Module 02

Pandas for DataFrame Manipulation

Data Science Developer

Purwadhika
Startup and Coding School

Outline

- Adding New Data
- Deleting Data
- More about Index
- Multi-Index
- Sorting

Using Numpy and Pandas

```
In [1]: import pandas as pd
        import numpy as np
```

Purwadhika
Startup and Coding School

Adding New Data

Add a New Row

In [19]:
```
df.loc['new']=[1,2,3,4]
df
```

Out[19]:

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057
E	0.190794	1.978757	2.605967	0.683509
new	1.000000	2.000000	3.000000	4.000000

Add a New Column

```
In [9]: df['new'] = df['W'] + df['Y']

In [10]: df
```

Out[10]:

	W	X	Y	Z	new
A	2.706850	0.628133	0.907969	0.503826	3.614819
B	0.651118	-0.319318	-0.848077	0.605965	-0.196959
C	-2.018168	0.740122	0.528813	-0.589001	-1.489355
D	0.188695	-0.758872	-0.933237	0.955057	-0.744542
E	0.190794	1.978757	2.605967	0.683509	2.796762

Add a New Column

Insert

```
In [39]: df.insert(2,'new',[1,2,3,4,5])
```

```
In [40]: df
```

Out[40]:

	W	X	new	Y	Z
A	2.706850	0.628133	1	0.907969	0.503826
B	0.651118	-0.319318	2	-0.848077	0.605965
C	-2.018168	0.740122	3	0.528813	-0.589001
D	0.188695	-0.758872	4	-0.933237	0.955057
E	0.190794	1.978757	5	2.605967	0.683509

Deleting Data

Removing Columns
without inplace

In [11]:
```
df.drop('new',axis=1)
```

Out[11]:

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057
E	0.190794	1.978757	2.605967	0.683509

In [12]:
```
# Not inplace unless specified!
df
```

Out[12]:

	W	X	Y	Z	new
A	2.706850	0.628133	0.907969	0.503826	3.614819
B	0.651118	-0.319318	-0.848077	0.605965	-0.196959
C	-2.018168	0.740122	0.528813	-0.589001	-1.489355
D	0.188695	-0.758872	-0.933237	0.955057	-0.744542
E	0.190794	1.978757	2.605967	0.683509	2.796762

Removing Columns
with inplace

```
In [13]: df.drop('new',axis=1,inplace=True)

In [14]: df
```

Out[14]:

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057
E	0.190794	1.978757	2.605967	0.683509

Removing Rows

Same with drop columns, the difference is the axis:

```
In [17]: df.drop('E',axis=0)
```

Out[17]:

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057

More about Index

More Index Details

In [29]:
```
df
```

Out[29]:

	W	X	Y	Z
A	2.706850	0.628133	0.907969	0.503826
B	0.651118	-0.319318	-0.848077	0.605965
C	-2.018168	0.740122	0.528813	-0.589001
D	0.188695	-0.758872	-0.933237	0.955057
E	0.190794	1.978757	2.605967	0.683509

In [30]:
```
# Reset to default 0,1...n index
df.reset_index()
```

Out[30]:

	index	W	X	Y	Z
0	A	2.706850	0.628133	0.907969	0.503826
1	B	0.651118	-0.319318	-0.848077	0.605965
2	C	-2.018168	0.740122	0.528813	-0.589001
3	D	0.188695	-0.758872	-0.933237	0.955057
4	E	0.190794	1.978757	2.605967	0.683509

More Index Details

In [34]:
```
newind = 'CA NY WY OR CO'.split()
newind
```

Out[34]:
```
['CA', 'NY', 'WY', 'OR', 'CO']
```

In [35]:
```
df['States'] = newind
```

In [36]:
```
df
```

Out[36]:

	W	X	Y	Z	States
A	2.706850	0.628133	0.907969	0.503826	CA
B	0.651118	-0.319318	-0.848077	0.605965	NY
C	-2.018168	0.740122	0.528813	-0.589001	WY
D	0.188695	-0.758872	-0.933237	0.955057	OR
E	0.190794	1.978757	2.605967	0.683509	CO

In [37]:
```
df.set_index('States')
```

Out[37]:

	W	X	Y	Z
States				
CA	2.706850	0.628133	0.907969	0.503826
NY	0.651118	-0.319318	-0.848077	0.605965
WY	-2.018168	0.740122	0.528813	-0.589001
OR	0.188695	-0.758872	-0.933237	0.955057
CO	0.190794	1.978757	2.605967	0.683509

