Wrangling Data in SQL: OpenStreetMap Edmonton and Area

# Map Area:

I chose to complete this project on the Edmonton, Alberta, Canada area, accessible at:

* [OpenStreetMap](https://www.openstreetmap.org/relation/2564500)
* [MapZen](https://mapzen.com/data/metro-extracts/metro/edmonton_canada/)

# Challenges Encountered during Wrangling:

I started investigating the data on a small subset of the original file, using sample.py and a k of 100 to create and osm file that was 1% of the original dataset (which is 825.5 MB). The resulting sample\_k100.osm file is 8.3 MB.

## Geobase Tags:

Running tags.py against the sample osm revealed that there were no tags with problem characters, but there were some that were classified as “other”. On further investigation, these tags had capitalization in them, and were all coming from Geobase tags: “geobase:acquisitionTechnique” and “geobase:datasetName”. The source of these tags was the [Canadian Geospatial Data Sets](https://canadiangis.com/geospatial/geobase). Ultimately, since the data contained in these tags was irrelevant to my ultimate goals of investigation, I decided to ignore them when parsing the osm file to the csv files.

## Tag Names:

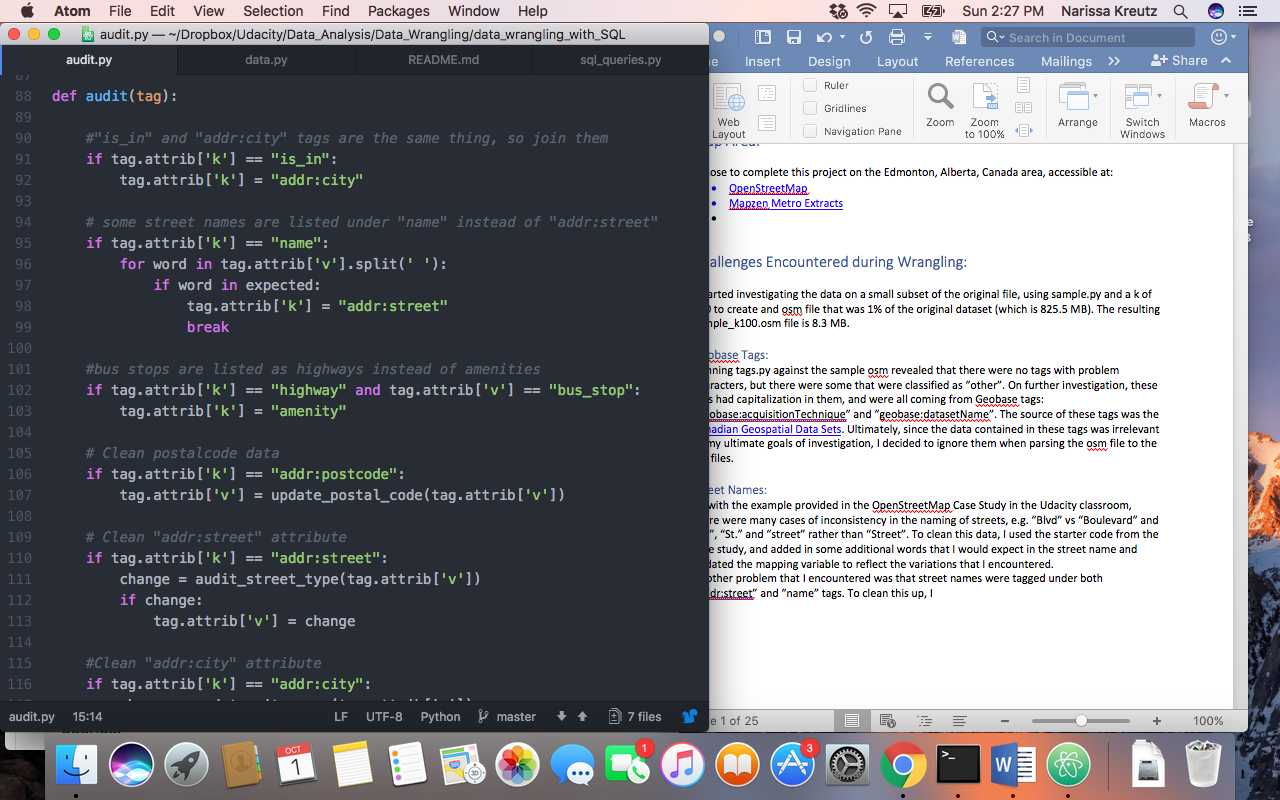
After this, I decided to take a look at what tag names were present in my dataset. There are 65 total possible “k” values, including: 'addr:city', 'addr:housenumber', 'addr:interpolation', 'addr:postcode', 'addr:province', 'addr:street', 'amenity', 'attribution', 'barrier', 'bicycle', 'bridge', 'building', 'building:levels', 'cables', 'contact:phone', 'created\_by', 'crossing', 'electrified', 'fixme', 'foot', 'footway', 'frequency','gauge', 'geobase:acquisitionTechnique', 'geobase:datasetName', 'geobase:uuid', 'highway', 'intermittent', 'is\_in', 'lamp\_type', 'landuse', 'lanes', 'lanes:backward', 'lanes:forward', 'layer', 'leisure', 'lit', 'maxspeed', 'name', 'natural', 'office', 'oneway', 'opening\_hours', 'phone', 'place', 'population', 'power', 'railway', 'ref', 'resource', 'restriction', 'service', 'shop', 'source', 'sport', 'surface', 'tunnel', 'type', 'voltage', 'water', 'waterway', 'website', 'wikidata', and 'wikipedia'.

