Data Retrieval

- Install the following python modules
- html5lib xlrd openpyxl sqlalchemy pymysql pymongo lxml

CSV and Text files

- read_csv, read_table, to_csv

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

In [2]:
```
%cat read_write_data/file_01.csv
```

```
white,red,blue,green,animal
1,5,2,3,cat
2,7,8,5,dog
3,3,6,7,horse
2,2,8,3,duck
4,4,2,1,mouse
```

In [3]:
```
df = pd.read_csv('read_write_data/file_01.csv')
df
```

Out[3]:

	white	red	blue	green	animal
0	1	5	2	3	cat
1	2	7	8	5	dog
2	3	3	6	7	horse
3	2	2	8	3	duck
4	4	4	2	1	mouse

In [4]:
```
df.shape
```

Out[4]:
```
(5, 5)
```

In [5]:
```
pd.read_table('read_write_data/file_01.csv')
```

Out[5]:

	white,red,blue,green,animal
0	1,5,2,3,cat
1	2,7,8,5,dog
2	3,3,6,7,horse
3	2,2,8,3,duck
4	4,4,2,1,mouse

In [6]:
```
pd.read_table('read_write_data/file_01.csv', sep=',')
```
Out[6]:

	white	red	blue	green	animal
0	1	5	2	3	cat
1	2	7	8	5	dog
2	3	3	6	7	horse
3	2	2	8	3	duck
4	4	4	2	1	mouse

In [7]:
```
# header

%cat read_write_data/file_02.csv
```
```
1,5,2,3,cat
2,7,8,5,dog
3,3,6,7,horse
2,2,8,3,duck
4,4,2,1,mouse
```
In [8]:
```
pd.read_csv('read_write_data/file_02.csv')
```
Out[8]:

	1	5	2	3	cat
0	2	7	8	5	dog
1	3	3	6	7	horse
2	2	2	8	3	duck
3	4	4	2	1	mouse

In [9]:
```
pd.read_csv('read_write_data/file_02.csv', header=None)
```
Out[9]:

	0	1	2	3	4
0	1	5	2	3	cat
1	2	7	8	5	dog
2	3	3	6	7	horse
3	2	2	8	3	duck
4	4	4	2	1	mouse

In [10]:
```
pd.read_csv('read_write_data/file_02.csv',
            names=['white','red','blue','green','animal'])
```
Out[10]:

	white	red	blue	green	animal
0	1	5	2	3	cat
1	2	7	8	5	dog
2	3	3	6	7	horse
3	2	2	8	3	duck
4	4	4	2	1	mouse

In [11]:
```
# Hierarchical structure

%cat read_write_data/file_03.csv
```

```
color,status,item1,item2,item3
black,up,3,4,6
black,down,2,6,7
white,up,5,5,5
white,down,3,3,2
white,left,1,2,1
red,up,2,2,2
red,down,1,1,4
```

In [12]:
```
df = pd.read_csv('read_write_data/file_03.csv')
df
```

Out[12]:

	color	status	item1	item2	item3
0	black	up	3	4	6
1	black	down	2	6	7
2	white	up	5	5	5
3	white	down	3	3	2
4	white	left	1	2	1
5	red	up	2	2	2
6	red	down	1	1	4

In [13]:
```
df = pd.read_csv('read_write_data/file_03.csv',
                 index_col=['color','status'])
df
```

Out[13]:

		item1	item2	item3
color	status			
black	up	3	4	6
	down	2	6	7
white	up	5	5	5
	down	3	3	2
	left	1	2	1
red	up	2	2	2
	down	1	1	4

In [14]:
```
df.shape
```

Out[14]:
```
(7, 3)
```

In [15]:
```
df.loc['black']
```

Out[15]:

	item1	item2	item3
status			
up	3	4	6
down	2	6	7

In [16]:
```
df.loc['black', 'down']
```

Out[16]:
```
item1     2
item2     6
item3     7
Name: (black, down), dtype: int64
```

In [17]:
```
# spaces or tabs in random order

%cat read_write_data/file_04.txt
```

```
white red   blue         green
1  5                2  3
2  7      8   5
   3   3   6    7
```

In [18]:
```
pd.read_table('read_write_data/file_04.txt')
```

Out[18]:

	white red blue	green
1 5	NaN	2 3
2 7	8 5	NaN
3 3 6 7	NaN	NaN

In [19]:
```
# Use regular expression for separator

pd.read_table('read_write_data/file_04.txt', sep='\s+',
             engine='python')
```

