Assignment 3

July 15, 2017

---

*You are currently looking at **version 1.5** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the Jupyter Notebook FAQ course resource.*

```
In [6]: import numpy as np
        import pandas as pd
        # import os
        # os.chdir('/Users/wangqi/Documents/PyCharm/Data_Science_in_Python')
```

1 Assignment 3 - More Pandas

This assignment requires more individual learning then the last one did - you are encouraged to check out the pandas documentation to find functions or methods you might not have used yet, or ask questions on Stack Overflow and tag them as pandas and python related. And of course, the discussion forums are open for interaction with your peers and the course staff.

1.0.1 Question 1 (20%)

Load the energy data from the file `Energy Indicators.xls`, which is a list of indicators of energy supply and renewable electricity production from the United Nations for the year 2013, and should be put into a DataFrame with the variable name of **energy**.

Keep in mind that this is an Excel file, and not a comma separated values file. Also, make sure to exclude the footer and header information from the datafile. The first two columns are unneccessary, so you should get rid of them, and you should change the column labels so that the columns are:

`['Country', 'Energy Supply', 'Energy Supply per Capita', '% Renewable']`

Convert `Energy Supply` to gigajoules (there are 1,000,000 gigajoules in a petajoule). For all countries which have missing data (e.g. data with "...") make sure this is reflected as `np.NaN` values.

Rename the following list of countries (for use in later questions):

`"Republic of Korea": "South Korea", "United States of America": "United States", "United Kingdom of Great Britain and Northern`

Ireland": "United Kingdom", "China, Hong Kong Special Administrative Region": "Hong Kong"

There are also several countries with numbers and/or parenthesis in their name. Be sure to remove these,

e.g.

'Bolivia (Plurinational State of)' should be 'Bolivia',

'Switzerland17' should be 'Switzerland'.

Next, load the GDP data from the file world_bank.csv, which is a csv containing countries' GDP from 1960 to 2015 from World Bank. Call this DataFrame **GDP**.

Make sure to skip the header, and rename the following list of countries:

"Korea, Rep.": "South Korea", "Iran, Islamic Rep.": "Iran", "Hong Kong SAR, China": "Hong Kong"

Finally, load the Sciamgo Journal and Country Rank data for Energy Engineering and Power Technology from the file scimagojr-3.xlsx, which ranks countries based on their journal contributions in the aforementioned area. Call this DataFrame **ScimEn**.

Join the three datasets: GDP, Energy, and ScimEn into a new dataset (using the intersection of country names). Use only the last 10 years (2006-2015) of GDP data and only the top 15 countries by Scimagojr 'Rank' (Rank 1 through 15).

The index of this DataFrame should be the name of the country, and the columns should be ['Rank', 'Documents', 'Citable documents', 'Citations', 'Self-citations', 'Citations per document', 'H index', 'Energy Supply', 'Energy Supply per Capita', '% Renewable', '2006', '2007', '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015'].

This function should return a DataFrame with 20 columns and 15 entries.

```
In [3]: def answer_one():
            #loading the .xls file of interest
            energy = pd.read_excel('Energy Indicators.xls','Energy', na_values = ['
            #slicing data, we just keep one part of the whole worksheet for that th
            #contains irrelevant information
            energy = energy.iloc[16:243, 2:]
            #rename the columns
            energy.columns = ['Country', 'Energy Supply', 'Energy Supply per Capita
            energy['Energy Supply'] = energy['Energy Supply'] * 1000000
            #wo need to have a look at the names before we start transforming them
            #'United States of America' in energy['Country'].unique()
            #maybe there are some other issues, i.e., maybe the country names are n
            #remove numbers from strings represents countries.
            energy['Country'] = energy['Country'].str.replace('\d+', '')
            #'United States of America' in energy['Country'].unique()
            #ok, now numbers have been removed
            #but there are still parentheses.
            energy['Country'] = energy['Country'].str.replace(r'\(.*\)+', '')
            the_dict = {"Republic of Korea": "South Korea",
                        "United States of America": "United States",
                        "United Kingdom of Great Britain and Northern Ireland": "Un
                        "China, Hong Kong Special Administrative Region": "Hong Kon
            #rename countries using dictionary
            energy['Country'] = energy['Country'].replace(the_dict, regex = True)
```

```
    energy['Country'] = energy['Country'].str.strip()
    energy[['Energy Supply', 'Energy Supply per Capita', '% Renewable']]\
    = energy[['Energy Supply', 'Energy Supply per Capita', '% Renewable']].


    #read gdp data
    GDP = pd.read_csv('world_bank.csv')
    #GDP.head()

    #change column names
    GDP.columns = GDP.iloc[3,]

    #slice the part that is of interest
    GDP = GDP.iloc[4:, ].reset_index(drop = True)

    GDP.columns = list(GDP.columns[:4]) + list(map(int, GDP.columns[4:]))
    another_dict = {"Korea, Rep.": "South Korea",
                    "Iran, Islamic Rep.": "Iran",
                    "Hong Kong SAR, China": "Hong Kong"}
    GDP['Country Name'] = GDP['Country Name'].replace(another_dict, regex =


    #read ScimEn data
    ScimEn = pd.read_excel('scimagojr-3.xlsx', 'Sheet1')

    #--------------------preparing for merging dataframeS------------------
    ScimEn_15 = ScimEn.iloc[:15, :]
    GDP_10 = GDP.loc[:,['Country Name']+ list(range(2006, 2016))]
    GDP_10.columns = ['Country'] + list(map(str, list(range(2006, 2016))))
    #ScimEn = ScimEn.set_index('Country', drop = True)
    #energy = energy.set_index('Country', drop = True)
    #GDP_10 = GDP_10.set_index('Country', drop = True)

    return pd.merge(pd.merge(ScimEn_15, energy, on = 'Country'), GDP_10, or
                                        .set_index('Country', drop = 'True')
```

```
In [60]: df = answer_one(); df.head()
```

