
(http://www.pieriandata.com)

Operations

There are lots of operations with pandas that will be really useful to you, but don't fall into any distinct category. Let's show them here in this lecture:

In [52]:

```
import pandas as pd
df = pd.DataFrame({'col1':[1,2,3,4],'col2':[444,555,666,444],'col3':['abc','def','ghi',
'xyz']})
df.head()
```

Out[52]:

	col1	col2	col3
0	1	444	abc
1	2	555	def
2	3	666	ghi
3	4	444	xyz

Info on Unique Values

In [53]:

```
df['col2'].unique()
```

Out[53]:

```
array([444, 555, 666])
```

In [54]:

```
df['col2'].nunique()
```

Out[54]:

```
3
```

In [55]:

```
df['col2'].value_counts()
```

Out[55]:

```
444    2
555    1
666    1
Name: col2, dtype: int64
```

Selecting Data

In [56]:

```
#Select from DataFrame using criteria from multiple columns
newdf = df[(df['col1']>2) & (df['col2']==444)]
```

In [57]:

```
newdf
```

Out[57]:

	col1	col2	col3
3	4	444	xyz

Applying Functions

In [58]:

```
def times2(x):
    return x*2
```

In [59]:

```
df['col1'].apply(times2)
```

Out[59]:

```
0    2
1    4
2    6
3    8
Name: col1, dtype: int64
```

In [60]:

```
df['col3'].apply(len)
```

Out[60]:

```
0    3
1    3
2    3
3    3
Name: col3, dtype: int64
```

In [61]:

```
df['col1'].sum()
```

Out[61]:

```
10
```

Permanently Removing a Column

In [62]:

```
del df['col1']
```

In [63]:

```
df
```

Out[63]:

	col2	col3
0	444	abc
1	555	def
2	666	ghi
3	444	xyz

Get column and index names:

In [64]:

```
df.columns
```

Out[64]:

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

In [65]:

```
df.index
```

Out[65]:

```
RangeIndex(start=0, stop=4, step=1)
```

Sorting and Ordering a DataFrame:

In [66]:

```
df
```

Out[66]:

	col2	col3
0	444	abc
1	555	def
2	666	ghi
3	444	xyz

In [67]:

```
df.sort_values(by='col2') #inplace=False by default
```

Out[67]:

	col2	col3
0	444	abc
3	444	xyz
1	555	def
2	666	ghi

Find Null Values or Check for Null Values

In [68]:

```
df.isnull()
```

Out[68]:

	col2	col3
0	False	False
1	False	False
2	False	False
3	False	False

In [69]:

```
# Drop rows with NaN Values
df.dropna()
```

Out[69]:

	col2	col3
0	444	abc
1	555	def
2	666	ghi
3	444	xyz

Filling in NaN values with something else:

In [71]:

```
import numpy as np
```

In [72]:

```
df = pd.DataFrame({'col1':[1,2,3,np.nan],
                   'col2':[np.nan,555,666,444],
                   'col3':['abc','def','ghi','xyz']})
df.head()
```

Out[72]:

	col1	col2	col3
0	1.0	NaN	abc
1	2.0	555.0	def
2	3.0	666.0	ghi
3	NaN	444.0	xyz

In [75]:

```
df.fillna('FILL')
```

Out[75]:

	col1	col2	col3
0	1	FILL	abc
1	2	555	def
2	3	666	ghi
3	FILL	444	xyz

In [89]:

```
data = {'A':['foo','foo','foo','bar','bar','bar'],
     'B':['one','one','two','two','one','one'],
       'C':['x','y','x','y','x','y'],
       'D':[1,3,2,5,4,1]}

df = pd.DataFrame(data)
```

In [90]:

```
df
```

Out[90]:

	A	B	C	D
0	foo	one	x	1
1	foo	one	y	3
2	foo	two	x	2
3	bar	two	y	5
4	bar	one	x	4
5	bar	one	y	1

In [91]:

```
df.pivot_table(values='D',index=['A', 'B'],columns=['C'])
```

Out[91]:

	C	x	y
A	**B**		
bar	**one**	4.0	1.0
	two	NaN	5.0
foo	**one**	1.0	3.0
	two	2.0	NaN

Great Job!