OpenStreetMap Sample Project
Data Wrangling with MongoDB

Olivier Bézie

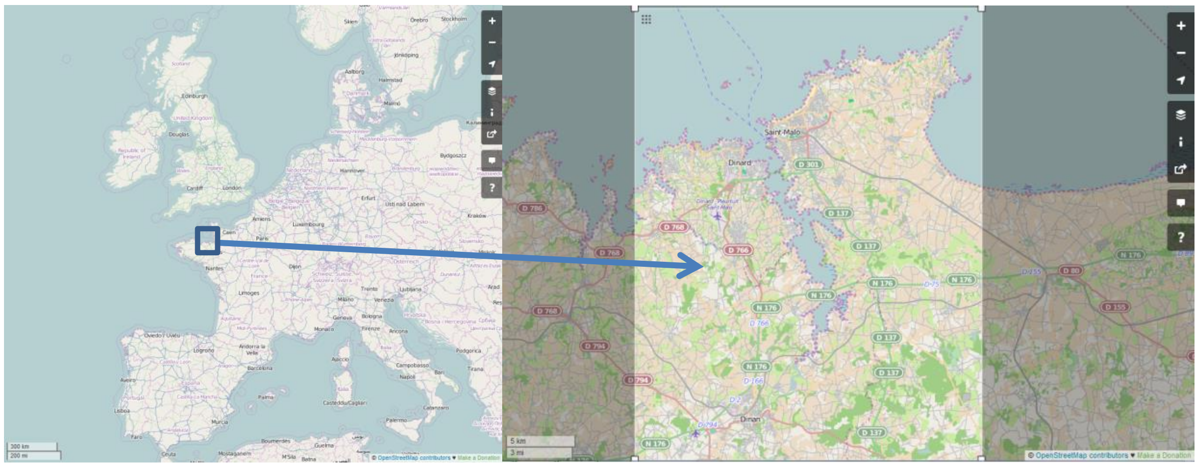
29 June 2015

Map Area: La Rance, Brittany, France

Table of Contents

Introduction

I have chosen to work on the OpenStreetMap data for the area of La Rance in the Brittany region in France where I used to spend my summer vacation. The area goes from the city of Dinan at the source of the river La Rance to its month with the port of Saint-Malo on one side and the seaside city of Dinard.



The boundary of the area is defined by the following coordinates:
minlat="48.4267" minlon="-2.1636" maxlat="48.7277" maxlon="-1.8141"

The cities in this area being rather small town, I decided to a to the scope the seaside town of Saint-Lunaire and the small port of Cancale famous for its oysters.

The OpenStreetMap data for this area have been downloaded manually using the "Overpass API".

1. Problems Encountered in the Map

After initially downloading the sample size of the La Rance area I have run it against a provisional from_Dinan_to_StMalo_Osm2Json.py file. To reduce the focus of the data cleaning work this program have filtered the osm source data to retain only the data related to the “node” and “way” main tags and the single key attributes or key attributes with single “:”. The data cleaning exercise for the excluded key attributes will be the subject of another project. In this project we will focus on the key attributes and their values which are related to location (like address). I noticed four main problems with the data, which I will discuss in the following order:

- While filtering the data to produce the json file I have found some attributes are with upper letters: NOM, CEMT, school:FR. I will check if they cannot be matched and replaced with standard attributes as documented in the OSM Map Features.
- I listed the city names on MongoDB and could identify that the city of Saint-Malo is spelled 2 different ways.
- Looking at the key attributes (filtered as mentioned above) of the downloaded data I also found a key attribute address which is redundant with the key attributes “addr:housenumber” and “addr:street”.
- Listing the house number value in mongoDB have shown some discrepancies in the way a number followed by a letter or a word is displayed.
- Once the data loaded in MongoDB, I realized that the key “type” is used in the osm data and then creates a conflict with the “type” key used to differentiate node and way tags in the json used to load the data in MongoDB

The following steps has been followed to address those problems:

1. Run the Osm2Json python script and load the generated json into MongoDB
2. Analyse the data consistency in the mongo shell or running aggregate queries from a python script
3. Include additional cleaning steps in the Osm2Json python script and go back to step

1.1 Key attributes with upper letter

Several attributes with upper letters have been detected while filtering and transforming the osm source data into the json that has been loaded in MongoDB. These attributes are:

- school:FR found amongst the “node” data attributes
- NOM, CEMT, school:FR found amongst the “way” data attributes

