OpenStreetMap Project
Data Wrangling with MongoDB

Matthew Bellisimo

Map Area: Columbus, OH, United States

1. Problems Encountered in the Map

After downloading a small sample size of the Columbus area and running it against a python audit script, I noticed a number of problems with the data. The most significant of which were:

- Over-abbreviated street names (“'S Court St'”)
- Incomplete Address Data
- Some state and postal codes were logged Incorrectly (Ohio vs. OH ,'postcode': 'Sunbury, OH 43074')
- Shaped JSON was not standardized.

Over-abbreviated Street Names

As I was writing my shaping script in python, I would periodically test to make sure the code was returning the correct values. Through this process I realized that a large number of street names were overly abbreviated. I used a custom dictionary to update all the problematic address substrings, such that 'S Court St' became ‘South Court Street’.

Incorrectly logged State and Postal Codes

The same testing process as mentioned previously led me to discover some other issues. Many postcode and state values were improperly logged. For instance, a number of postcode values included street and state, and a number of state values were not abbreviated to the two-letter standard. Luckily these were easy fixes. For postcode, I used regex to take the entire sequence of numbers at the end of the string, and then use that to replace the original value. For the state code, I chose to replace any values that started with the letters ‘Oh’ and were larger than 2 characters in length, with the abbreviated street code of ‘OH’. This way all the values would be standardized.

The update process here, involved taking the entire sequence of numbers at the end of the problematic strings.

So

'postcode’: 'Sunbury, OH 43074'

Became 'postcode’:

43074

Incomplete Address Data

Unfortunately, there were many blank address values in the Columbus Dataset. To make matters slightly more frustrating, in many cases where an address value did exist, it was only partially filled out.

Ex:

```
{‘street’: 'Livingston Avenue', 'housenumber': '5000'}
{‘city’: 'Worthington', 'street': 'North High Street', 'housenumber': '7172'}
```

There wasn’t an easy fix to this issue, due to the fact that in order to complete the address data set, I would have to infer the non-existing values from the ones that were already imputed. As I believed this task outside the scope of this project I decided to leave it alone for now.

2. Data Overview

This section contains basic statistics about the dataset and the MongoDB queries used to gather them.

File sizes

columbus_ohio.osm ....170.59 MB
columbus_ohio.osm.json....... 163.45 MB

#Number of documents

```
 > db.ColumbusOH.find().count()
 752045
```

# Number of nodes

```
>db.ColumbusOH.find({'type':'node'}).count() 7
>52045
```

#Number of ways

```
>db.ColumbusOH.find({'type':’way’}).count()
>   0
```

#Number of unique users

```
>db.ColumbusOH.distinct('created.user').length
>625
```

#Top 1 contributing user

```
        > db. ColumbusOH.aggregate([{"$group":{"_id":"$created.user",
"count":{"$sum":1}}},{'$sort':{'count':-1}},{'$limit':1

        { "_id" : "woodpeck_fixbot", "count" : 419023 }
```

# Number of users appearing only once (having 1 post)

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

{ "_id" : 1, "num_users" : 115 }
```

3. Additional Ideas

Total Docs: 752045

- Top user contribution percentage (‘woodpeck_fixbot’) - 55.72%
- Combined top 5 users' contribution (‘woodpeck_fixbot’,’Vid the Kid’,’TimC’,’TIGERcnl’,’Minh Nguyen’) - 79.90%
- Combined Top 10 users contribution – 87.43%
- Combined number of users making up only 5% of posts – 584 (about 93.6% of all users)

Contributor statistics indicate potential gamification

The contribution of users seems incredibly skewed. While some of this can be explained by automated map editing, only 1 out of 10 of the top 10 contributors include the word ‘bot’ in their user names, which implies the remaining 9 users contributed to OSM manually. Given that the top 10 contributors make up over 87% of the total contribution, and that the 9 manual users out of these top 10 make up 31.7% that number, it probably makes sense that there is some serious gamification going on. This isn’t surprising, and from my experience is more or less the standard with crowd driven projects.

Additional data exploration using MongoDB queries

#Top 10 appearing amenities

```
>db.ColumbusOH.aggregate([{'$match':{'amenity':{'$exists':1}}},{'$group':{'_id':'$a
menity','count':{'$sum':1}}},{'$sort':{'count':-1}},{'$limit':10}])

{ "_id" : "place_of_worship", "count" : 1202 }
{ "_id" : "school", "count" : 829 }
{ "_id" : "grave_yard", "count" : 477 }
{ "_id" : "restaurant", "count" : 310 }
{ "_id" : "post_office", "count" : 203 }
{ "_id" : "fast_food", "count" : 185 }
{ "_id" : "fuel", "count" : 115 }
{ "_id" : "parking", "count" : 75 }
{ "_id" : "library", "count" : 59 }
{ "_id" : "fire_station", "count" : 56 }
```

# Largest religion

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

{ "_id" : "christian", "count" : 1141 }
```

#Number of Restaurants

```
>       db.ColumbusOH.find({'amenity':{'$exists':1},'amenity':'restaurant'}).count()
310
```

#Most Popular Cuisine

```
>db.ColumbusOH.aggregate([{'$match':{'amenity':{'$exists':1},'amenity':'restaurant'
}},{'$group':{'_id':'$cuisine','count':{'$sum':1}}},{'$sort':{'count':1}},{'$limit':10}])

{ "_id" : "spanish", "count" : 1 }
{ "_id" : "pasta", "count" : 1 }
{ "_id" : "caribbean", "count" : 1 }
{ "_id" : "Vegan", "count" : 1 }
{ "_id" : "Japanese_Sushi_Hibachi", "count" : 1 }
{ "_id" : "donuts", "count" : 1 }
{ "_id" : "subs", "count" : 1 }
{ "_id" : "Vietnamese", "count" : 1 }
{ "_id" : "Asian", "count" : 1 }
{ "_id" : "burger,_shakes", "count" : 1 }
```

Conclusion

Columbus OH is a city of well over 800,000 people, so it's really interesting to me that a total of 625 user have been responsible for mapping almost the entirety of the city. Granted that half of these contributions were done via an automated bot, it's still surprising that only 9 users contributed 31.7% of 752,045 documents. Of course the downside of this is human error. Take for example the list two Mongo Queries. Over 310 restaurants were logged, but if you try to break them down by type to rank them, you notice that there is no standard for consistency. It's very hard to believe that there is 1 Japanese restaurant in Columbus; it's very easy to believe that there are many Asian themed restaurants, all logged slightly differently.

I can think of a couple of ways to help improve the data set. First, I believe that if there were pre-defined category options, for instance with types of restaurants, that could go a long way towards making the information more relevant. I actually think from the end users standpoint, it actually hurts that the information is so specific, as it makes the data less readable. This process could probably be done programmatically by the application, for instance if there was an existing mapping dictionary that grouped key words in the type description with categories that they were commonly associated with. Another thing I can think of is to require all the components of address data to be mandatory fields required for submission. This way there wouldn't be partial entries, such as only have street name and house number. Given that the whole purpose of OSM is to provide an accurate open source mapping solution, this doesn't seem like an unreasonable demand.