20250415_01

April 15, 2025

[5]:
```
import pandas as pd

# Create students data
students = pd.DataFrame({'student_id':[101, 102, 103, 104], 'name':['Alice',␣
 ↪'Bob', 'Charile', 'David'], 'gender':['F', 'M', 'M', 'M']})

# Create scores data, which contains some error
scores = pd.DataFrame({'student_id':[101, 102, 104, 105], 'midterm_score':[85,␣
 ↪90, 78, 88]})
```

[7]:
```
students.head()
```

[7]:

	student_id	name	gender
0	101	Alice	F
1	102	Bob	M
2	103	Charile	M
3	104	David	M

[9]:
```
scores.head()
```

[9]:

	student_id	midterm_score
0	101	85
1	102	90
2	104	78
3	105	88

[23]:
```
# inner join: intersection of datas
merged_inner = pd.merge(students, scores, on = 'student_id') # Merged based on␣
 ↪student_id, 'how = ' is set to 'inner' by default
print(merged_inner)
```

	student_id	name	gender	midterm_score
0	101	Alice	F	85
1	102	Bob	M	90
2	104	David	M	78

[25]:
```
# left join: only need left to have the data, and the same logic apply to␣
 ↪'right'
```

```
merged_left = pd.merge(students, scores, on = 'student_id', how = 'left')
print(merged_left)
```

```
   student_id     name gender  midterm_score
0         101    Alice      F           85.0
1         102      Bob      M           90.0
2         103  Charile      M            NaN
3         104    David      M           78.0
```

[19]:
```
# outer join: union of datas
merged_outer = pd.merge(students, scores, on = 'student_id', how = 'outer')
print(merged_outer)
```

```
   student_id     name gender  midterm_score
0         101    Alice      F           85.0
1         102      Bob      M           90.0
2         103  Charile      M            NaN
3         104    David      M           78.0
4         105      NaN    NaN           88.0
```

[33]:
```
# indicator: shows where the data came from
merged_indicator = pd.merge(students, scores, on = 'student_id', how = 'outer',␣
 ↪indicator = True)
print(merged_indicator)
```

```
   student_id     name gender  midterm_score      _merge
0         101    Alice      F           85.0        both
1         102      Bob      M           90.0        both
2         103  Charile      M            NaN   left_only
3         104    David      M           78.0        both
4         105      NaN    NaN           88.0  right_only
```

[75]:
```
# Adding new datas into existing categories
df1 = pd.DataFrame({'student_id':[201, 202], 'name':['Eve', 'Frank'], 'gender':
 ↪['F', 'M']})

df_combined = pd.concat([students, df1], ignore_index = True) # 'axis = ' is␣
 ↪set to '0' by default
print(df_combined)
```

```
   student_id     name gender
0         101    Alice      F
1         102      Bob      M
2         103  Charile      M
3         104    David      M
4         201      Eve      F
5         202    Frank      M
```

[79]:
```
# Adding new datas that need new categories, pretty much like 'join'
df2 = pd.DataFrame({'club':['Math', 'Science', 'Music', 'Drama']})

df_joined = pd.concat([students, df2], axis = 1) # 'ignore_index = ' is set to␣
 ↪'False' by default
print(df_joined)
```

```
   student_id     name gender     club
0         101    Alice      F     Math
1         102      Bob      M  Science
2         103  Charile      M    Music
3         104    David      M    Drama
```

[89]:
```
# Specify datasets name
df_concat = pd.concat([students, df1], keys=['original', 'new']) # also,␣
 ↪'ignore_index' will override 'keys'.
print(df_concat)
```

```
            student_id     name gender
original 0         101    Alice      F
         1         102      Bob      M
         2         103  Charile      M
         3         104    David      M
new      0         201      Eve      F
         1         202    Frank      M
```

[103]:
```
# Exercise
finals = pd.DataFrame({'student_id':[101, 103, 104, 106], 'final_score':[88,␣
 ↪77, 85, 90]})
merged_all_outer = pd.merge(merged_outer, finals, on = 'student_id', how =␣
 ↪'outer') # only two datas at a time
print(merged_all_outer)
```

```
   student_id     name gender  midterm_score  final_score
0         101    Alice      F           85.0         88.0
1         102      Bob      M           90.0          NaN
2         103  Charile      M            NaN         77.0
3         104    David      M           78.0         85.0
4         105      NaN    NaN           88.0          NaN
5         106      NaN    NaN            NaN         90.0
```

[109]:
```
merged_all_left = pd.merge(merged_left, finals, on = 'student_id', how = 'left')
print(merged_all_left)
```

```
   student_id     name gender  midterm_score  final_score
0         101    Alice      F           85.0         88.0
1         102      Bob      M           90.0          NaN
2         103  Charile      M            NaN         77.0
3         104    David      M           78.0         85.0
```

[113]:
```
# just in case if the original data is not in order
merged_all_left = merged_all_left.sort_values('student_id').reset_index(drop =␣
↪True)
print(merged_all_left)
```

```
   student_id     name gender  midterm_score  final_score
0         101    Alice      F           85.0         88.0
1         102      Bob      M           90.0          NaN
2         103  Charile      M            NaN         77.0
3         104    David      M           78.0         85.0
```