# Project 3: Data Wrangle OpenStreetMap

---

For this project city of Kyiv, Ukraine (link to osmap (https://www.openstreetmap.org/relation/421866)), was chosen because the author of the project is originated and almost all her life lived in this city.

## Section 0. References

---

Problems with the incoding while saving to .json

https://docs.python.org/2/library/json.html (https://docs.python.org/2/library/json.html)

Documentation about creating and querying to geoindex in mongoDB:

http://docs.mongodb.org/manual/tutorial/build-a-2dsphere-index/ (http://docs.mongodb.org/manual/tutorial/build-a-2dsphere-index/)
http://docs.mongodb.org/manual/tutorial/query-a-2dsphere-index/ (http://docs.mongodb.org/manual/tutorial/query-a-2dsphere-index/)

Using $near operator:

http://docs.mongodb.org/manual/reference/operator/query/near/#op._S_near (http://docs.mongodb.org/manual/reference/operator/query/near/#op._S_near)

---

## Section 1. Problems encountered in the map

In fact, I hadn't encountered any massive problems with the data, that can be cured programmatically.

Initial feeling was that there are also can be some problems with the street naming (e.g. shortenings like "вулиця" -> "вул.", "проспект" -> "Пр"). So I collected a dictionary with the "street types" in keys in all street names with this type in value (script streets.py). Doing so

with the sample file (kyiv_ukraine_sample.osm) I found only 2 items (streets) that didn't end with the conventional street name. Keys of the output dictionary:

```
Орача
тупик
шосе
Васильковская
вулиця
провулок
площа
набережна
проспект
узвіз
бульвар
```

Problems are "Орача" and "Васильковская". It is really names of the streets, but without the indicator - what kind of street they are - street, boulevard, ... so can be fixed only by hand. As it is really small number of this cases (2 of 813 entries with streets) I decided to leave that as is.

Then, while running script shaping and converting .osm to .json I caught problems with the wrong encoding of the output file. After investigating it became clear, that problem is in occuring while writing to .json in json.dumps. Fixed with adding the ensure_ascii=False parameter.

---

## Section 2. Data overview

```
Originally downloaded and unzipped file:

kyiv_ukraine.osm ............... 255.5 MB,

Transformed file:

kyiv_ukraine_osm_sd2.json ...... 328.1 MB.
```

---

Note - structure of .json was changed so that geoingex can be built:

originally:

node["pos"] = [longitude, latitude]

changed to:

node["loc"]["coordinates"] = [longitude, latitude]

---

**Number of documents in database**:

```
> db.kyiv_ukraine_2ds.find().count()
> 1316124
```

**Number of nodes**:

```
> db.kyiv_ukraine_2ds.find({"type" : "node"}).count()
> 1146884
```

**Number of ways**:

```
> db.kyiv_ukraine_2ds.find({"type" : "way"}).count()
> 169057
```

Case with the **unique users** was somewhat interesting.

I calculated the number of unique uids in two ways - in the initial .xml (exercise from Lesson 6) and in the mongo database, imported from shaped .json:

db.kyiv_ukraine_2ds.distinct("created.uid").length

First result - 1502, second - 1483. In both cases it is obviously to few users for such a big city (around 3 million people), but what the cause of difference? Well, I saved uids in two .txt files, found difference ($ comm -23 uids_python.txt uids_mongo.txt ), got uids list, and greped entries, which was modified by those uids. It appeared, that they added only **relations** components.

In any case, our community must do much for the localization of OSM project to recruit new uids. Because language barrier surely is a problem fore many and some pages in wiki are not translated (e.g. https://wiki.openstreetmap.org/wiki/OSM_XML (https://wiki.openstreetmap.org/wiki/OSM_XML)).

---

Further I decided to investigate region of the city, that was familiar for me to check for the complitness.

