Chapter 2 - Data Preparation Basics

Segment 4 - Concatenating and transforming data

```
import numpy as np
import pandas as pd

from pandas import Series, DataFrame
```

```
DF_obj = pd.DataFrame(np.arange(36).reshape(6,6))
DF_obj
```

	0	1	2	3	4	5
0	0	1	2	3	4	5
1	6	7	8	9	10	11
2	12	13	14	15	16	17
3	18	19	20	21	22	23
4	24	25	26	27	28	29
5	30	31	32	33	34	35

```
DF_obj_2 = pd.DataFrame(np.arange(15).reshape(5,3))
DF_obj_2
```

```
    0   1   2
```

Concatenating data

```
1   3   4   5
```

```
pd.concat([DF_obj, DF_obj_2], axis=1)
#pd.concat([DF_obj, DF_obj-1],axis=1) #adding new columns and getting added in the existing rows
```

	0	1	2	3	4	5	0	1	2
0	0	1	2	3	4	5	0.0	1.0	2.0
1	6	7	8	9	10	11	3.0	4.0	5.0
2	12	13	14	15	16	17	6.0	7.0	8.0
3	18	19	20	21	22	23	9.0	10.0	11.0
4	24	25	26	27	28	29	12.0	13.0	14.0
5	30	31	32	33	34	35	NaN	NaN	NaN

```
pd.concat([DF_obj, DF_obj_2])
#pd.concat([df1,df2]) #default axis is 0 which means adding new rows with added to existing columns
```

	0	1	2	3	4	5

Transforming data

Dropping data

3	18	19	20	21.0	22.0	23.0

```
DF_obj.drop([0, 2])
#df.drop([1,2]) #dropping 1st and 2nd rows
```

	0	1	2	3	4	5
1	6	7	8	9	10	11
3	18	19	20	21	22	23
4	24	25	26	27	28	29
5	30	31	32	33	34	35

```
DF_obj.drop([0, 2], axis=1)
#pd.drop([1,2],axis=1)  #dropping 1st and 2nd columns when specified axis=1
```

	1	3	4	5
0	1	3	4	5
1	7	9	10	11
2	13	15	16	17
3	19	21	22	23
4	25	27	28	29
5	31	33	34	35

Adding data

```
series_obj = Series(np.arange(6))
```

```
#series_obj.name = "added_variable"
series_obj
```

```
0    0
1    1
2    2
3    3
4    4
5    5
dtype: int64
```

```
variable_added = DataFrame.join(DF_obj, series_obj)
variable_added
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-13-25d53ed65a7a> in <module>()
----> 1 variable_added = DataFrame.join(DF_obj, series_obj)
      2 variable_added
```

1 frames

```
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py in _join_compat(self, other, on, how, lsuffix, rsuffix, sort)
   9115         if isinstance(other, Series):
   9116             if other.name is None:
-> 9117                 raise ValueError("Other Series must have a name")
   9118             other = DataFrame({other.name: other})
   9119 

ValueError: Other Series must have a name
```

SEARCH STACK OVERFLOW

```
added_datatable = variable_added.append(variable_added, ignore_index=False)
added_datatable
```

	0	1	2	3	4	5	added_variable
0	0	1	2	3	4	5	0
1	6	7	8	9	10	11	1
2	12	13	14	15	16	17	2
3	18	19	20	21	22	23	3
4	24	25	26	27	28	29	4
5	30	31	32	33	34	35	5
0	0	1	2	3	4	5	0
1	6	7	8	9	10	11	1

```
added_datatable = variable_added.append(variable_added, ignore_index=True)
added_datatable
```

	0	1	2	3	4	5	added_variable
0	0	1	2	3	4	5	0
1	6	7	8	9	10	11	1
2	12	13	14	15	16	17	2
3	18	19	20	21	22	23	3
4	24	25	26	27	28	29	4
5	30	31	32	33	34	35	5
6	0	1	2	3	4	5	0
7	6	7	8	9	10	11	1
8	12	13	14	15	16	17	2
9	18	19	20	21	22	23	3
10	24	25	26	27	28	29	4
11	30	31	32	33	34	35	5

Sorting data

```
DF_sorted = DF_obj.sort_values(by=(5), ascending=[False])
DF_sorted
```

```
    0   1   2   3   4   5
5  30  31  32  33  34  35
4  24  25  26  27  28  29
3  18  19  20  21  22  23
2  12  13  14  15  16  17
1   6   7   8   9  10  11
0   0   1   2   3   4   5
```