Purwadhika
Startup and Coding School

More Index Details

```
In [38]: df
```

Out[38]:

	W	X	Y	Z	States
A	2.706850	0.628133	0.907969	0.503826	CA
B	0.651118	-0.319318	-0.848077	0.605965	NY
C	-2.018168	0.740122	0.528813	-0.589001	WY
D	0.188695	-0.758872	-0.933237	0.955057	OR
E	0.190794	1.978757	2.605967	0.683509	CO

```
In [39]: df.set_index('States',inplace=True)
```

```
In [40]: df
```

Out[40]:

	W	X	Y	Z
States				
CA	2.706850	0.628133	0.907969	0.503826
NY	0.651118	-0.319318	-0.848077	0.605965
WY	-2.018168	0.740122	0.528813	-0.589001
OR	0.188695	-0.758872	-0.933237	0.955057
CO	0.190794	1.978757	2.605967	0.683509

Purwadhika
Startup and Coding School

Multi-Index

Multi-Index

Creating Multi-Index

In [4]:
```
# Index Level
outside=['Jakarta', 'Jakarta','Jakarta',
         'Surabaya','Surabaya','Surabaya']
inside=[1,2,3,1,2,3]
hier_index= list(zip(outside,inside))
hier_index
```

Out[4]:
```
[('Jakarta', 1),
 ('Jakarta', 2),
 ('Jakarta', 3),
 ('Surabaya', 1),
 ('Surabaya', 2),
 ('Surabaya', 3)]
```

In [5]:
```
hier_index=pd.MultiIndex.from_tuples(hier_index)
hier_index
```

Out[5]:
```
MultiIndex([( 'Jakarta', 1),
            ( 'Jakarta', 2),
            ( 'Jakarta', 3),
            ('Surabaya', 1),
            ('Surabaya', 2),
            ('Surabaya', 3)],
           )
```

Purwadhika
Startup and Coding School

Multi-Index

Creating DataFrame with Multi-Index

```
In [8]: df=pd.DataFrame(np.random.randint(1,100,(6,2)),
                        index=hier_index,
                        columns=['Restorant A', 'Restorant B'])
        df
```

Out[8]:

		Restorant A	Restorant B
Jakarta	**1**	37	73
	2	12	33
	3	51	30
Surabaya	**1**	36	58
	2	74	70
	3	1	6

Purwadhika
Startup and Coding School

Multi-Index
Indexing and Selecting

```
In [10]: df.loc['Jakarta']
```

Out[10]:

	Restorant A	Restorant B
1	37	73
2	12	33
3	51	30

```
In [12]: df.loc['Jakarta'].loc[1]
```

```
Out[12]: Restorant A    37
         Restorant B    73
         Name: 1, dtype: int32
```

```
In [24]: df.xs('Jakarta')
```

Out[24]:

	Restorant A	Restorant B
1	55	5
2	54	54
3	58	59

```
In [25]: df.xs(['Jakarta',1])
```

```
Out[25]: Restorant A    55
         Restorant B     5
         Name: (Jakarta, 1), dtype: int32
```

Purwadhika
Startup and Coding School

Multi-Index and Index Hierarchy

Index Name

```
In [14]: df.index.names
```

```
Out[14]: FrozenList([None, None])
```

```
In [26]: df.index.names=['City','Location']
         df
```

Out[26]:

		Restorant A	Restorant B
City	**Location**		
Jakarta	**1**	55	5
	2	54	54
	3	58	59
Surabaya	**1**	39	13
	2	77	29
	3	56	83

Purwadhika
Startup and Coding School

Multi-Index
Indexing and Selecting

```
In [18]: df.xs(1,level='Location')
```

Out[18]:

City	Restorant A	Restorant B
Jakarta	37	73
Surabaya	36	58

Purwadhika
Startup and Coding School

Sorting

Sorting by any columns

```
df.sort_values('name')
```

	name	gender	hire date	gross salary
300207	Dina Rebaine	Female	2015-03-20	15000000
200210	Marko Mendell	Male	2018-07-04	15000000
100111	Raven Bierman	Female	2016-12-04	7000000
200211	Takahiro Momota	Male	2016-11-18	12000000
100112	Valter Havers	Male	2018-04-13	7000000
200312	Yahiko Tilemans	Male	2017-05-26	20000000

```
df.sort_values('name',ascending = False)
```

	name	gender	hire date	gross salary
200312	Yahiko Tilemans	Male	2017-05-26	20000000
100112	Valter Havers	Male	2018-04-13	7000000
200211	Takahiro Momota	Male	2016-11-18	12000000
100111	Raven Bierman	Female	2016-12-04	7000000
200210	Marko Mendell	Male	2018-07-04	15000000
300207	Dina Rebaine	Female	2015-03-20	15000000