Out[60]:

Country	Rank	Documents	Citable documents	Citations
China	1	127050	126767	597237
United States	2	96661	94747	792274
Japan	3	30504	30287	223024
United Kingdom	4	20944	20357	206091
Russian Federation	5	18534	18301	34266

Country	Self-citations	Citations per document	H index

```
China                          411683                    4.70      138
United States                  265436                    8.20      230
Japan                           61554                    7.31      134
United Kingdom                  37874                    9.84      139
Russian Federation              12422                    1.85       57

                    Energy Supply  Energy Supply per Capita  % Renewable
Country
China                1.271910e+11                     93.0     19.75491
United States        9.083800e+10                    286.0     11.57098
Japan                1.898400e+10                    149.0     10.23282
United Kingdom       7.920000e+09                    124.0     10.60047
Russian Federation   3.070900e+10                    214.0     17.28868

                              2006          2007          2008          2009
Country
China                3.992331e+12  4.559041e+12  4.997775e+12  5.459247e+12
United States        1.479230e+13  1.505540e+13  1.501149e+13  1.459484e+13
Japan                5.496542e+12  5.617036e+12  5.558527e+12  5.251308e+12
United Kingdom       2.419631e+12  2.482203e+12  2.470614e+12  2.367048e+12
Russian Federation   1.385793e+12  1.504071e+12  1.583004e+12  1.459199e+12

                              2010          2011          2012          2013
Country
China                6.039659e+12  6.612490e+12  7.124978e+12  7.672448e+12
United States        1.496437e+13  1.520402e+13  1.554216e+13  1.577367e+13
Japan                5.498718e+12  5.473738e+12  5.569102e+12  5.644659e+12
United Kingdom       2.403504e+12  2.450911e+12  2.479809e+12  2.533370e+12
Russian Federation   1.524917e+12  1.589943e+12  1.645876e+12  1.666934e+12

                              2014          2015
Country
China                8.230121e+12  8.797999e+12
United States        1.615662e+13  1.654857e+13
Japan                5.642884e+12  5.669563e+12
United Kingdom       2.605643e+12  2.666333e+12
Russian Federation   1.678709e+12  1.616149e+12
```

In [8]:
```
energy = pd.read_excel('Energy Indicators.xls','Energy', na_values = ["..."
energy = energy.iloc[16:243, 2:]
energy.columns = ['Country', 'Energy Supply', 'Energy Supply per Capita', '
energy['Energy Supply'] = energy['Energy Supply'] * 1000000
#wo need to have a look at the names before we start transforming them
#'United States of America' in energy['Country'].unique()
#maybe there are some other issues, i.e., maybe the country names are not s
#remove numbers from strings represents countries.
energy['Country'] = energy['Country'].str.replace('\d+', '')
#'United States of America' in energy['Country'].unique()
```

```
        #ok, now numbers have been removed
        #but there are still parentheses.
        energy['Country'] = energy['Country'].str.replace(r'\(.*\)+', '')
        the_dict = {"Republic of Korea": "South Korea",
                    "United States of America": "United States",
                    "United Kingdom of Great Britain and Northern Ireland": "United
                    "China, Hong Kong Special Administrative Region": "Hong Kong"}
        #rename countries using dictionary
        energy['Country'] = energy['Country'].replace(the_dict, regex = True)
        energy['Country'] = energy['Country'].str.strip()


        #read gdp data
        GDP = pd.read_csv('world_bank.csv')
        #GDP.head()

        #change column names
        GDP.columns = GDP.iloc[3,]

        #slice the part that is of interest
        GDP = GDP.iloc[4:, ].reset_index(drop = True)

        GDP.columns = list(GDP.columns[:4]) + list(map(int, GDP.columns[4:]))
        another_dict = {"Korea, Rep.": "South Korea",
                        "Iran, Islamic Rep.": "Iran",
                        "Hong Kong SAR, China": "Hong Kong"}
        GDP['Country Name'] = GDP['Country Name'].replace(another_dict, regex = Tru


        #read ScimEn data
        ScimEn = pd.read_excel('scimagojr-3.xlsx', 'Sheet1')
```

```
In [9]: 'China' in energy['Country']
```

```
Out[9]: False
```

```
In [10]: 'China' in energy['Country'].unique()
```

```
Out[10]: True
```

```
In [11]: 'Iran' in energy['Country'].unique()
```

```
Out[11]: True
```

```
In [12]: energy['Country'] = energy['Country'].str.strip()
```

1.0.2 Question 2 (6.6%)

The previous question joined three datasets then reduced this to just the top 15 entries. When you joined the datasets, but before you reduced this to the top 15 items, how many entries did you lose?

This function should return a single number.

```
In [9]: %%HTML
        <svg width="800" height="300">
          <circle cx="150" cy="180" r="80" fill-opacity="0.2" stroke="black" stroke
          <circle cx="200" cy="100" r="80" fill-opacity="0.2" stroke="black" stroke
          <circle cx="100" cy="100" r="80" fill-opacity="0.2" stroke="black" stroke
          <line x1="150" y1="125" x2="300" y2="150" stroke="black" stroke-width="2'
          <text  x="300" y="165" font-family="Verdana" font-size="35">Everything bu
        </svg>
```

```
<IPython.core.display.HTML object>
```

```
In [13]: def answer_two():
             return 156
```

```
In [14]: answer_two()
```

```
Out[14]: 156
```

```
In [290]: energy.shape
```

```
Out[290]: (227, 4)
```

```
In [291]: GDP.shape
```

```
Out[291]: (264, 60)
```

```
In [292]: ScimEn.shape
```

```
Out[292]: (191, 8)
```

Answer the following questions in the context of only the top 15 countries by Scimagojr Rank (aka the DataFrame returned by `answer_one()`)

1.0.3 Question 3 (6.6%)

What is the average GDP over the last 10 years for each country? (exclude missing values from this calculation.)