Out[19]:

	white	red	blue	green
0	1	5	2	3
1	2	7	8	5
2	3	3	6	7

In [20]:
```
%cat read_write_data/file_05.txt
```

```
000END123AAA122
001END124BBB321
002END125CCC333
```

In [21]:
```
# Extract numerical parts

pd.read_table('read_write_data/file_05.txt', sep='\D+',
              header=None, engine='python')
```

Out[21]:

	0	1	2
0	0	123	122
1	1	124	321
2	2	125	333

In [22]:
```
%cat read_write_data/file_06.txt
```

```
########### LOG FILE ############
This file has been generated by automatic system
white,red,blue,green,animal
12-Feb-2015: Counting of animals inside the house
1,5,2,3,cat
2,7,8,5,dog
13-Feb-2015: Counting of animals outside the house
3,3,6,7,horse
2,2,8,3,duck
4,4,2,1,mouse
```

In [23]:
```
# Skip lines

pd.read_table('read_write_data/file_06.txt', sep=',',
              skiprows=[0,1,3,6])
```

Out[23]:

	white	red	blue	green	animal
0	1	5	2	3	cat
1	2	7	8	5	dog
2	3	3	6	7	horse
3	2	2	8	3	duck
4	4	4	2	1	mouse

In [24]:
```
pd.read_csv('read_write_data/file_06.txt',
              skiprows=[0,1,3,6])
```

Out[24]:

	white	red	blue	green	animal
0	1	5	2	3	cat
1	2	7	8	5	dog
2	3	3	6	7	horse
3	2	2	8	3	duck
4	4	4	2	1	mouse

In [25]:
```
%cat read_write_data/file_02.csv
```

```
1,5,2,3,cat
2,7,8,5,dog
3,3,6,7,horse
2,2,8,3,duck
4,4,2,1,mouse
```

In [26]:
```python
# read only a portion of the file

pd.read_csv('read_write_data/file_02.csv',
              skiprows=2, header=None)
```

Out[26]:

	0	1	2	3	4
0	3	3	6	7	horse
1	2	2	8	3	duck
2	4	4	2	1	mouse

In [27]:
```python
pd.read_csv('read_write_data/file_02.csv',
              skiprows=2, nrows=1, header=None)
```

Out[27]:

	0	1	2	3	4
0	3	3	6	7	horse

In [28]:
```
%cat read_write_data/file_01.csv
```

```
white,red,blue,green,animal
1,5,2,3,cat
2,7,8,5,dog
3,3,6,7,horse
2,2,8,3,duck
4,4,2,1,mouse
```

In [29]:
```python
# read in chunks

out = []

pieces = pd.read_csv('read_write_data/file_01.csv',chunksize=2)

for piece in pieces:
    print(piece, '\n')
    out.append(piece['red'].sum())

print(out)
```

```
   white  red  blue  green animal
0      1    5     2      3    cat
1      2    7     8      5    dog

   white  red  blue  green animal
2      3    3     6      7  horse
3      2    2     8      3   duck

   white  red  blue  green animal
4      4    4     2      1  mouse

[12, 5, 4]
```

In [30]:
```
# Using List Comprehension

pieces = pd.read_csv('read_write_data/file_01.csv',chunksize=2)

[piece['red'].sum() for piece in pieces]
```

Out[30]:
```
[12, 5, 4]
```

Writing Data in CSV format

In [31]:
```
frame = pd.DataFrame(
    np.arange(16).reshape((4,4)),
    index=['red', 'blue', 'yellow', 'white'],
    columns=['ball','pen','pencil','paper'])

frame
```

Out[31]:

	ball	pen	pencil	paper
red	0	1	2	3
blue	4	5	6	7
yellow	8	9	10	11
white	12	13	14	15

In [32]:
```
frame.to_csv('read_write_data/file_07.csv')

%cat read_write_data/file_07.csv
```

```
,ball,pen,pencil,paper
red,0,1,2,3
blue,4,5,6,7
yellow,8,9,10,11
white,12,13,14,15
```

In [33]:
```
frame.to_csv('read_write_data/file_07b.csv',
            header=True, index=False)

%cat read_write_data/file_07b.csv
```

```
ball,pen,pencil,paper
0,1,2,3
4,5,6,7
8,9,10,11
12,13,14,15
```

In [34]:
```
frame.to_csv('read_write_data/file_07c.csv',
            header=False, index=False)

%cat read_write_data/file_07c.csv
```

```
0,1,2,3
4,5,6,7
8,9,10,11
12,13,14,15
```

In [35]:
```
frame3 = pd.DataFrame(
    [
        [6,np.nan,np.nan,6,np.nan],
        [8,np.nan,np.nan,np.nan,np.nan],
        [10,np.nan,np.nan,np.nan,np.nan],
        [20,np.nan,np.nan,20.0,np.nan],
        [22,np.nan,np.nan,19.0,np.nan]
    ],
    index=['blue','green','red','white','yellow'],
    columns=['ball','mug','paper','pen','pencil'])

frame3
```

Out[35]:

	ball	mug	paper	pen	pencil
blue	6	NaN	NaN	6.0	NaN
green	8	NaN	NaN	NaN	NaN
red	10	NaN	NaN	NaN	NaN
white	20	NaN	NaN	20.0	NaN
yellow	22	NaN	NaN	19.0	NaN

In [36]:
```
frame3.to_csv('read_write_data/file_08.csv',
            header=True, index=True)

%cat read_write_data/file_08.csv
```

```
,ball,mug,paper,pen,pencil
blue,6,,,6.0,
green,8,,,,
red,10,,,,
white,20,,,20.0,
yellow,22,,,19.0,
```

In [37]:
```
df = pd.read_csv('read_write_data/file_08.csv', index_col=0)
df
```

Out[37]:

	ball	mug	paper	pen	pencil
blue	6	NaN	NaN	6.0	NaN
green	8	NaN	NaN	NaN	NaN
red	10	NaN	NaN	NaN	NaN
white	20	NaN	NaN	20.0	NaN
yellow	22	NaN	NaN	19.0	NaN

In [38]:
```
df.dropna(axis=1)
```
Out[38]:

	ball
blue	6
green	8
red	10
white	20
yellow	22

In [39]:
```
frame3.to_csv('read_write_data/file_08b.csv', na_rep='Nan',
            header=True, index=True)

%cat read_write_data/file_08b.csv
```
```
,ball,mug,paper,pen,pencil
blue,6,Nan,Nan,6.0,Nan
green,8,Nan,Nan,Nan,Nan
red,10,Nan,Nan,Nan,Nan
white,20,Nan,Nan,20.0,Nan
yellow,22,Nan,Nan,19.0,Nan
```
In [40]:
```
df = pd.read_csv('read_write_data/file_08b.csv', index_col=0)
df
```
Out[40]:

	ball	mug	paper	pen	pencil
blue	6	Nan	Nan	6.0	Nan
green	8	Nan	Nan	Nan	Nan
red	10	Nan	Nan	Nan	Nan
white	20	Nan	Nan	20.0	Nan
yellow	22	Nan	Nan	19.0	Nan

In [41]:
```
df.dropna(axis=1)
```
Out[41]:

	ball	mug	paper	pen	pencil
blue	6	Nan	Nan	6.0	Nan
green	8	Nan	Nan	Nan	Nan
red	10	Nan	Nan	Nan	Nan
white	20	Nan	Nan	20.0	Nan
yellow	22	Nan	Nan	19.0	Nan

```
In [42]: df = pd.read_csv('read_write_data/file_08b.csv', index_col=0, na_values='Nan')
         df
```

Out[42]:

	ball	mug	paper	pen	pencil
blue	6	NaN	NaN	6.0	NaN
green	8	NaN	NaN	NaN	NaN
red	10	NaN	NaN	NaN	NaN
white	20	NaN	NaN	20.0	NaN
yellow	22	NaN	NaN	19.0	NaN

```
In [43]: df.dropna(axis=1)
```

Out[43]:

	ball
blue	6
green	8
red	10
white	20
yellow	22

Writing Data to HTML

```
In [44]: frame = pd.DataFrame(np.arange(10,14).reshape(2,2))
         frame
```

Out[44]:

	0	1
0	10	11
1	12	13

```
In [45]: print(frame.to_html())
```

```
<table border="1" class="dataframe">
  <thead>
    <tr style="text-align: right;">
      <th></th>
      <th>0</th>
      <th>1</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>0</th>
      <td>10</td>
      <td>11</td>
    </tr>
    <tr>
      <th>1</th>
      <td>12</td>
      <td>13</td>
    </tr>
  </tbody>
</table>
```

```
In [46]: frame = pd.DataFrame( np.random.random((4,4)),
                               index = ['white','black','red','blue'],
                               columns = ['up','down','right','left'])
         frame
```

Out[46]:

	up	down	right	left
white	0.394032	0.721311	0.059766	0.103326
black	0.960509	0.693363	0.454659	0.499577
red	0.148201	0.439648	0.475753	0.758542
blue	0.218904	0.832922	0.328875	0.964803

```
In [47]: s = ['<HTML>']
         s.append('<HEAD><TITLE>My DataFrame</TITLE></HEAD>')
         s.append('<BODY>')
         s.append(frame.to_html())
         s.append('</BODY></HTML>')
         html = ''.join(s)
```

