Liam Juskevice

FishBase Overview

Welcome to FishBase, a web scraper and database designed to accumulate easily digestible data on tropical fish. Originally this project was intended to have a front end allowing users to quickly browse through fish species by name or certain specifications such as region/country of origin, temperament, tank size, parameters, etc. However, it became increasingly clear that this one man team would need considerably more than 20 hours to make a well-designed front end when gathering this data by itself proves to be such a time-intensive task. As such, the project currently consists of a web scraping program and database that can be used to set-up such a front end. Now let’s take a bit of a deeper dive.

**Implementation**

This project is created using ruby programs and a MySQL database. I use Ruby for the web scraping program largely because I have some experience using Ruby for similar tasks in the past. It would probably not be too difficult to write a similar program in Python or almost anything else. The most important software libraries for this task are *Nokogiri* and *Mechanize*. Nokogiri is an HTML and XML parser that helps isolate specific elements in a web page and work with them to create meaningful results. Mechanize is a useful tool for automating web page interactions such as filling out forms or navigating to different pages. In the case of this project I only used Mechanize to navigate to different pages of the seriouslyfish.com knowledge base to collect data on every fish.

The MySQL database is filled out by the ruby programs to contain all the data the web scraping program gathers. There is no particular meaning to it being a MySQL database as this is once again largely a comfort pick. One could attain similar results using any other type of database considering the simplicity of this database’s structure. The primary table is called “Fish” which has a row for each fish species and contains columns on relevant information such as species name, tank size, temperature, etc. Then there is the countries table, which simply contains all countries in the world and which continent they’re in. There’s the regions table, which is simply a list of continent (besides Antarctica) and their ID.

These two tables are important because they allow me to make the relationship table fishCountries and to specify which region a fish is from. These two tables are rather simple in their use. FishCountries has a fish ID and a country ID, and each row is a country a fish is found in. As it is perfectly possible for fish to be from multiple countries I could not simply have country as another row in the Fish table. It would be bad form for a countries column to be some long comma separated list or something similar.

**Instructions**

Running this program is pretty simple, but you will need to set your computer up so that it can operate MySQL databases locally. I will not write detailed instructions on how to do so here, because it would depend on your operating system and it would be easier to simply refer you to this resource: <https://dev.mysql.com/doc/mysql-installation-excerpt/5.7/en/>

Once you have that set up, you can use the sql file provided to import the database. Note that this database should be called “FishBase” on your computer to make the scraper run properly. If not you may have to reconfigure line 11 on srslyFish.rb to fit the name of the database. Once the database is properly installed and configured running the scraping code is quite simple. You run srslyFish.rb in the terminal or any equivalent that can run ruby programs, input your database password, and let the scraper do its thing. In cases where it stops midway through you can modify the categories array found on line 13 and delete/move certain entries so the program doesn’t have to start from the very beginning again. Simple as that.