GÁBOR GALGÓCZ

OpenStreetMap Sample Project

Data Wrangling with MongoDB

Map area: Dublin, Ireland

Data source: https://mapzen.com/data/metro-extracts
https://s3.amazonaws.com/metro-extracts.mapzen.com/dublin_ireland.osm.bz2
Accessed: 25 / 07 / 2015

1. Problems encountered

The first problem I encountered during the project work was the selection of the city to work with. I've originally chosen Budapest, which is my hometown, but I encountered two problems. The first was a minor problem, the file size was considerable, so working with the file was slow. The second and bigger issue was that once I started cleaning up the data, I noticed that dozens and dozens of errors can be found in the data, which can be attributed to the special accented characters used in the Hungarian language (öüóőúéáűí). Apparently many character encoding issues appeared, and cleaning this up would take considerable time from my side, because I should dive into several possible bad encodings of the correct characters. (Bad encodings are a result of not using UTF-8 but several other character tables used by different generations of Windows and other operating systems. This means that one correct character usually has several mis-encoded versions in the data.)

After realizing that the accented characters would create considerable manual work to clean them, I decided to choose an English speaking location instead, but one which has several unconventional words to designate locations. Also, to keep the file size small, I chose a smaller city than Budapest. I explored some other options and finally decided to go with Dublin, Ireland.

Dublin has an unexpectedly high number of unique ways to describe locations, my guess is that it is a consequence of its rich historical heritage. So after adding my expected values in a list, I had to create a new list for all the names which were correct, but unexpected. I named this list "approved", because often I had to check with Google Maps that the location's name exists and it is spelled correctly. Once I cross-referenced the name with Google Maps, I either added it to the "approved" list, or to the "mapping" dictionary to correct it.

Common errors in the dataset were the following:

TYPOGRAPHICAL ERRORS:

"Nouth": "North"
"Roafd": "Road"
"Sreet": "Street"

SHORTENED NAMES:

"Ave": "Avenue"
"St": "Street"

GÁBOR GALGÓCZ

INCONSISTENT CAPITALIZATION:

“o connell street”: “O’Connell Street”

MISSPELLING OF WORDS OF FOREIGN ORIGIN:

“Heidleberg”: “Heidelberg”

EXTRA INFORMATION ADDED TO THE STREET NAME:

“Newbrook Road, Donghmede”: “Newbrook Road”
“Supple Park 32-39”: “Supple Park”

FULL LIST OF CORRECTIONS, THE OUTPUT OF AUDIT.PY

```
The Drive, castletown => The Drive, Castletown
Bervstede => Berystede
The Rise 14-28 => The Rise 14-28
The Rise 1-13 => The Rise 1-13
Strand Rd. => Strand Road
26 => 26
Heidleberg => Heidelberg
residential => Residential
The Rise,Belgard heights => The Rise,Belgard Heights
Ballinclea heights => Ballinclea Heights
Serpentine Avenue, Ballsbridge, Dublin 4 => Serpentine Avenue,
Ballsbridge, Dublin 4
Kill Avevnue => Kill Avenue
Supple Park 27-31 => Supple Park 27-31
Manor Court 1-9 => Manor Court 1-9
Supple Park 48- => Supple Park 48-
Hyde park => Hyde Park
Parliament Sreet => Parliament Street
The Ward, Ashbourne Rd, Dublin => The Ward, Ashbourne Rd, Dublin
Manor Court 10-21 => Manor Court 10-21
Aungier St => Aungier Street
Parknasilloge => Parknasilla
Supple Park 32-39 => Supple Park 32-39
Supple Park 40-44 => Supple Park 40-44
Newbrook Road, Donghmede => Newbrook Road, Donaghmede
Saint Mobhi Road => Saint Mobhi Road
Hanbury lane => Hanbury Lane
Warner's lane => Warner's Lane
Bayside Boulevard Nouth => Bayside Boulevard North
Old Dublin Roafd => Old Dublin Road
Balally, Dundrum => Balally, Dundrum
Aspencourt => Aspen Court
o connell street => o connell Street
Charlestown Shopping Cente => Charlestown Shopping Center
First Ave => First Avenue
Spruce Ave => Spruce Avenue
Griffith Ave => Griffith Avenue
Novara road => Novara Road
```

2. Overview of the data

FILE SIZE

dublin_ireland.osm - 239 MB
dublin_ireland.osm.json - 267 MB

After following the steps that were detailed in Lesson 6, I cleaned the data, converted it to JSON and imported it into MongoDB. Then using the following queries I was able to give a statistical overview of the OSM data.

NUMBER OF DOCUMENTS

The Dublin OSM file contains 1189242 documents.

```
db.dublin.count()
```

NUMBER OF NODES

The Dublin OSM file contains 1013116 nodes.

```
db.dublin.find({ "type": { "$in":['node']} }).count()
```

NUMBER OF WAYS

The Dublin OSM file contains 176094 ways.

```
db.dublin.find({ "type": { "$in":[‘way']} }).count()
```

NUMBER OF UNIQUE USERS

The Dublin OSM has been edited by 1038 unique users.

```
len(db.dublin.distinct("created.user”))
```

TOP CONTRIBUTORS

```
db.dublin.aggregate([{"$group":{"_id":"$created.user", "count":{"$sum":
1}}}, {"$sort": {“count":-1}},{"$limit":5}])
```

The list of top contributors:

```
[{u'count': 229595, u'_id': u'Nick Burrett'},
 {u'count': 181712, u'_id': u'mackerski'},
 {u'count': 150386, u'_id': u'Dafo43'},
```

```
{u'count': 134867, u'_id': u'brianh'},
{u'count': 57368, u'_id': u'Conormap'}]
```

MOST POPULAR CUISINES

By querying the amenity type “restaurant”, we can easily make a list of the most popular cuisines, based on the occurence of the restaurants. Note that several restaurants have no data filled in about the cuisine they offer, so these results are mostly just for illustrate the OSM data, not for statistical analysis.

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

Results:

```
{u'_id': u'italian', u'count': 61},
{u'_id': u'indian', u'count': 41},
{u'_id': u'chinese', u'count': 35},
{u'_id': u'pizza', u'count': 23},
{u'_id': u'asian', u'count': 16},
{u'_id': u'regional', u'count': 16},
{u'_id': u'thai', u'count': 15},
{u'_id': u'international', u'count': 9},
{u'_id': u'japanese', u'count': 8}]}
```

3. Other ideas about the datasets

It turned out that there is a need to check many of the entries, to filter out misspellings and other incorrect data that I listed above. I was thinking about building a script that could use another freely available database to compare the entries, maybe the Google Maps API would be useful for this. So our cleaning process could have a step when it is querying the Google Maps API to see if the entry can be found in the city or not. This way we could make the cleaning faster, to sort out all the entries which are really incorrect from the ones that are actually correct but they are so unusual that they are not included in our “expected” list.

Another idea that I came up with is to get the expected list of names from some service, so we don’t have to create it manually. While I was still working with the Budapest OSM data, I found that the Hungarian post has a webpage to find addresses in Hungary, and they have a dropdown menu with all the options like “street”, “road”, “square”, a total of at least 40 values. By scraping the values of this dropdown list from the source, we can easily create the list of expected values. Unfortunately I don’t know a similar webpage in Ireland so I couldn’t test my idea.