In [48]:
```python
html_file = open('read_write_data/myFrame.html','w')
html_file.write(html)
html_file.close()

%cat read_write_data/myFrame.html
```

```
<HTML><HEAD><TITLE>My DataFrame</TITLE></HEAD><BODY><table border="1" class="datafram
e">
  <thead>
    <tr style="text-align: right;">
      <th></th>
      <th>up</th>
      <th>down</th>
      <th>right</th>
      <th>left</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>white</th>
      <td>0.394032</td>
      <td>0.721311</td>
      <td>0.059766</td>
      <td>0.103326</td>
    </tr>
    <tr>
      <th>black</th>
      <td>0.960509</td>
      <td>0.693363</td>
      <td>0.454659</td>
      <td>0.499577</td>
    </tr>
    <tr>
      <th>red</th>
      <td>0.148201</td>
      <td>0.439648</td>
      <td>0.475753</td>
      <td>0.758542</td>
    </tr>
    <tr>
      <th>blue</th>
      <td>0.218904</td>
      <td>0.832922</td>
      <td>0.328875</td>
      <td>0.964803</td>
    </tr>
  </tbody>
</table></BODY></HTML>
```

Reading from HTML

In [49]:

```
frames = pd.read_html('read_write_data/myFrame.html',
                      index_col=0,
                      flavor='html5lib')
frames[0]
```

Out[49]:

	up	down	right	left
white	0.394032	0.721311	0.059766	0.103326
black	0.960509	0.693363	0.454659	0.499577
red	0.148201	0.439648	0.475753	0.758542
blue	0.218904	0.832922	0.328875	0.964803

In [50]:

```
frames = pd.read_html('http://www.fdic.gov/bank/individual/failed/banklist.html',
                                  flavor='html5lib')
len(frames)
```

Out[50]: 1

In [51]:

```
frames[0]
```

Out[51]:

	Bank Name	City	ST	CERT	Acquiring Institution	Closing Date	Updated Date
0	The Enloe State Bank	Cooper	TX	10716	Legend Bank, N. A.	May 31, 2019	August 22, 2019
1	Washington Federal Bank for Savings	Chicago	IL	30570	Royal Savings Bank	December 15, 2017	July 24, 2019
2	The Farmers and Merchants State Bank of Argonia	Argonia	KS	17719	Conway Bank	October 13, 2017	August 12, 2019
3	Fayette County Bank	Saint Elmo	IL	1802	United Fidelity Bank, fsb	May 26, 2017	January 29, 2019
4	Guaranty Bank, (d/b/a BestBank in Georgia & Mi...	Milwaukee	WI	30003	First-Citizens Bank & Trust Company	May 5, 2017	March 22, 2018
...	...	...	...	...	...	...	...
551	Superior Bank, FSB	Hinsdale	IL	32646	Superior Federal, FSB	July 27, 2001	August 19, 2014
552	Malta National Bank	Malta	OH	6629	North Valley Bank	May 3, 2001	November 18, 2002
553	First Alliance Bank & Trust Co.	Manchester	NH	34264	Southern New Hampshire Bank & Trust	February 2, 2001	February 18, 2003
554	National State Bank of Metropolis	Metropolis	IL	3815	Banterra Bank of Marion	December 14, 2000	March 17, 2005
555	Bank of Honolulu	Honolulu	HI	21029	Bank of the Orient	October 13, 2000	March 17, 2005

556 rows × 7 columns

In [52]:

```
frames = pd.read_html('https://projects.fivethirtyeight.com/global-club-soccer-rankings/
                                 flavor='html5lib')
len(frames)
```

Out[52]: 1

In [53]:
```python
frames[0]
```

Out[53]:

	Unnamed: 0_level_0	Unnamed: 1_level_0	Unnamed: 2_level_0	Unnamed: 3_level_0	Unnamed: 4_level_0	Team rating		
	Rank	1-week change	team	League	League country	off.	def.	spi
0	1	NaN	Man. City	Premier League	England	3.3	0.2	95.6
1	2	NaN	Bayern Munich	Bundesliga	Germany	3.2	0.4	92.9
2	3	NaN	Liverpool	Premier League	England	2.9	0.3	92.9
3	4	NaN	PSG	Ligue 1	France	2.8	0.4	89.8
4	5	NaN	Barcelona	La Liga	Spain	2.8	0.4	89.5
...	...	...	...	...	...	...	...	...
624	625	-1.0	Macclesfield	League Two	England	0.2	2.2	7.0
625	626	1.0	Morecambe	League Two	England	0.3	2.5	6.8
626	627	-2.0	Walsall	League Two	England	0.2	2.3	6.6
627	628	-2.0	Port Vale	League Two	England	0.2	2.3	6.4
628	629	-1.0	C.S. Switchbacks	USL Championship	USA	0.2	2.4	5.6

