Project 3: Data Wrangling With MongoDB, Montreal

Map Area: Greater Montreal (Quebec, Canada), Custom Borders

Introduction

I have picked the Montreal, Quebec Canada greater area to work on. My main data-cleaning-focus was over detecting corrupt phone numbers and postal codes, detecting language mix in street names (English and French) and standardizing all street names to be in English. In the case when a corrupt data is found, a flag field within the document would be created called "has_corrupt_data", and another field called "corrupt_fields" is created, and the corrupt data is appended to that latter field. I have preferred to do this over deleting the data so there would be a chance to manually correct the data later on.

Problems Encountered

The first problem that was encountered is the Montreal map from MapZen extract: it included a huge border, clearly much greater than the city and its immediate suburbs. To overcome this, I have manually selected a smaller frame that included only the Montreal island, its southern shore suburbs, Laval island and the northern shore of Laval. I have downloaded the map using the overpass API. There was an old link for maps download that did not work, but I have notified the OSM organization, and they have quickly fixed it.

The next problem comes from the fact that this part of the world is French speaking. Of course this is not a problem by itself, but the problem was the mixing of street names between English and French. So for the same street, we could have a node called "Rue Sainte-Catherine", and another one "Saint Catherine Street".

After that, phone numbers of a node were sometimes included under the tag "phone", some other times it was "contact:phone".

Sometimes the phone numbers were written in letters, and this is a bit tedious to adjust (Because I have to programmatically find out if the letters were part of the phone number or were just non-standard letters trying to explain the phone number (Like indicating the hours to call, if it is a toll free number..etc). This is an example of such a phone number: 514-844-HERO (4376). The other problem with phone numbers is too much spaces separating the parts, like "514 899 – 9979". I chose not handle these and declare them as corrupt to avoid a potential conflict with fields containing more than one phone number.

Another problem that was found is that sometimes the data were entered to the wrong field. For example, one of the documents had as a phone number a website (petro-canada.ca)

Abbreviations were frequent in street names, both in English and French. There was even some tricky abbreviations: St. . Montreal is city with a French heritage, so it has a Catholic heritage and a lot of the streets are named after Saints. The word "Saint" is sometimes abbreviated, using either St. or Ste., the latter in the case of a female Saint. What makes this tricky is that St. is also used as an abbreviation for "Street". So there was a need to recognize when the abbreviation St. meant Saint or Street.

There was other tricky abbreviation-like problem that I could not programmatically handle. For example, there is a street called "Impasse J. E. Bernier". There is no way that I would know if E. would mean East or just the name of the street (Well, the name is actually the way I spelled, not abbreviated. Surprisingly!). For now, my script would still turn this last example to “Impasse J. East Bernier”, so this can be considered as a known bug in my submission.

Another problem was that since a lot of the names were written in French, the encoding raised a problem. I had to use the UTF8 encoding in Python, which is not used by the language by default.

The next problem is translating the street qualifiers. Rue, Boulevard and Avenue for example are easily translated (Street, Boulevard and Avenue, respectively); but some other qualifiers were not obvious to me, like Ascente and Descente for example (They literally mean an uphill and downhill. Literal translation was not appropriate in my opinion, so they were left as is for now). I have posted over the OSM (Open Street Map) forum asking for help, but there was no helpful reply so far.

Other noteworthy problems include:

- At many instances, the street name included also an apartment number, like (Bureau 141, Suite 101..etc). I have left these untouched for now.
- Sometimes the separator between the two parts that make up the postal code was not a single character as I assumed at first in my regular expression. I had to readjust my regex for that.
- Not all entries are standardized. For example, some entries were written with words separated by spaces, others by underscores. The OSM standard specifies when to use each way, but these guidelines were not always respected.

Data Overview

Data Size

greater_montreal.osm : approx. 314 MB (321,660 KB)
Note: The MapZen extract was 800,919 KB, clearly a much larger area was covered.

Number of documents: 1,480,814

Top Contributor: minewman

Type	Count
TAGS	
Nodes	1,240,288
member	19,992
nd	1,568,724
tag	1,404,837
way	240,526
relation	1,142
bounds	1
note	1
meta	1
Users	
Unique Users	1445

Users Appearing Just Once	289
Top Contributor’s contributions	483,543
Select Amenities	
Parking	1848 (Most Frequent)
School	1428
Restaurant	1128
Places of Worship	524
Fast Food	412
Cafe	380
Bank	277
Casino	1
Maker Space	1
Select Cuisines	
None	693
Burger	107
Pizza	70
Italian	59
Sandwich	56
Chinese	40
French	36
Select Denominations	
None	325
Catholic	82
Roman Catholic	29
Orthodox	8
Greek Orthodox	2
Ukrainian Orthodox	2

In the denominations, I have selected some which I think should have been labeled under the same label (Roman Catholic and Catholic for sure. Maybe not for the Greek\Ukrainian since maybe they pray in their native language and this is why qualifying them might be necessary)

An interesting thing to note: according to one of the OSM forum members, street names should be written in the language of the street sign. So what I have done while cleaning the data is not entirely correct, since I have converted the names into French. The right way was to have the English name under the tag “name:en”. For me I did not do that, although can easily be done, since I did not intend to apply the changes to the actual OSM. I have posted that I am doing some clean up and that if anyone is interested for this they can contact me, but no one contacted me until the time I am writing these letters.

Some additional surveyed data were the top brands when it comes to cafes and fast food. So here is the result:

Top 3 fast food chain:

Chain	Count
McDonald's	65
Subway	58
Tim Hortons	32

Top 3 fast food chain:

Chain	Count
Tim Hortons	55
Second Cup	32
Tim Starbucks	18

NOTE: All the queries used to generate these statistics are found within the submission's function called map_stats(). I did not included them within the report because they would have made the file too large.

Additional Ideas

An improvement method is automated typo checking. The idea I have in mind is by building a dictionary of the available street names, either using geoindexing (Same street would have nodes next to each other) or by frequency analysis of street names. If a street name is found, let's say, more than 10 times – it will be considered as a correct name and will be added to the dictionary of street names. If a street name is not in the dictionary, it can be substituted by the most similar name within the street names dictionary (That may involve using a data structure like a Trie to get the correct one.). In my opinion, this is doable – especially if we involve a human operator to confirm suggested changes before writing them to the database, but it would still require a good amount of effort that I felt was beyond what was required for the project.

The other thing to mention is that my gut feeling about the Montreal map within OSM is that it is far from comprehensive. Probably there are not enough mappers within the city, since there are some tags that are obviously too low. Montreal is a multicultural city, with a lot of immigrants – very diverse. For example, although it has its proper China Town, Chinese restaurants count were low. Only two dumpling places for instance. This makes the statistics extracted from here unreliable.

Conclusion

I think that data cleaning is context dependent. We should have a question in mind that we want to answer and under the light of that question we can better structure the data. For example, in all what I did during the project, I did not have any interest in the “ways”, so maybe I should have not included them into the database from the first place.

The other thing about the process of cleaning data is that it is an iterative process. The more I run the data cleanup and look at the result, the more ideas I have about how to better do it. For example, for the top 3 cafes and fast food chain above, Tim Horton was included in both. I know that Tim Horton offers the same service – so it is actually both fast food and café chain. In the next iteration I would unite both Tim Hortons under the same category. And who knows, maybe then I would find something better to do. This is probably a never ending process, so I think a very important question to identify from the context of our reason to use the data from the first place is: How much cleaning is enough? I do not think that there is a standard answer for this, but it will change from one project to another.

The other thing, which is a bit negative, the OSM forum is not very active. They did help a little bit, but with only one reply to my post.

For concluding about the data, the data needs a real long term effort to be cleaned, beyond the time allocated for just one project. I have detected many problems, like corrupt phone numbers (Missing a digit or more), corrupt postal codes and non-standard data. I liked the process of finding them, but doing the actual cleaning is not an easy task to be performed programmatically. The process of cleaning, although in a lot of cases can be resolved programmatically using simple algorithms, needs a minimum amount of “intelligence” to produce dependable results. This intelligence might be a human operator who corrects the data manually, it can be signaling to the user who entered the data that there is something wrong with his\her entry and that they must correct them, or even some sort of a machine learning algorithm who can correct the data in a better way than an automated script that can handle a finite cases of possibilities.