A cleaning method is a proposed and applied only for the key NOM and CEMT

a) NOM

NOM is present only once as an attribute and its value shows that it can be replaced with the standard attribute “name” (which make sense since “nom” is the French word for “name”)):

```
> db.osmData.distinct("NOM").length
1
> db.osmData.distinct("NOM")
[ "Crématorium de Saint-Malo" ]
```

This has been done as a cleaning step in the Osm2Json program:

```
 #replace the local key "NOM" with a standard key "name"
  if key == "NOM":
            node["name"] = value
```

and the result verified in MongoDB:

```
  > db.osmData.distinct("NOM")
  [ ]
  > db.osmData.find({"name":"Crématorium de Saint-Malo"})
```

{ "_id" : ObjectId("55900b38cd346c01de17db3d"), "amenity" : "crematorium", "node_refs" : ["1832468530", "1832468631", "3238617521", "3238617523", "3238617532", "3238617533", "3238617534", "3238617536", "3238617537", "3238617542", "3238617543", "1832468521", "1832469076", "1832468530"], "name" : "Crématorium de Saint-Malo", "created" : { "changeset" : "27516689", "user" : "Rom1", "version" : "1", "uid" : "337375", "timestamp" : "2014-12-16T21:20:07Z" }, "pos" : [0, 0], "phone" : "+33 2 99888288", "source" : "E-mégalis Bretagne et collectivités territoriales bretonnes - 2012", "address" : { }, "waterway" : { }, "type" : "way", "id" : "317580953" }

b) CEMT

CEMT is the abbreviation of Conférence européenne des ministres des Transports. It is a European organization building synergies across the ministry of transportation. The value of this key attributes classifies the navigability of the rivers. It could be proposed to split the information under “waterway” by creating some sub-key attribute like “waterway:classtype” (value=’CEMT’) and “waterway:class” (value=’0’). The existing waterway values will be copied in “waterway:value”.

```
> db.osmData.find({"CEMT":"0"}).limit(1)
```

{ "_id" : ObjectId("558aa02e8a43f55ef53f0c31"), "node_refs" : ["1599610013", “1599610024", "1599610034", "1599610044", "1599610049"], "name" : "Canal d'Ille-et-Rance", "created" : { "changeset" : "25171315", "user" : "Verdy_p", “version": "10", "uid" : "90780", "timestamp" :

"2014-09-01T22:16:25Z" }, "CEMT" : "0", "pos" : [0, 0], "waterway" : "canal", "type" : "way", "id" : "42762906" }

This has been added as a cleaning step in the Osm2Json python program:

```
#include this river info under the standard key "waterway"
elif key == "waterway":
        node["waterway"]["value"] = value
elif key == "CEMT":
        node["waterway"]["classtype"] = key
        node["waterway"]["class"] = value
```

and the result verified in MongoDB:

```
> db.osmData.distinct("waterway")
[
     …
    {
        "classtype" : "CEMT",
        "class" : "0",
        "value" : "canal"
    },
    {
        "value" : "river"
    },
    {
        "classtype" : "CEMT",
        "class" : "0",
        "value" : "river"
    },
     …
 ]
```

c) school:FR

« school » is not a key attribute but an attribute value in the OSM map features list. On the other hand the values shown under the key attribute “school:FR” indicate the different school level available in the French education system and there is no corresponding key in the OSM map features.

```
> db.osmData.distinct("school:FR")
[
    "primaire;maternelle",
    "primaire",
    "lycée",
    "élémentaire",
    "maternelle",
    "collège",
    "secondaire"
]
```

Looking at one document containing key attribute “school:FR”, I could see that the “amenity” key value is correctly set to “school”. Therefore nothing is broken. The opportunity to move the key attribute “school:FR” under “amenity:school:FR” could eventually be discussed in the OSM forums. I don’t propose any cleaning solution here.

```
> db.osmData.find({"school:FR":"primaire"}).limit(1)
{ "_id" : ObjectId("558aa00c8a43f55ef52e620a"), "amenity" : "school", "name" : "
École primaire catholique", "created" : { "changeset" : "26561674", "user" : "Ve
rdy_p", "version" : "3", "uid" : "90780", "timestamp" : "2014-11-05T01:24:19Z" }
, "operator:type" : "private", "pos" : [ 48.6779103, -1.9127143 ], "school:FR" :
 "primaire", "type" : "node", "id" : "370807980" }
```

1.2 City name wrongly spelled

It seems that there is a lot of missing streets in the different city nodes and ways of the La Rance area. OSM search (boulevard Lhotelier, Dinard, France) of the place where I spend my vacation in Dinard street “boulevard Lhotelier” does not provide any result.

