OpenStreetMap Sample Project
Data Wrangling with MongoDB

Tim Lindsey

1. Problems Encountered in the Map

2. Data Overview

3. Additional Ideas

1. Problems Encountered in the Map

The data set downloaded had two primary issues after investigating a sample of the downloaded set from Map Zen

- Minimal address information
- Unique encoding requirements for the document

Minimal Address information

The data in the data set in general is pretty well formed, there are no significant lapses in data or inappropriate data polluting the set. For that reason I was comfortable injecting all “nd” and “tag” information into the json output (as compared to lesson 6 where a good bit of data was instructed to be ignored). However I noticed a few areas where there was text based input where raw data was expected, such as:

<tag k="addr:postcode" v="There are two separate postcodes for Garðabær and they are stored in admin_level=8 relations."/>

Here I’d expect just a zip code but instead it appears to be text based instruction.

It's also worth noting it seemed some data input wasn't extremely thorough. For instance:

```
<way id="5153835" version="12" timestamp="2011-11-27T15:44:00Z" changeset="9968302"
uid="53579" user="Dabbi">
        <nd ref="1519709430"/>
        <nd ref="1519709438"/>
        <nd ref="1519709481"/>
        <nd ref="1519709533"/>
        <nd ref="1519709520"/>
        <nd ref="1519709461"/>
        <nd ref="1519709462"/>
        <nd ref="1519709456"/>
        <nd ref="1519709515"/>
        <nd ref="1208295328"/>
        <nd ref="1519709483"/>
        <nd ref="1208295455"/>
        <tag k="oneway" v="yes"/>
        <tag k="highway" v="primary_link"/>
</way>
```

In a large part of the data set it appeared common to include important data such as "maxspeed" on roads. However here it appears user Dabbi does not follow this practice. Presumably this is due to loose requirements that leads to significant variation from user to user.

As one method to alleviate the lack of address info, I added a dictionary to the code that had all postal code information for Iceland and the region associated with said postal code. If a node had postal code info, I extracted the associated region from the dict and applied it to the node. This will ensure that while we may not be able to discern city we can at least tie it to the associated region in Iceland.

Phone Numbers Inconsistent

There were two issues with phone numbers in the data set:

1) The keys came across as both 'phone' and 'contact:phone'

2) The format was inconsistent (some numbers had a leading plus sign, there was random white space, etc).

Some simple data cleaning before posting the JSON resolved these issues and made the phone number presentation more concise.

Unique Encoding Requirements For The Document

Because of some of the unique characters involved in the language from the data set (it's all in icelandic), I had to before some character encoding to move the data to Unicode then into the proper encoding before output into the json output file. After research it appears this would have been slightly easier in Python 3+ as it defaults to Unicode rather than UTF-8. Once example of a troublesome line:

<tag k="addr:street" v="Hljóðalind"/>

2. Data Overview

Here is some general info on the data used in this lesson.

File Size:

reykjavik_iceland.osm - 181.6MB

reykjavik_iceland.json - 176.8MB

Example of Data Imported Into Mongo:

Data structure of single user

> *db.collection1.findOne()*


```
{
        "_id" : ObjectId("56b05556398a2ff21ff6b7e6"),
        "id" : "12885854",
        "pos" : [
```

```
                "-22.5434936",
                "63.9721673"
        ],
        "created" : {
                "timestamp" : "2013-09-07T14:43:51Z",
                "changeset" : "17719085",
                "user" : "MdMax",
                "version" : "5",
                "uid" : "136345"
        }
}
```


Submissions by an active user “MdMax”

db.collection1.find({},{"user": "MdMax"})


```
{ "_id" : ObjectId("56b05556398a2ff21ff6b7e6") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7e7") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7e8") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7e9") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7ea") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7eb") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7ec") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7ed") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7ee") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7ef") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f0") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f1") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f2") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f3") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f4") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f5") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f6") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f7") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f8") }
{ "_id" : ObjectId("56b05556398a2ff21ff6b7f9") }
```


Total records for user ‘Kvarrval’ sorted descending by timestamp (note disk use must be enabled due to size of list)

db.collection1.aggregate([{$match: {'created.user': 'Kjarrval'}},{$sort: {'created.timestamp': -1}}],{allowDiskUse: true})


```
{ "_id" : ObjectId("56b05561398a2ff21ffc89e7"), "id" : "1619587074", "pos" : [ "-21.755284", "64.157507" ], "created" : { "changeset" : "36631154", "user" : "Kjarrval", "version" : "5", "uid" : "209717", "timestamp" : "2016-01-17T11:04:34Z" } }
{ "_id" : ObjectId("56b05561398a2ff21ffc8b31"), "id" : "1619588599", "pos" : [ "-21.779077", "64.152653" ], "created" : { "version" : "5", "uid" : "209717", "timestamp" : "2016-01-17T11:04:34Z", "changeset" : "36631154", "user" : "Kjarrval" } }
```

{ "_id" : ObjectId("56b05569398a2ff21f003b91"), "id" : "2310300358", "pos" : ["-21.7552934", "64.1575245"], "created" : { "changeset" : "36631154", "user" : "Kjarrval", "version" : "2", "uid" : "209717", "timestamp" : "2016-01-17T11:04:34Z" } }
{ "_id" : ObjectId("56b05569398a2ff21f003cf1"), "id" : "2310427886", "pos" : ["-21.7790926", "64.1526999"], "created" : { "version" : "3", "uid" : "209717", "timestamp" : "2016-01-17T11:04:34Z", "changeset" : "36631154", "user" : "Kjarrval" } }
...