This function should return a Series named `avgGDP` with 15 countries and their average GDP sorted in descending order.

```
In [15]: def answer_three():
             Top15 = answer_one()
             return np.mean(Top15.loc[:,'2006':], axis = 1).sort_values(ascending =
```

```
In [16]: answer_three()
```

```
Out[16]: Country
         United States         1.536434e+13
         China                 6.348609e+12
         Japan                 5.542208e+12
         Germany               3.493025e+12
         France                2.681725e+12
         United Kingdom        2.487907e+12
         Brazil                2.189794e+12
         Italy                 2.120175e+12
         India                 1.769297e+12
         Canada                1.660647e+12
         Russian Federation    1.565459e+12
         Spain                 1.418078e+12
         Australia             1.164043e+12
         South Korea           1.106715e+12
         Iran                  4.441558e+11
         dtype: float64

In [17]: df = answer_one()

In [18]: np.mean(df, axis = 1).sort_values(ascending = False)

Out[18]: Country
         United States         7.686714e+12
         China                 3.180664e+12
         Japan                 2.772053e+12
         Germany               1.747176e+12
         France                1.341392e+12
         United Kingdom        1.244349e+12
         Brazil                1.095505e+12
         Italy                 1.060414e+12
         India                 8.863085e+11
         Canada                8.308453e+11
         Russian Federation    7.842652e+11
         Spain                 7.092853e+11
         Australia             5.822907e+11
         South Korea           5.539076e+11
         Iran                  2.108723e+11
         dtype: float64

In [19]: np.mean(df.loc['Iran'][10:])

Out[19]: 444155754051.09497
```

1.0.4 Question 4 (6.6%)

By how much had the GDP changed over the 10 year span for the country with the 6th largest average GDP?

This function should return a single number.

```
In [20]: def answer_four():
             Top15 = answer_one()
             avg_Top15 = answer_three()
             gdp_uk_2006 = Top15.loc[avg_Top15.index[5], '2006']
             gdp_uk_2015 = Top15.loc[avg_Top15.index[5], '2015']
             return gdp_uk_2015 - gdp_uk_2006

In [21]: answer_four()

Out[21]: 246702696075.3999

In [22]: Top15 = answer_one()
         avg_Top15 = answer_three()

In [23]: avg_Top15.index[5]

Out[23]: 'United Kingdom'

In [24]: Top15.loc[avg_Top15.index[5]]

Out[24]: Rank                        4.000000e+00
         Documents                   2.094400e+04
         Citable documents           2.035700e+04
         Citations                   2.060910e+05
         Self-citations              3.787400e+04
         Citations per document      9.840000e+00
         H index                     1.390000e+02
         Energy Supply               7.920000e+09
         Energy Supply per Capita    1.240000e+02
         % Renewable                 1.060047e+01
         2006                        2.419631e+12
         2007                        2.482203e+12
         2008                        2.470614e+12
         2009                        2.367048e+12
         2010                        2.403504e+12
         2011                        2.450911e+12
         2012                        2.479809e+12
         2013                        2.533370e+12
         2014                        2.605643e+12
         2015                        2.666333e+12
         Name: United Kingdom, dtype: float64

In [25]: gdp_uk_2006 = Top15.loc[avg_Top15.index[5], '2006']
         gdp_uk_2015 = Top15.loc[avg_Top15.index[5], '2015']

In [318]: #growth_gdp_uk = (gdp_uk_2015 - gdp_uk_2006) / gdp_uk_2015; growth_gdp_uk

Out[318]: 0.092525074471689742

In [26]: gdp_uk_2015 - gdp_uk_2006

Out[26]: 246702696075.3999
```

1.0.5 Question 5 (6.6%)

What is the mean `Energy Supply per Capita`?
This function should return a single number.

```
In [27]: def answer_five():
             Top15 = answer_one()
             avg_energy_per_capita = np.mean(Top15['Energy Supply per Capita'])
             return avg_energy_per_capita
```

```
In [28]: answer_five()
```

```
Out[28]: 157.59999999999999
```

```
In [29]: Top15
```

```
Out[29]:                     Rank  Documents  Citable documents  Citations  \
         Country
         China                  1     127050             126767     597237
         United States          2      96661              94747     792274
         Japan                  3      30504              30287     223024
         United Kingdom         4      20944              20357     206091
         Russian Federation     5      18534              18301      34266
         Canada                 6      17899              17620     215003
         Germany                7      17027              16831     140566
         India                  8      15005              14841     128763
         France                 9      13153              12973     130632
         South Korea           10      11983              11923     114675
         Italy                 11      10964              10794     111850
         Spain                 12       9428               9330     123336
         Iran                  13       8896               8819      57470
         Australia             14       8831               8725      90765
         Brazil                15       8668               8596      60702

                              Self-citations  Citations per document  H index  \
         Country
         China                        411683                    4.70      138
         United States                265436                    8.20      230
         Japan                         61554                    7.31      134
         United Kingdom                37874                    9.84      139
         Russian Federation            12422                    1.85       57
         Canada                        40930                   12.01      149
         Germany                       27426                    8.26      126
         India                         37209                    8.58      115
         France                        28601                    9.93      114
         South Korea                   22595                    9.57      104
         Italy                         26661                   10.20      106
         Spain                         23964                   13.08      115
         Iran                          19125                    6.46       72
```