To decrease the size of the resulting csv files and time needed to run data.py on the osm file, I decided to “ignore” tags that I deemed irrelevant to my investigation by including an ignore\_list variable in my data.py file, comparing tags to that list, and ignoring the tag if it was part of that list. I decided to add 19 tags to that list.

## Street Names:

As with the example provided in the OpenStreetMap Case Study in the Udacity classroom, there were many cases of inconsistency in the naming of streets, e.g. “Blvd” vs “Boulevard” and “ST”, “St.” and “street” rather than “Street”. To clean this data, I used the starter code from the case study, and added in some additional words that I would expect in the street name and updated the mapping variable to reflect the variations that I encountered.

Another problem that I encountered was that street names were tagged under both “addr:street” and “name” tags. To clean this up, I converted any “name” tag that was actually a street name using this function, which compares the content of the name tag to the expected words in the street name:



## Bus Stops:

When looking through the “name” tag, I decided to investigate what some of the 4-digit values referred to. What I came across was this:

<tag k="name" v="2210" />

<tag k="highway" v="bus\_stop" />

So bus stops are being listed as highways… and the bus stop number is being listed as a name. I wondered if maybe these bus stops were located on a highway, and investigated a few using the [Edmonton Transit Website](https://etslive.edmonton.ca/hiwire?.a=iLocationLookup&.s=09f37a3itd7tk), but this was not the case, many were residential or transit center stops. So, I decided to change the tag to “amenity” within my audit function if the value was “bus\_stop” as I feel that is a better representation of a bus stop. One thing that I did not tackle in this project, but feel would have been worthwhile is also changing the “name” tag to “bus\_stop\_number” so that the node would look like this instead:

