OpenStreetMap Sample Project Data Wrangling with MongoDB

Map area: Vancouver, BC, Canada
https://www.openstreetmap.org/relation/1852574#map=12/49.2573/-123.1241

I chose the Vancouver BC area as this is where I reside. The ability to lean on my domain knowledge of this locale proved helpful as I learned how to wrangle data in this project. This area consists of 115 km^2 and just over 600,000 people. It is the third most populous city in Canada, but the most densely populated Canadian municipality and fourth in North America, behind New York, San Francisco, and Mexico City. Vancouver City is also known for being one of the most ethnically and linguistically diverse city in Canada.

Problems encountered in your map

The data was surprisingly clean. I expected to see a lot more inconsistencies, but with the exception of minor typos, street abbreviations, and field names with colons the data seemed ready to be imported to MongoDB.

Street name typos While very low in occurrence, this type of inconsistency would have gone unnoticed had I not have intimate knowledge of the map area. I added these to my mapping dictionary to correct for. An example of such a mistake is, `Jervis => Jarvis`.

Street name abbreviations While auditing the dataset, I noticed instances where street names were in various formats while contextually being similar. These were also accounted for in the mapping dictionary to standardize the address. The following is a snippet of the mapping dictionary where four variations of `Highway` were corrected for:

```
"Hwy": "Highway", "Hwy.": "Highway", "HWY": "Highway", "HIGHWAY": "Highway"
```

Field names with colon During the data audit, multiple instances of field names with double colons were identified using regular expression. It seemed unlikely that the lower_colon regex would catch all colon instances, so I used a slight modification to remove the first colon.

```
double_colon = re.compile(r'^([a-z]|_)*:([a-z]|_)*:([a-z]|_)*$')
```

Additional note Manually sifting through the json export, I noticed that the phone numbers were in different formats. While, this was not explored in this project, the `phonenumbers` python library could have been used to scrub through the phone numbers.

A further point to note about the cleanliness of this dataset was the absence of problem characters.

Overview of the Data

File sizes `vancouver_canada.osm: 154.5 MB` `vancouver_canada.osm.json: 179.2 MB`

This section contains basic statistic about the dataset and the MongoDB queries used on them. The following are snippets of queries found in `query.py`:

Number of nodes

```
db.openstreetmap.find({"type": "node"}).count()

3390650
```

Number of ways

```
db.openstreetmap.find({"type": "way"}).count()

685670
```

Number of total documents

```
db.openstreetmap.find().count()

4076435
```

It seems that there are 115 more total documents that the sum of nodes and ways. This is discussed further below.

Top contributer

```
db.openstreetmap.aggregate([{
        '$group': {
            '_id': '$created.user',
            'count': {
                '$sum': 1
            }
        }
    }, {
        '$sort': {
            'count': -1
        }
    }, {
        '$limit': 1
    }])

[{u'_id': u'keithonearth', u'count': 965035}]
```

The user keithonearth is very active on OpenStreetMap and is a huge contributor to the British Columbia province in general.

Number of unique users contributing

```
db.openstreetmap.aggregate([{
        '$group': {
            '_id': '$created.user',
            'count': {
                '$sum': 1
            }
        }
    }, {
        '$group': {
            '_id': '$count',
            'num_users': {
                '$sum': 1
            }
        }
    }, {
        '$sort': {
            '_id': 1
        }
    }, {
        '$limit': 1
    }])

[{u'_id': 5, u'num_users': 156}]
```

Unsurprisingly, there are a large number of unique contributors.

Top 10 building types

```
db.openstreetmap.aggregate([{
        '$match': {
            'building': {
                '$exists': 1
            }
        }
    }, {
        '$group': {
            '_id': '$building',
            'count': {
                '$sum': 1
            }
        }
    }, {
        '$sort': {
            'count': -1
        }
    }, {
        '$limit': 10
    }])

[{u'_id': u'yes', u'count': 543035},
 {u'_id': u'residential', u'count': 9060},
 {u'_id': u'apartments', u'count': 5980},
 {u'_id': u'house', u'count': 3525},
 {u'_id': u'commercial', u'count': 1025},
 {u'_id': u'entrance', u'count': 565},
 {u'_id': u'roof', u'count': 545},
 {u'_id': u'school', u'count': 445},
 {u'_id': u'retail', u'count': 400},
 {u'_id': u'greenhouse', u'count': 175}]
```

There were an odd number of `yes` entries for buildings. As I investigated in the json file, they seems to be just an *other* category. So I ignored adjusting it to use the amenity name instead.

