Wrangling Canberra, ACT, Australia OpenStreetMap data!

Author: Jae Hee Lee

1. Brief Process Overview

I have chosen Canberra because this is the place where my University is located in.

Convert OSM XML Data to JSON Data	Store JSON data to MongoDB	Do data exploration via querying MongoDB

2. Converting OSM XML data to JSON data

Getting OSM XML Data

I downloaded data from Mapzen. They have already prepared metro extracts for Canberra, ACT, Australia.

Wrangling OSM XML Data

This data cleaning process was probably one of the most difficult parts of this project. Luckily, only addr: key part had problems.

1. Street Name Issue

There were a lot of inconsistencies. Here are few examples:

- Grose Street Deakin (Deakin is suburb)
- Tez Automotive (Not a street)
- 12 (just digit nothing really. This can be a house number instead)
- Waramanga Shopping Centre (Name of the shopping centre)
- Thoji Street Unit 4 (Unit 4 should be omitted)

First, I tried to use regular expressions to make all street name consistent but I concluded that it is not possible to simply come up with one regular expression to cover all possibilities. After extracting unique street names, I realized that the street type is fairly limited (i.e. in total there are only 24 possibilities). Thus, I made an exhaustive list of expected street types. Regarding the abbreviations of street types (e.g. St. -> Street). I have mapped them and

updated the name so that every name has full street type.

Second, there were few instances where two streets were given within one attr:street key. In this case, I have looked them up and have chosen the main street.

2. Postcode Issue

There were only two errors. Therefore, I have just explicitly cleaned them (i.e. from 'Yarralumla ACT 2600' to '2600' and from 'ACT 2604' to '2604')

3. City Issue

There were only five errors. Therefore, I have just explicitly cleaned them (refer to code).

4. Housenumber Issue

The only concern I had was some of the entries used 'ndash' symbol (which is similar to asci minus -) Therefore, I had to replace ndash to minus in order to eliminate encoding error. However, in Python 2.7, we cannot replace ndash to minus using replace function as ndash is invalid. As such, I encoded the Unicode character to UTF-8 and then replace \xe2\x80\x93 with -.

Each JSON data has the following data structure:

```
{
  "created": {
    "changeset": "12319201",
    "user": "OSMF Redaction Account",
    "version": "3",
    "uid": "722137",
    "timestamp": "2012-07-19T10:42:11Z"
  },
  "type": "node",
  "pos": [
    -35.3410089,
    149.1023313
  ],
  "address": {
    "street": "Kitchener Street",
    "housenumber": "117"
  },
  "id": "330149955"
}
```

3. Data Overview

File Sizes

- ◆ canberra_australia.osm (102,038 KB)
- ◆ canberra_australia.osm.json (157,490 KB)

I have created a database called „openstreet" and created a collection called „canberra" and then I have imported canberra_australia.osm.json data and performed some basic queries:

General Information regarding the area selected

Total documents inserted:

db.canberra.find().count() → **548,585 documents**

Total document with the type of node:

db.canberra.find({"type":"node"}).count() → **500,387 documents**

Total documents with the type of way:

db.canberra.find({"type":"way"}).count() → **48,186 documents**

The top –most contributed user:

db.canberra.aggregate([{"$group" : {"_id" : "$created.user", "count" : {"$sum" : 1} } },

                    {"$sort" : {"count" : -1}},

                    {"$limit" : 1}]) → **numbfew (90681 documents)**

The number of unique users:

len(db.canberra.distinct("created.user")) → **492**
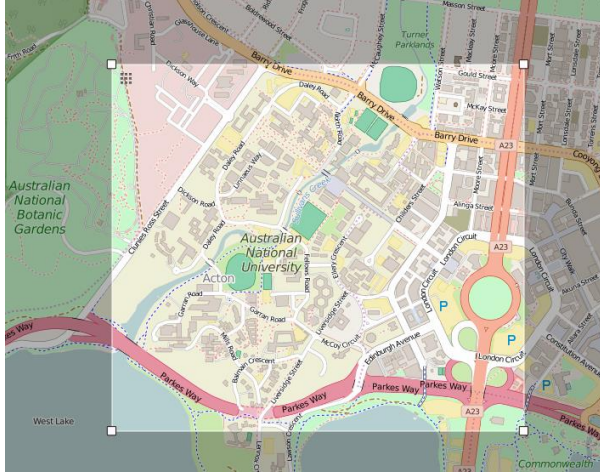
The number of unique streets:

len(db.canberra.distinct("address.street")) → **318**

The number of unique postcodes:

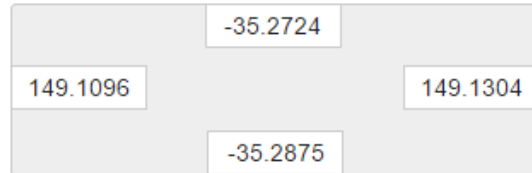len(db.canberra.distinct("address.postcode")) → **29**

Specific Information

I wanted to see nodes or ways in the area selected below:

The precise position for above area is

Although I could have used the Overpass API to download a custom square area, I have just used the MongoDB query using PyMongo to extract the desired data.

```
map_query = {    "pos.0" : {"$gte": -35.2875, "$lte":-35.2724},

                 "pos.1" : {"$gte": 149.1096, "$lte": 149.1304} }

map_projection = {"_id" : 0, "created.uid" : 1}

map_contributed_users = db.canberra.find(map_query, map_projection)
```

The number of nodes and ways within this area:

```
count_specific_area_nodes = map_contributed_users.count()  →  8002
```

The number of users who contributed in giving information about nodes and ways within this area:

```
unique_users = set()

for user in map_contributed_users:

        unique_users.add(user['created']['uid'])

len(unique_users)  →  89
```

I have lived in Yarralumla and Braddon so I would like to find out information regarding things in these suburbs. The postcode for Yarralumla and Braddon are 2600 and 2612 respectively.

```
postcode_query = {"type" : "node", "address.postcode" : {"$in" : ['2600', '2612']}}

postcode_projection = {"_id" : 0, "created.uid" : 1}

specific_area_nodes = db.canberra.find(postcode_query, postcode_projection)
```

The number of users who contributed to giving information about Yarralumla and Braddon:

```
unique_users = set()

for user in specific_area_nodes:

    unique_users.add(user['created']['uid'])

len(unique_users)  →  5
```

The number of nodes and ways in 2600 and 2612:

```
count_specific_area_nodes = specific_area_nodes.count()  →  7
```

Top 3 biggest religion in Canberra:

```
list(db.canberra.aggregate([{"$match" : {"amenity" : {"$exists": 1} ,

                             "amenity" : "place_of_worship"}},

                            {"$group" : {"_id" : "$religion", "count": {"$sum" : 1}}},

                            {"$sort" : {"count" : -1}},

                            {"$limit" : 3}]))  →  Christian (109), Buddhist (4), Islam (3)
```

The 10 amenities in Canberra:

```
list(db.canberra.aggregate([{"$match" : {"amenity" : {"$exists" : 1}}},

                 {"$group" : {"_id" : "$amenity", "count" : {"$sum" : 1}}},

                 {"$sort" : {"count" : -1}},

                 {"$limit" : 10}]))   →  Parking (1154), School (211), Restaurant (204), Drinking Water (192),
```
Toilets (168), Café (158), Bench (145), Bicycle Parking (131), Fast Food (126), Place of Worship (121)

The top 5 streets in Canberra:

```
list(db.canberra.aggregate([{"$match" : {"address.street" : {"$exists" : 1}}},

                  {"$group" : {"_id" : "$address.street", "count" : {"$sum" : 1}}},

                  {"$sort" : {"count" : -1}}, {"$limit" : 5}]))   →  Enderby Street (49), Beasley Street (37), Delma
```
View (34), The Valley Avenue (30), Henry Kendall Street (26)

This result does not match the reality because Enderby Street in Mawson is pretty secluded place. I would now like to question the 'quality' of data.

The question is: **is the data outdated?**

In answering this question, I initially searched for the most top 3 frequent date in which these nodes or ways were created and 271 data were recorded in June 2007, 174 in October 2007 and 113 in May 2008. With the data, it is not sufficient to say the whole data are outdated because compared to the total number of documents it is almost infinitesimal.

I will regard the 'up-to-date' data as anything that is published after 2013.

The number of nodes and ways created between 2014 and 2016 (20 February):

time_query = { "created.timestamp" : {"$gte": '2014-01-01T00:00:00Z', "$lte": '2016-02-20T00:00:00Z'}}

time_projection = {"_id" : 0, "created.uid" : 1}

specific_time_nodes = db.canberra.find(time_query, time_projection).count() → **158528 (28.9%)**

The number of nodes and ways created before 2014:

time_query = { "created.timestamp" : {"$lt": '2014-01-01T00:00:00Z'}}

time_projection = {"_id" : 0, "created.uid" : 1}

specific_time_nodes = db.canberra.find(time_query, time_projection).count() → **390057 (71.1%)**

Therefore, the data itself are not really reflecting the current situation in my opinion, even though one may claim that I am being too strict. In order to improve the data, people need to induce people in Canberra to update the OpenStreetMap data.

When I played games, I invested my time primarily to improve the ranking of my character within that game. By applying the same principle to this map contribution 'game', I believe that the data will become more robust. For example, OpenStreetMap and the ACT State Government can collaborate by giving good incentives to users who contribute the most. For example, when I was working as an ambassador at Google, I tried to improve the Google Map by hosting Google Map Competition and the person who contributed the most received a Nexus smartphone (in fact he contributed a lot!). Without giving a prize, I do not think that people will participate. The main reason for this is that ordinary people prefer using Google Map instead of OpenStreetMap. As it is almost good for the state government to invest in this area, I believe that this can be a good opportunity for both parties.

Conclusion

Despite the **wrangling** process, as investigated above, the data (i.e. XML OML) is itself **incomplete** and **outdated** to some extent. As such, the exploration above may not necessarily match the reality. For other metrics for assessing data quality, the following points are my conclusion:

- The data is **valid** because it does conform to the schema.

- The data is **consistent**.

- The data itself is **accurate**.

In order to improve the '**completeness**' of data, there must be some other forms of incentives to attract users from Canberra to more actively contribute to the map.