Streamlining data entry for museums <https://youtu.be/TweoSjKARA8>

When the museum gets donations, we fill out a data template that we then use to batch upload information into our database, TMS. The batch file process is only good for text entries and is not great for certain projects, specifically photo uploading and geography/ localities.

When we get photo files or take photos, they are usually named starting with IMG and then a string of numbers. There is a photo upload module where it will take an access file and upload images according to the template. However, the photos need to have the object primary key attached to it as well as the object number so the module knows which object to attach the photo to. This means we have to manually open all the photos to see what images objects are in the images, then rename the image and finally adding it to our access file so that they can be uploaded. As you can see it's not the easiest or quickest process because we have to label each photo one by one. I worked backwards to create an Excel file that had two columns, one column was the name of what the file was called originally and then the second column was the name of the object number or what I want the image to be named.

My Python code takes this csv and compares it to a list of all the files within the directory. If the first column finds a match with another filename, then it will rename and save the file to its destination directory. At the end of this, it will create a CSV file that has all required information for the template, including the name of the object name, the object ID, the file path which I can then use to upload onto TMS. I used very basic libraries for this: shutil, globil, os and csv for this part.

The second part involves the localities look up. When we get donations, we have a team of interns that fill out the location data in an excel file. They then look up the equivalent location in TMS using a thesaurus. This process is very slow because it cannot be done in a batch process and therefore can only be done one specimen at a time. I created a python code that mimicked this process by creating an index of localities, then matching that up against the localities data in the excel file which returned a value of what it thought was the closest locality.

The biggest difficulty was that any index match I do should not give me an exact one to one because the locations are written too differently every time. For example, Massachusetts could be written as is, MA or Mass. Because the syntax differs every time, I needed to find a way that we could match the localities with what was in the Excel file with the localities already in our database using a fuzzy look up or near matches.

I found a library called Whoosh that would allow me to do so. Whoosh allows me to do near matches, so if I type in ‘Boston USA’ , it won’t return only Boston USA. Instead, it will return all instances of Boston USA. The fuzzy look up algorithm is flexible enough to bring up multiple matches even though none of them are exact one to one match.

After all this, python produces a couple SQL queries for me so that I can pick which one I think is the most accurate. I enter the query directly into the SQL database. This is not a perfect solution but the only way I can batch upload localities.

In the future, I hope to be able to create a full list of results for locations; however, I realized I needed to clean the localities database before I could go any further with the search matches.