<tag k="bus\_stop\_number" v="2210" />

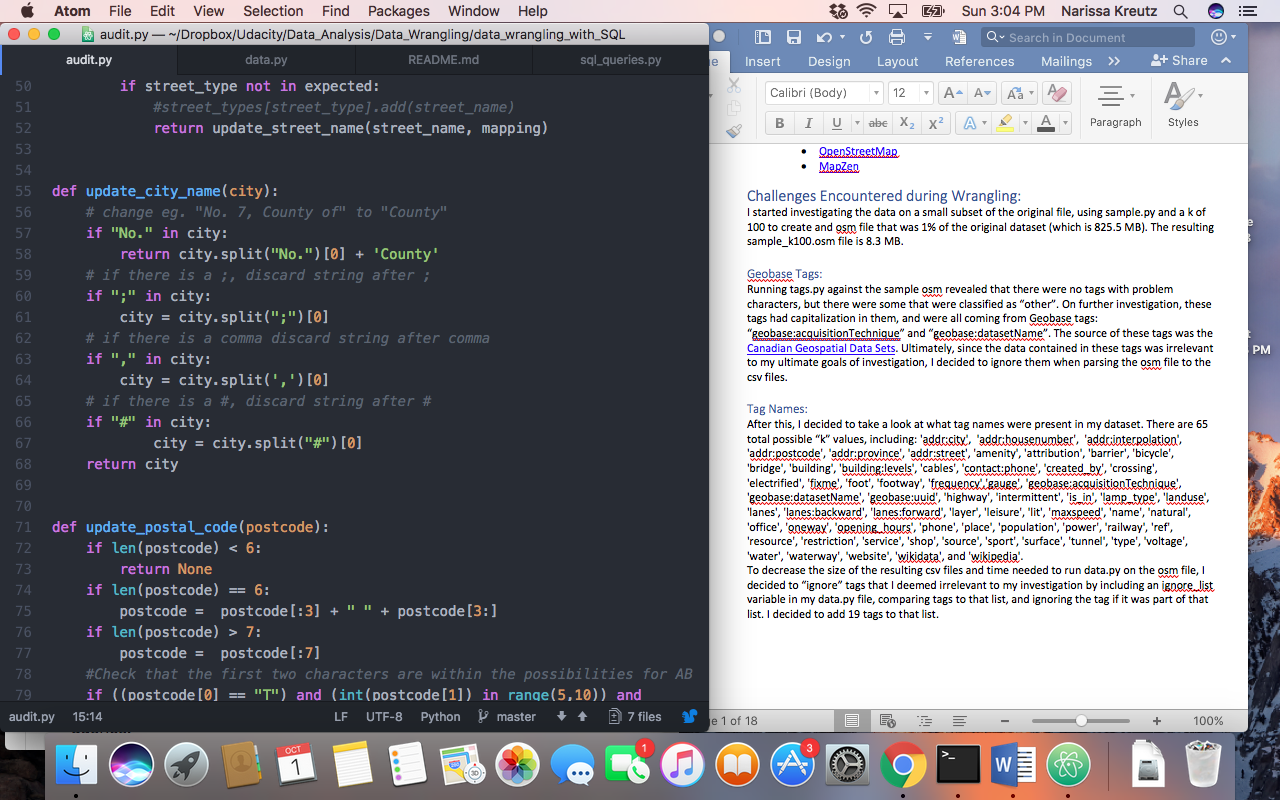
<tag k="amenity" v="bus\_stop" />

However, this change would be considered out of scope in difficulty for the purposes of this project, so I left it as is.

## Cities:

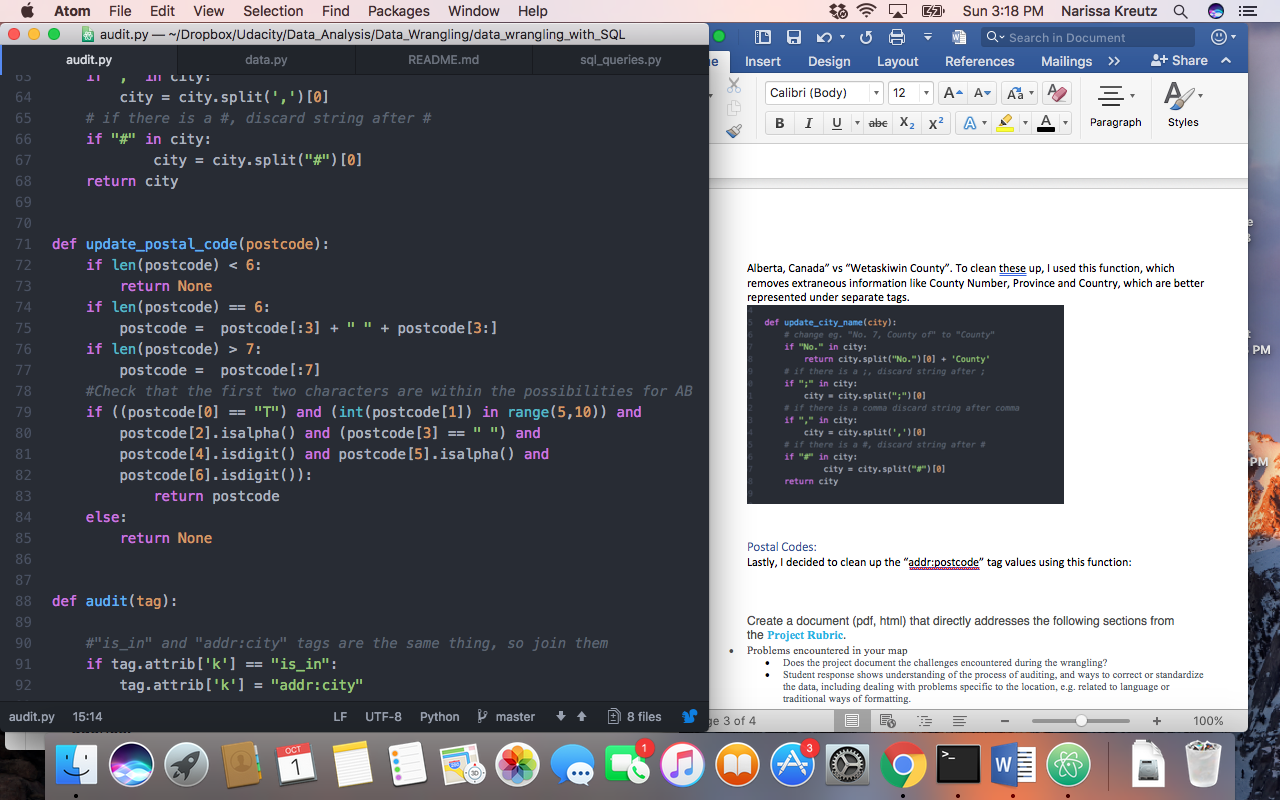
When looking through the values that certain tags held, I realized that the “addr:city” tag and “is\_in” tag held the same information. There were no nodes that held both an “addr:city” and an “is\_in” tag simultaneously. I decided to combine these two tags under the “addr:city” tag.