Australia	15606	10.28	107
Brazil	14396	7.00	86

Country	Energy Supply	Energy Supply per Capita	% Renewable
China	1.271910e+11	93.0	19.754910
United States	9.083800e+10	286.0	11.570980
Japan	1.898400e+10	149.0	10.232820
United Kingdom	7.920000e+09	124.0	10.600470
Russian Federation	3.070900e+10	214.0	17.288680
Canada	1.043100e+10	296.0	61.945430
Germany	1.326100e+10	165.0	17.901530
India	3.319500e+10	26.0	14.969080
France	1.059700e+10	166.0	17.020280
South Korea	1.100700e+10	221.0	2.279353
Italy	6.530000e+09	109.0	33.667230
Spain	4.923000e+09	106.0	37.968590
Iran	9.172000e+09	119.0	5.707721
Australia	5.386000e+09	231.0	11.810810
Brazil	1.214900e+10	59.0	69.648030

Country	2006	2007	2008	2009
China	3.992331e+12	4.559041e+12	4.997775e+12	5.459247e+12
United States	1.479230e+13	1.505540e+13	1.501149e+13	1.459484e+13
Japan	5.496542e+12	5.617036e+12	5.558527e+12	5.251308e+12
United Kingdom	2.419631e+12	2.482203e+12	2.470614e+12	2.367048e+12
Russian Federation	1.385793e+12	1.504071e+12	1.583004e+12	1.459199e+12
Canada	1.564469e+12	1.596740e+12	1.612713e+12	1.565145e+12
Germany	3.332891e+12	3.441561e+12	3.478809e+12	3.283340e+12
India	1.265894e+12	1.374865e+12	1.428361e+12	1.549483e+12
France	2.607840e+12	2.669424e+12	2.674637e+12	2.595967e+12
South Korea	9.410199e+11	9.924316e+11	1.020510e+12	1.027730e+12
Italy	2.202170e+12	2.234627e+12	2.211154e+12	2.089938e+12
Spain	1.414823e+12	1.468146e+12	1.484530e+12	1.431475e+12
Iran	3.895523e+11	4.250646e+11	4.289909e+11	4.389208e+11
Australia	1.021939e+12	1.060340e+12	1.099644e+12	1.119654e+12
Brazil	1.845080e+12	1.957118e+12	2.056809e+12	2.054215e+12

Country	2010	2011	2012	2013
China	6.039659e+12	6.612490e+12	7.124978e+12	7.672448e+12
United States	1.496437e+13	1.520402e+13	1.554216e+13	1.577367e+13
Japan	5.498718e+12	5.473738e+12	5.569102e+12	5.644659e+12
United Kingdom	2.403504e+12	2.450911e+12	2.479809e+12	2.533370e+12
Russian Federation	1.524917e+12	1.589943e+12	1.645876e+12	1.666934e+12
Canada	1.613406e+12	1.664087e+12	1.693133e+12	1.730688e+12
Germany	3.417298e+12	3.542371e+12	3.556724e+12	3.567317e+12

```
India                 1.708459e+12  1.821872e+12  1.924235e+12  2.051982e+12
France                2.646995e+12  2.702032e+12  2.706968e+12  2.722567e+12
South Korea           1.094499e+12  1.134796e+12  1.160809e+12  1.194429e+12
Italy                 2.125185e+12  2.137439e+12  2.077184e+12  2.040871e+12
Spain                 1.431673e+12  1.417355e+12  1.380216e+12  1.357139e+12
Iran                  4.677902e+11  4.853309e+11  4.532569e+11  4.445926e+11
Australia             1.142251e+12  1.169431e+12  1.211913e+12  1.241484e+12
Brazil                2.208872e+12  2.295245e+12  2.339209e+12  2.409740e+12

                              2014          2015
Country
China                 8.230121e+12  8.797999e+12
United States         1.615662e+13  1.654857e+13
Japan                 5.642884e+12  5.669563e+12
United Kingdom        2.605643e+12  2.666333e+12
Russian Federation    1.678709e+12  1.616149e+12
Canada                1.773486e+12  1.792609e+12
Germany               3.624386e+12  3.685556e+12
India                 2.200617e+12  2.367206e+12
France                2.729632e+12  2.761185e+12
South Korea           1.234340e+12  1.266580e+12
Italy                 2.033868e+12  2.049316e+12
Spain                 1.375605e+12  1.419821e+12
Iran                  4.639027e+11           NaN
Australia             1.272520e+12  1.301251e+12
Brazil                2.412231e+12  2.319423e+12
```

1.0.6 Question 6 (6.6%)

What country has the maximum % Renewable and what is the percentage?

This function should return a tuple with the name of the country and the percentage.

```
In [30]: def answer_six():
             Top15 = answer_one()
             Top15 = Top15.sort_values(by = '% Renewable', ascending = False).reset
             return Top15['Country'][0], Top15['% Renewable'][0]
```

```
In [31]: answer_six()
```

```
Out[31]: ('Brazil', 69.648030000000006)
```

1.0.7 Question 7 (6.6%)

Create a new column that is the ratio of Self-Citations to Total Citations. What is the maximum value for this new column, and what country has the highest ratio?

This function should return a tuple with the name of the country and the ratio.

```
In [32]: Top15.columns
```

```
Out[32]: Index(['Rank', 'Documents', 'Citable documents', 'Citations', 'Self-citati
                'Citations per document', 'H index', 'Energy Supply',
                'Energy Supply per Capita', '% Renewable', '2006', '2007', '2008',
                '2009', '2010', '2011', '2012', '2013', '2014', '2015'],
               dtype='object')

In [33]: def answer_seven():
             Top15 = answer_one()
             Top15['self_citations_ratio'] = Top15['Self-citations'] / Top15['Citat
             Top15 = Top15.sort_values(by = 'self_citations_ratio', ascending = Fal
             return Top15['Country'][0], Top15['self_citations_ratio'][0]

In [34]: answer_seven()

Out[34]: ('China', 0.68931261793894216)
```

1.0.8 Question 8 (6.6%)

Create a column that estimates the population using Energy Supply and Energy Supply per capita.
What is the third most populous country according to this estimate?

This function should return a single string value.

```
In [35]: def answer_eight():
             Top15 = answer_one()
             if (Top15['Energy Supply'].isnull().values.any() == False) & \
                             (Top15['Energy Supply per Capita'].isnull().values.any
                 Top15['Estimated_pop_by_energy'] = Top15['Energy Supply'] / \
                                                    Top15['Energy Supply per Capita
                 Top15 = Top15.sort_values('Estimated_pop_by_energy', ascending = F
                 Top15_1 = Top15.iloc[:3]
             return Top15_1.index[2]

In [36]: answer_eight()

Out[36]: 'United States'

In [37]: Top15 = answer_one()
         Top15['Estimated_pop_by_energy'] = Top15['Energy Supply'] / Top15['Energy
         Top15 = Top15.sort_values('Estimated_pop_by_energy', ascending = False)
         Top15_1 = Top15.iloc[:3]

In [38]: Top15_1.index[2]

Out[38]: 'United States'

In [39]: Top15['Energy Supply'].isnull().values.any()

Out[39]: False
```

1.0.9 Question 9 (6.6%)

Create a column that estimates the number of citable documents per person. What is the correlation between the number of citable documents per capita and the energy supply per capita? Use the `.corr()` method, (Pearson's correlation).

This function should return a single number.

(Optional: Use the built-in function `plot9()` to visualize the relationship between Energy Supply per Capita vs. Citable docs per Capita)

```
In [40]: Top15.head(2)
```

```
Out[40]:          Rank  Documents  Citable documents  Citations  Self-citations  \
         Country
         China       1     127050             126767     597237          411683
         India       8      15005              14841     128763           37209

                   Citations per document  H index  Energy Supply  \
         Country
         China                       4.70      138   1.271910e+11
         India                       8.58      115   3.319500e+10

                   Energy Supply per Capita  % Renewable           ...             \
         Country                                                  ...
         China                         93.0     19.75491           ...
         India                         26.0     14.96908           ...

                            2007          2008          2009          2010          2
         Country
         China    4.559041e+12  4.997775e+12  5.459247e+12  6.039659e+12  6.612490e
         India    1.374865e+12  1.428361e+12  1.549483e+12  1.708459e+12  1.821872e

                            2012          2013          2014          2015  \
         Country
         China    7.124978e+12  7.672448e+12  8.230121e+12  8.797999e+12
         India    1.924235e+12  2.051982e+12  2.200617e+12  2.367206e+12

                   Estimated_pop_by_energy
         Country
         China               1.367645e+09
         India               1.276731e+09

         [2 rows x 21 columns]
```

```
In [41]: def answer_nine():
             Top15 = answer_one()
             # if (Top15['Energy Supply'].isnull().values.any() == False) & (Top15
             Top15['Estimated_pop_by_energy'] = Top15['Energy Supply'] / \
                                                Top15['Energy Supply per Capita']
             Top15['citable_documents_per_capita'] = Top15['Citable documents'] / \
```

```
                                                Top15['Estimated_pop_by_energy']
            the_corr_coef = Top15['citable_documents_per_capita']\
                                              .corr(Top15['Energy Supply per Capita'])

            return the_corr_coef
```

```
In [42]: answer_nine()
```

```
Out[42]: 0.79400104354429435
```

```
In [43]: def plot9():
             import matplotlib as plt
             %matplotlib inline

             Top15 = answer_one()
             Top15['PopEst'] = Top15['Energy Supply'] / Top15['Energy Supply per Ca
             Top15['Citable docs per Capita'] = Top15['Citable documents'] / Top15[
             Top15.plot(x='Citable docs per Capita', y='Energy Supply per Capita',
                        kind='scatter', xlim=[0, 0.0006])
```

```
In [19]: #plot9() # Be sure to comment out plot9() before submitting the assignment
```

1.0.10 Question 10 (6.6%)

Create a new column with a 1 if the country's % Renewable value is at or above the median for all countries in the top 15, and a 0 if the country's % Renewable value is below the median.

This function should return a series named `HighRenew` whose index is the country name sorted in ascending order of rank.

```
In [44]: def answer_ten():
             Top15 = answer_one()
             Top15['HighRenew'] = Top15['% Renewable'] >= Top15['% Renewable'].medi
             Top15['HighRenew_num'] = Top15['HighRenew'] * 1
             return Top15['HighRenew_num']
```

```
In [534]: # Top15_1 = Top15[Top15['HighRenew_num'] == 1]
          # Top15_1
```

```
In [45]: Top15['HighRenew'] = Top15['% Renewable'] >= Top15['% Renewable'].median()
         Top15['HighRenew_num'] = Top15['HighRenew'] * 1
         Top15.head(2)
```

```
Out[45]:          Rank  Documents  Citable documents  Citations  Self-citations  \
         Country
         China       1     127050             126767     597237          411683
         India       8      15005              14841     128763           37209

                  Citations per document  H index  Energy Supply  \
         Country
```

```
China                        4.70      138   1.271910e+11
India                        8.58      115   3.319500e+10

         Energy Supply per Capita  % Renewable      ...                 200
Country                                           ...
China                        93.0     19.75491      ...        5.459247e+1
India                        26.0     14.96908      ...        1.549483e+1

                  2010          2011          2012          2013          2
Country
China    6.039659e+12  6.612490e+12  7.124978e+12  7.672448e+12  8.230121e
India    1.708459e+12  1.821872e+12  1.924235e+12  2.051982e+12  2.200617e

                  2015  Estimated_pop_by_energy  HighRenew  HighRenew_num
Country
China    8.797999e+12             1.367645e+09       True              1
India    2.367206e+12             1.276731e+09      False              0

[2 rows x 23 columns]
```

```
In [533]: # Top15['HighRenew'] = [1 if Top15['% Renewable'] >= Top15['% Renewable']
```

```
In [46]: Top15['% Renewable'] >= Top15['% Renewable'].median()
```

```
Out[46]: Country
         China                 True
         India                False
         United States        False
         Brazil                True
         Russian Federation    True
         Japan                False
         Germany               True
         Iran                 False
         United Kingdom       False
         France                True
         Italy                 True
         South Korea          False
         Spain                 True
         Canada                True
         Australia            False
         Name: % Renewable, dtype: bool
```

1.0.11 Question 11 (6.6%)

Use the following dictionary to group the Countries by Continent, then create a dateframe that displays the sample size (the number of countries in each continent bin), and the sum, mean, and std deviation for the estimated population of each country.

```
ContinentDict  = {'China':'Asia',
                  'United States':'North America',
```

```
                    'Japan':'Asia',
                    'United Kingdom':'Europe',
                    'Russian Federation':'Europe',
                    'Canada':'North America',
                    'Germany':'Europe',
                    'India':'Asia',
                    'France':'Europe',
                    'South Korea':'Asia',
                    'Italy':'Europe',
                    'Spain':'Europe',
                    'Iran':'Asia',
                    'Australia':'Australia',
                    'Brazil':'South America'}
```

This function should return a DataFrame with index named Continent ['Asia', 'Australia', 'Europe', 'North America', 'South America'] and columns ['size', 'sum', 'mean', 'std']

```
In [47]: def answer_eleven():
             Top15 = answer_one()
             Top15['Estimated_pop_by_energy'] = Top15['Energy Supply'] / \
                                                Top15['Energy Supply per Capita']
             ContinentDict  = {'China':'Asia',
                               'United States':'North America',
                               'Japan':'Asia',
                               'United Kingdom':'Europe',
                               'Russian Federation':'Europe',
                               'Canada':'North America',
                               'Germany':'Europe',
                               'India':'Asia',
                               'France':'Europe',
                               'South Korea':'Asia',
                               'Italy':'Europe',
                               'Spain':'Europe',
                               'Iran':'Asia',
                               'Australia':'Australia',
                               'Brazil':'South America'}
             Top15 = Top15.reset_index()
             Top15['Country'] = Top15['Country'].replace(ContinentDict, regex = Tru
             return Top15[['Country', 'Estimated_pop_by_energy']].groupby(['Country
                                 .agg(['size', 'sum', 'mean', 'std'])['Estimated_pop

In [48]: answer_eleven()

Out[48]:                size           sum          mean           std
         Country
         Asia              5  2.898666e+09  5.797333e+08  6.790979e+08
         Australia         1  2.331602e+07  2.331602e+07           NaN
         Europe            6  4.579297e+08  7.632161e+07  3.464767e+07
```

```
North America     2  3.528552e+08  1.764276e+08  1.996696e+08
South America     1  2.059153e+08  2.059153e+08           NaN
```

```
In [50]: ContinentDict  = {'China':'Asia',
                           'United States':'North America',
                           'Japan':'Asia',
                           'United Kingdom':'Europe',
                           'Russian Federation':'Europe',
                           'Canada':'North America',
                           'Germany':'Europe',
                           'India':'Asia',
                           'France':'Europe',
                           'South Korea':'Asia',
                           'Italy':'Europe',
                           'Spain':'Europe',
                           'Iran':'Asia',
                           'Australia':'Australia',
                           'Brazil':'South America'}
```

```
In [51]: dddd = answer_one()
         dddd['Estimated_pop_by_energy'] = dddd['Energy Supply'] / dddd['Energy Sup
         dddd = dddd.reset_index()
         dddd['Country'] = dddd['Country'].replace(ContinentDict, regex = True)
         #dddd
```

Out[51]:

	Country	Rank	Documents	Citable documents	Citations
0	Asia	1	127050	126767	597237
1	North America	2	96661	94747	792274
2	Asia	3	30504	30287	223024
3	Europe	4	20944	20357	206091
4	Europe	5	18534	18301	34266
5	North America	6	17899	17620	215003
6	Europe	7	17027	16831	140566
7	Asia	8	15005	14841	128763
8	Europe	9	13153	12973	130632
9	Asia	10	11983	11923	114675
10	Europe	11	10964	10794	111850
11	Europe	12	9428	9330	123336
12	Asia	13	8896	8819	57470
13	Australia	14	8831	8725	90765
14	South America	15	8668	8596	60702

	Self-citations	Citations per document	H index	Energy Supply
0	411683	4.70	138	1.271910e+11
1	265436	8.20	230	9.083800e+10
2	61554	7.31	134	1.898400e+10
3	37874	9.84	139	7.920000e+09
4	12422	1.85	57	3.070900e+10

5	40930	12.01	149	1.043100e+10
6	27426	8.26	126	1.326100e+10
7	37209	8.58	115	3.319500e+10
8	28601	9.93	114	1.059700e+10
9	22595	9.57	104	1.100700e+10
10	26661	10.20	106	6.530000e+09
11	23964	13.08	115	4.923000e+09
12	19125	6.46	72	9.172000e+09
13	15606	10.28	107	5.386000e+09
14	14396	7.00	86	1.214900e+10

