Stage 1: The World Wide Web and World Wide Geography

In this stage, you will write code to download the data files, load the data to Pandas DataFrames, and answer various questions about the data.

Question 1: what is the total area across all the countries in the dataset?

Write code to pull the data from here (do not manually download): https://tyler.caraza-harter.com/cs301/spring19/data/countries.json

Hint 1: `pd.read_json(URL)` will return a DataFrame by downloading the JSON file from online at URL. If the downloaded JSON contains a list of dictionaries, each dictionary will be a row in the DataFrame.

Hint 2: review how to extract a single column as a Series from a DataFrame. You can add all the values in a Series with the `.sum()` method.

Keep the DataFrame in a variable named `countries`.

Question 2: what is the first URL in the `manifest.txt` page?

Write some code to download the list from here (do not manually download): https://tyler.caraza-harter.com/cs301/spring19/data/capitals/manifest.txt

---

Complete the following function:

```
def get_json(url):
    pass
```

Requirements:

- the first time `get_json` is called for a given URL, it should download the JSON content at that URL using `requests.get`, decode it, and return the result as a Python structure (list or dict, depending on the contents)
- the first time `get_json` is called for a given URL, it should also save the contents to a file on your computer (for example, if URL is `https://tyler.caraza-harter.com/cs301/spring19/data/capitals/Brazil_Peru.json`, it might save a local file named `Brazil_Peru.json` in the current directory)
- subsequent times `get_json` is called, it should return the data from the previously-downloaded file
- `get_json` should determine whether or not it is being called the first time for a given URL by using `os.path.exists` to check whether the data has previously been downloaded

Test your function (e.g., make sure you can call `get_json("https://tyler.caraza-harter.com/cs301/spring19/data/capitals/Brazil_Peru.json")` more than once).

For every file in `manifest.txt`, use `get_json` to fetch and decode the contents to a Python dictionary. Collect all these dictionaries in a list named `capital_rows`.

Question 3: what is `capital_rows`?

Create a DataFrame with `capitals = DataFrame(capital_rows)`. Use `capitals` and `countries` to answer the following questions.

Question 4: what is the capital of Bermuda?

Now we can begin to answer more complex questions using our newly constructed DataFrame.

Our first task will be determining the capital of Bermuda.

Hint: you can use fancy indexing to extract the row where the `Country` equals "China". Then, extract the `Capital` Series, from which you can grab the only value with the Series.item() function.

Question 5: Which country's capital is Maputo?

Question 6: which 5 countries have the southern-most capitals?

Produce a Python list of the 5, with southernmost first.

Hint: look at the documentation examples of how to sort a DataFrame with the sort_values function.

Question 7: which 3 countries have the northern-most capitals?

Question 8: for "birth-rate" and "death-rate", what are various summary statistics (e.g., mean, max, standard deviation, etc)?

Format: use the describe function on a DataFrame that contains `birth-rate` and `death-rate` columns. You may include summary statistics for other columns in your output, as long as your summary table has stats for birth and death.

Question 9: for columns `literacy` and `phones`, what are various summary statistics?

Format: a table generated by the `describe` function.

In some countries, it is standard to use commas instead of periods to indicate decimals. The `literacy` and `phone` data is formatted this way (i.e., decimal numbers represented as strings, with commas for decimals). You'll need to reformat the data to use periods (instead of commas), then convert the column of strings to a column of floats.

Hint: learn how to use the
astype
and
replace
Pandas functions.

Question 10: what is the largest land-locked country in Europe?

A "land-locked" country is one that has zero coastline. Largest is in terms of **area**.

Question 11: what is the largest land-locked country in Africa?

Question 12: what is the largest land-locked country in South America?

Question 13: what is the distance between Camp Randall Stadium and the Wisconsin State Capital?

This isn't related to countries, but it's a good warmup for the next problems. Your answer should be about 1.433899492072933 miles.

Assumptions:

- the latitude/longitude of Randall Stadium is 43.070231,-89.411893
- the latitude/longitude of the Wisconsin Capital is 43.074645,-89.384113
- use the Haversine formula: http://www.movable-type.co.uk/scripts/gis-faq-5.1.html
- the radius of the earth is 3956 miles
- answer in miles

If you find code online that computes the Haversine distance for you, great! You are allowed to use it as long as (1) it works and (2) you cite the source with a comment. Note that we won't help you troubleshoot Haversine functions you didn't write yourself during office hours, so if you want help, you should start from scratch on this one.

If you decide to implement it yourself (it's fun!), here are some tips:

- review the formula: http://www.movable-type.co.uk/scripts/gis-faq-5.1.html
- remember that latitude and longitude are in degrees, but sin, cos, and other Python math functions usually expect radians. Consider math.radians
- This means that before you do anything with the long and latitudes make sure to convert them to radians as your FIRST STEP
- people often use x^N to mean x raised to the Nth power. Make sure you write it as x**N in Python.

Question 14: what is the distance between India and Brazil?

For the coordinates of a country, use its capital.

Question 15: What are the distances between Chile, Guyana, and Colombia?

Your result should be DataFrame with 3 rows (for each country) and 3 columns (again for each country). The value in each cell should be the distance between the country of the row and the country of the column. For a general idea of what this should look like, open the `expected.html` file you downloaded. When displaying the distance between a country and itself, the table should should NaN (instead of 0).

Question 16: what is the distance between every pair of South American countries?

Your result should be a table with 12 rows (for each country) and 12 columns (again for each country). The value in each cell should be the distance between the country of the row and the country of the column. For a general idea of what this should look like, open the expected.html file you downloaded. When displaying the distance between a country and itself, the table should should NaN (instead of 0).

Question 17: what is the most central South American country?

This is the country that has the shortest average distance to other South American countries.

Hint 1: check out the following Pandas functions:

- DataFrame.mean
- Series.sort_values (note this is not the same as the DataFrame.sort_values function you've used before)

Hint 2: a Pandas Series contains indexed values. If you have a Series `s` and you want just the values, you can use `s.values`; if you want just the index, you can use `s.index`. Both of these objects can readily be converted to lists.

Question 18: What is the least central South American country?

This one has the largest average distance to other countries.

Question 19: how close is each country in South America to it's nearest neighbor?

The answer should be in a table with countries as the index and two columns: `nearest` will contain the name of the nearest country and `distance` will contain the distance to that nearest country.

Hint 1: find a Series of numerical data you can experiment with (perhaps from one of the DataFrames you've been using for this project). If your Series is named `s`, try running `s.min()`. Unsurprisingly, this returns the smallest value in the Series. Now try running `s.idxmin()`. What does it seem to be doing?

Hint 2: if you run `df.min()` on a DataFrame, Pandas applies that function to every column Series in the DataFrame. The returned value is a Series. The index of the returned Series contains the columns of the DataFrame, and the values of the returned Series contain the minimum values. If you run `df.idxmin()` on a DataFrame, the returned values contain indexes from the DataFrame.

Question 20: how far is each country in South America to it's furthest neighbor?

The answer should be in a table with countries as the index and two columns: `furthest` will contain the name of the furthest country and `distance` will contain the distance to that nearest country