To analyze the extend of missing streets we will focus on the 5 cities in scope: Dinan, Dinard, Saint-Lunaire, Saint-Malo and Cancale

First I have tried to find the key attributes holding the city values. I looked into the key attributes “is_in.city”, “place.city” and “addr:city. Only the latter provided some results. I also realized that the city of Saint-Malo has at least 2 different spellings: “Saint-Malo” and “Saint Malo”. This last should be corrected to “Saint-Malo” in OSM.

```
]> db.osmData.distinct("address.city").sort()
[
        "Cancale",
        "Dinan",
        "Dinard",
        "Hirel",
        "La Gouesnicre",
        "La Richardais",
        "La Ville-cs-Nonais",
        "Lancieux",
        "Miniac-Morvan",
        "Plerguer",
        "Plesder",
        "Pleurtuit",
        "Saint Malo",
        "Saint-Coulomb",
        "Saint-Jouan-des-Guérets",
        "Saint-Lunaire",
        "Saint-Malo"
]
```

This has been taken into account Osm2Json python script and all “Saint Malo” occurrence have been changed to “Saint-Malo”: A dictionary maps the different way a city can be named to the right one and a function makes the change. If any other wrong city name is found, the cleaning can happen by just adding the correct mapping in the dictionary

```
rightCityName = {"Saint Malo": "Saint-Malo"}
def correctCitySpelling(city):
    if city in rightCityName.keys():
        return rightCityName[city]
    else:
         return city
```

The result is verified in MongoDB:

```
> db.osmData.distinct("address.city").sort()
[
        "Cancale",
        "Dinan",
        "Dinard",
        "Hirel",
        "La Gouesnière",
        "La Richardais",
        "La Ville-ès-Nonais",
        "Lancieux",
        "Miniac-Morvan",
        "Plerguer",
        "Plesder",
        "Pleurtuit",
        "Saint-Coulomb",
        "Saint-Jouan-des-Guérets",
        "Saint-Lunaire",
        "Saint-Malo"
]
```

I have had also a look at the list of attribute selected at the time the source osm file has been transformed into json and could not see other eventually custom attributes that could hold a city name. It is safe to say that if the city name is present it will be with the “addr:city” key

1.3 Redundant “address” key attribute

I have done a similar exercise to identify where the street address is stored. I have identified 2 key attributes: “address.city” and “address”. The “address” key is incorrectly used in OSM. This has been taken into account in the cleaning steps of the Osm2Json script and all addresses with the key “address” has been moved to the keys “address.housenumber”, “address.street”:

```
def parseAddress(value):
     [housenumber, street] = value.split(', ')
      return housenumber, street
```

An assumption has been made for the above code that the standard way to write a French street address is respected, i.e. there is a comma to delimit the house number from the street name. The proposed parsing function should be reviewed if other cases are caught.

1.4 Incorrect house numbers

The house number can be followed by a letter (A, B ,…) or a word (bis, ter) to indicate the repetition of the same number in the same street (bis, ter). Looking in MondoDB I have noticed that sometime this letter or word is directly put as a suffix of the number. The good practice is to have a blank in between:

```
> db.osmData.distinct("address.housenumber").sort()
[
        "1",
        "1 B",
        "1 C",
        "1 bis",
        "10",
        …
        "103",
        "103 bis",
        "105",
          "…
        "109",
        "10bis",
        "11",
        "11 bis",
        "110",
          …
        "118",
        "11Bb",
        "11T",
        "12",
        "12 bis",
          …
  ]
```

This has been addressed in the Osm2Json python script to make sure that a space is inserted:

```
def correctHouseNumber(value):
    i = 0
    for char in value:
        if re.search(r"[0-9]", char):
            i += 1
            continue
        else:
            break
    if re.search(r"[a-z]|[A-Z]", char):
        value = value[:i] + " " + value[i:]

     return value
```

The result is verified in MongoDB:

```
 > db.osmData.distinct("address.housenumber").sort()
 [
        "1",
        "1 A",
        "1 B",
        "1 C",
        "1 D",
        "1 bis",
        "1 ter",
        "10",
        "10 bis",
         "100",
          …
        "103",
        "103 bis",
        "105",
          …
        "11",
        "11 Bb",
         "11 T",
          …
   ]
```

1.5 “type” attribute

In the OSM source “type” is used as a key attribute and is in conflict with the json key “type” that has been used to map the “node” and “way” attribute values. This has been identified when querying the “type” key in MongoDB. An extra “gas” type is displayed

```
> db.osmData.distinct("type")
[ "node", "way", "gas" ]
```

Fortunately such an attribute is used only in one tag of type way. It does not corrupt the data that has been loaded in MongoDB.

```
> db.osmData.find({"type":"gas"}).count()
1
> db.osmData.find({"type":"gas"})
{ "_id" : ObjectId("558aa0358a43f55ef5415a40"), "node_refs" : [ "2493743501", "2493743506",
"2493743504", "2493743499", …, "2493742589" ], "created" : { "timestamp" : "2013-10-
12T14:01:45Z", "changeset" : "18315540", "user" : "David Crochet","version" : "1", "uid" : "50624" },
"man_made" : "pipeline", "pos" : [ 0, 0 ], "location" : "underground", "type" : "gas", "id" : "241783579" }
```

On the other this problem might occur while cleaning other map area to I have changed json key “type” into “basic_elem”, re-run the Osm2Json script and reloaded the data in MongoDB:

```
> db.osmData.distinct("basic_elem")
[ "node", "way" ]
> db.osmData.find({"type":"gas"})
> db.osmData.find({"type":"gas"})
{ "_id" : ObjectId("559044b3e1b561109abe0543"), "node_refs" : [ "2493743501", "2493743506",
"2493743504", …, "2493742591", "2493742589" ], "created" : { "version": "1", "uid" : "50624",
"timestamp" : "2013-10-12T14:01:45Z", "changeset" : "18315540", "user" : "David Crochet" },
"man_made" : "pipeline", "pos" : [ 0, 0 ], "basic_elem" : "way", "location" : "underground", "type" :
"gas", "id" : "241783579" }
```

2. Data Overview

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

File sizes

from_Dinan_to_ StMalo.osm .... 251 MB
from_Dinan_to_ StMalo.json .... 283 MB

# Number of documents

```
> db.osmData.find().count()
1265892
```

# Number of nodes

```
> db.osmData.find({"type":"node"}).count()
1100161
```

# Number of ways

```
> db.osmData.find({"type":"way"}).count()
165730
```

# Number of unique users

```
> db.osmData.distinct("created.user").length
408
```

# Top 1 contributing user

```
> db.osmData.aggregate([{"$group":{"_id":"$created.user",
“count":{"$sum":1}}}, {“$sort":{"count":1}}, {"$limit":1}])
{ "_id" : "destjena", "count" : 428340 }]
```

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

```
> db.osmData.aggregate([{"$group":{"_id":"$created.user",
"count":{"$sum":1}}}, {"$group":{"_id":"$count", "num_users":{"$sum":1}}}, {"$sort":{"_id":1}},
{"$limit":1}])
{ "_id" : 1, "num_users" : 68 }
   # “_id” represents postcount
```

3. Additional Ideas

House number in way

There is then a lot of missing street information and certainly a lot of missing node and way information in general. A closer into the cities that has been chosen for this project just confirms the above statement:

	Cancale	Dinan	Dinard	Saint-Lunaire	Saint-Malo
#address in ways		9	2		174
#house number in ways		8	2		172
#addesses in nodes	4		1	1	6
#house number in nodes	2		1		4
# of street name (formal source)	276	250	369	178	1157

```
(  db.osmData.aggregate([{"$match": {"address.street": {"$exists": 1}}}, {"$group": {"_id":
{"element": "$basic_elem", "city": "$address.city"}, "count": {"$sum": 1}}}])
db.osmData.aggregate([{"$match": {"address.housenumber": { "$exists": 1}}},{"$group": {"_id":
{"element": "$basic_elem", "city": "$address.city"}, "count": { "$sum": 1}}}]) )
```

I don’t think that the “way“ element is the right place to indicate the house number. This information rather pertains to the “node” element. But the house number info appears in the “way” rather than in the “node”. I could think of a python script to move back the house number information along with the corresponding address in the “node” and upload the changes using the JOSM upload utility.

Missing address information

The # of address (formal source) has been extracted from an official electronic directory showing all the streets in a given city. The corresponding web page for each of the 5 cities has been manually saved locally and I have run a python program using the elementTree library to extract the streets and compute their numbers.

For each of the cities there is a big discrepancy between the number of streets in OSM and the actual number of streets available in the official source. I have been thinking of developing a script (javascript) using the google google.maps.GeocoderRequest API but it is considered as a copyrighted resource and cannot be used for OSM. Another way to contribute would be to get in touch with the top contributors in each city around a specific objective – for example update all ways element with the appropriate name street or update the nodes with all the restaurant in your city. However most of them does not seem to be very active:

```
> db.osmData.aggregate([{"$match": {"address.city": {"$in": ["Cancale", "Dinan", "Dinard",
"Saint-Lunaire", "Saint-Malo"]}}}, {"$group": {"_id":  {"user": "$created.user", "city":
"$address.city"}, "count": {"$sum":1}}}, {"$sort": {"count": -1}}])
{ "_id" : { "user" : "xylome", "city" : "Saint-Malo" }, "count" : 137 }
{ "_id" : { "user" : "cquest", "city" : "Saint-Malo" }, "count" : 19 }
{ "_id" : { "user" : "Pasfan", "city" : "Saint-Malo" }, "count" : 8 }
{ "_id" : { "user" : "Gnafron", "city" : "Dinan" }, "count" : 7 }
{ "_id" : { "user" : "botdidier2020", "city" : "Saint-Malo" }, "count" : 2 }
{ "_id" : { "user" : "Maruku", "city" : "Saint-Malo" }, "count" : 2 }
{ "_id" : { "user" : "JulienBalas", "city" : "Saint-Malo" }, "count" : 2 }
{ "_id" : { "user" : "Springbank35", "city" : "Saint-Malo" }, "count" : 2 }
{ "_id" : { "user" : "Pasfan", "city" : "Dinard" }, "count" : 1 }
{ "_id" : { "user" : "indiscipline", "city" : "Dinan" }, "count" : 1 }
{ "_id" : { "user" : "Rom1", "city" : "Saint-Malo" }, "count" : 1 }
{ "_id" : { "user" : "caralax", "city" : "Dinard" }, "count" : 1 }
{ "_id" : { "user" : "MatthieuMartin", "city" : "Dinan" }, "count" : 1 }
{ "_id" : { "user" : "Kioska Journo", "city" : "Saint-Malo" }, "count" : 1 }
{ "_id" : { "user" : "ericboulet35", "city" : "Saint-Malo" }, "count" : 1 }
{ "_id" : { "user" : "orklah", "city" : "Saint-Malo" }, "count" : 1 }
{ "_id" : { "user" : "Olyon", "city" : "Cancale" }, "count" : 1 }
{ "_id" : { "user" : "Cyrano25550", "city" : "Saint-Malo" }, "count" : 1 }
{ "_id" : { "user" : "Marcussacapuces91", "city" : "Saint-Malo" }, "count" : 1 }
{ "_id" : { "user" : "Cyrano25550", "city" : "Cancale" }, "count" : 1 }
{ "_id" : { "user" : "Gnafron", "city" : "Dinard" }, "count" : 1 }
{ "_id" : { "user" : "Niels Elgaard Larsen", "city" : "Saint-Malo" }, "count" :1 }
{ "_id" : { "user" : "It's so funny", "city" : "Saint-Malo" }, "count" : 1 }
{ "_id" : { "user" : "Acor78", "city" : "Cancale" }, "count" : 1 }
{ "_id" : { "user" : "nbossard", "city" : "Saint-Lunaire" }, "count" : 1 }
{ "_id" : { "user" : "Kalaallit Nunaat", "city" : "Cancale" }, "count" : 1 }
{ "_id" : { "user" : "mappy_luzern", "city" : "Saint-Malo" }, "count" : 1 }
```

Other statistics:

The lack of address information is confirmed with these very low percentage of ways or nodes containing address information:

# nodes	1,100,161
# address in nodes	716
Address ratio in nodes	0.06%

#ways	165,730
#address in ways	593
Address ratio in ways	0.4%

The number ways and nodes were given by a simple count:

```
> db.osmData.find({"type":"node"}).count()
1100161
```

```
> db.osmData.find({"type":"way"}).count()
165730
```

The aggregate function has been used to calculate the number of addresses in nodes and ways:

```
> db.osmData.aggregate({"$match": {"address.street": {"$exists": 1}}},{"$group": {"_id":
"$basic_elem","count": { "$sum": 1}}})
 { "_id" : "way", "count" : 593 }
 { "_id" : "node", "count" : 716 }
```

Conclusion

The information available for the area is rather poor despite the fact that nodes and ways geographical positions are there. A lot of improvement of the existing data can still be completed but an important external contribution seems to be the key to add new information to each nodes and ways.

References

http://wiki.openstreetmap.org/wiki/Map_Features - The list of the most common key attributes used in OSM

http://docs.mongodb.org/manual/reference/operator/aggregation/ - MongoDB aggregation pipeline operators

http://stackoverflow.com/questions/16607359/mongo-shell-in-windows-7-unicode-text-could-not-be-correctly-displayed - The default character set of my MS windows shell terminal does not support Unicode characters resulting in some misleading conclusion on the source of typos in the MongoDB shell output. A Unicode compatible set (lucida) must be chosen