629 rows × 8 columns

In [54]:
```python
frames[0]['Team rating']
```

Out[54]:

	off.	def.	spi
0	3.3	0.2	95.6
1	3.2	0.4	92.9
2	2.9	0.3	92.9
3	2.8	0.4	89.8
4	2.8	0.4	89.5
...	...	...	...
624	0.2	2.2	7.0
625	0.3	2.5	6.8
626	0.2	2.3	6.6
627	0.2	2.3	6.4
628	0.2	2.4	5.6

629 rows × 3 columns

In [55]:
```
frames[0].xs("team", level=1, axis=1)
```

Out[55]:

	Unnamed: 2_level_0
0	Man. City
1	Bayern Munich
2	Liverpool
3	PSG
4	Barcelona
...	...
624	Macclesfield
625	Morecambe
626	Walsall
627	Port Vale
628	C.S. Switchbacks

629 rows × 1 columns

In [56]:
```
idx = pd.IndexSlice
idx

frames[0].loc[:, idx[:, 'team']]
```

Out[56]:

	Unnamed: 2_level_0
	team
0	Man. City
1	Bayern Munich
2	Liverpool
3	PSG
4	Barcelona
...	...
624	Macclesfield
625	Morecambe
626	Walsall
627	Port Vale
628	C.S. Switchbacks

629 rows × 1 columns

JSON Data

In [57]:
```
frame = pd.DataFrame(np.arange(16).reshape(4,4),
                     index=['white','black','red','blue'],
                     columns=['up','down','right','left'])
frame
```

Out[57]:

	up	down	right	left
white	0	1	2	3
black	4	5	6	7
red	8	9	10	11
blue	12	13	14	15

In [58]:
```
frame.to_json('read_write_data/frame1.json', orient='columns')
%cat read_write_data/frame1.json
```

```
{"up":{"white":0,"black":4,"red":8,"blue":12},"down":{"white":1,"black":5,"red":9,"blu
e":13},"right":{"white":2,"black":6,"red":10,"blue":14},"left":{"white":3,"black":7,"r
ed":11,"blue":15}}
```

In [59]:
```
pd.read_json('read_write_data/frame1.json', orient='columns')
```

Out[59]:

	up	down	right	left
white	0	1	2	3
black	4	5	6	7
red	8	9	10	11
blue	12	13	14	15

In [60]:
```
frame.to_json('read_write_data/frame2.json', orient="index")
%cat read_write_data/frame2.json
```

```
{"white":{"up":0,"down":1,"right":2,"left":3},"black":{"up":4,"down":5,"right":6,"lef
t":7},"red":{"up":8,"down":9,"right":10,"left":11},"blue":{"up":12,"down":13,"right":1
4,"left":15}}
```

In [61]:
```
pd.read_json('read_write_data/frame2.json', orient="index")
```

Out[61]:

	down	left	right	up
black	5	7	6	4
blue	13	15	14	12
red	9	11	10	8
white	1	3	2	0

In [62]:
```
frame.to_json('read_write_data/frame3.json', orient="records")
%cat read_write_data/frame3.json
```

```
[{"up":0,"down":1,"right":2,"left":3},{"up":4,"down":5,"right":6,"left":7},{"up":8,"do
wn":9,"right":10,"left":11},{"up":12,"down":13,"right":14,"left":15}]
```

In [63]:
```
frame.to_json('read_write_data/frame4.json', orient="values")
%cat read_write_data/frame4.json
```

```
[[0,1,2,3],[4,5,6,7],[8,9,10,11],[12,13,14,15]]
```

In [64]:
```
frame.to_json('read_write_data/frame5.json', orient="split")
%cat read_write_data/frame5.json
```

```
{"columns":["up","down","right","left"],"index":["white","black","red","blue"],"data":
[[0,1,2,3],[4,5,6,7],[8,9,10,11],[12,13,14,15]]}
```