**Number of houses on my street**:

```
> db.kyiv_ukraine_2ds.find({"address.street" : "Лайоша Гавро вулиця",
"building" : {"$exists" : 1}}).count()
> 27
```

There are, actually, more buildings on my street. What is

**Near my house**:

```
> db.kyiv_ukraine_2ds.find( { loc :
                            { $near :
                             { $geometry :
                                { type : "Point" ,
                                  coordinates : [ 30.5213587, 50.453716
] } ,
                               $maxDistance : 500
                        } } } ,  amenity : {"$exists" : 1}})
```

Types of objects:

```
> "cafe"
"atm"
"veterinary"
"post_office"
"post_office;bank;atm"
"drinking_water"
"pharmacy"
"post_office"
```

```
Those are really exist, but in real life there are some more. So from
this investigation I assume, that completeness of data for my city is
insufficient. It coincide with not very great number of contributors
and popularity of maps in my country. Which is a pity.
```

---

# Section 3. Other ideas about the datasets

---

**(Added idea)**:

As it was said previously - there are only about 1500 contributors in Kyiv. Of course, we need more, but, as these guys have an experience of working with OSMaps, maybe it would worth to remind them, that they have not added or corrected something for quite a long time? Moreover, to sort them by oldness of the last edit? Then we can somehow show them the reminder, or rating of the newest editions. Everybody likes to see their names on top, so this will be kind of gamification.

So I found uids, whose last editions are the oldest. Pipeline:

```
db.kyiv_ukraine_2ds.aggregate([{"$match" : {"created.timestamp":
{"$lte": "2014-09-24"}}}, {"$group": {_id : "$created.uid",
last_change : {"$max" : "$created.timestamp"}}}, {"$sort" :
{"last_change" : 1}}, {"$limit" : 15}])
```

output:

```
{ "_id" : "229", "last_change" : "2007-04-07T10:51:16Z" }
{ "_id" : "6157", "last_change" : "2007-08-29T10:25:00Z" }
{ "_id" : "11810", "last_change" : "2008-01-30T03:41:27Z" }
{ "_id" : "682", "last_change" : "2008-04-07T11:17:50Z" }
{ "_id" : "33449", "last_change" : "2008-04-11T15:02:27Z" }
{ "_id" : "33055", "last_change" : "2008-04-14T13:41:41Z" }
{ "_id" : "33536", "last_change" : "2008-04-21T11:36:16Z" }
{ "_id" : "69865", "last_change" : "2008-09-26T16:02:57Z" }
{ "_id" : "36214", "last_change" : "2008-10-02T09:19:48Z" }
{ "_id" : "33503", "last_change" : "2008-10-19T07:39:40Z" }
{ "_id" : "59359", "last_change" : "2008-10-19T14:01:42Z" }
{ "_id" : "12459", "last_change" : "2008-10-24T01:39:36Z" }
{ "_id" : "17497", "last_change" : "2008-11-03T09:43:51Z" }
{ "_id" : "60146", "last_change" : "2008-11-03T13:25:20Z" }
{ "_id" : "24126", "last_change" : "2008-11-03T23:16:17Z" }
```

So we can remind them (e.g. by e-mail) that "you've done great job previously,..., please, as you are experiensed user - make some more and be at the top of constantly active contributors".

---

*I leave my previous idea here, I think it is somehow better. Maybe there is a chance to restore street name from coordinates of node, and I'm still thinking about it*.

My idea was to investigate proportion of the amenities in the street to the length (in number of houses) of the street. I assume, that for the most streets the number of pharmacies or kindergartens has to be proportional to the dimension of the street for convenience of the

population of that street. But, as I figured out from the previous investigation, for the most amenities only coordinates included in the information, not the name of the street. And number of amenities over the whole city is too few:

```
> db.kyiv_ukraine_2ds.find({"amenity" : {"$exists" : 1}}).count()
11942
> db.kyiv_ukraine_2ds.find({"amenity" : {"$exists" : 1},
"address.street" : {"$exists" : 1}}).count()
838
```

example of the amenity entry on my street:

```
{ "_id" : ObjectId("55fb87b6926b0f43bbafe0ab"), "loc" : { "type" :
"Point", "coordinates" : [ 30.5069079, 50.4949295 ] }, "amenity" :
"cafe", "type" : "node", "id" : "258925275", "created" : { "user" :
"КомЯра", "version" : "2", "uid" : "29320", "timestamp" : "2011-09-
13T16:53:33Z", "changeset" : "9290837" } }
```

so I can't really prossed with my idea, because of the poor completness.

Usefulness from the output has to be to identify those streets with the lack of some amenities (comparing to mean or median value). Reason for this lack can be either some amenities not marked on the map, and contributors from the street, seeing the alert, can cure the matter, or what street was not built taking into account needs of population, so this can raise social awareness.

In [ ]: