ID   Name  CourseID       CourseName  Marks
0   1  Ramiz       101           Python  88.00
1   2   Aman       102               AI  87.25
2   3   Neha       101           Python  95.00
3   4   Zara       103     Data Science  76.00
4   5  Karan       104  Web Development  90.00

Course-wise Statistics:
                Marks
                 mean    max    min count
CourseName
AI              87.25  87.25  87.25     1
Data Science    76.00  76.00  76.00     1
Python          91.50  95.00  88.00     2
Web Development 90.00  90.00  90.00     1

Top Scorers (Marks >= 90):
    Name       CourseName  Marks
2   Neha           Python   95.0
4  Karan  Web Development   90.0
```

In [ ]: