Wrangle OpenStreetMap data

Udacity Data Analyst Nanodegree - Project 3

Overview

I chose Cologne, http://www.openstreetmap.org/#map=11/50.9387/6.8740, as my preferred map area, because before moving to Milan, Italy, where I currently live, Cologne used to be my home for several years. I know this city quite well and I am curious to improve its OSM data.
File size is about 376 MB:

```
!ls -lh cologne_germany_sample.osm
-rw-r--r--  1 stefan  staff    38M Nov 10 22:08
cologne_germany_sample.osm
```

Problems encountered

A brief summary of the problems encountered during data exploration:

- The **city key does not only store the value "Köln"** (Cologne in German), but information regarding districts and cities other than Cologne: {'Bergisch Gladbach', 'Hürth', 'Köln', 'Köln Rath/Heumar', 'Köln-Nippes'}
- Addresses are surprisingly well formated and are **not** considered an issue
- Two keys are used to store information regarding Cologne' districts (**addr:suburb and addr:district**) and postal codes (**postal_code vs postcode**)
- Opening hours **lack a standard schema** for notation (see examples provided below)
- Analysis of problem char keys (see below) gave a good overview of potentially interesting data points, however problem char values were often not real issues given the German language (consider the ß in Straße)
- Interestingly, OSM users themselves reported data point to be fixed using the **FIXME** tag

Results of my audit function audit_keys() using regular expressions defined within the course:

```
{'lower': 75642, 'lower_colon': 76328, 'other': 1744, 'problemchars':
457}
```

This seems like a huge number of issues, but interestingly **most of lower keys and values were not issues at all**:

First 5 items of **lower** keys

```
{'abandoned',
 'across',
 'administrative',
 'adult_gaming_centre',
```

```
 'advertising',
```

First 5 items of **lower** values:

```
{'addr:city',
 'addr:country',
 'addr:district',
 'addr:housename',
 'addr:housenumber',
```

As far as **problem chars keys** were concerned, the following items were retrieved

```
{'step.condition',
 'step.height',
 'step.length',
 'strassen-nrw:abs': ',
 'surface.material''}
```

Problem Chars values were much harder to judge, since often a key considered problematic was correct according to German spelling. The following three findings are pretty clear:

```
{'51143,51145',
 'Köln Rath/Heumar'
 'Köln-Nippes'}
```

The same is true for various items within the “**other**” category:

```
2014-06-03: Umbau, Haus vollständig eingerüstet\n2016-03-02: 2016-03-02:
Gebäude ist fertiggestellt, es wird ein Mieter für das Ladengeschäft
gesucht',
'Gedankengut...',
Mo-Fr 10:00-13:00, 14:00-18:00; Sa 10:30-13:00; Su, PH off'
```

Fixing problems

In order to fix the problems encountered above, I wrote my own function fix_problems(), which took a mapping (for problem char keys) and the previously identified problem chars values as input for the cleaning.Programmatically cleaning problem chars values was difficult, since often items identified as problem chars were actually valid, i.e the name-of-a-street can not be replaced by name_of_a_street in German, since the former version is the correct way to spell it. Therefore I decided to only clean via mapping, because doing it differently had done more harm than good. Further, findings in the lower and lower_colon category were not considered an issue at all. Plus, if one would want to clean the opening times values, he would first have to define a standard schema.

The following fixes were applied:

```
{'51143,51145': '51143',
 'Köln Rath/Heumar': 'Köln',
 'Köln-Nippes': 'Köln',
 'step.condition': 'step:condition',
 'step.height': 'step:height',
```

```
'step.length': 'step:length',
'strassen-nrw:abs': 'strassen_nrw:abs',
'surface.material': 'surface:material'}
```

Analyzing data in MongoDB

Summary statistic	**Value**	**Query used**
Number of documents in database	187163	db.find().count()
Number of nodes	156596	db.find({"type": "node"}).count()
Number of ways	30563	db.find({"type": "way"}).count()
Number of unique users	886	len(db.distinct("created.user"))
Number of shops	453	shop_query = db.aggregate([{"$match": {"shop": {"$exists": 1}}}, {"$group": {"_id": None, "count": {"$sum": 1}}}])
Count number of documents in cologne-ehrenfeld district (the district I used to live in)	308	ehrenfeld_query = db.aggregate([{"$match": {"address.postcode": "50823"}}, {"$group": {"_id": None, "count": {"$sum": 1}}}])

Other ideas about the data set

Additional analysis

How much content did the top 5 users create?

