# Wrangling OpenStreetMap data of Edmonton, AB, Canada

## *Project objective*

In this project, OpenStreetMap data of down town and surrounding area of Edmonton, AB, Canada has been cleaned. The following link shows the greater Edmonton area. http://www.openstreetmap.org/relation/2564500. From this map, I manually chose downtown and surrounding area ( http://www.openstreetmap.org/export#map=12/53.5140/-113.5344).

## *Auditing and cleaning data*

### Auditing name of the tag k

Names of the k attributes found to be inconsistent. For instance for postal codes both 'addr:postcode' and 'postal_code' are used. So names of the k attributes were audited in the 'audit_k_value.py' file. Another inconsistent naming is for phone numbers, where both 'phone' and 'contact:phone' have been used.

### Auditing latitude and longitude

The part of the Edmonton I selected lies between the latitude 53.4251 and 53.6593, and longitude -113.7085 and -113.3446. Latitudes and longitudes in all the nodes fall within this range (audit_lat_lon.py). Since latitudes and longitudes have been assigned automatically, it is less likely that there will be errors in this entry

### Auditing postal code

A Canadian postal code is a six-character string that are alphanumeric. They are in the format *A1A 1A1*, with a space separating the third and fourth characters[1].
There were some unexpected postal codes. Among these, I converted "T6E 4R9 Commute to Downtown Edmonton  4 min  24 min  8 min  30 min View Routes Check Availability Favorite Map Nearby Apartment" to "T6E 4R9" and "AB T6E4S6" to "T6E 4S6". The tags with the postal codes 'AB T5J', 'Alberta T6G', 'T5J', 'T6E', 'T6G' were ignored as these are incorrect postal codes.

#### *Cleaning postal codes*

Postal codes were in different format in open street data (e.g., *A1A1A1, A1A-1A1, a1a1a1*). I changed all of them to the format *A1A 1A1* ('data_cleaning.py, file).

### Auditing phone numbers

Phone numbers have been audited to see if there are any unusual phone numbers such as phone numbers having less than 10 digits, phone numbers not having Edmonton area code (780), or phone numbers having characters other than ()-\s+.0123456789 (audit_phone.py).

The following phone numbers do not conform to the above rules.

'1 (866) 912-3560' – Toll free number, so left unchanged.

'+1-587-520-6338' – Could be a branch of a restaurant that has its main office located elsewhere, so left unchanged.
'+16133544904' - this entire node seem to be wrong as city is entered as Greater Napanee. So entire node has been ignored.
'+14503497898' - this entire node seem to be wrong as city is entered as Greater St-Jean-sur-Richelieu. So entire node has been ignored.
'+17053281190' - this entire node seem to be wrong as city is entered as Lindsay. So entire node has been ignored.
'1-866-540-4468' - Toll free number of a hotel in Edmonton, so left unchanged.

***Cleaning phone numbers:***
As per Language Portal of Canada, phone numbers are to be written with a hyphen between each sequence, as follows: 1-NPA-NXX-XXXX or NPA-NXX-XXXX[2]. All phone numbers have been cleaned to 1-NPA-NXX-XXXX format ('data_cleaning.py, file).

**Audit geographic details**
I audited the data to verify if city is Edmonton, province is Alberta and country is Canada in all tags ('audit_geography.py').
***Unexpected city names***: 'Edmonton, AB', 'Greater Napanee', 'St-Jean-sur-Richelieu', 'Lindsay'.
Edmonton, AB is correct name, and just needs to be changed to Edmonton. The nodes containing city names 'Greater Napanee', 'St-Jean-sur-Richelieu', 'Lindsay' have been ignored (phone numbers and postal code also suggest that this node contain information that is not related to Edmonton).
***Unexpected province names***: 'AVB', 'Aberta', 'Alberta;AB'
The first two cases seem to typo, and in the third case both full name and code is mentioned for the province, therefore, all three will be just converted to AB.
***Unexpected country names*****:** None

***Cleaning geographic information***
The geographic data has been cleaned to the following standard format ('data_cleaning.py, file).
City = Edmonton
Province = AB
Country = Canada

**Auditing house numbers**
House numbers have been audited to check if they contain only numbers, or hyphenated numbers (audit_house_number.py).
***Unexpected house numbers*****: 'P.O. Box 21100'**, '10200 Suite 1259', '111000 Suite K6', **'1240, 5555'**, **'10103;10360'**, '8525.0', '#200 10150', **'109 Street'**, 'Main Address\t10015', 'Suite 220-10423'
Tags with house numbers in bold above have been ignored as it is difficult to ascertain correct house numbers. The rest have been changed appropriately (see mapping dictionary in the "clean_house_number" function in the 'data_cleaning.py' file).

