

DATA WRANGLING WITH PANDAS

DEFINING A DATAFRAME

OPTION 1

```
DataFrame([[ 'a', 'b', 'c' ],
            [ 'd', 'e', 'f' ],
            [ 'g', 'h', 'i' ]],
            index = [ 1, 2, 3],
            columns = [ 'col1', 'col2', 'col3'])
```

	col1	col2	col3
1	a	b	c
2	d	e	f
3	g	h	i

OPTION 2

```
DataFrame({ 'col1': [ 'a', 'd', 'g' ],
            'col2': [ 'b', 'e', 'h' ],
            'col3': [ 'c', 'f', 'i' ]},
            index = [ 1, 2, 3])
```

RETRIEVING DATA

0 based indexing

	0	1	2
	col1	col2	col3
0	1	a	b
1	2	d	e
2	3	g	h

Column labels

Index labels

```
Int64Index([1, 2, 3], dtype='int64')
```

```
df.columns df.index
```

```
Index(['col1', 'col2', 'col3'], dtype='object')
```

```
df.values
```

```
array([[ 'a', 'b', 'c'],
       [ 'd', 'e', 'f'],
       [ 'g', 'h', 'i']], dtype = object)
```

```
df.loc[ [ 0:2 ], [ 0:2 ] ]
```

	col1	col2
1	a	b
2	d	e

```
df.loc[ [ 1,2 ], [ 'col1', 'col2' ] ]
```

COMBINING DATAFRAMES

```
pd.merge(df_1, df_2, how = ' ', on= ' ')
```

	col1	col2
1	A	2
2	K	9
3	Z	4

+

	col1	col3
1	A	3
2	K	8
3	X	1

= ?

```
pd.merge( df_1, df_2,
            how = 'left',
            on= 'col1' )
```

	col1	col2	col3
1	A	2	3
2	K	9	8
3	Z	4	NaN

```
pd.merge( df_1, df_2,
            how = 'right',
            on= 'col1' )
```

	col1	col2	col3
1	A	2	3
2	K	9	8
3	X	NaN	1

```
pd.merge( df_1, df_2,
            how = 'inner',
            on= 'col1' )
```

	col1	col2	col3
1	A	2	3
2	K	9	8

```
pd.merge( df_1, df_2,
            how = 'outer',
            on= 'col1' )
```

	col1	col2	col3
1	A	2	3
2	K	9	8
3	Z	4	NaN
4	X	NaN	1

Inner

Left

Right

Outer



col1				

```
pd.groupby( by = 'col1' )
```


RESHAPING DATAFRAMES

```
df.melt()
```

transform columns into rows


```
df.pivot()
```

transform rows into columns


```
pd.concat([df,df])
```



```
pd.concat([df_1,df_2],
            axis = 1)
```


Adding column
df['col3'] = ['A', 'B']

			col3

Deleting columns
df.drop('col3', axis=1)

Adding row
df.loc[2] = ['A', 'B']

Deleting row
df_4.drop(2, axis = 0)

DEALING WITH NULL VALUES

```
df.dropna()
```



```
df_na_1.dropna(axis=1)
```