Permanently saved the result

```
df.sort_values('name',ascending = False,inplace=True)
```

```
df
```

	name	gender	hire date	gross salary
200312	Yahiko Tilemans	Male	2017-05-26	20000000
100112	Valter Havers	Male	2018-04-13	7000000
200211	Takahiro Momota	Male	2016-11-18	12000000
100111	Raven Bierman	Female	2016-12-04	7000000
200210	Marko Mendell	Male	2018-07-04	15000000
300207	Dina Rebaine	Female	2015-03-20	15000000

Purwadhika
Startup and Coding School

Sorting by more than one columns

```
df.sort_values(by = ['gender','name'])
```

	name	gender	hire date	gross salary
300207	Dina Rebaine	Female	2015-03-20	15000000
100111	Raven Bierman	Female	2016-12-04	7000000
200210	Marko Mendell	Male	2018-07-04	15000000
200211	Takahiro Momota	Male	2016-11-18	12000000
100112	Valter Havers	Male	2018-04-13	7000000
200312	Yahiko Tilemans	Male	2017-05-26	20000000

```
df.sort_values(by = ['gender','name'],
    ascending = [False,False])
```

	name	gender	hire date	gross salary
200312	Yahiko Tilemans	Male	2017-05-26	20000000
100112	Valter Havers	Male	2018-04-13	7000000
200211	Takahiro Momota	Male	2016-11-18	12000000
200210	Marko Mendell	Male	2018-07-04	15000000
100111	Raven Bierman	Female	2016-12-04	7000000
300207	Dina Rebaine	Female	2015-03-20	15000000

Permanently saved the result

```
df.sort_values(
    by = ['gender','name'],
    ascending = [False,False],
    inplace = True
)
```

```
df
```

	name	gender	hire date	gross salary
200312	Yahiko Tilemans	Male	2017-05-26	20000000
100112	Valter Havers	Male	2018-04-13	7000000
200211	Takahiro Momota	Male	2016-11-18	12000000
200210	Marko Mendell	Male	2018-07-04	15000000
100111	Raven Bierman	Female	2016-12-04	7000000
300207	Dina Rebaine	Female	2015-03-20	15000000

Purwadhika
Startup and Coding School

Sorting by Index

```
df
```

	name	gender	hire date	gross salary
300207	Dina Rebaine	Female	2015-03-20	15000000
200210	Marko Mendell	Male	2018-07-04	15000000
100111	Raven Bierman	Female	2016-12-04	7000000
200211	Takahiro Momota	Male	2016-11-18	12000000
100112	Valter Havers	Male	2018-04-13	7000000
200312	Yahiko Tilemans	Male	2017-05-26	20000000

```
df.sort_index()
```

	name	gender	hire date	gross salary
100111	Raven Bierman	Female	2016-12-04	7000000
100112	Valter Havers	Male	2018-04-13	7000000
200210	Marko Mendell	Male	2018-07-04	15000000
200211	Takahiro Momota	Male	2016-11-18	12000000
200312	Yahiko Tilemans	Male	2017-05-26	20000000
300207	Dina Rebaine	Female	2015-03-20	15000000

Permanently saved the result

```
df.sort_index(inplace=True)
```

```
df
```

	name	gender	hire date	gross salary
100111	Raven Bierman	Female	2016-12-04	7000000
100112	Valter Havers	Male	2018-04-13	7000000
200210	Marko Mendell	Male	2018-07-04	15000000
200211	Takahiro Momota	Male	2016-11-18	12000000
200312	Yahiko Tilemans	Male	2017-05-26	20000000
300207	Dina Rebaine	Female	2015-03-20	15000000

Purwadhika
Startup and Coding School

References

- Add one row to Pandas DataFrame. https://stackoverflow.com/questions/10715965/add-one-row-to-pandas-dataframe
- Reset Index in Pandas DataFrame. https://www.geeksforgeeks.org/reset-index-in-pandas-dataframe/
- MultiIndex / advanced indexing. https://pandas.pydata.org/pandas-docs/stable/user_guide/advanced.html
- What is Data Sorting. https://www.displayr.com/what-is-data-sorting/

Purwadhika
Startup and Coding School