# OpenStreetMap Sample Project

# Data Wrangling using MongoDb

Area of interest Southampton- England

## 1 Area of interest:

Southampton is a city located in south coast of England, about 121 Km from south west of London, and 31 Km north west of Portsmouth. This city is the largest one in the ceremonial county of Hampshire with an estimated population of 253,651 and superficial of 51,47km$^{2.}$


I downloaded my data from this link [https://mapzen.com/data/metro-extracts/metro/southampton_england/](https://mapzen.com/data/metro-extracts/metro/southampton_england/)
the file was sized about 66.2 MB and named southampton.osm

## 2 Problems encountered with the data:

After I downloaded the data, I noticed several problems that needed to be corrected.

### Problematic characters, and the abbreviations.

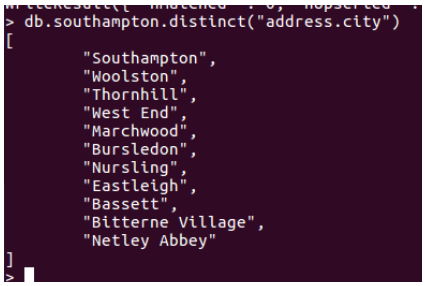
I started by spotting problematic characters and ignoring the tags that contains these ones, then switching any abbreviation in the address to the full name.

### the villages in Southampton were not in the same format.

I had to clean the city, some had Southampton doubled while others didn't , so It was important to chose one and apply it on all the data. This was done using the code working_on_database.py

Old format :


New format :




### the address was mentioned as "is_in" in some documents.

There was some fields that contained "is_in" instead of address and city, so that needed to be switched to the format we choose, using a function update_other that deleted the field and set a new one.

# 3 Wrangling procedure

This part is to explain what has been done to the osm file before inserting it in mongoDB.
The first script is create_json.py, this script do the following things:

- Open the southampton.osm file
- Iterate on the elements to do the following things :
  - -First check the type of the elements, a node or a way based on the first tag.
  - -For elements with way type, we gather the nodes in an array called node_refs
  - -An array called "pos" is created with the longitude and latitude of each node
  - -Filtrate the tags that have a field named "addr:", and correct it into the suitable format then add it to the dictionary.
- Write the json file under southampton.osm.json name

The output of json should gives:

{"amenity": "school",
"name": "Bitterne C of E Junior School",
"created": {"changeset": "36995847",
"version": "3",
"uid": "1540938",
"timestamp": "2016-02-04T11:28:02Z"},
"pos": ["50.9134359", "-1.3620393"],
"address": {"city": "Southampton",
"street": "Brownlow Avenue",
"postcode": "SO19 7BX"},
"type": "node",

"id": "807931773"}

Then generate a json file that has the nodes and the ways in the format we want, with another script called from_json_to_mongo.py we load the json in mongoDB

### Problems while loading json in mongoDB:

While trying to load my data on mongodb, I had troubles because the format wasn't json, weird because I didn't change the process_map function, so the only solution that I have got ( not really classy) is to turn it manually, means that I created a list instead of append it to one, so I added " ," to the fo.write(json.dumps(el)+","+"\n") and then I went to my json file to delete the last ", " , I also added '[]' . that was the only solution I got after many tries, I would appreciate if there is a better solution or the reason why I get this problem.

# 4 Data Overview :

### File sizes

Southampton.osm……....66.2 MB
Southampton.osm.json …...69.2 MB

### Number of documents

>db.southampton.find().count()
6407031

### Number of nodes:

>db.southampton.find({"type":"node"}).count()

5391071

### Number of ways:

>db.southampton.find({"type":"way"}).count()

1015860

### Number of unique users

>db.southampton.distinct("created.user").length
381

### Top 3 contributing user

>db.southampton.aggregate([{"$group":{"_id":"$created.user","count":{"$sum":1}}},{"$sort":{"count":-1}},{"$limit":3}])

{ "_id" : null, "count" : 4412991 }
{ "_id" : "0123456789", "count" : 480460 }
{ "_id" : "pcman1985", "count" : 284140 }

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

> db.southampton.aggregate([{"$group":{"_id":"$created.user", "count":{"$sum":1}}}, {"$group":{"_id":"$count", "num_users":{"$sum":1}}}, {"$sort":{"_id":1}}, {"$limit":1}])

{ "_id" : 17, "num_users" : 75 }

# 5 Additional ideas:

#top bicycle_parking kinds

db.southampton.aggregate([{"$match":{"amenity":{"$exists":1}, "amenity":"bicycle_parking"}}, {"$group":{"_id":"$bicycle_parking", "count":{"$sum":1}}},{"$sort":{"count":-1}}, {"$limit":3}])

{ "_id" : "stands", "count" : 3860 }

{ "_id" : null, "count" : 1320 }

{ "_id" : "wall_loops", "count" : 380 }

# top parking kind of spaces for cars

db.southampton.aggregate([{"$match":{"amenity":{"$exists":1}, "amenity":"parking"}}, {"$group": {"_id":"$parking", "count":{"$sum":1}}},{"$sort":{"count":-1}}, {"$limit":3}])

{ "_id" : null, "count" : 11140 }

{ "_id" : "surface", "count" : 2200 }

{ "_id" : "multi-storey", "count" : 260 }

# Conclusion

In this project we corrected and cleaned data provided from osm of Southampton in England, then downloaded it in mongodb. Southampton data still needs a lot of improvement from users, as there is many missing values and incomplete data .