In [65]:
```
# more general json data

%cat read_write_data/books.json
```

```
[
  {"writer": "Mark Ross",
    "nationality": "USA",
    "books": [
          {"title": "XML Cookbook", "price": 23.56},
          {"title": "Python Fundamentals", "price": 50.70},
          {"title": "The NumPy library", "price": 12.30}
         ]
   },

  {"writer": "Barbara Bracket",
    "nationality": "UK",
    "books": [
          {"title": "Java Enterprise", "price": 28.60},
          {"title": "HTML5", "price": 31.35},
          {"title": "Python for Dummies", "price": 28.00}
         ]
   }
 ]
```

In [66]:
```
from pandas.io.json import json_normalize, loads
```

In [67]:
```
file = open('read_write_data/books.json', 'r')
text = file.read()
text = loads(text)
print(text)
```

```
[{'writer': 'Mark Ross', 'nationality': 'USA', 'books': [{'title': 'XML Cookbook', 'pr
ice': 23.56}, {'title': 'Python Fundamentals', 'price': 50.7}, {'title': 'The NumPy li
brary', 'price': 12.3}]}, {'writer': 'Barbara Bracket', 'nationality': 'UK', 'books':
[{'title': 'Java Enterprise', 'price': 28.6}, {'title': 'HTML5', 'price': 31.35}, {'ti
tle': 'Python for Dummies', 'price': 28.0}]}]
```

In [68]:
```
json_normalize(text, 'books')
```

Out[68]:

	title	price
0	XML Cookbook	23.56
1	Python Fundamentals	50.70
2	The NumPy library	12.30
3	Java Enterprise	28.60
4	HTML5	31.35
5	Python for Dummies	28.00

In [69]:
```
json_normalize(text, 'books', 'writer')
```

Out[69]:

	title	price	writer
0	XML Cookbook	23.56	Mark Ross
1	Python Fundamentals	50.70	Mark Ross
2	The NumPy library	12.30	Mark Ross
3	Java Enterprise	28.60	Barbara Bracket
4	HTML5	31.35	Barbara Bracket
5	Python for Dummies	28.00	Barbara Bracket

In [70]:
```
frame = json_normalize(text, 'books', ['nationality', 'writer'])
frame
```

Out[70]:

	title	price	nationality	writer
0	XML Cookbook	23.56	USA	Mark Ross
1	Python Fundamentals	50.70	USA	Mark Ross
2	The NumPy library	12.30	USA	Mark Ross
3	Java Enterprise	28.60	UK	Barbara Bracket
4	HTML5	31.35	UK	Barbara Bracket
5	Python for Dummies	28.00	UK	Barbara Bracket

In [71]:
```
frame.set_index(['writer', 'nationality'])
```

Out[71]:

		title	price
writer	nationality		
	USA	XML Cookbook	23.56
Mark Ross	USA	Python Fundamentals	50.70
	USA	The NumPy library	12.30
	UK	Java Enterprise	28.60
Barbara Bracket	UK	HTML5	31.35
	UK	Python for Dummies	28.00

XML Data

```
In [72]: %cat read_write_data/books.xml

<?xml version="1.0"?>
<Catalog>
   <Book id="ISBN9872122367564">
      <Author>Ross, Mark</Author>
      <Title>XML Cookbook</Title>
      <Genre>Computer</Genre>
      <Price>23.56</Price>
      <PublishDate>2014-22-01</PublishDate>
   </Book>
   <Book id="ISBN9872122367564">
      <Author>Bracket, Barbara</Author>
      <Title>XML for Dummies</Title>
      <Genre>Computer</Genre>
      <Price>35.95</Price>
      <PublishDate>2014-12-16</PublishDate>
   </Book>
</Catalog>
```

```
In [73]: from lxml import objectify
```

```
In [74]: xml = objectify.parse('read_write_data/books.xml')
         xml

Out[74]: <lxml.etree._ElementTree at 0x110b582c8>
```

```
In [75]: root = xml.getroot()
         root

Out[75]: <Element Catalog at 0x1134155c8>
```

```
In [76]: root.Book

Out[76]: <Element Book at 0x113415fc8>
```

```
In [77]: root.Book.Author

Out[77]: 'Ross, Mark'
```

```
In [78]: root.getchildren()

Out[78]: [<Element Book at 0x113415fc8>, <Element Book at 0x1133ffec8>]
```

```
In [79]: [book.Author for book in root.getchildren()]

Out[79]: ['Ross, Mark', 'Bracket, Barbara']
```

```
In [80]: [child.tag for child in root.Book.getchildren()]

Out[80]: ['Author', 'Title', 'Genre', 'Price', 'PublishDate']
```

```
In [81]: [child.text for child in root.Book.getchildren()]

Out[81]: ['Ross, Mark', 'XML Cookbook', 'Computer', '23.56', '2014-22-01']
```

```
In [82]: root.Book.attrib.keys()

Out[82]: ['id']
```