There were also some city names that required cleaning. For example “Wetaskiwin No. 10, county of”, “Wetaskiwin No. 10, county of, Alberta” and “Wetaskiwin No. 10, county of, Alberta, Canada” vs “Wetaskiwin County”. To clean these up, I used this function, which removes extraneous information like County Number, Province and Country, which are better represented under separate tags.



## Postal Codes:

Lastly, I decided to clean up the “addr:postcode” tag values using this function:



[Edmonton Area Postal Codes](https://en.wikipedia.org/wiki/List_of_T_postal_codes_of_Canada) are 7 characters long, and follow the pattern T, Digit in range 5 to 9, Letter, Space, Number, Letter, Number. I discarded any values that were under 6 characters long, and discarded extraneous characters from values that were over 7 characters in length. Sometimes the value was 6 characters in length due to a missing space, so I added a space to values with a length of 6 characters. Then I took the postcode and compared it to the pattern. If it still did not follow the exact pattern, it would not be a valid postal code, and therefore I discarded it as a value.

## Statistical Overview of the Data:

Create a document (pdf, html) that directly addresses the following sections from the [**Project Rubric**](https://review.udacity.com/#!/projects/3168208620/rubric).

* Overview of the Data
* Are overview statistics of the dataset computed?

Database queries are used to provide a statistical overview of the dataset, like:

* size of the file
* number of unique users
* number of nodes and ways
* number of chosen type of nodes, like cafes, shops etc.

Additional statistics not in the list above are computed. For SQL submissions some queries make use of more than one table.

* The submission document includes the database queries and statistics from above.
* Other ideas about the datasets
* Are ideas for additional improvements included?
* Submission document includes one or more additional suggestions for improving the data or its analysis. The suggestions are backed up by at least one investigative query.
* Are benefits and problems with additional improvements discussed?
* Submission document includes thoughtful discussion about the benefits as well as some anticipated problems in implementing the improvement.
* Submission document is long enough to thoroughly answer the questions asked without giving unnecessary detail. A good general guideline is that the question responses should take about 3-6 pages.

amenity produces:

set(['fast\_food', 'toilets', 'school', 'atm', 'bar', 'restaurant', 'police', 'parking\_entrance', 'recycling', 'car\_wash', 'pharmacy', 'bench', 'parking', 'fuel', 'place\_of\_worship', 'post\_office', 'cafe', 'bank'])

ok so that looks better, still not as much variation as I would have thought, but much better There are no surprises here or duplications

used stackoverflow.com to do some data manipulation

used live help to understand scope of the project and forums to learn how best to import a csv into SQL

sqlite3 edmonton\_openstreetmaps.db

sqlite> .mode csv

sqlite> .import nodes\_tags.csv nodes\_tags

sqlite> .import nodes.csv nodes

sqlite> .import ways\_nodes.csv ways\_nodes

sqlite> .import ways\_tags.csv ways\_tags

sqlite> .import ways.csv ways

cities represented:

SQL queries:

What city has the largest amount of fast food restaurants per capita?