Top 5 amenities

```
db.openstreetmap.aggregate([{
        '$match': {
            'amenity': {
                '$exists': 1
            }
        }
    }, {
        '$group': {
            '_id': '$amenity',
            'count': {
                '$sum':1
            }
        }
    }, {
        '$sort': {
            'count':-1
        }
    }, {
        '$limit': 5
    }])

[{u'_id': u'parking', u'count': 4545},
 {u'_id': u'restaurant', u'count': 2460},
 {u'_id': u'bench', u'count': 2415},
 {u'_id': u'cafe', u'count': 1410},
 {u'_id': u'fast_food', u'count': 1035}]
```

This was expected - Vancouverites do love being healthy and the outdoors, which explains the high number of benches and restaurants compared to fast food.

Top 3 cafe

```
db.openstreetmap.aggregate([{
        '$match': {
            'amenity':
                'cafe'
        }
    }, {
        '$group': {
            '_id': '$name',
            'count': {
                '$sum':1
            }
        }
    }, {
        '$sort': {
            'count':-1
        }
    }, {
        '$limit': 3
    }])

[{u'_id': u'Starbucks', u'count': 255},
 {u'_id': u'Starbucks Coffee', u'count': 115},
 {u'_id': u'Tim Hortons', u'count': 65}]
```

The high number of cafes peaked my interest, hence this query. Though it comes to little surprise that Starbucks is everywhere in Vancouver compared to the very Canadian, Tim Hortons. On a side note, the amenities names needs to be scrubbed.

Top 2 fast food

```
db.openstreetmap.aggregate([{
          '$match': {
              'amenity': {
                  '$exists': 1
              },
              'amenity':
                  'fast_food'
          }
      }, {
          '$group': {
              '_id': '$name',
              'count': {
                  '$sum':1
              }
          }
      }, {
          '$sort': {
              'count':-1
          }
      }, {
          '$limit': 2
      }])

[{u'_id': u'Subway', u'count': 165}, {u'_id': u"McDonald's", u'count': 100}]
```

This query was out of plain curiosity and is inline with the claim that Vancouverites are generally health conscious. Thus the prominence of Subway over McDonald's.

Other ideas about the datasets

The data validation and queries performed were primarily on *node* and *way* tags with a focus on address locations. There were a significant number of *nd* reference tags that were ignored. Depending on the application of this dataset for future analysis, it would be worthwhile to dive into this more and uncover the different tags and see if there are any discrepancies. As noted above, there were 115 more total documents than the sum of nodes and ways. It is a bit perplexing that this would occur; perphaps, the data already contained a `type` field that was interfering with the script.

It would also be interesting to dive into other areas of the data to assess cleanliness, such as postal codes, phone numbers and coordinate data. The phone numbers were fairly consistent but did contain some inconsistencies that could have been improved. A few years back Vancouver's telecomm providers started offering the *778* area code in addition to *604*, and recently, there were talks of offering another area code to supply demand. This dataset could be used to observe the distribution of generally available phone numbers. Perhaps, businesses could be given different area codes to easily distinguish themselves from an individual caller.

Lastly, there are ample more opportunities to clean this data beyond this cursory excercise as seen in the *Top 3 cafe* query. Nevertheless, the dataset is fairly clean as noted above. OpenStreepMap is crowdsourced and is therefore dependant on users' input, which could lead to incomplete or outdated information. It would be interesting to compare these results with Google Maps' API.

References

- Parsing large XML files
- Python XML documentation
- Vancouver wiki
- Installing MongoDB
- Importing JSON into MongoDB
- Discusison Forum