Data Wrangling Project

Area of Study: City of Montreal and surrounding environs, Quebec, Canada

https://www.openstreetmap.org/export#map=9/45.5593/-73.7238

Using the Overpass API the following coordinate ranges were selected. Latitude range (45.8403, 45.2739) Longitude Range(-74.4214, -73.0261)

I live in a rural town in upstate New York and Montreal is the largest and closest metropolitan area. I have enjoyed exploring Montreal and it's surrounding areas by car, bicycle and on foot. This amazing area has so much to offer I thought it would be interesting to explore the data.

Issues and items of interest encountered in the Map

- Four keys contained problem characters, specifically a period and a space. These are the four invalid keys ['addr.source:street', 'addr.source:housenumber', 'Rendu 3D', 'service road']
- PostCode missing or incorrect format
- Incorrect city names found in node and way child tags having ”k=addr:city”.
- Way tags with key values such as Canvec:code and Canvec:roadclass. While these are not problems they are of interest and are discussed below.
- A myriad of issues with street names and types
- Untagged unconnected Nodes
- Challenges with UTF-8 data not displaying correctly in the SQLite3 terminal

Problem characters in key fields

Four problem keys were found during the audit phase. Additional auditing was performed using the program problemcharsOnFull.py to get more information about these keys. The problems all occurred in children tags of "node" elements. The keys containing addr.source:street and addr.source:housenumber were added by the same user and the value for both was "survey". The other two problems contained just as mysterious data and therefore the decision was made to not include these keys in the csv files.

PostCode missing or incorrect format

A tag containing the key attribute "addr:postcode" was missing for most nodes; most notably for data resulting from CanVec imports. I had hoped to fix this but after doing some research decided this component could not be fixed programmatically at this time for the following reasons:

- The function found at https://github.com/inkjet/pypostalcode could have helped somewhat but only would have obtained the FSA code which is the first three characters of Canadian postal codes.
- The documentation at http://wiki.openstreetmap.org/wiki/Key:postal_code regarding licensing and legally obtaining the correct postal code indcates the only correct method to obtain the correct postal code is quite involved. Per their documentation *"Postcodes from*

scratch? Does this mean knocking on people's doors and asking what their postcode is? Yes... That's the way we build our map"

The program fixpostalcodes.py was used along with regular expressions to fix some of the postal codes by adding a space between third and fourth characters, and/or changing lower case letters to uppercase.

Incorrect city names found in node child tags having ”k=addr:city”

This was disovered at the audit phase and required three steps to resolve. First the program check_city.py was run over the entire osm file to create cities.csv. This file contains every city name encountered along with the number of times found.

The next step required importing the file to Excel specifying UTF-8 format and doing research to identify the correct name. In some instances it was clear a typographical error had been made and in others the Canadian Postal Service postal code lookup at https://www.canadapost.ca/cpo/mc/personal/postalcode/fpc.jsf was useful. The Excel file was modified to contain an additional column which contained the correct name.

Examples of mappings:

- In the case of Montreal which occurred 177 times , Montréal was selected to replace it.
- St was replaced with Saint or Sainte depending on the gender of the saint in question.

The last step was to add a function to the creatingCSVs2.py program to build a dictionary of cities needing modification from the Excel file and then modifying the get_tags function in it to fix the city name using the dictionary.

Once the data was in the SQL files the program code montrealvariations.py was run to verify that the variations of Montreal had been resolved. Below is a sample of the output.

Montréal 80419
Mont-Royal 1938
Deux-Montagnes 1643
Montréal-Ouest 459
Montréal-Est 423

CanVec data

During the audit thousands of records were found to have a tag with key="source" and value containing CanVec followed by a release number. According to http://wiki.openstreetmap.org/wiki/CanVec, "CanVec is a digital cartographic reference product produced by Natural Resources Canada(NRCan)." This source was definitely the largest contributor to the data encountered and any additional tags that are associated with this source did not impact my procedures or methods of auditing or fixing.

Problems with street names

An audit of the OSM file found 15,475 street names(see streets.csv). Due to the number of streets and variability of names it seemed best to fix some of the issues once the data was in the SQL format. Issues that were discovered during the audit:

- Abbreviated directions at the end of street names (O or O. for Ouest, S or S. for Sud, E or E. for Est, N or N. for Nord). Some street names actually have these abbreviations in them and are correct for the locale(e.g. Rue Laurent-O.David, and Rue Jean-E. Bouvy).
- Abbreviations such as Blvd and Ave and their variations were fixed.
- The audit indicated that the expected street types in Quebec are quite different than in the United States and include Rue, Croissant, Rang, and Chemin.
- Street rather than Rue exists in over 40 street records but since other parts of the name were in English rather than French they are left "as is" at this time.
- If a street name contained the abbreviation St it was left alone as we can't be certain if the intent was Saint or Street.
- The program fix_street_abbrev.py was used to fix the street names and write the original names along with the new to the file corrected_streets.csv
- A sample of results written to the corrected_streets.csv file(edited for easier reading):

original	corrected	count
Boulevard de la Concorde O	Boulevard de la Concorde Ouest	1
Ontario E.	Ontario Est	4
St-Charles blvd	St-Charles Boulevard	3

Untagged unconnected nodes

According to http://wiki.openstreetmap.org/wiki/Untagged_unconnected_node, "An untagged unconnected node is a node which is not part of any ways or relations (unconnected) and which does not have any tags on it". The documentation indicates that these nodes should be deleted if they are older than several days. This issue was analyzed once the data was in SQL tables.

