# OpenStreetMap Data Wrangling Case Study

## OSM Dataset

The OSM dataset I have chosen is of my current hometown, Toronto, Ontario. I chose this location because I'd like to learn more about my hometown and would like to teach others more about it. Also I hope to move on from my boring government job to start my carrer as a data scientist in this great city!

The dataset was exported as a metro extract from Mapzen: https://mapzen.com/data/metro-extracts/metro/toronto_canada/ (https://mapzen.com/data/metro-extracts/metro/toronto_canada/)

## Problems Encountered in the Dataset

After downloading the dataset from Mapzen, I extracted a sample dataset to view and perform a quick analysis of the data. I ran multiple functions to view the formatting consistency of tag attributes and noticed the following issues:

- Abbreviated street names (e.g. Bremner Blvd, Front Str.)
- Inconsistent province format (e.g. Ontario vs ON)
- Inconsistent postal code format (e.g. "M5J0A8" vs "M5J 0A8")
- Inconsistent building colour format (e.g. grey vs #F2F2F2)

### Abbreviated Street Names

As I audited the street names, I noticed most of the problems were indicated in the OSM case study. There were street names with abbreviated endings and there were also lots of abbreviation that I didn't think about until I looked at all of the distinct street name endings. I also noticed there are street names with multiple abbreviations, such as "Wellington St. E" or "St. John St." Only the street names with an ending abbreviation were identified and fixed.

After I exported the data to the way_tags csv file, I examined street names by filtering the "key" column by "street" and noticed that street names are used by two different "type" values: "addr" and "destination". After learning this I included the "destination:street" key to the is_street_name function to make sure I corrected the street names for this tag key.

### Inconsistent Province Format

I explored the "addr:province" key value and noticed there were inconsistencies with the way "Ontario" was inputted. The values were either "Ontario" or "ON". The standard format for a Canadian proince as indicated on the OpenStreetMap wiki is an uppercase two-letter abbreviation. Therefore, all instances of "Ontario" were replaced with "ON".

### Inconsistent Postal Code Format

The standard postal code format in Canada is a 6 character alphanumeric string. The first, third, and fifth characters are letters, and the oher characters are digits. The Canada Post (Canada's mail service crown corporation) displays postal codes with a space in between the third and fourth character of a postal (e.g. "M5J 0A8"). In the openstreetmap csv files, I noticed most postal codes have the space in between the third and fourth characters, however some do not have the space. To be consistent with Canada Post, I changed all postal codes with no space in between the characters to have a space.

I also audited for any unusual postal code formats such as the 5-digit US ZIP code and other strings that did not follow the format I explained in the previous paragraph, but I did not discover any usual strings.

### Inconsistent Building Colour Format

As I was exploring the way_tags csv file I noticed the format of the colour key for buildings was inconsistent. Some colors were represented with words (e.g. black, white, red) and other colours were represented in hexadecimals. This makes it difficult for someone to make comparisons.

## Converting to CSV Files

After I identifed problems in the toronto_osm dataset to fix, I created scripts to fix these problems as they were being converted to CSV files. I used the preparing_for_database python file to convert the text files to CSV while fixing the issues identified with the improving_street_names python file.

## Coverting to Database

Once the CSV files were created, I then created a SQL schema to organize and convert the data in the CSV files into database tables for me to run SQL queries.

## Exploring the Dataset

## Size of Files

- toronto_canada.osm -> 1.13GB
- database.db -> 691MB
- nodes.csv -> 399MB
- nodes_tags.csv -> 87MB
- ways.csv -> 43MB
- ways_nodes -> 135MB
- ways_tags -> 94MB

## Counting Tags

In [2]:
```
import iterative_parsing
iterative_parsing.count_tags("toronto_canada.osm")
```

Out[2]:
```
{'bounds': 1,
 'member': 148654,
 'nd': 5818706,
 'node': 5083151,
 'osm': 1,
 'relation': 9560,
 'tag': 4992810,
 'way': 754071}
```

## Number of Unique User IDs

In [9]:
```
import sqlite3
db = sqlite3.connect("database.db")
c = db.cursor()
query = "select count(distinct(x.uid))\
        from (select uid from nodes union all select uid from ways) x;"
c.execute(query)
print c.fetchall()
```

```
[(2657,)]
```

## Number of Nodes

In [6]:
```
query = "select count(*) from nodes;"
c.execute(query)
print c.fetchall()
```

```
[(5083151,)]
```

## Number of Ways

In [7]:
```
query = "select count(*) from ways;"
c.execute(query)
print c.fetchall()
```

```
[(754071,)]
```

## Top 5 Contributing Users

In [10]:
```
query = "select x.user, count(*) as num\
        from (select user from nodes union all select user from ways) x\
        group by x.user\
        order by num desc\
        limit 5;"
c.execute(query)
print c.fetchall()
```

```
[(u'andrewpmk', 3394211), (u'Kevo', 484571), (u'MikeyCarter', 474275), (u'Boo
tprint', 208858), (u'Victor Bielawski', 142906)]
```

## Number of Dentist Establishments

In [12]:
```
query = "select count(*) from nodes_tags where value='dentist';"
c.execute(query)
print c.fetchall()
```

```
[(690,)]
```

## Number of Optometrist Establishments

In [16]:
```
query = "select count(*) from nodes_tags where value='optometrist';"
c.execute(query)
print c.fetchall()
```

```
[(25,)]
```

This 25 figure seems suspicious. Maybe the toronto OSM data still has to be populated with most optometrist establishments.