In [83]:
```
# Assuming at least one child

def etree2df(root):

    column_names = root.getchildren()[0].attrib.keys()

    column_names += [child.tag for child in root.getchildren()[0].getchildren()]

    xmlframe = pd.DataFrame(columns=column_names)

    for j in range(0, len(root.getchildren())):

        obj = root.getchildren()[j]

        texts = obj.attrib.values()
        texts += [child.text for child in obj.getchildren()]

        row = dict(zip(column_names, texts))
        row_s = pd.Series(row)
        row_s.name = j

        xmlframe = xmlframe.append(row_s)

    return xmlframe
```

In [84]:
```
etree2df(root)
```

Out[84]:

	id	Author	Title	Genre	Price	PublishDate
0	ISBN9872122367564	Ross, Mark	XML Cookbook	Computer	23.56	2014-22-01
1	ISBN9872122367564	Bracket, Barbara	XML for Dummies	Computer	35.95	2014-12-16

Excel Data

In [85]:
```
pd.read_excel('read_write_data/file01_data.xlsx', index_col=0)
```

Out[85]:

	white	red	green	black
a	12	23	17	18
b	22	16	19	18
c	14	23	22	21

In [86]:
```
pd.read_excel('read_write_data/file01_data.xlsx', 'Sheet2', index_col=0)
```

Out[86]:

	yellow	purple	blue	orange
A	11	16	44	22
B	20	22	23	44
C	30	31	37	32

In [87]:

```
# use index - 0, 1, ...

pd.read_excel('read_write_data/file01_data.xlsx', 1, index_col=0)
```

Out[87]:

	yellow	purple	blue	orange
A	11	16	44	22
B	20	22	23	44
C	30	31	37	32

In [88]:

```
frame = pd.DataFrame(np.arange(16).reshape(4,4),
                    index=['white','black','red','blue'],
                    columns=['up','down','right','left'])
frame
```

Out[88]:

	up	down	right	left
white	0	1	2	3
black	4	5	6	7
red	8	9	10	11
blue	12	13	14	15

In [89]:

```
frame.to_excel('read_write_data/file02_data.xlsx')
```

In [90]:

```
pd.read_excel('read_write_data/file02_data.xlsx', index_col=0)
```

Out[90]:

	up	down	right	left
white	0	1	2	3
black	4	5	6	7
red	8	9	10	11
blue	12	13	14	15

Pickle - Python Object Serialization

In [91]:

```
frame = pd.DataFrame(np.arange(16).reshape(4,4),
                    index=['up','down','left','right'])
frame
```

Out[91]:

	0	1	2	3
up	0	1	2	3
down	4	5	6	7
left	8	9	10	11
right	12	13	14	15

In [92]:

```
frame.to_pickle('read_write_data/frame.pkl')
```

In [93]:
```
pd.read_pickle('read_write_data/frame.pkl')
```

Out[93]:

	0	1	2	3
up	0	1	2	3
down	4	5	6	7
left	8	9	10	11
right	12	13	14	15

Databases

In [94]:
```
from sqlalchemy import create_engine
from pandas.io import sql
```

In [95]:
```
engine = create_engine('sqlite:///foo.db')
```

In [96]:
```
frame = pd.DataFrame(
    np.arange(20).reshape(4,5),
    columns=['white','red','blue','black','green'])

frame
```

Out[96]:

	white	red	blue	black	green
0	0	1	2	3	4
1	5	6	7	8	9
2	10	11	12	13	14
3	15	16	17	18	19

In [97]:
```
sql.execute('DROP TABLE IF EXISTS colors', engine)
frame.to_sql('colors',engine, index=False)
```

In [98]:
```
pd.read_sql('colors', engine)
```

Out[98]:

	white	red	blue	black	green
0	0	1	2	3	4
1	5	6	7	8	9
2	10	11	12	13	14
3	15	16	17	18	19

In [99]:
```
pd.read_sql_query('SELECT white, blue FROM colors', engine)
```

Out[99]:

	white	blue
0	0	2
1	5	7
2	10	12
3	15	17

In [100]:
```
pd.read_sql_query('SELECT name FROM sqlite_master WHERE type="table";', engine)
```

Out[100]:

	name
0	colors

MongoDB database

In [101]:
```
from pymongo import MongoClient
```

In [102]:
```
url = 'mongodb://cs602_user:cs602_secret@ds115768.mlab.com:15768/cs602db';

client = MongoClient(url)
```

