Open Street Map:

Case study for data wrangling and SQL database

By: Rakpong Kittinaradorn

Date: 18 November 2016

Map area

Bangkok, Thailand

https://www.openstreetmap.org/relation/92277#map=10/13.5862/100.6332

https://mapzen.com/data/metro-extracts/metro/bangkok_thailand/

This area is my hometown and also one of the best place in the world for food lover. I am interested in querying information about food and restaurant.

List of Files

> - slice.py: sample data to manageable size
> - sample.osm: sample data
> - parse.py: count number of each tags
> - way_tag_unique_k.py: examine unique 'k' tag in way element
> - investigate_tag.py: investigate pattern in 'k' tag
> - audit.py: list all unexpected street name and postal code
> - pre_database.py: code for updating data and write csv files for the database

Problems Encountered in Map Data

The whole dataset is large (~360 Mb), so I sample it for preliminary exploration. After running parse.py, way_tag_unique_k.py, investigate_tag.py and looking into sample.osm, I

found 3 main problems needed to be addressed before import it into the database.

- Inconsistent translation and transliteration from Thai to English (ex. Temple vs. Wat (transliteration of Temple (วัด))
- Street name appears in multiple syntaxes
- Abbreviated street name (ex. Rd. for Road)
- Incorrect postal code form

Inconsistent translation and transliteration

Thai words are usually translated into English but some entries has its transliteration instead. The example are the word “Wat (วัด)” which can be translated into “Temple” and “Thanon (ถนน)” can be translated into “Road”. Note that transliteration word is usually a prefix while translated word is a suffix, i.e. Thanon Sukhumvit is mapped to Sukhumvit Road.

Street name appears in multiple syntaxes

Street name appears in two different format. The first one is in parent element ‘node’. Street name is in its child element “tag” with attribute ‘k’ = ‘addr:street’.

```
<node … >
        <tag k="name" v="Elle Tha Pra Chan" />
        <tag k="amenity" v="restaurant" />
        <tag k="cuisine" v="thai" />
        <tag k="addr:street" v="Thanon Maharat" />
        <tag k="addr:housenumber" v="172" />
</node>
```

The second one is in parent element ‘way’. When it has child element ‘tag’ with ‘k’ = ‘highway’ and another one with ‘k’ = ‘name:en’. The street name is in the one with ‘k’ = ‘name:en’.

```
<way … >
    <nd ref="249643562" />
    …
    <nd ref="249643519" />
ถนนดำรงรักษ์ k="name" v="            " />
    <tag k="highway" v="residential" />
    <tag k="name:en" v="Thanon Rachchadamnern" />
ถนนดำรงรักษ์ k="name:th" v="            " />
</way>
```

In cleaning street name, I need to take into account both formats.

Abbreviated street name

Some street names use abbreviation form (ex. Rd.) while most of them is in its full form (ex. Road). I list all abbreviation by running audit.py and wrote a script to map all abbreviated one to its full format.

The updating code for both abbreviation and transliteration are as follows.

```
mapping1 = {"St": "Street",
            "St.": "Street",
            "Rd.": "Road",
            "Rd": "Road",
            "rd": "Road",
            "rd.": "Road",
            "Ave": "Avenue",
            "Ave.": "Avenue"
            }

mapping2 = {"thanon": "Road",
            "Thanon": "Road",
            "Wat": "Temple"
            }

def update_name(word_in, mapping1, mapping2):
    name_list = word_in
        .replace(')','')
        .replace(' (',';')
        .replace('(',';')
        .split(';')
    for i, name in enumerate(name_list):
        parts = name.split(' ')
        if parts[-1] in mapping1:
            parts[-1] = mapping1[parts[-1]]
        if parts[0] in mapping2:
            parts.append(mapping2[parts[0]])
            parts.pop(0)
        name_list[i] = " ".join(parts)
        if i != 0:
            name_list[i] += ')'
    word_out = " (".join(name_list)
    return word_out
```

Incorrect postal code form

Postal code in Thailand should have 5 digits. Some postal codes are not in this form. I will update them by keeping the first 5 digits and cut everything else with this code.

```
def update_postcode(word_in):
    word_out = re.match(r'[0-9]{5}',word_in).group(0)
    return word_out
```

Data Exploration using SQL queries

After data cleaning, I print csv files by running pre_database.py and import it into sql database. The importing procedure are as follows.

```
>sqlite3 test.db
sqlite>.mode csv [table name]
sqlite>.import [file name] [table name]
```

Following are the explorations of this dataset by SQL queries.

Number of nodes

```
SELECT COUNT(*) FROM nodes;
```

1679039

Number of ways

```
SELECT COUNT(*) FROM ways;
```

226371

Number of unique users

```
SELECT COUNT(DISTINCT(e.uid))
FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) as e;
```

2070

Top 10 amenities

```
SELECT value, COUNT(*) as num
FROM nodes_tags
WHERE key='amenity'
GROUP BY value
ORDER BY num DESC
LIMIT 10;
```

Amenities	Count
restaurant	2558
cafe	1263

Amenities	Count
bar	1077
fuel	952
place_of_worship	867
bank	566
atm	523
telephone	512
parking	403
school	345

Food related (restaurant, cafe, bar) amenities come at the top as expected. Place of worship is also numerous as Bangkok has a lot of Buddhist temple.

Top cuisine

```
SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
    JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='restaurant') i
    ON nodes_tags.id=i.id
WHERE nodes_tags.key='cuisine'
GROUP BY nodes_tags.value
ORDER BY num DESC
LIMIT 10;
```

Cuisine	Count
thai	249
japanese	92
indian	78
international	41
italian	39
vegetarian	38
pizza	37

Cuisine	Count
regional	37
chinese	33
seafood	33

As a local, I agree Thai and Japanese are the most numerous. Indian though is surely should not be the third place. I am certain Chinese and Western food are much easier to find in Bangkok.

Top religion

```
SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
    JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='place_of_worship')
    ON nodes_tags.id=i.id
WHERE nodes_tags.key='religion'
GROUP BY nodes_tags.value
ORDER BY num DESC
LIMIT 5;
```

Religion	Count
buddhist	625
christian	27
muslim	21
hindu	11
taoist	8

Additional Ideas

Guideline for transliteration

As I have seen inconsistencies between transliteration and translation, it would be nice if Open Street Map can provide some guidelines about this issue to the contributors. This would make data cleaner from the source level.

Conclusion

The information in Open Street Map is rich but it is far from complete. As Indian cuisine comes at the third place, I think the distribution of the contributor might not sparse enough. To gather more information, a game like Pokemon Go might help motivating local people to walk around and help completing map in process.