OpenStreetMap Project
Data Wrangling with MongoDB
Bhavya Garg
Map Area: DC-Baltimore

INTRODUCTION

I chose this particular area because it is my neighbor city. This domain knowledge was very helpful as I could clean the data with some context instead of doing guesswork or doing extensive research on another city. I chose the DC-Baltimore, instead of the Richmond city from where i belong to because of the limited data that has been populated so far for Richmond on OpenStreetMaps and I needed to analyze a dataset larger than 50 MB for this project. Below is the list of .py files which shows my full work of this project.

Section 1: Problems Encountered in the Map

Unexpected Tags

mapparser.py was used to count occurrences of each tag, with a result:

- member : 4765
- nd : 488617
- node : 418014
- osm : 1
- relation : 335
- tag : 251958
- way : 40621

Then in Users.py, I find out the number of unique users who have contributed to the map in this particular area!

Street type abbreviations standardized

I changed many street types to a non-abbreviated form after discovering these issues in the audit (before importing into MongoDB). For example, below are the 3 types of street which I standardized by adding them to the mapping dictionary in the audit file (Audit.py).

'St': 'Street', 'St.': 'Street', 'ST': 'Street'

Other than Street below are the list of changes i made to the variable mapping :

- "Ave": "Avenue",
- "NW": "North West",
- "E": "East",
- "E.":"East",

- "W.":"West",
- "N.":"North",
- "S.":"South",
- "West":"West",
- "Northeast":"North East",
- "SE" :"South East",
- "Blvd": "Boulevard",
- "Blvd.":"Boulevard",
- "CIrcle":"Circle",
- "Hwy": "Highway",
- "Rd.": "Road",
- "St.":"Street",
- "St" :"Street",
- "ST": "Street",
- "Rdt" :"Road",
- "Ln" : "Lane",
- "SW" : "South West",
- "Ct": "Court",
- "Cir": "Circle",
- "AVE": "Avenue"

Section 2: Data Overview

This section contains basic statistics about the dataset and the MongoDB queries used to gather them. The queries are included in query.py.

File sizes:

```
> collection.dataSize()
106068294
```

- sample.osm: 90.3 MB
- sample.osm.json: 97.6 MB

Number of documents:

```
> collection.count()
449119
```

Number of nodes and ways:

```
> collection.find({'type':'node'}).count()
418014
```

```
> collection.find({'type':'way'}).count()
31105
```

Number of unique users:

```
collection.distinct("created.user").length
1176
```

Top 10 contributing user:

```
> collection.aggregate([{"$group":{ "_id":"$created.user",
"count":{"$sum":1}}},{"$sort":{"count":-1}},{"$limit":10}]).pretty()

{ "_id" : "woodpeck_fixbot", "count" : 94731 }
{ "_id" : "asciiphil", "count" : 86544 }
{ "_id" : "aude", "count" : 64797 }
{ "_id" : "mdroads", "count" : 15728 }
{ "_id" : "kriscarle", "count" : 15031 }
{ "_id" : "wonderchook", "count" : 13585 }
{ "_id" : "JoshD", "count" : 10824 }
{ "_id" : "RJCorazza", "count" : 10454 }
{ "_id" : "Evanator", "count" : 9666 }
{ "_id" : "westendguy", "count" : 9204 }
```

Number of users contributing only once:

```
> collection.aggregate([{
... ...             '$group': {
... ...                 '_id': '$created.user',
... ...                 'count': {
... ...                     '$sum': 1
... ...                 }
... ...             }
... ...         }, {
... ...             '$group': {
... ...                 '_id': '$count',
... ...                 'num_users': {
... ...                     '$sum': 1
... ...                 }
... ...             }
... ...         }, {
... ...             '$sort': {
... ...                 '_id': 1
... ...             }
... ...         }, {
... ...             '$limit': 1
... ...         }])
{ "_id" : 1, "num_users" : 320 }
```

5 Most common street address:

```
> collection.aggregate([{
... ...             '$match': {
... ...                 'address.street': {
... ...                     '$exists': 1
... ...                 }
... ...             }
... ...         }, {
... ...             '$group': {
... ...                 '_id': '$address.street',
... ...                 'count': {
... ...                     '$sum': 1
... ...                 }
... ...             }
... ...         }, {
... ...             '$sort': {
... ...                 'count': -1
... ...             }
... ...         }, {
... ...             '$limit': 5
... ...         }])
{ "_id" : "Connecticut Avenue North West", "count" : 45 }
{ "_id" : "13th Street North West", "count" : 45 }
{ "_id" : "Q Street North West", "count" : 41 }
{ "_id" : "16th Street North West", "count" : 41 }
{ "_id" : "Georgia Avenue North West", "count" : 41 }
```