**Auditing street types**
The code is in AuditStreetNames.py file.

***Unexpected street types*:**

Avenue)
104th
North-west
St
Rd
North-West
street
Northwest
W
AVE
Blvd
Ave
Rue
avenue
806
NW

Auditing street type was little tricky as some names contain prefixes like NW, North-west etc. These street names have been further audited.

***Unexpected street types in street with suffixes such as North-west***

'Circle'
'Close'
'Southridge'
'Wynd'

***Unexpected street types in street with suffixes such as NW***

Ave.
Ave
St
St.

***Cleaning street names***

The words in the above lists have been changed to appropriate types (please see dictionary ‘mapping’ in ‘ImproveStreetNames.py’ file) except the following cases. The word ‘Avenue)’ is in 'Buena Vista Road (87th Avenue)' and it is correct as this street has two names, so left unchanged. Rue is also left unchanged as it is correct street name (Rue is French word for street). Suffix ‘W’ appeared in one street name, but information in this node (postal code, phone number and city name) suggest that this belong to Lindsay city. So whole node has been ignored.

The words 'Circle', 'Close', 'Southridge' and 'Wynd', although look strange as a street name, in Edmonton a street name can have these words as street types, and hence they are left unchanged.

In streets with suffixes, Northwest, and North-West, I found no problems with street names. These suffixes have been changed to NW to be consistent (‘data_cleaning.py, file).

## *Preparing CSV files for database*

Openstreet data has been parsed, cleaned and written to csv files in the 'data.py' file. One problem I faced is that when I ran this script I got '*ParseError*: *unclosed token'* error. However, when I restarted spyder, and reran the script it ran and created all CSV files just fine.

## *Overview of the data*

***Size of the files***
edmonton_canada.osm – 67.267 MB
openstreet.db – 36.602 MB
nodes.csv – 26.570 MB
nodes_tags.csv – 1.797 MB
ways.csv – 2.879 MB
ways_tags.csv – 2.424 MB
ways_nodes.csv - 8.744 MB

## *SQL code*

SQL tables have been created using the script in the 'sql_create_tables_1.py' and 'sql_create_tables_2.py' files[3].

***Number of nodes***
```
sqlite> SELECT COUNT(*) FROM nodes;
310940
```

***Number of ways***
```
sqlite> SELECT COUNT(*) FROM ways;
47432
```

***Number of unique users***
```
sqlite> SELECT COUNT(DISTINCT(e.uid))
FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) e;
242
```

***Available Amenities in this area (in ways)***
```
sqlite> SELECT value, COUNT(*) as num
FROM ways_tags
WHERE key='amenity'
GROUP BY value
ORDER BY num DESC;
```

***Available Amenities in this area (in nodes)***
```
sqlite> SELECT value, COUNT(*) as num
FROM nodes_tags
WHERE key='amenity'
```

```
GROUP BY value
ORDER BY num DESC;
```

***Top 10 users who generated the most numbers of node items (tags)***

```
sqlite> SELECT nodes.user, count(*) as num
FROM nodes, nodes_tags
WHERE nodes.id = nodes_tags.id
GROUP BY nodes.user
ORDER BY num desc
LIMIT 10;
```

## *Further improvement of the dataset*

1. Zip codes (postal codes) in this data set can be compared against databases containing zip-codes of Edmonton to ascertain their accuracy. I found the following two sources that gives list of Edmonton postal codes.

   http://www.findthepostalcode.com/location.php?province=AB&location=Edmonton
   https://www.zip-codes.com/m/canadian/city.asp?city=edmonton

   Unfortunately, both of these sources are not up-to-date. On digging deeper, I found the following interesting article, wherein, Canada Post sued a company to keep ownership of postal code list. The article also says Canada Post charges companies approximately $5,500 a year for the postalcode information.

   http://www.torontosun.com/2012/04/13/canada-post-sues-to-keep-ownership-of-postal-code-list

   Hence, I did not proceed further with this task. However, I wrote a code to do this task (postcode_validity.py) when authentic list of postal codes are available.

   It looks like Canada Post is building a postal code database (http://open.canada.ca/en/suggested-datasets/postal-code-database). However, it is not known how we can access this database

2. We can use current business directories to verify information like whether a business still exists as shown in the openstreet data, whether the phone number given is correct, or whether there is any update to the other information entered in openstreet map.

## *References*

1. https://en.wikipedia.org/wiki/Postal_codes_in_Canada
2. https://en.wikipedia.org/wiki/National_conventions_for_writing_telephone_numbers#Canada
3. https://discussions.udacity.com/t/creating-db-file-from-csv-files-with-non-ascii-unicode-characters/174958/7
4. http://www.findthepostalcode.com/location.php?province=AB&location=Edmonton
5. https://www.zip-codes.com/m/canadian/city.asp?city=edmonton

6. http://www.torontosun.com/2012/04/13/canada-post-sues-to-keep-ownership-of-postal-code-list

7. http://open.canada.ca/en/suggested-datasets/postal-code-database