# Open street map: data wrangling with MongoDB

**Xin Pang**

**Map Area**: Antwerp, Belgium
**Data source**: https://s3.amazonaws.com/metro-extracts.mapzen.com/antwerp_belgium.osm.bz2

## 1. Problems in the dataset

After running the python script for sampling from the full-sized dataset, following problems are found in the sample data.

1. **Different key names for postcode**
   Some are named `addr:postcode`, while some are named `postal_code`. To solve this problem I changed all tags containing the k value of `postal_code` to `addr:postcode` and then process it in the same way as processing `addr:postcode`.

2. **Different key names for contacts**
   Same as the previous problem, in some of the tags the k value is `email`, whereas in other tags the k value is `contact:email`. To solve this problem, following steps are taken.
   a. `contact:email` -> `email`
   b. `contact:phone` -> `phone`
   c. `contact:website` -> `website`

3. **Phone numbers in different format**
   Following formats of phone numbers are found. To make the phone number format consistent, 3.e. is chosen.
   a. +32 (0)3 887 46 12
   b. +32 3 344 77 44
   c. +3234571247
   d. 03 233 26 27
   e. 032354418

4. **Different format of city names**
   a. Capital letters: `Berchem` vs. `berchem`
   b. Some city names are in Dutch, and some in English: `Antwerpen` vs. `Antwerp`
   c. City names with remarks that this city is part of the Antwerp distrct: '`Berchem (Antwerpen)`'
   d. Similar with 4.c., but with hyphen: '`Hoboken-Antwerpen`'
   e. Similar with 4.c., but with comma: '`Borgerhout, Antwerp`'
   f. City names containing special characters: '`<verschillend>`'
   Following regular expression is applied such that all the above mentioned problems can be solved.

```
r'[\(|\-|,|\s]+antwerp[\)]*'
```

5. **Some postcodes are in the Netherlands**
   The postcode in Belgium are four digits e.g. '3001', whereas in Netherlands it is four digits followed by 2 letters e.g. '5611CH'. When doing a check in the data sample, I found several post

codes from the Netherlands, such as 4645BD. This address comes from a city called Putte. After locating it in the map, it seems that people confused this town in the Netherlands with Putte in Belgium, because they have the same name. Therefore all postcodes from the Netherlands should be removed from the dataset. To do this, a simple regular expression is applied such that all nodes containing Dutch postcodes are removed.

## 2. Data overview

1. File size
    a. osm file: 250M
    b. json file after cleaning: 265M

All data are stored in the database named `'osm'` (DB_NAME = "osm") with collection name `'antwerp'` (COLLECTION = "antwerp").

2. Number of documents

```
> db[COLLECTION].find().count()
1281090
```

3. Number of nodes

```
> db[COLLECTION].find({"type":"node"}).count()
1063128
```

4. Number of ways

```
> db[COLLECTION].find({"type":"way"}).count()
217962
```

5. Number of users

```
> len(db[COLLECTION].distinct("created.user"))
947
```

6. Number of cities

```
> db[COLLECTION].aggregate([
                        {"$match":{"address.city":{"$exists":True}}},
                        {"$group":{"_id":"$address.city",
                                   "count":{"$sum":1} } },
                        {"$sort":{"count":-1} } ])
48
```

## 3. Other ideas

1. Number of users and their contributions

```
> db[COLLECTION].aggregate([
                        {"$group":{"_id":"$created.user",
                                   "count":"$sum": 1} } },
                        {"$sort" :{"count": -1} },
                        {"$group":{"_id":"$count",
                                   "count_of_users": {"$sum": 1}}} ])
```

Following section of code plots the number of users, based on how many documents they contribute to OSM. From the plot it can be seen that most of the users contribute only 1 document, and the majority of users contribute less than 10 documents. It is very rare that users contribute at the level of $10^5$ documents. Given this, one possible solution could be to provide multiple ways of login, especially connect with location based services such as Foursquare, to enable users to upload more data when using those apps.

```
> plot_basic = ggplot(aes(x='_id', y='count_of_users'),
                          data=documents_by_user)
```

```
> plot = plot_basic + geom_point() \
                    + geom_line() \
                    + scale_x_log() \
                    + scale_y_continuous(limits=(0,250)) \
                    + xlab('Number of documents uploaded to OSM') \
                    + ylab('Number of users')
> print plot
```