The top 5 users created 70% of the documents in the database (130536/187163)

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

What are the top 5 amenities?

Cologne seems to be a city where parking is difficult, since the top amenity is considered parking (with bicycle parking at the fourth position). Further, apart from restaurants and benches, post boxes are listed

```
amenity_query = db.aggregate([
        {"$match": {"amenity": {"$exists": 1}}},
        {"$group": {"_id": "$amenity", "count" : {"$sum": 1}}},
        {"$sort": {"count": -1}},
        {"$limit" : 5}
    ])
```

Which are the top 3 cuisines?

Although only a few documents contain cuisine tags, the result is not unusual for Cologne. Being a city with a numerous Italian and Turkish community, top three cuisines are considered Italian, Turkish (Kebab is my proxy for Turkish here) and German.

```
cuisine_query = db.aggregate([
        {"$match": {"cuisine": {"$exists": 1}}},
        {"$group": {"_id": "$cuisine", "count" : {"$sum": 1}}},
        {"$sort": {"count": -1}},
        {"$limit" : 3}
    ])
```

Which historic sites exist in Cologne?

The most frequent historic site is labeled memorial, probably related to world war 2

```
historic_query = db.aggregate([
        {"$match": {"historic": {"$exists": 1}}},
        {"$group": {"_id": "$historic", "count" : {"$sum": 1}}},
        {"$sort": {"count": -1}}
    ])
```

What are the top 5 shops?

No surprises here, the kiosk a.k.a. "büdchen", which means small shop where you can buy alcohol late at night in German, is the top shop in Cologne

```
shop_query2 = db.aggregate([
        {"$match": {"shop": {"$exists": 1}}},
        {"$group": {"_id": "$shop", "count" : {"$sum": 1}}},
        {"$sort": {"count": -1}},
        {"$limit" : 5}
    ])
```

Which leisure venues exist in Cologne?

After consulting with Google translate, I learned that pitch is considered a synonym to playground. Thus having pitch as the most frequently listed leisure venue in Cologne makes sense

```
leisure_query = db.aggregate([
        {"$match": {"leisure": {"$exists": 1}}},
        {"$group": {"_id": "$leisure", "count" : {"$sum": 1}}},
        {"$sort": {"count": -1}}
    ])
```

How many gay venues exist in Cologne?

Apparently there exist only 1 gay venue in our sample, which is clearly a strange result for Cologne (the capital of gays in Germany).

```
gay_query = db.aggregate([
        {"$match": {"gay": {"$exists": 1}}},
        {"$group": {"_id": None, "count" : {"$sum": 1}}},
    ])
```

Actions to improve data quality

Besides taking care of the issues listed under problems encountered, Cologne's OSM data could be further enhanced by an organizational measure. Using a meetup group, one could find like minded people that come together once a month and do a OSM hackathon: They would identify one or two particular problems with Cologne's OSM data and thrive to solve these during the hackathon. Doing so, a lot of currently existing problems could be fixed easily. Further, by fixing problems OSM data gets more valuable for businesses, which may in return sponsor some stuff (drinks or food) for the hackathon. The major challenge here is to motivate users to come together regularly and to convince businesses of the value of OSM and open data in general.

Conclusion

After my short review of Cologne's OSM data I am surprised of the general quality of the data, especially related to street names. Additionally OSM users tag documents which need to be fixed with corresponding tags (FIXME), which makes it easy for new users to contribute. Two problems exist which could be fixed easily (postal code and city tags), further opening times are recorded lacking a standard input format. Considering Cologne's status as the capital of gay people in Germany, it is surprising that only one document carries the tag "gay".

Resources

- Udacity course materials
- Python 3 documentation
- MongoDB documentation
- MongoDB driver documentation