Determine highest speed asphalt based roads

> db.collection1.aggregate([{$match: {'surface': 'asphalt'}},{$sort: {'maxspeed': -1}}],{allowDiskUse: true})

{ "_id" : ObjectId("56b0556d398a2ff21f027468"), "maxspeed" : "90", "node_refs" : "2374403836", "name" : "Reykjanesbraut", "created" : { "user" : "Kjarrval", "version" : "17", "uid" : "209717", "timestamp" : "2014-07-25T02:33:16Z", "changeset" : "24341135" }, "oneway" : "yes", "surface" : "asphalt", "address" : { }, "ref" : "41", "id" : "4425119", "highway" : "primary", "network" : "S" }
{ "_id" : ObjectId("56b0556e398a2ff21f02798f"), "maxspeed" : "90", "node_refs" : "43051676", "name" : "Krxfdsuvxedkurvegur", "created" : { "version" : "8", "uid" : "110639", "timestamp" : "2015-08-07T16:45:57Z", "changeset" : "33184726", "user" : "aighes" }, "surface" : "asphalt", "smoothness" : "excellent", "address" : { }, "ref" : "42", "id" : "23296272", "highway" : "secondary", "network" : "T" }
{ "_id" : ObjectId("56b0556e398a2ff21f027d68"), "maxspeed" : "90", "node_refs" : "819262027", "name" : "Suourlandsvegur", "created" : { "changeset" : "34618260", "user" : "Baron Harkonnen", "version" : "16", "uid" : "2704608", "timestamp" : "2015-10-13T18:09:58Z" }, "oneway" : "no", "surface" : "asphalt", "address" : { }, "ref" : "1", "id" : "26756984", "highway" : "trunk", "network" : "S" }
{ "_id" : ObjectId("56b0556e398a2ff21f0286c3"), "maxspeed" : "90", "node_refs" : "60652128", "name" : "xdeingvallavegur", "created" : { "changeset" : "12140360", "user" : "olvagor", "version" : "10", "uid" : "63969", "timestamp" : "2012-07-07T14:17:10Z" }, "surface" : "asphalt", "lit" : "no", "address" : { }, "lanes" : "2", "ref" : "36", "id" : "37845929", "highway" : "primary", "network" : "S" }
{ "_id" : ObjectId("56b0556e398a2ff21f028f9d"), "maxspeed" : "90", "node_refs" : "2374403835", "name" : "Reykjanesbraut", "created" : { "version" : "10", "uid" : "136345", "timestamp" : "2013-09-07T14:43:50Z", "changeset" : "17719085", "user" : "MdMax" }, "oneway" : "yes", "surface" : "asphalt", "address" : { }, "ref" : "41", "id" : "61492888", "highway" : "primary", "network" : "S" }
{ "_id" : ObjectId("56b0556f398a2ff21f0305e7"), "maxspeed" : "90", "node_refs" : "1815985319", "name" : "xdeingvallavegur", "created" : { "user" : "Stalfur", "version" : "8", "uid" : "149223", "timestamp" : "2015-06-07T01:18:46Z", "changeset" : "31779970" }, "surface" : "asphalt", "lit" : "no", "address" : { }, "lanes" : "2", "ref" : "36", "id" : "108449379", "highway" : "primary", "network" : "S" }
{ "_id" : ObjectId("56b05570398a2ff21f033f33"), "maxspeed" : "90", "node_refs" : "1743566654", "name" : "Krxfdsuvxedkurvegur", "created" : { "version" : "3", "uid" : "1643116", "timestamp" : "2015-10-18T17:49:09Z", "changeset" : "34716813", "user" : "fell3" }, "surface" : "asphalt", "smoothness" : "excellent", "address" : { }, "ref" : "42", "id" : "162473180", "highway" : "secondary", "network" : "T" }

Unique Users

db.collection1.distinct("created.user").length
559

Total Records in The Document

865913

3. Additional Ideas

Some consideration of ideas to add to help make the dataset more complete:

Motivate users to complete parameters completely and correctly:

Overall it's clear the users have taken good care to keep the data set clear and pretty consistent. As compared to the data set from lesson 6 it seems that there are far fewer anomalies to account for. Also, a nice bonus was that there were no street names that needed to be changed from abbreviation to the full name. However, there are a few instances where there appear to be inconsistencies in data loading. Here is one example:

<tag k="addr:housenumber" v="2"/>

<tag k="addr:housenumber_1" v="4"/

These are 2 tags side by side in one of the datasets. It does not appear as though this would be a valid address input and the issue was either introduced by human error or by some sort of bug in the system that compiles this data. It would appear enforcing stricter parameters around these inputs would help clean the data up front.

Increase datapoints

While the data is clean and a good effort it appears by the users, it seems like the available data is a little light. It may be possible to help motivate new or existing users to provide additional data in the future by offering local discounts, perhaps announcing “winners” and offering prized through some sort of locally help competition, or encouraging locales to get the word out about the program and drive their own activity in openmaps.

Conclusion

Getting data from a place like iceland provides some interesting insights into how a project like this translates overseas. Considering the project is largely volunteer run the data itself is extremely clean and practical to access and load into the database. However it is a somewhat light dataset as opposed to some of the other regions and I believe that to be able to fully compare and find use of this data the users will have to populate this region with more useful data points or new users will have to get interested in the project to help fill the Reykjavik dataset in more completely.