Sort cities by count, descending

```
> collection.aggregate([{
... ... ...             '$match': {
... ... ...                 'address.city': {
... ... ...                     '$exists': 1
... ... ...                 }
... ... ...             }
... ... ...         }, {
... ... ...             '$group': {
... ... ...                 '_id': '$address.city',
... ... ...                 'count': {
... ... ...                     '$sum': 1
... ... ...                 }
... ... ...             }
... ... ...         }, {
... ... ...             '$sort': {
... ... ...                 'count': -1
... ... ...             }
... ... ...         }])
{ "_id" : "Leesburg", "count" : 137 }
{ "_id" : "Sterling", "count" : 15 }
{ "_id" : "Frederick", "count" : 13 }
{ "_id" : "Alexandria", "count" : 9 }
{ "_id" : "Baltimore", "count" : 9 }
{ "_id" : "Towson", "count" : 8 }
{ "_id" : "Silver Spring", "count" : 7 }
{ "_id" : "Pikesville", "count" : 5 }
{ "_id" : "Arlington", "count" : 5 }
{ "_id" : "Columbia", "count" : 5 }
{ "_id" : "Cockeysville", "count" : 5 }
{ "_id" : "Churchton", "count" : 4 }
{ "_id" : "Washington", "count" : 4 }
{ "_id" : "Mertford Street", "count" : 4 }
{ "_id" : "Deale", "count" : 3 }
{ "_id" : "Springfield", "count" : 3 }
{ "_id" : "Westminster", "count" : 3 }
{ "_id" : "Vienna", "count" : 3 }
{ "_id" : "Kensington", "count" : 3 }
{ "_id" : "DC", "count" : 2 }
```

Nodes without addresses:

```
> collection.aggregate([{
... ...             '$match': {
... ...                 'type': 'node',
... ...                 'address': {
... ...                     '$exists': 0
... ...                 }
... ...             }
... ...         }, {
... ...             '$group': {
... ...                 '_id': 'Nodes without addresses',
... ...                 'count': {
... ...                     '$sum': 1
... ...                 }
... ...             }
... ...         }])
{ "_id" : "Nodes without addresses", "count" : 416566 }
```

Sort postcodes by count, descending

```
> collection.aggregate([{
... ... ...             '$match': {
... ... ...                 'address.postcode': {
... ... ...                     '$exists': 1
... ... ...                 }
... ... ...             }
... ... ...         }, {
... ... ...             '$group': {
... ... ...                 '_id': '$address.postcode',
... ... ...                 'count': {
... ... ...                     '$sum': 1
... ... ...                 }
... ... ...             }
... ... ...         }, {
```

```
... ... ...             '$sort': {
... ... ...                 'count': -1
... ... ...             }
... ... ...         }])
{ "_id" : "20176", "count" : 106 }
{ "_id" : "21043", "count" : 42 }
{ "_id" : "20175", "count" : 32 }
{ "_id" : "21093", "count" : 28 }
{ "_id" : "21228", "count" : 21 }
{ "_id" : "21204", "count" : 17 }
{ "_id" : "20164", "count" : 14 }
{ "_id" : "21703", "count" : 13 }
{ "_id" : "21045", "count" : 11 }
{ "_id" : "21208", "count" : 9 }
{ "_id" : "20878", "count" : 8 }
{ "_id" : "21042", "count" : 8 }
{ "_id" : "21044", "count" : 7 }
{ "_id" : "22314", "count" : 6 }
{ "_id" : "21286", "count" : 6 }
{ "_id" : "21030", "count" : 6 }
{ "_id" : "21046", "count" : 6 }
{ "_id" : "21061", "count" : 5 }
{ "_id" : "20701", "count" : 5 }
{ "_id" : "21224", "count" : 4 }
```

Conclusion

There are still several opportunities for cleaning and validation that I left unexplored. Of note, the data set is populated only from one source: OpenStreetMaps. While this crowdsourced repository pulls from multiple sources, some of data is potentially outdated. It would have been an interesting exercise to pull missing information from Google Map API,since every node has latitude-longitude coordinates.
The benefits in implementing the improvement is now we have a Street type abbreviations in a standardized manner . As of now i don't see any problems in implementing the improvement.