P3 – OpenStreetMap Project: Data Wrangling with MongoDB

Map Area: Lexington, KY

James Wheaton

December 14th, 2015

Contents

1 Problems Encountered in the Map

After downloading the Map Zen dataset for Lexington, Kentucky, I audited the data using some techniques from Problem Set 6. The following data inconsistencies were noticed:

- Abbreviated street names
- Inconsistent capitalization
- Data entry problems with ZIP codes
- Data entry problems with street name

1.1 Abbreviated Street Names

Using the regular expression to detect the street type, I audited the street names to find abbreviated or irregular street types. I used the `expected` array of street types from Problem Set 6, and then updated it with more street types that are present in the Lexington area. I cross-referenced strange street names like "Esplanade" and "Headley Green" with Google Maps. A subsequent run returned the following problems:

```
{'120': set(['Versailles Road #120']),
 '168': set(['West Lowry Lane #168']),
 '600': set(['East Vine Street, Suite 600']),
 'Ave': set(['524 Angliana Ave']),
 'Ave.': set(['E. Euclid Ave.']),
 'Blvd': set(["Man o' War Blvd", 'Polo Club Blvd']),
 'Blvd.': set(['Martin Luther King Blvd.']),
 'Courth': set(['Deltino Courth']),
 'Ct': set(['Belfair Ct']),
 'DR': set(['NEEDLERUSH DR']),
 'Dr': set(['Beaumont Circle Dr', 'Nichols Park Dr', 'Parkway Dr']),
 'Dr.': set(['Southland Dr.', 'Tates Creek Centre Dr.', 'Walden Dr.']),
 'Ln': set(['Norman Ln', 'W Lowry Ln', 'W. Lowry Ln']),
 'NE': set(['New Circle Road NE']),
 'Rd': set(['E New Circle Rd', 'East Reynolds Rd', ...]),
 'Rd.': set(['Redding Rd.', 'Tates Creek Rd.', 'W. New Circle Rd.']),
 'St': set(['E Main St', 'E. Main St', 'Jefferson St']),
 'St.': set(['N. Limestone St.', 'S Limestone St.']),
 'Ter': set(['Columbia Ter']),
 'broadway': set(['North broadway']),
 'drive': set(['cooper drive'])}
```

The apartment/suite numbers are fine and will be fixed. One spelling error was found: "Courth" should be "Court." The abbreviations, including the cardinal direction abbreviations, will be replaced with their full word equivalents.

1.2 Inconsistent Capitalization

From the above street name audit, street names were found which are in all caps or all lowercase. An updating function will be run over the street names to properly capitalize them.

With the following query, a Mongo query was ran to visually inspect all the street names and find any other problems:

```
db.lexington_orig.aggregate([{$group: { _id: "$address.street"}}])
```

Some examples of inconsistent capitalization are: "south Broadway," "meijer Way," and "west Main Street."

1.3 Data entry problems with ZIP codes

A Mongo query was ran to inspect the ZIP (postal) codes for this area.

```
db.lexington_orig.aggregate([{$match: {"address":{$exists:1}}},
{$group: { _id: "$address.postcode", count: {$sum: 1}}}, {$sort:
{ count: 1}}])
```

The results show that there is one ZIP code entered as "KY" (Kentucky, the state) and 297 entries with no ZIP code (`null`). Adding the ZIP codes is out of the scope of this project: it would require checking the address against USPS or Google Maps API.

There are ZIP codes which include the extra four numbers. We will ignore these and assume they are correct. Perhaps these could be split into another field for better consistency and ease of querying.

1.4 Data entry problems with street name

One error was found with the street names where the house number was included in `addr:street`, for "524 Angliana Avenue." This was fixed by removing "524 " from the street name and moving it to `housenumber`.

2 Data Overview

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

2.1 File Sizes

```
lexington_kentucky.osm ........  66 MB
lexington_kentucky.osm.json ...  72 MB
```

2.2 Number of documents

```
> db.lexington_orig.find().count()
346737
```

2.3 Number of nodes

```
> db.lexington_orig.find({''type'':''node''}).count()
320817
```

2.4 Number of ways

```
> db.lexington_orig.find({''type'':''way''}).count()
25920
```

2.5 Number of unique users

```
> db.lexington_orig.distinct(''created.user'').length
537
```

2.6 Top 1 contributing user

```
> db.lexington_orig.aggregate([{$group: { _id: "$created.user", count:
{ $sum: 1 }}}, {$sort: { "count": -1}}, {$limit: 1}])
[ { "_id" : "woodpeck_fixbot", "count" : 175583 } ]
```

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

```
> db.lexington_orig.aggregate([{$group: { _id: "$created.user", count:
{ $sum: 1 }}}, {$group: {_id: "$count", num_users: {$sum:1}}},
{$sort: { _id: 1}}, {$limit: 1}])
[ { "_id" : 1, "num_users" : 90 } ]
# ‘‘_id’’ represents postcount
```

3 Additional Ideas

3.1 Top 10 appearing amenities

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

[ { "_id" : "parking", "count" : 546 },
{ "_id" : "restaurant", "count" : 376 },
{ "_id" : "place_of_worship", "count" : 171 },
{ "_id" : "bicycle_parking", "count" : 129 },
{ "_id" : "fast_food", "count" : 115 },
{ "_id" : "school", "count" : 110 },
{ "_id" : "fuel", "count" : 88 },
{ "_id" : "university", "count" : 82 },
{ "_id" : "bank", "count" : 69 },
```

```
{ "_id" :  "cafe", "count" :  40 } ]
```

An interesting insight from this data: this city seems to be bicycle-friendly, judging by the number of bicycle parking places.

3.2 Top 10 cuisines available

```
> db.lexington.aggregate([{$match:  {"cuisine":{$exists:1}}}, {$group:
{_id:  "$cuisine", count:  {$sum:1}}}, {$sort:  {count:  -1}}, {$limit:
10}])

[ { "_id" :  "burger", "count" :  39 },
{ "_id" :  "american", "count" :  38 },
{ "_id" :  "sandwich", "count" :  25 },
{ "_id" :  "mexican", "count" :  23 },
{ "_id" :  "pizza", "count" :  22 },
{ "_id" :  "coffee_shop", "count" :  18 },
{ "_id" :  "regional", "count" :  18 },
{ "_id" :  "ice_cream", "count" :  18 },
{ "_id" :  "italian", "count" :  15 },
{ "_id" :  "chinese", "count" :  13 } ]
```

3.3 Banks available

```
> db.lexington.aggregate([{$match:  {"amenity":"bank"}}, {$group:  {_id:
"$name", count:  {$sum:1}}}, {$sort:  {count:  -1}}])

[ { "_id" :  null, "count" :  16 },
{ "_id" :  "Fifth Third Bank", "count" :  6 },
{ "_id" :  "Chase", "count" :  4 },
{ "_id" :  "PNC Bank", "count" :  3 },
{ "_id" :  "Chase Bank", "count" :  3 },
{ "_id" :  "Central Bank", "count" :  3 },
{ "_id" :  "Whitaker Bank", "count" :  3 },
{ "_id" :  "Kentucky Bank", "count" :  2 },
{ "_id" :  "Farmer's Bank", "count" :  2 },
{ "_id" :  "US Bank", "count" :  2 },
{ "_id" :  "5/3 Bank ATM", "count" :  2 },
{ "_id" :  "PNC", "count" :  1 },
{ "_id" :  "United Bank", "count" :  1 },
{ "_id" :  "Branch Banking and Trust Company", "count" :  1 },
{ "_id" :  "Republic Bank", "count" :  1 },
{ "_id" :  "First State Financial", "count" :  1 },
{ "_id" :  "First State Financial Bank", "count" :  1 },
{ "_id" :  "American Founders Bank", "count" :  1 },
{ "_id" :  "Peoples Exchange Bank", "count" :  1 },
```

```
{ "_id" :  "UK Credit Union ATM", "count" :  1 },
{ "_id" :  "BB&T ATM", "count" :  1 },
{ "_id" :  "Greater Kentucky Credit Union Inc.", "count" :  1 },
{ "_id" :  "Wells Fargo", "count" :  1 },
{ "_id" :  "Bank of the Bluegrass & Trust Co.", "count" :  1 },
{ "_id" :  "Peoples Bank", "count" :  1 },
{ "_id" :  "Republic", "count" :  1 },
{ "_id" :  "Greater Kentucky Credit Union, Inc.", "count" :  1 },
{ "_id" :  "Traditional Bank", "count" :  1 },
{ "_id" :  "Forcht Bank", "count" :  1 },
{ "_id" :  "University of Kentucky Federal Credit Union", "count" :
1 },
{ "_id" :  "Republic Bank & Trust", "count" :  1 },
{ "_id" :  "City National Bank", "count" :  1 },
{ "_id" :  "Central Bank (Drive-Thru)", "count" :  1 },
{ "_id" :  "Community Trust Bank", "count" :  1 } ]
```

This result shows some naming inconsistencies, e.g. "Republic Bank," "Republic," and "Republic Bank & Trust."

3.4 Determining a bicycle-friendly score of Lexington

The data revealed there are a large number of bicycle parking facilities in Lexington. This fact agrees with my intuition that college campus cities have more bicyclists.

As a hypothetical person wanting to move to Lexington without a car, I would want to know how bicycle-friendly this city is. I would want ample access to bicycle parking, lanes, and repair stations.

The OpenStreetMap data includes information about bicycle parking, including what type, e.g. stand, loops, rack, etc. It also includes details about the bicycle repair stations, e.g. whether they have a chain tool available or not. Unfortunately, I see a limited amount of data about bicycle lanes on the streets, compared to the number of ways.

```
> db.lexington.find({"bicycle":"yes"}).count()
221
> db.lexington.find({"bicycle":"no"}).count()
119
> db.lexington.find({"bicycle":{$exists:1}}).count()
341
> db.lexington.distinct("bicycle")
[ "no", "yes", "dismount" ]
> db.lexington.aggregate([{$match:  {"type":"way"}}, {$group:  {_id:
"$name", count:  {$sum:1}}}, {$sort:  {count:  -1}}]).result.length
7908 # 341/7908 = 4.3%
```

We could map out all bicycle-related places over Lexington, with special icons for each, using OpenStreetMap or Google Maps API. This would provide a useful visualization for the user.

A bicycle-friendly city score could be calculated by figuring the number of bicycle parking or repair stations within each square mile (density) of Lexington. The densities could be summed and normalized to a 0-100 score. This score would only be calculable and useful when the same process is done for other cities in America.

4 Conclusion

The cleanliness of this dataset was no worse than expected. The usual errors suggested by the Problem Set 6 were present, e.g. street type abbreviations. Unsurprisingly, some place names were inconsistent, and would require fixing by hand. There was a large number of ZIP codes missing which can be fixed using a program tied with the USPS address verification API. The user "woodpeck_fixbot" was the most active in this area, where the same user appeared as #2 in the sample project (Charlotte, NC)

The Lexington area OSM data is off to a good start, with details such as whether a bicycle repair station has a chain repair tool, bicycle lanes, and number of lanes. I would have to investigate further into OSM to see what other types of data can be entered.