Michael Rabe

05.02.2021

Wrangle OpenStreetMap Data

1 Map Area

Berlin, Germany

- https://www.openstreetmap.org/#map=14/52.5186/13.41135
- https://overpass-api.de/api/map?bbox=13.3735,52.4994,13.4492,52.5378

Berlin is the capital city of germany and not far away from my hometown. Several times per year i visit Berlin. My brother is living in Berlin since many years.

2 Audit

The following fields have been checked using the ***audit.py*** file.

key	Findings	Description
"addr:street"	No	Street names in germany are very diverse. The extracted osm data did not show any problematic value.
"addr:city"	No	All available address data for citiy was assigned to “Berlin”.
"addr:country"	No	All available address data for country was assigned to “DE” (**De**utschland).
"addr:suburb"	**Yes**	**Berlin is subdivided into 12 boroughs, which are again subdivided in individual districts.** **For around 50% of the entries a borough was defined and for the other 50 % a district was given.** **To simplify later SQL queries I decided to harmonize the suburb field and use only borough names (see next chapter for more details).**
"addr:postcode"	No	All counted postcodes are located within the Berlin postcode range.
"country"	No	Available data for country has shown multiple nationalities. The osm data was extracted from berlin city center where,the german government is located. Close to the german government many diplomatic missions are located. Since embassies are property of the embassy country it the different nationalities shown up are not unusual.
"cuisine"	**Yes**	**Several entries per field divided by semicolons. In general difficult to clean/harmonize since the values are user defined. Nevertheless some typo mistakes are identified and corrected as well as some crystal clear double definition are harmonized (see next chapter for more details).**

Michael Rabe

05.02.2021

3 Problems Encountered in the Map

As discussed in chapter 2 two problems have been identified during data screening:

- Inconsistent “addr:suburb” values
- Multiple cuisine values separated by semicolon, typo mistakes in cuisine nationality entries

3.1 Inconsistent “addr:suburb” Values

Berlin is divided into 12 main districts also called boroughs. Below each borough several districts are assigned. A good overview can be found here:

https://de.wikipedia.org/wiki/Verwaltungsgliederung_Berlins#Bezirke

The osm data shows randomly the borough name or district name. I decided to harmonize the names to have only the higher level borough name in my database.

I have send a request to the above mentioned webpage and extracted with BeautifulSoup the berlin borough data and stored that in a dictionary (see ***add_help_functions.py***)

SQL Query and Output:

```
SELECT value,
       count( * ) AS count
  FROM (
           SELECT *
             FROM nodes_tags
           UNION ALL
           SELECT *
             FROM ways_tags
       )
 WHERE [key] = "suburb"
 GROUP BY value
 ORDER BY count DESC;

                    borough  count
0                    Mitte   8172
1  Friedrichshain-Kreuzberg   4410
2                   Pankow   3007
```

3.2 Multiple “cuisine” Values

Cuisine values showing multiple entries divided by a semicolon. To solve this the value is split into a list and the entries have been screened against the cuisine_mapping dict.

Finally a corrected list was returned and for each list entry a row was dropped into the tags csv file.

Top 10 cusine:

```
SELECT value,
       count( * ) AS count
  FROM (
           SELECT *
             FROM nodes_tags
           UNION ALL
           SELECT *
             FROM ways_tags
       )
 WHERE [key] = "cuisine"
 GROUP BY value
 ORDER BY count DESC
 limit 10

       cuisine  count
0      italian    138
1  coffee_shop     94
2       german     82
3        asian     74
4   vietnamese     68
5       burger     63
6     regional     50
7        pizza     48
8        kebab     42
9        sushi     41
```

4 Data Overview and Additional Ideas

This section contains basic statistics about the dataset, the SQL queries used to gather them, and some additional ideas about the data in context.

4.1 File sizes

```
Berlin_OSM_v2.osm     76.3 MB
nodes.csv             20.5 MB
nodes_tags.csv         7.8 MB
ways.csv               2.2 MB
ways_nodes.csv         7.6 MB
ways_tags.csv          6.0 MB
berlin_osm.db         40.5 MB
```

4.2 Number of nodes

```
SELECT count( * ) AS count
  FROM nodes;

   Node Count
0      261523
```

4.3 Number of ways

```
SELECT count( * ) AS count
  FROM ways;

   Way Count
0      38657
```

4.4 Number of unique users

```
SELECT count( * )
  FROM (
           SELECT user,
                  count( * ) AS count
             FROM (
                      SELECT user
                        FROM nodes
                      UNION ALL
                      SELECT user
                        FROM ways
                  )
            GROUP BY user
            ORDER BY count DESC
       );

   user
0  1705
```

4.5 Top 10 contributing users

```
SELECT user,
       count( * ) AS count
  FROM (
           SELECT user
             FROM nodes
           UNION ALL
           SELECT user
             FROM ways
       )
 GROUP BY user
 ORDER BY count DESC
 LIMIT 10

         user  count
0  atpl_pilot  65588
1    Bot45715  16918
2   sandrow75  14011
3     joe2812  12397
4     toaster  10652
5       MorbZ  10593
6   kartonage  10396
7  kartograph   7114
8        anbr   6720
9        kjon   6664
```

5 Other Ideas About the Dataset

While screening the address data, I have observed that some addresses are not complete. As shown below the counted numbers for different address keys are not equal as I have assumed it should be. Of course sometimes in germany not all addresses have a housenumber if they are located at a square for example. But postcode, country and street should be the minimum of what must be in the osm data.

```
SELECT [key],
       count( * ) AS count
  FROM (
           SELECT *
             FROM nodes_tags
           UNION ALL
           SELECT *
             FROM ways_tags
       )
 WHERE type = "addr"
 GROUP BY [key]
 ORDER BY count DESC
 LIMIT 6;

      addr key  count
0  housenumber  16984
1       street  16845
2     postcode  16817
3         city  16796
4      country  15656
5       suburb  15589
```

I run a short study using **add_missing_addr.py** to identify one of the locations. I catched the node shown below, where the city and postcode are missing.

```
69226073
{'addr:housenumber': '104',
 'addr:street': 'Leipziger Straße',
 'meta': {'lat': '52.5104026',
          'lon': '13.3894102',
          'ref_node': 69226073,
          'tag': 'node'}}
```

The database quality could be improved with **geopy** which allows to get missing information and add it to the osm data before passing the data into the database.

https://geopy.readthedocs.io/en/stable/

https://nominatim.openstreetmap.org/ui/reverse.html

With geopy we can send latitude and longitud to the openstreetmap service nominatim service. For node 69226073 and lat/lon: ('52.5104026' , 13.3894102') location address can be returned:

```
Sixt, 104, Leipziger Straße, Mitte, Berlin, 10117, Deutschland
```

Postcode and city name could now be easily extracted and added.

Disadvantage is the low performance if to many requests are required.

6 Additional Data Exploration

Number of trees:

```
SELECT count( * ) AS count
  FROM nodes_tags
 WHERE value = "tree";

   Tree Count
0       10166
```

Top 10 Amenities:

```
SELECT value,
       count( * ) AS count
  FROM (
           SELECT *
             FROM nodes_tags
           UNION ALL
           SELECT *
             FROM ways_tags
       )
 WHERE [key] = "amenity"
 GROUP BY value
 ORDER BY count DESC
 limit 10

           amenity  count
0  bicycle_parking   1800
1            bench   1200
2       restaurant    896
3             cafe    486
4     waste_basket    413
5          parking    367
6        fast_food    285
7  vending_machine    228
8              atm    185
9     kindergarten    178
```

Berlin is a green city: A lot of trees, a lot of benches to rest and of course thousands of bicycles which needs parking areas.

7 Conclusion

The berlin osm data I have extracted was in a very good quality from my perspective. One of the problem was, that the node/way data was incomplete. But such problems could be solved by doing a proper and much deeper cleaning/screening than I did here. For example geopy could help to add missing address data to the osm.

In my opinion the user has too much possibility for user defined inputs. I would appreciate a stricter and harmonized approach.