Pandas

In [ ]:
```
stu_details = {'stu_id':[1,2,3,4,5],
               'stu_name':['A','B','C','D','E'],
               'Region':['N','S','N','E','W']}

stu_marks = {'st_id':[1,2,3,4,6],
             'sub':['Py','Tab','Sql','Py','Tab'],
             'marks':[35,32,36,37,33]}
```

In [5]:
```
stu_details
```

Out[5]:
```
{'stu_id': [1, 2, 3, 4, 5],
 'stu_name': ['A', 'B', 'C', 'D', 'E'],
 'Region': ['N', 'S', 'N', 'E', 'W']}
```

In [6]:
```
stu_marks
```

Out[6]:
```
{'st_id': [1, 2, 3, 4, 6],
 'sub': ['Py', 'Tab', 'Sql', 'Py', 'Tab'],
 'marks': [35, 32, 36, 37, 33]}
```

In [7]:
```
import pandas as pd
```

In [8]:
```
pd.__version__
```

Out[8]:
```
'1.1.3'
```

In [9]:
```
stu_details = pd.DataFrame(stu_details)
```

In [10]:
```
type(stu_details)
```

Out[10]:
```
pandas.core.frame.DataFrame
```

In [11]:
```
stu_details
```

Out[11]:

	stu_id	stu_name	Region
0	1	A	N
1	2	B	S
2	3	C	N
3	4	D	E
4	5	E	W

In [12]:
```
stu_marks = pd.DataFrame(stu_marks)
```

In [13]:
```
stu_marks
```

Out[13]:

	st_id	sub	marks
0	1	Py	35
1	2	Tab	32
2	3	Sql	36
3	4	Py	37
4	6	Tab	33

In [14]:
```
"""if the student id is same(st_id) on both side.
   use "onnst_id" --> this is used for any same column while joining.
   other wise use in this fashion shown below line"""

pd.merge(stu_details,stu_marks, left_on='stu_id',right_on='st_id', how='inner')
```

Out[14]:

	stu_id	stu_name	Region	st_id	sub	marks
0	1	A	N	1	Py	35
1	2	B	S	2	Tab	32
2	3	C	N	3	Sql	36
3	4	D	E	4	Py	37

In [15]:
```
pd.merge(stu_details,stu_marks, left_on='stu_id',right_on='st_id', how='left')
```

Out[15]:

	stu_id	stu_name	Region	st_id	sub	marks
0	1	A	N	1.0	Py	35.0
1	2	B	S	2.0	Tab	32.0
2	3	C	N	3.0	Sql	36.0
3	4	D	E	4.0	Py	37.0
4	5	E	W	NaN	NaN	NaN

In [16]:
```
pd.merge(stu_details,stu_marks, left_on='stu_id',right_on='st_id', how='right')
```

Out[16]:

	stu_id	stu_name	Region	st_id	sub	marks
0	1.0	A	N	1	Py	35
1	2.0	B	S	2	Tab	32
2	3.0	C	N	3	Sql	36
3	4.0	D	E	4	Py	37
4	NaN	NaN	NaN	6	Tab	33

In [17]:
```
pd.merge(stu_details,stu_marks, left_on='stu_id',right_on='st_id', how='outer')
```

Out[17]:

	stu_id	stu_name	Region	st_id	sub	marks
0	1.0	A	N	1.0	Py	35.0
1	2.0	B	S	2.0	Tab	32.0
2	3.0	C	N	3.0	Sql	36.0
3	4.0	D	E	4.0	Py	37.0
4	5.0	E	W	NaN	NaN	NaN
5	NaN	NaN	NaN	6.0	Tab	33.0

In [18]:
```
stu_details_Nag = {'st_id':[1,2,3,4,5],
                   'stu_name':['A','B','C','D','E'],
                   'sub':['Py','Tab','Sql','Py','Tab']}

stu_details_US = {'st_id':[6,7,8,9,10],
                  'stu_name':['T','Y','U','I','Z'],
                  'sub':['Tab','Tab','Sql','Py','Py']}
```

In [19]:
```
stu_details_Nag = pd.DataFrame(stu_details_Nag)
```

In [20]:
```
stu_details_Nag
```

Out[20]:

	st_id	stu_name	sub
0	1	A	Py
1	2	B	Tab
2	3	C	Sql
3	4	D	Py
4	5	E	Tab

In [21]:
```
stu_details_US = pd.DataFrame(stu_details_US)
```

In [22]:
```
stu_details_US
```

Out[22]:

	st_id	stu_name	sub
0	6	T	Tab
1	7	Y	Tab
2	8	U	Sql
3	9	I	Py
4	10	Z	Py

In [26]:
```
student_details = pd.concat([stu_details_Nag,stu_details_US])
```

In [27]:
```
student_details
```

Out[27]:

	st_id	stu_name	sub
0	1	A	Py
1	2	B	Tab
2	3	C	Sql
3	4	D	Py
4	5	E	Tab
0	6	T	Tab
1	7	Y	Tab
2	8	U	Sql
3	9	I	Py
4	10	Z	Py

In [25]:
```
student_details.reset_index()
```

Out[25]:

	index	st_id	stu_name	sub
0	0	1	A	Py
1	1	2	B	Tab
2	2	3	C	Sql
3	3	4	D	Py
4	4	5	E	Tab
5	0	6	T	Tab
6	1	7	Y	Tab
7	2	8	U	Sql
8	3	9	I	Py
9	4	10	Z	Py

In [ ]:
```
```

In [ ]:
```
```