Final Project: Web Service APIs

This final project should serve to bring together almost all of the topics we have been covering throughout this class, from manipulating tidy data to designing and storing data in a relational database to working with hierarchical data to the underpinnings of client-server computing and working with Provider APIs.

It is also intended to be self-defined. Do some exploration and find data from a provider that you are interested in investigating.

The final products of this project are two-fold:

1. You will produce and turn in a Notebook that gives documented processing code, organized as a set of functions, that shows your Python programming steps of data acquisition, traversing data and building normalized (tidy) data tables, the optional export of the tidy data in a well-designed relational database, and the export of the tidy data as CSV files to be used to enable visualizations. It may also include code for some preliminary data exploration, like histograms or scatter plots used to better understand the data.

2. You will produce an essay in which you develop a **data story**. A data story is a single, compact, thesis that drives your work from start (inquiry) to end (conclusion). You will have earlier phases of data exploration. These earlier phases SHOULD NOT generally become part of your essay. Use the data exploration phase to find a compelling storyline in the data you can utilize for your project. Typically, this essay is composed in its own Notebook that consists entirely of Markdown cells along with inclusion of figures generated through some visualization tool, such as Tableau, but other document composition toos are possible. Regardless of tool, you are to turn in a **PDF** of the resultant essay.

Requirements

The requirements of this project include the following:

1. Work with a Web API designed by a Data System provider.
    - The selected provider must require authentication, but this authentication may either be by API key or through the more involved `OAuth`.
    - The API must use HTTP for its requests and responses.
    - See https://github.com/public-apis/public-apis for a fairly comprehensive table listing of public and semi-public APIs along with the type(s) of authentication used by the API.
2. Find a second data source, which could be openly provided, or could be obtained through web scraping techniques, that complements the data from 1.
3. Use the skills/knowledge from the Hierarchical Data Models unit to parse and extract tables of information from XML, JSON, and HTML from your providers. You **must** end with **tidy** data tables (with explicitly documented functional dependency), and have these tidy tables in `pandas`.
4. (Extra Credit/Optional) Build a relational database from your data sources. This includes a sound database schema, table schema, and data population.
5. Practice good software development techniques:
    - Practice functional abstraction and associated code documentation
    - Program defensively, checking for error codes/returns in *every* client/server interaction, and handling the error in a reasonable way
6. Give the acquired data meaning through your data story and essay to ask and present findings for at least two interesting questions about the data.

Process

Students will work as individuals on this project, and the project will be due the last day of regular exams by **11:59pm, Dec. 3rd**. Students are **strongly encouraged** to start early and to document their progress through a journaling notebook.

The project specifics are left to the creativity and design by the student involved, but must satisfy all of the above goals. Explore the Providers and APIs, perhaps coming up with providers that are not included in the provided lists, and think about data that excites/intrigues you.

A minimum standard for questions/visualization/data exploration is two questions and three visualizations of moderate complexity that depict the data in a way conducive to answering the questions.

Some OAuth Data Providers

1. Facebook
    - **Graph API** allows querying and posting from a program and includes ability to get lists of friends and many other
2. LinkedIn
    - Uses OAuth 2
    - RESTful APIs
3. Fitbit
4. Tumblr
5. GitHub
6. Twitter
    - Note that this uses OAuth 1, not 2, so there will be additional learning curve
    - There are streaming APIs, but these will require additional complexity on the programming side, and to be useful, you will want to start early so that you have sufficient time to collect data on a stream to allow for interesting analysis
7. Dropbox
    - Data is relative to whatever one deposits into Dropbox, so this will need something additional to generate interesting data.
8. Google Drive
    - Same comment as for `Dropbox`