JSON examples and exercise

- get familiar with packages for dealing with JSON
- study examples with JSON strings and files
- work on exercise to be completed and submitted

- reference: http://pandas-docs.github.io/pandas-docs-travis/io.html#json
- data source: http://jsonstudio.com/resources/

In [6]:
```python
import pandas as pd
```

imports for Python, Pandas

In [7]:
```python
import json
from pandas.io.json import json_normalize
```

JSON example, with string

- demonstrates creation of normalized dataframes (tables) from nested json string
- source: http://pandas-docs.github.io/pandas-docs-travis/io.html#normalization

In [8]:
```python
# define json string
data = [{'state': 'Florida',
         'shortname': 'FL',
         'info': {'governor': 'Rick Scott'},
         'counties': [{'name': 'Dade', 'population': 12345},
                      {'name': 'Broward', 'population': 40000},
                      {'name': 'Palm Beach', 'population': 60000}]},
        {'state': 'Ohio',
         'shortname': 'OH',
         'info': {'governor': 'John Kasich'},
         'counties': [{'name': 'Summit', 'population': 1234},
                      {'name': 'Cuyahoga', 'population': 1337}]}]
```

In [9]:
```python
# use normalization to create tables from nested element
json_normalize(data, 'counties')
```

Out[9]:

	name	population
0	Dade	12345
1	Broward	40000
2	Palm Beach	60000
3	Summit	1234
4	Cuyahoga	1337

In [10]:
```python
# further populate tables created from nested element
json_normalize(data, 'counties', ['state', 'shortname', ['info', 'governor']])
```

Out[10]:

	name	population	state	shortname	info.governor
0	Dade	12345	Florida	FL	Rick Scott
1	Broward	40000	Florida	FL	Rick Scott
2	Palm Beach	60000	Florida	FL	Rick Scott
3	Summit	1234	Ohio	OH	John Kasich
4	Cuyahoga	1337	Ohio	OH	John Kasich

JSON example, with file

- demonstrates reading in a json file as a string and as a table
- uses small sample file containing data about projects funded by the World Bank
- data source: http://jsonstudio.com/resources/

In [11]:
```python
# load json as string
json.load((open('data/world_bank_projects_less.json')))
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-11-721b6769f6f5> in <module>
      1 # load json as string
----> 2 json.load((open('data/world_bank_projects_less.json')))

FileNotFoundError: [Errno 2] No such file or directory: 'data/world_bank_projects_less.json'
```

In [ ]:
```python
# load as Pandas dataframe
sample_json_df = pd.read_json('data/world_bank_projects_less.json')
sample_json_df
```

JSON exercise

Using data in file 'data/world_bank_projects.json' and the techniques demonstrated above,

1. Find the 10 countries with most projects
2. Find the top 10 major project themes (using column 'mjtheme_namecode')
3. In 2. above you will notice that some entries have only the code and the name is missing. Create a dataframe with the missing names filled in.

Import library files

In [2]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import json
from pandas.io.json import json_normalize
```

In [3]:
```python
df = pd.read_json("world_bank_projects.json")
```

In [4]:
```python
df.head()
```

Out[4]:

	sector	supplementprojectflg	projectfinancialtype	prodline	mjtheme	idacommamt	impagency
0	[{'Name': 'Primary education'}, {'Name': 'Seco...	N	IDA	PE	[Human development]	130000000	MINISTRY OF EDUCATION
1	[{'Name': 'Public administration- Other social...	N	OTHER	RE	[Economic management, Social protection and ri...	0	MINISTRY OF FINANCE
2	[{'Name': 'Rural and Inter-Urban Roads and Hig...	Y	IDA	PE	[Trade and integration, Public sector governan...	6060000	MINISTRY OF TRANSPORT AND COMMUNICATIONS
3	[{'Name': 'Other social services'}]	N	OTHER	RE	[Social dev/gender/inclusion, Social dev/gende...	0	LABOR INTENSIVE PUBLIC WORKS PROJECT PMU
4	[{'Name': 'General industry and trade sector'}...	N	IDA	PE	[Trade and integration, Financial and private ...	13100000	MINISTRY OF TRADE AND INDUSTRY

5 rows × 50 columns

In [5]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 50 columns):
sector                      500 non-null object
supplementprojectflg        498 non-null object
projectfinancialtype        500 non-null object
prodline                    500 non-null object
mjtheme                     491 non-null object
idacommamt                  500 non-null int64
impagency                   472 non-null object
project_name                500 non-null object
mjthemecode                 500 non-null object
closingdate                 370 non-null object
totalcommamt                500 non-null int64
id                          500 non-null object
mjsector_namecode           500 non-null object
docty                       446 non-null object
sector1                     500 non-null object
lendinginstr                495 non-null object
countrycode                 500 non-null object
sector2                     380 non-null object
totalamt                    500 non-null int64
mjtheme_namecode            500 non-null object
boardapprovaldate           500 non-null object
countryshortname            500 non-null object
sector4                     174 non-null object
prodlinetext                500 non-null object
productlinetype             500 non-null object
regionname                  500 non-null object
status                      500 non-null object
country_namecode            500 non-null object
envassesmentcategorycode    430 non-null object
project_abstract            362 non-null object
approvalfy                  500 non-null int64
projectdocs                 446 non-null object
lendprojectcost             500 non-null int64
lendinginstrtype            495 non-null object
theme1                      500 non-null object
grantamt                    500 non-null int64
themecode                   491 non-null object
borrower                    485 non-null object
sectorcode                  500 non-null object
sector3                     265 non-null object
majorsector_percent         500 non-null object
board_approval_month        500 non-null object
theme_namecode              491 non-null object
countryname                 500 non-null object
url                         500 non-null object
source                      500 non-null object
projectstatusdisplay        500 non-null object
ibrdcommamt                 500 non-null int64
sector_namecode             500 non-null object
_id                         500 non-null object
dtypes: int64(7), object(43)
memory usage: 195.4+ KB
```

In [6]:
```python
df.dtypes
```

Out[6]:
```
sector                      object
supplementprojectflg        object
projectfinancialtype        object
prodline                    object
mjtheme                     object
idacommamt                   int64
impagency                   object
project_name                object
mjthemecode                 object
closingdate                 object
totalcommamt                 int64
id                          object
mjsector_namecode           object
docty                       object
sector1                     object
lendinginstr                object
countrycode                 object
sector2                     object
totalamt                     int64
mjtheme_namecode            object
boardapprovaldate           object
countryshortname            object
sector4                     object
prodlinetext                object
productlinetype             object
regionname                  object
status                      object
country_namecode            object
envassesmentcategorycode    object
project_abstract            object
approvalfy                   int64
projectdocs                 object
lendprojectcost              int64
lendinginstrtype            object
theme1                      object
grantamt                     int64
themecode                   object
borrower                    object
sectorcode                  object
sector3                     object
majorsector_percent         object
board_approval_month        object
theme_namecode              object
countryname                 object
url                         object
source                      object
projectstatusdisplay        object
ibrdcommamt                  int64
sector_namecode             object
_id                         object
dtype: object
```

In [7]:
```python
df.describe()
```

Out[7]:

	idacommamt	totalcommamt	totalamt	approvalfy	lendprojectcost	grantamt	ibrdcommamt
count	5.000000e+02	5.000000e+02	5.000000e+02	500.000000	5.000000e+02	5.000000e+02	5.000000e+02
mean	3.542136e+07	7.271386e+07	6.828146e+07	2013.108000	1.547241e+08	4.432400e+06	3.286010e+07
std	7.681431e+07	1.234705e+08	1.242662e+08	0.722066	4.764211e+08	2.023307e+07	1.089197e+08
min	0.000000e+00	3.000000e+04	0.000000e+00	1999.000000	3.000000e+04	0.000000e+00	0.000000e+00
25%	0.000000e+00	5.000000e+06	0.000000e+00	2013.000000	6.472500e+06	0.000000e+00	0.000000e+00
50%	0.000000e+00	2.500000e+07	2.000000e+07	2013.000000	3.500000e+07	0.000000e+00	0.000000e+00
75%	3.700000e+07	9.045000e+07	8.625000e+07	2013.000000	1.021250e+08	1.695000e+06	0.000000e+00
max	6.000000e+08	1.307800e+09	1.307800e+09	2015.000000	5.170000e+09	3.650000e+08	1.307800e+09

Finding 10 countries with most projects

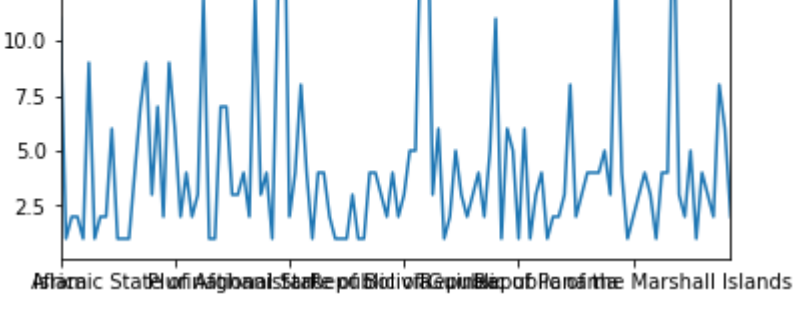
In [8]:
```python
df.countryname.value_counts().head(10)
```

Out[8]:
```
Republic of Indonesia              19
People's Republic of China         19
Socialist Republic of Vietnam      17
Republic of India                  16
Republic of Yemen                  13
Kingdom of Morocco                 12
Nepal                              12
People's Republic of Bangladesh    12
Republic of Mozambique             11
Africa                             11
Name: countryname, dtype: int64
```

plotting graph for 10 countries with most projects

In [9]:
```python
df.countryname.value_counts().sort_index().plot()
```

Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b4196ca4a8>
```



Finding Top 10 major project themes

In [10]:
```python
df.mjtheme_namecode.value_counts().head(10)
```

Out[10]:
```
[{'code': '11', 'name': 'Environment and natural resources management'}, {'code': '11', 'name': 'Environment and natural resources management'}]
12
[{'code': '8', 'name': 'Human development'}, {'code': '11', 'name': ''}]
11
[{'code': '8', 'name': 'Human development'}, {'code': '8', 'name': 'Human development'}]
8
[{'code': '4', 'name': 'Financial and private sector development'}, {'code': '4', 'name': 'Financial and private sector development'}]
6
[{'code': '2', 'name': 'Public sector governance'}, {'code': '2', 'name': 'Public sector governance'}, {'code': '2', 'name': 'Public sector governance'}]
6
[{'code': '8', 'name': 'Human development'}, {'code': '7', 'name': 'Social dev/gender/inclusion'}]
5
[{'code': '8', 'name': 'Human development'}, {'code': '8', 'name': 'Human development'}, {'code': '8', 'name': 'Human development'}, {'code': '8', 'name': 'Human development'}, {'code': '8', 'name': 'Human development'}]      5
[{'code': '11', 'name': 'Environment and natural resources management'}, {'code': '11', 'name': ''}]
5
[{'code': '11', 'name': 'Environment and natural resources management'}, {'code': '4', 'name': ''}]
5
[{'code': '4', 'name': 'Financial and private sector development'}, {'code': '5', 'name': 'Trade and integration'}]
5
Name: mjtheme_namecode, dtype: int64
```

In [11]:
```python
with open(r'D:\json files\data_wrangling_json\data\world_bank_projects.json') as  f :
    df1 = json.load(f)
```

In [12]:
```python
df2 = json_normalize(df1, 'mjtheme_namecode', ['id'])
```

Created DataFrame with missing values

In [13]:
```python
df2.head()
```

Out[13]:

	code	name	id
0	8	Human development	P129828
1	11		P129828
2	1	Economic management	P144674
3	6	Social protection and risk management	P144674
4	5	Trade and integration	P145310

In [15]:
```python
df2.sort_values('code', inplace=True)
df2
```

Out[15]:

	code	name	id
458	1	Economic management	P130925
1114	1		P131094
454	1	Economic management	P130925
453	1	Economic management	P128573
1257	1	Economic management	P133230
1260	1	Economic management	P133339
1212	1	Economic management	P121019
249	1	Economic management	P132948
1057	1	Economic management	P144633
841	1	Economic management	P122793
204	1	Economic management	P126034
2	1	Economic management	P144674
223	1	Economic management	P145018
803	1	Economic management	P127332
205	1	Economic management	P126034
212	1		P128282
222	1	Economic management	P145865
220	1	Economic management	P144917
1056	1	Economic management	P144633
175	1	Economic management	P118027
1045	1	Economic management	P133791
497	1	Economic management	P133663
1235	1	Economic management	P131440
1230	1	Economic management	P129465
1229	1	Economic management	P129465
900	1	Economic management	P143819
648	1	Economic management	P129825
647	1	Economic management	P129825
1078	1	Economic management	P131234
1218	1	Economic management	P130824
...	...	...	...
447	9	Urban development	P126899
739	9	Urban development	P127079
356	9	Urban development	P127543
1342	9	Urban development	P130174
341	9	Urban development	P144357
1347	9	Urban development	P131394
570	9	Urban development	P119039
535	9	Urban development	P120370
1473	9	Urban development	P128763
1470	9	Urban development	P128763
1495	9	Urban development	P126321
1095	9	Urban development	P130819
1104	9	Urban development	P130972
414	9	Urban development	P119063
1103	9	Urban development	P130972
194	9	Urban development	P128768
460	9		P117394
320	9	Urban development	P122950
940	9	Urban development	P127955
732	9	Urban development	P127036
471	9		P126817
930	9	Urban development	P126489
513	9	Urban development	P127163
917	9	Urban development	P143770
650	9	Urban development	P123384
318	9	Urban development	P122950
669	9	Urban development	P127303
1303	9	Urban development	P130528
1166	9	Urban development	P101289
1102	9	Urban development	P130972

1499 rows × 3 columns

Filled missing values with their respective values

In [16]:
```python
df3 = df2.replace({'':np.nan})
df3.fillna(method='ffill', inplace=True)
df3
```

Out[16]:

	code	name	id
458	1	Economic management	P130925
1114	1	Economic management	P131094
454	1	Economic management	P130925
453	1	Economic management	P128573
1257	1	Economic management	P133230
1260	1	Economic management	P133339
1212	1	Economic management	P121019
249	1	Economic management	P132948
1057	1	Economic management	P144633
841	1	Economic management	P122793
204	1	Economic management	P126034
2	1	Economic management	P144674
223	1	Economic management	P145018
803	1	Economic management	P127332
205	1	Economic management	P126034
212	1	Economic management	P128282
222	1	Economic management	P145865
220	1	Economic management	P144917
1056	1	Economic management	P144633
175	1	Economic management	P118027
1045	1	Economic management	P133791
497	1	Economic management	P133663
1235	1	Economic management	P131440
1230	1	Economic management	P129465
1229	1	Economic management	P129465
900	1	Economic management	P143819
648	1	Economic management	P129825
647	1	Economic management	P129825
1078	1	Economic management	P131234
1218	1	Economic management	P130824
...	...	...	...
447	9	Urban development	P126899
739	9	Urban development	P127079
356	9	Urban development	P127543
1342	9	Urban development	P130174
341	9	Urban development	P144357
1347	9	Urban development	P131394
570	9	Urban development	P119039
535	9	Urban development	P120370
1473	9	Urban development	P128763
1470	9	Urban development	P128763
1495	9	Urban development	P126321
1095	9	Urban development	P130819
1104	9	Urban development	P130972
414	9	Urban development	P119063
1103	9	Urban development	P130972
194	9	Urban development	P128768
460	9	Urban development	P117394
320	9	Urban development	P122950
940	9	Urban development	P127955
732	9	Urban development	P127036
471	9	Urban development	P126817
930	9	Urban development	P126489
513	9	Urban development	P127163
917	9	Urban development	P143770
650	9	Urban development	P123384
318	9	Urban development	P122950
669	9	Urban development	P127303
1303	9	Urban development	P130528