Number of Nodes - just looking at nodes table

sqlite> select count(*) from nodes;
2173940

Number of Useful Nodes - create a view to only select nodes that are associated with at least a way or a tag.

sqlite> CREATE VIEW useful_nodes AS SELECT nodes.id, count(nodes_tags.id) AS tag _count, count(ways_nodes.node_id) AS ways_count FROM nodes LEFT JOIN nodes_tags ON nodes.id = nodes_tags.id LEFT JOIN ways_nodes ON nodes.id = ways_nodes.node_i d GROUP BY nodes.id HAVING tag_count > 0 or ways_count > 0;

sqlite> select count(*) from useful_nodes;
2173485

As the counts above indicate 455 nodes exist without a way or tag associated with them. Before actually deleting these nodes from OpenStreetMap further analysis would certainly be required. We would first need to determine if any relations existed for them as our pull from the OSM file did not deal with relations. Additionally the date on the nodes should also be checked to determine if they are actually older information.

Approximate File sizes:

File	Size	File	Size
montreal_map.osm	539 MB	ways.csv	22 MB
ways_tags.csv	34 MB	nodes.csv	180 MB
ways_nodes.csv	64 MB	nodes_tags.csv	47 MB
new_montreal database	521 MB		

Number of Unique Users and Top 5 contributing users

Unique users obtained with the following query.

sqlite> SELECT COUNT(DISTINCT(u.uid)) FROM (SELECT uid FROM nodes UNION SELECT u id FROM ways) u;
1794

The inner query combines the nodes and ways tables and includes duplicates and the outer select counts the number of times the users name is encountered. sqlite> SELECT c.user, COUNT(*) as count FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) c GROUP BY c.user ORDER by count DESC LIMIT 5;

User	Count
minewman	502330
canvec_fsteggink	248902
andrewpmk	236561
jfd553	199678
procyon43	140494

Top 5 sources

This query combines nodes_tags and ways_tags to determine the top 10 sources of where the data came from
sqlite> SELECT u.value, count(*) as count FROM (SELECT value FROM nodes_tags WHE RE key="source" UNION ALL SELECT value FROM ways_tags WHERE key="source") u GROU P BY u.value ORDER BY count DESC LIMIT 5;

Source	Count
NRCan-CanVec-10.0	313131
NRCan-CanVec-7.0	114442
CanVec_Import_2009	21228
NRCan-CanVec-8.0	18986
Bing	5502

This query makes it very clear that CanVec imports are the main contributors to Montreal's OpenStreetMap. Granted, this query does not account for data that may have been added to OpenStreetMap without having a source specified but based on the counts I believe it is safe to say data from CanVec is the greatest contributor.

Number of ways, see number of nodes above in untagged nodes issue

sqlite> select count(*) from ways;
360828

Top 5 Cities

Due to issues with the SQLite3 terminal and Windows displaying Unicode data correctly small Python programs were used to query the SQL files. TopCities.py was used to determine the most popular cities.

City	Count
Montréal	80419
Laval	35666
Terrebonne	11363
Saint-Jean-sur-Richelieu	10105
Repentigny	8164

Additonal Ideas

- Due to the amount of addresses missing postal code it seems it would be useful to determine way to update this at least somewhat programmatically. It may involve actually purchasing address lists from Canada Post
- As mentioned earlier some time could be spent determining which unconnected nodes could be deleted from OpenStreetMap
- The following query was used to check the date of ways

sqlite> SELECT w.date, count(*) as count FROM (SELECT substr(timestamp,0,8) as ate FROM ways) w GROUP BY w.date ORDER BY count DESC LIMIT 5; 2013-08|64226 2012-08|39790 2015-05|17723 2011-05|17461 2015-07|15422

Since major construction of highways and exits have been added in and around Montreal it may be useful to check if some of these changes have been reflected OpenStreetMap. Perhaps looking at the data from 2013-08 and 2012-08 to make sure the ways are still valid. According to http://wiki.openstreetmap.org/wiki/CanVec the last CanVec data was converted to OSM July of 2012 so reviewing Natural Resources Canada(NRCan) for more current information makes sense but this same page also warns, "Many experienced OSM users have inadvertently broken parts of the map while importing Canvec data" so great caution and working with the OpenStreetMap community would be needed before attempting to import data from NRCan.

Additional Exploration

- Top 10 Amenities As a cyclist I found the fact that the fourth most common amenity was for bicycle parking. Part of the reason this number may be so high is that Montreal does have bike sharing kiosks. The bixi website, http://montreal.bixi.com/en/who-we-are, 460 statioins exist throught the Montreal metropolitan area.

sqlite> SELECT value, count(*) as count FROM nodes_tags WHERE key="amenity" GROU P BY value ORDER BY count DESC LIMIT 5;
restaurant|1298
bench|544
fast_food|459
bicycle_parking|456
cafe|420

Conclusion

Montreal and the surrounding metropolitan areas seem to be well represented in OpenStreetMap. Eventhough a majority of the data is from Natural Resources Canada(NRCan) it does appear that others have contributed as well. It would be nice to see more data contributed from individuals with local knowledge but with that comes the potential for greater error and lack of standardization. Additionally great care would be needed to ensure that any additional information does not conflict with existing data.

Reference List

http://assemble.io/docs/Cheatsheet-Markdown.html

http://answers.microsoft.com/en-us/office/forum/office_2007-excel/accented-characters-dont-appear-corectly-in-excel/19ee3ece-9449-e011-8dfc-68b599b31bf5?auth=1

https://docs.python.org/2/library/

http://montreal.bixi.com/en/who-we-are

http://sebastianraschka.com/Articles/2014_sqlite_in_python_tutorial.html

https://www.sqlite.org/lang_corefunc.html

http://stackoverflow.com

http://www.techonthenet.com/sqlite/tables/create_table_as.php

http://www.tutorialspoint.com/sqlite/sqlite_unions_clause.htm