	Energy Supply per Capita	...	2007	\
0	93.0	...	4.559041e+12	
1	286.0	...	1.505540e+13	
2	149.0	...	5.617036e+12	
3	124.0	...	2.482203e+12	
4	214.0	...	1.504071e+12	
5	296.0	...	1.596740e+12	
6	165.0	...	3.441561e+12	
7	26.0	...	1.374865e+12	
8	166.0	...	2.669424e+12	
9	221.0	...	9.924316e+11	
10	109.0	...	2.234627e+12	
11	106.0	...	1.468146e+12	
12	119.0	...	4.250646e+11	
13	231.0	...	1.060340e+12	
14	59.0	...	1.957118e+12	

	2008	2009	2010	2011	2012
0	4.997775e+12	5.459247e+12	6.039659e+12	6.612490e+12	7.124978e+12
1	1.501149e+13	1.459484e+13	1.496437e+13	1.520402e+13	1.554216e+13
2	5.558527e+12	5.251308e+12	5.498718e+12	5.473738e+12	5.569102e+12
3	2.470614e+12	2.367048e+12	2.403504e+12	2.450911e+12	2.479809e+12
4	1.583004e+12	1.459199e+12	1.524917e+12	1.589943e+12	1.645876e+12
5	1.612713e+12	1.565145e+12	1.613406e+12	1.664087e+12	1.693133e+12
6	3.478809e+12	3.283340e+12	3.417298e+12	3.542371e+12	3.556724e+12
7	1.428361e+12	1.549483e+12	1.708459e+12	1.821872e+12	1.924235e+12
8	2.674637e+12	2.595967e+12	2.646995e+12	2.702032e+12	2.706968e+12
9	1.020510e+12	1.027730e+12	1.094499e+12	1.134796e+12	1.160809e+12
10	2.211154e+12	2.089938e+12	2.125185e+12	2.137439e+12	2.077184e+12
11	1.484530e+12	1.431475e+12	1.431673e+12	1.417355e+12	1.380216e+12
12	4.289909e+11	4.389208e+11	4.677902e+11	4.853309e+11	4.532569e+11
13	1.099644e+12	1.119654e+12	1.142251e+12	1.169431e+12	1.211913e+12
14	2.056809e+12	2.054215e+12	2.208872e+12	2.295245e+12	2.339209e+12

	2013	2014	2015	Estimated_pop_by_energy
0	7.672448e+12	8.230121e+12	8.797999e+12	1.367645e+09
1	1.577367e+13	1.615662e+13	1.654857e+13	3.176154e+08

```
2   5.644659e+12  5.642884e+12  5.669563e+12             1.274094e+08
3   2.533370e+12  2.605643e+12  2.666333e+12             6.387097e+07
4   1.666934e+12  1.678709e+12  1.616149e+12             1.435000e+08
5   1.730688e+12  1.773486e+12  1.792609e+12             3.523986e+07
6   3.567317e+12  3.624386e+12  3.685556e+12             8.036970e+07
7   2.051982e+12  2.200617e+12  2.367206e+12             1.276731e+09
8   2.722567e+12  2.729632e+12  2.761185e+12             6.383735e+07
9   1.194429e+12  1.234340e+12  1.266580e+12             4.980543e+07
10  2.040871e+12  2.033868e+12  2.049316e+12             5.990826e+07
11  1.357139e+12  1.375605e+12  1.419821e+12             4.644340e+07
12  4.445926e+11  4.639027e+11           NaN             7.707563e+07
13  1.241484e+12  1.272520e+12  1.301251e+12             2.331602e+07
14  2.409740e+12  2.412231e+12  2.319423e+12             2.059153e+08

[15 rows x 22 columns]
```

```
In [52]: dddd[['Country', 'Estimated_pop_by_energy']].groupby(['Country'])\
                .agg(['sum', 'mean', 'std', 'size'])['Estimated_pop_by_energy']
```

Out[52]:

Country	sum	mean	std	size
Asia	2.898666e+09	5.797333e+08	6.790979e+08	5
Australia	2.331602e+07	2.331602e+07	NaN	1
Europe	4.579297e+08	7.632161e+07	3.464767e+07	6
North America	3.528552e+08	1.764276e+08	1.996696e+08	2
South America	2.059153e+08	2.059153e+08	NaN	1

1.0.12 Question 12 (6.6%)

Cut % Renewable into 5 bins. Group Top15 by the Continent, as well as these new % Renewable bins. How many countries are in each of these groups?

This function should return a ***Series*** *with a MultiIndex of* `Continent`*, then the bins for* `% Renewable`*. Do not include groups with no countries.*

```
In [53]: def answer_twelve():
             Top15 = answer_one()
             ContinentDict  = {'China':'Asia',
                               'United States':'North America',
                               'Japan':'Asia',
                               'United Kingdom':'Europe',
                               'Russian Federation':'Europe',
                               'Canada':'North America',
                               'Germany':'Europe',
                               'India':'Asia',
                               'France':'Europe',
                               'South Korea':'Asia',
                               'Italy':'Europe',
                               'Spain':'Europe',
                               'Iran':'Asia',
```

```
                        'Australia':'Australia',
                        'Brazil':'South America'}
            Top15 = Top15.reset_index()
            Top15['Continent'] = Top15['Country'].replace(ContinentDict, regex = T
            Top15['binning_renew'] = pd.cut(Top15['% Renewable'], 5)
            return Top15[['Continent', 'Country', 'binning_renew']]\
                                        .groupby(['Continent', 'binning_renew']).s
```

In [54]:
```
type(answer_twelve())
```

Out[54]:
```
pandas.core.series.Series
```

In [55]:
```
Top15 = answer_one()
ContinentDict  = {'China':'Asia',
                  'United States':'North America',
                  'Japan':'Asia',
                  'United Kingdom':'Europe',
                  'Russian Federation':'Europe',
                  'Canada':'North America',
                  'Germany':'Europe',
                  'India':'Asia',
                  'France':'Europe',
                  'South Korea':'Asia',
                  'Italy':'Europe',
                  'Spain':'Europe',
                  'Iran':'Asia',
                  'Australia':'Australia',
                  'Brazil':'South America'}
Top15 = Top15.reset_index()
Top15['Continent'] = Top15['Country'].replace(ContinentDict, regex = True)
Top15['binning_renew'] = pd.cut(Top15['% Renewable'], 5)
```

In [56]:
```
Top15.head()
```

Out[56]:

	Country	Rank	Documents	Citable documents	Citations	\
0	China	1	127050	126767	597237	
1	United States	2	96661	94747	792274	
2	Japan	3	30504	30287	223024	
3	United Kingdom	4	20944	20357	206091	
4	Russian Federation	5	18534	18301	34266	

	Self-citations	Citations per document	H index	Energy Supply	\
0	411683	4.70	138	1.271910e+11	
1	265436	8.20	230	9.083800e+10	
2	61554	7.31	134	1.898400e+10	
3	37874	9.84	139	7.920000e+09	
4	12422	1.85	57	3.070900e+10	

	Energy Supply per Capita	...	2008	2009

```
0                      93.0        ...         4.997775e+12  5.459247e+12
1                     286.0        ...         1.501149e+13  1.459484e+13
2                     149.0        ...         5.558527e+12  5.251308e+12
3                     124.0        ...         2.470614e+12  2.367048e+12
4                     214.0        ...         1.583004e+12  1.459199e+12

           2010          2011          2012          2013          2014  \
0  6.039659e+12  6.612490e+12  7.124978e+12  7.672448e+12  8.230121e+12
1  1.496437e+13  1.520402e+13  1.554216e+13  1.577367e+13  1.615662e+13
2  5.498718e+12  5.473738e+12  5.569102e+12  5.644659e+12  5.642884e+12
3  2.403504e+12  2.450911e+12  2.479809e+12  2.533370e+12  2.605643e+12
4  1.524917e+12  1.589943e+12  1.645876e+12  1.666934e+12  1.678709e+12

           2015      Continent     binning_renew
0  8.797999e+12           Asia  (15.753, 29.227]
1  1.654857e+13  North America   (2.212, 15.753]
2  5.669563e+12           Asia   (2.212, 15.753]
3  2.666333e+12         Europe   (2.212, 15.753]
4  1.616149e+12         Europe  (15.753, 29.227]

[5 rows x 23 columns]
```

```
In [57]: Top15[['Continent', 'Country', 'binning_renew']]\
         .groupby(['Continent', 'binning_renew']).size()
```

```
Out[57]: Continent      binning_renew
         Asia           (2.212, 15.753]     4
                        (15.753, 29.227]    1
         Australia      (2.212, 15.753]     1
         Europe         (2.212, 15.753]     1
                        (15.753, 29.227]    3
                        (29.227, 42.701]    2
         North America  (2.212, 15.753]     1
                        (56.174, 69.648]    1
         South America  (56.174, 69.648]    1
         dtype: int64
```

1.0.13 Question 13 (6.6%)

Convert the Population Estimate series to a string with thousands separator (using commas). Do not round the results.

e.g. 317615384.61538464 -> 317,615,384.61538464

This function should return a Series `PopEst` whose index is the country name and whose values are the population estimate string.

```
In [58]: def answer_thirteen():
             Top15 = answer_one()
             Top15['Estimated_pop_by_energy'] = Top15['Energy Supply'] / \
                                                Top15['Energy Supply per Capita']
```

```
        Top15['Estimated_pop'] = Top15['Estimated_pop_by_energy']\
                                              .apply(lambda x: "{:,}".format(x))
        return Top15['Estimated_pop']
```

```
In [59]: answer_thirteen()
```

```
Out[59]: Country
         China                  1,367,645,161.2903225
         United States            317,615,384.61538464
         Japan                    127,409,395.97315437
         United Kingdom           63,870,967.741935484
         Russian Federation              143,500,000.0
         Canada                    35,239,864.86486486
         Germany                   80,369,696.96969697
         India                  1,276,730,769.2307692
         France                    63,837,349.39759036
         South Korea              49,805,429.864253394
         Italy                    59,908,256.880733944
         Spain                     46,443,396.2264151
         Iran                     77,075,630.25210084
         Australia                23,316,017.316017315
         Brazil                   205,915,254.23728815
         Name: Estimated_pop, dtype: object
```

1.0.14 Optional

Use the built in function `plot_optional()` to see an example visualization.

```
In [24]: def plot_optional():
             import matplotlib as plt
             %matplotlib inline
             Top15 = answer_one()
             ax = Top15.plot(x='Rank', y='% Renewable', kind='scatter',
                             c=['#e41a1c','#377eb8','#e41a1c','#4daf4a','#4daf4a','
                                '#4daf4a','#e41a1c','#4daf4a','#4daf4a','#e41a1c','
                             xticks=range(1,16), s=6*Top15['2014']/10**10, alpha=.

             for i, txt in enumerate(Top15.index):
                 ax.annotate(txt, [Top15['Rank'][i], Top15['% Renewable'][i]], ha='

             print("This is an example of a visualization that can be created to he
         This is a bubble chart showing % Renewable vs. Rank. The size of the bubbl
         2014 GDP, and the color corresponds to the continent.")
```

```
In [25]: #plot_optional() # Be sure to comment out plot_optional() before submittin
```