## Top Tourism Establishments

In [14]:
```
query = "select value, count(*) as num\
        from nodes_tags\
        where nodes_tags.key = 'tourism'\
        group by value\
        order by num desc\
        limit 20;"
c.execute(query)
print c.fetchall()
```

```
[(u'information', 330), (u'picnic_site', 244), (u'artwork', 155), (u'viewpoin
t', 119), (u'attraction', 104), (u'hotel', 63), (u'museum', 57), (u'motel', 2
7), (u'camp_site', 22), (u'guest_house', 13), (u'hostel', 7), (u'gallery',
 3), (u'caravan_site', 2), (u'yes', 2), (u'zoo', 2), (u'Hotel', 1), (u'chale
t', 1), (u'fish ladder', 1), (u'theme_park', 1), (u'winery', 1)]
```

## Top Bank Establishments

In [15]:
```
query = "select nodes_tags.value, count(*) as num\
        from nodes_tags\
            join (select distinct(id) from nodes_tags where value = 'bank' ) x
 on nodes_tags.id=x.id\
        where nodes_tags.key = 'name'\
        group by nodes_tags.value\
        order by num desc\
        limit 10;"
c.execute(query)
print c.fetchall()
```

```
[(u'TD Canada Trust', 232), (u'Scotiabank', 154), (u'CIBC', 132), (u'BMO Bank
of Montreal', 92), (u'RBC', 84), (u'RBC Royal Bank', 48), (u'RBC Financial Gr
oup', 39), (u'CIBC Banking Centre', 32), (u'BMO', 31), (u'HSBC', 26)]
```

## Top Highways

In [17]:
```
query = "select value, count(*) as num\
        from waystags\
        where waystags.key = 'highway'\
        group by value\
        order by num desc;"
c.execute(query)
print c.fetchall()
```

```
[(u'service', 82640), (u'residential', 81483), (u'footway', 36683), (u'second
ary', 27644), (u'path', 17186), (u'tertiary', 13388), (u'motorway_link', 340
7), (u'motorway', 2463), (u'unclassified', 2432), (u'secondary_link', 1526),
 (u'steps', 1510), (u'cycleway', 1437), (u'proposed', 1061), (u'constructio
n', 1053), (u'track', 1022), (u'primary', 736), (u'pedestrian', 272), (u'tert
iary_link', 142), (u'trunk', 46), (u'platform', 43), (u'primary_link', 39),
 (u'living_street', 26), (u'raceway', 23), (u'bridleway', 20), (u'corridor',
 14), (u'services', 3), (u'road', 2), (u'service_link', 2), (u'stop', 2),
 (u'abandoned', 1), (u'disused', 1), (u'elevator', 1), (u'parking_aile', 1),
 (u'residental', 1), (u'rest_area', 1), (u'tertiary_', 1), (u'traffic_islan
d', 1), (u'trunk_link', 1)]
```

# Additional Ideas

## More data for healthcare establishments

When I searched for professional healthcare establishments, such as optometrists, opthamologists, podiatrists, etc., in the value column of the nodes_tags file, there were either not many results (only 25 optometrists) or no results at all (opthamologists were not present in the dataset). For example, below is query for the number of chiropractors practising in Toronto:

In [19]:
```
query = "select count(*) from nodes_tags where value='opthamologist';"
c.execute(query)
print c.fetchall()
```

```
[(0,)]
```

In [18]:
```
query = "select count(*) from nodes_tags where value='chiropractor';"
c.execute(query)
print c.fetchall()
```

```
[(44,)]
```

## Benefits and Potential Issues

The query above returned 44 chiropractor establishments in Toronto. But if I use a website such as www.opencare.com, a database of healthcare professionals pracitising in some cities, the website lists there are over 700 chiropractors practising in Toronto. There could be more than chiropractor practising in one clinic, but it's not realistic to assume 700 chiropractors are practising in only 44 establishments.

Data from www.opencare.com or a similar source such as the College of Physicians and Surgeons of Ontario (CPSO) have data about family doctors and specialists, including their addresses, qualifications, contact information. This information can be scraped from CPSOs website with new nodes being created for establishments not existing in the OSM dataset and multiple tags for a node can be created if there is more than one type of healthcare professional working there. This will be of great benefit to the OSM database since informaton on CPSO is up-to-date. Everytime a Doctor wants to open up a new practise or practise at another Doctor's clinic, they have to register this information with CPSO.

One issue I can see arising with the use of CPSO's website for this information is the misleading "Professional Corporation Information" section of a Doctor's profile page. Many Doctor's have registered a corporation with CPSO which they use when they establish their own clinic. The "Professional Corporation Information" section also provides the address of the corporation. But what is misleading is there are business addresses listed under this section which are not owned or operated by the respective Doctor. In some instances, they are locations where the Doctor receives income from (hospital or someone elses clinic) which they which they claim through their corporation. It doesn't mean they own or operate a clinic through their corporation.

A problem that could occur in the OSM database is if someone mistakes these types of business addresses as a new node and uses the name of the Doctor's corporation as the name of the node instead of the actual name of the location such as the name of the hospital or healthcare clinic.