In [103]:
```
db = client.cs602db
db
```

Out[103]:
```
Database(MongoClient(host=['ds115768.mlab.com:15768'], document_class=dict, tz_aware=F
alse, connect=True), 'cs602db')
```

In [104]:
```
collection = db['zipcodes']
collection
```

Out[104]:
```
Collection(Database(MongoClient(host=['ds115768.mlab.com:15768'], document_class=dict,
tz_aware=False, connect=True), 'cs602db'), 'zipcodes')
```

In [105]:
```
len(list(collection.find()))
```

Out[105]:
```
29353
```

```
In [106]: list(collection.find())[:10]

Out[106]: [{'_id': '01012',
            'city': 'CHESTERFIELD',
            'loc': [-72.833309, 42.38167],
            'pop': 177,
            'state': 'MA'},
           {'_id': '01010',
            'city': 'BRIMFIELD',
            'loc': [-72.188455, 42.116543],
            'pop': 3706,
            'state': 'MA'},
           {'_id': '01020',
            'city': 'CHICOPEE',
            'loc': [-72.576142, 42.176443],
            'pop': 31495,
            'state': 'MA'},
           {'_id': '01013',
            'city': 'CHICOPEE',
            'loc': [-72.607962, 42.162046],
            'pop': 23396,
            'state': 'MA'},
           {'_id': '01007',
            'city': 'BELCHERTOWN',
            'loc': [-72.410953, 42.275103],
            'pop': 10579,
            'state': 'MA'},
           {'_id': '01011',
            'city': 'CHESTER',
            'loc': [-72.988761, 42.279421],
            'pop': 1688,
            'state': 'MA'},
           {'_id': '01026',
            'city': 'CUMMINGTON',
            'loc': [-72.905767, 42.435296],
            'pop': 1484,
            'state': 'MA'},
           {'_id': '01028',
            'city': 'EAST LONGMEADOW',
            'loc': [-72.505565, 42.067203],
            'pop': 13367,
            'state': 'MA'},
           {'_id': '01027',
            'city': 'MOUNT TOM',
            'loc': [-72.679921, 42.264319],
            'pop': 16864,
            'state': 'MA'},
           {'_id': '01022',
            'city': 'WESTOVER AFB',
            'loc': [-72.558657, 42.196672],
            'pop': 1764,
            'state': 'MA'}]
```

In [107]:
```python
zipcodes = pd.DataFrame(list(collection.find()), columns=['state', 'city', '_id', 'loc',

zipcodes = zipcodes.set_index(['state', 'city'])

zipcodes.loc['MA'].loc['BOSTON']
```

Out[107]:

	_id	loc	pop
city			
BOSTON	02108	[-71.068432, 42.357603]	3697
BOSTON	02109	[-71.053386, 42.362963]	3926
BOSTON	02111	[-71.0629, 42.350348]	3759
BOSTON	02115	[-71.092215, 42.342706]	25597
BOSTON	02110	[-71.051417, 42.357636]	957
BOSTON	02113	[-71.055958, 42.365656]	6698
BOSTON	02114	[-71.06823, 42.361111]	10246
BOSTON	02116	[-71.076798, 42.349201]	17459
BOSTON	02199	[-71.082543, 42.347873]	886
BOSTON	02210	[-71.046511, 42.348921]	308
BOSTON	02215	[-71.102689, 42.347088]	17769

In [108]:
```python
zipcodes.loc[('MA','BOSTON')]
```

```
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages/ipykerne
l_launcher.py:1: PerformanceWarning: indexing past lexsort depth may impact performanc
e.
  """Entry point for launching an IPython kernel.
```

Out[108]:

		_id	loc	pop
state	**city**			
	BOSTON	02108	[-71.068432, 42.357603]	3697
	BOSTON	02109	[-71.053386, 42.362963]	3926
	BOSTON	02111	[-71.0629, 42.350348]	3759
	BOSTON	02115	[-71.092215, 42.342706]	25597
	BOSTON	02110	[-71.051417, 42.357636]	957
MA	BOSTON	02113	[-71.055958, 42.365656]	6698
	BOSTON	02114	[-71.06823, 42.361111]	10246
	BOSTON	02116	[-71.076798, 42.349201]	17459
	BOSTON	02199	[-71.082543, 42.347873]	886
	BOSTON	02210	[-71.046511, 42.348921]	308
	BOSTON	02215	[-71.102689, 42.347088]	17769

In [ ]:
```python

```