OpenStreetMap Sample Project

Data Wrangling with MongoDB

Iuşan Andrei

Map Area: Bucharest, Romania

https://mapzen.com/data/metro-extracts

1. Problems encountered in the dataset

The problems observed are:

- Miss-spelled street names (1.3.1)
- Incorrectly marked amenities (1.3.2)

1.1 Inserting data into MongoDB

First I imported the data as it is, into MongoBD. This first step does not involve cleaning.

The document schema contains the fields `'created'`, `'id'`, `'type'`, and all the tags in the osm object. If `'type'` equals `'way'` we also have `'node_refs'` as an array containing the ids of the nodes that the way references; if type is `'relation'` we have `'members'` as an array of documents that contain references to the members of the relation (a relation references nodes and/or ways).

1.2 Inspecting the data

I counted the tags and printed them, sorting them in the descending order of their frequency. Most common tags out of about 97000 `'way'` elements in the dataset are:

```
'building': 45929
'highway': 41665
```

This suggests that about half of the `'way'` elements in the dataset represent buildings and about half of them represent roads.

I have tested this hypothesis by querying the database:

```
> db.bucharest.find({'type':'way', 'building':{'$exists':1}, 'addr:street':
{'$exists':1}}).count()
 1548
> db.bucharest.find({'type':'way', 'highway':{'$exists':1}, 'addr:street':
{'$exists':1}}).count()
 8
```

This confirmed my initial observation: a building has an address field, but the '`addr:street`' field is not required for a street. The street name appear in the `'name'` tag of the elements that also contain a `'highway'` tag. There are 8 exceptions though that need to be fixed.

1.3 Auditing

1.3.1 Street names

In Romania, the street type appears before the street name (e.g. “Bulevardul Republicii”, while in English this would be “Republicii Boulevard”). I checked for the first word in street names, both in the address field of the buildings and in the name field of the elements with a 'highway' tag.

I wrote a script to audit the street names, and to update them in the database. For this dataset, street names are not abbreviated, but a number of misspellings were present. I corrected them, and at the same time, I modified the schema. I created an 'address' field for the buildings, that contain a dictionary with street name, street number, and other address info.

Some elements can be fixed programmatically, but some are miss-tagged, for instance an element has the tag 'highway' and in the 'name' tag, the word 'forest'. Those elements need to be inspected one at a time since they are outliers, each having a unique way of encoding some data.

1.3.2 Amenities

I queried the database for shops and found those results.

```
db.bucharest.find({'amenity':'shop'}).count()
6
db.bucharest.find({'shop':{'$exists':1}}).count()
2334
```

This is surprising since I believe the shops should also have the tag 'amenity'. I came up with a method to update all amenities programatically. Here are a few updates made by the algorithm:

```
'shop': 2334,
'atm': 430,
'parking': 281
```

2. Data Overview

This section contains statistics about the dataset, obtained using MongoDB queries.

Size of file:
```
bucharest_romania.osm .......... 206 MB
bucharest_romania.osm.json ...... 230 MB
```

Number of documents:
```
> db.bucharest.find().count()
928429
```

Number of nodes
```
> db.bucharest.find({'type':'node'}).count()
829812
```

Number of ways
```
> db.bucharest.find({'type':'way'}).count()
97165
```

Number of relations
```
> db.bucharest.find({'type':'relation'}).count()
1452
```

Number of users
```
> users = db.bucharest.aggregate([{'$group':{'_id':'$created.user','count':{'$sum':1}}},
{'$sort':{'count':-1}}])
> users = users['result']
> len(users)
749
```

Top 5 contributing users:
```
> users[:5]
[{u'_id': u'razor74', u'count': 272851},
 {u'_id': u'dincaionclaudiu', u'count': 138865},
 {u'_id': u'baditaflorin', u'count': 84928},
 {u'_id': u'Mircea Toader', u'count': 74937},
 {u'_id': u'Romania CLC import bot', u'count': 66724}]
```

Number of shops (after cleaning)
```
> db.bucharest.find({'amenity':'shop'}).count()
2336
```

3. Other Ideas

Another great way of cleaning and standardizing the dataset is to bring in more data from different sources. Since every node is required to have the position and a large number of nodes does not contain address information, it is worth exploring the applicability of reverse geocoding services, like: http://www.doogal.co.uk/BatchReverseGeocoding.php

This service allows converting multiple latitude, longitude pairs to addresses in one step. This can be used for adding addresses to the nodes that do not contain an address. The service could also be used to automate the process, such that when a new node is added to the map, a script will check it and update the address.

There are also datasets with postal codes available like:

- http://date.gov.ro/organization/posta-romana
- http://www.coduripostale.ro/en

Those datasets can be used to automatically update a large amount of data.

Conclusion

After inspecting this dataset it is clear that the data is not standardized. The data has been cleaned to some extent in this assignment, but further cleaning is possible. Most elements have been cleaned programatically, but others - those that have unique tags - require special attention if we are to preserve as much of the data as possible.