OpenStreetMap Data Case Study

Map Area

Gunsan, Jeollabuk-do, Republic of Korea

- https://www.openstreetmap.org/export#map=10/35.9769/126.2219 (https://www.openstreetmap.org/export#map=10/35.9769/126.2219)

This map was downloaded at http://overpass-api.de/query_form.html (http://overpass-api.de/query_form.html) using overpass API query form: `(node(35.6584, 125.4968, 36.2974, 126.9456);<;);out meta;`

Since Gunsan is my home, I want to know what a database query reveals. I want an opportunity to contribute to the improvement of OpenStreetMap.org.

Problems Encountered in the Map

After initially downloading an original OSM of Gunsan, it is sampled to the 1/20 of the orignal size using `sample_osm.py`. After processing of the sample against the data.py file, I noticed five main problems with the data, which I will discuss in the following order:

- Deprecated format of postal codes*("399-5", “573-350”.)*
- Deprecated foramt of address*("흑암동", "금봉동".)*
- Impractical and hard-to-read romanization of Korean names*("Uiryobeopinganggyeonguiryojaedandasaranghyoyoyangbyeongwon", "Janghangseongnugabyeongwonaptaeksihochuljeonhwa", "Malgeunsingyeongjeongsingwachoemyeonkeullinik".)*

Postal Code

The postal code system of the Republic of Korea was revised three times since it was first enacted on July 1, 1970, and the final revision took place in 2015(ref (https://ko.wikipedia.org/wiki/%EB%8C%80%ED%95%9C%EB%AF%BC%EA%B5%AD%EC%9D%98_%E
In the last revision, the current five-digit system replaced the former six-digit system.

On the map, I looked at zip codes that did not match the current postal code format and how much they were used.

```
sqlite> select tags.value, count(*) as count
from (select * from nodes_tags union all select * from ways_tags) tags
where tags.key = 'postcode' and (tags.value like '%-%' or length(tags.valu
e) != 5)
group by tags.value
order by count desc;
541102|3
333707|1
345-804|1
399-5|1
560-033|1
560-500|1
566-844|1
573-350|1
811-5|1
856-1|1
872-12|1
```

Address Format

The new address format based on the road was first enacted on April 5, 2007, and fully used since 2014. (https://goo.gl/CKqs9e (https://goo.gl/CKqs9e)) However the address format based on the district still remains at the map. The old address format can be identified with the following "동(Dong)" at the and of the district name. `<tag k="name" v="흑암동" />` These old-fashioned address formats are found on maps so much that the new address format seems to have not been firmly established yet.

I looked into how many old address formats are coming up. The obstacle to the programmatic search is that '동' is also used as a building number notation; `<tag k="name" v="105동" />`. This type of building number is so diverse that it is almost impossible to process programmatically.

```
sqlite> select tags.value, count(*) as count
from (select * from nodes_tags union all select * from ways_tags) tags
where tags.key = 'name' and tags.value like '%동' and tags.value not like
'%아파트%' and tags.value glob '[^0-9]*'
group by tags.value order by count desc;
가동|3
교동|3
금암동|3
나동|3
나성동|3
중동|3
B동|2
다동|2
동산동|2
반월동|2
...
```

Korean Romanization Notation

"k" tags with the value "name:ko_rm" is Korean romanization notation. It is intended to provide a help foreigners to read Korean names. However, lots of values are hard to read aloud. Camel case or spacing is strongly required. The followings are the 10 longest romanizations without spacing.

```
sqlite> select length(tags.value), tags.value
from (select * from nodes_tags union all select * from ways_tags) tags
where tags.key = 'ko_rm' and tags.value not like '% %'
order by length(tags.value) desc
limit 10;

58|Uiryobeopinganggyeonguiryojaedandasaranghyoyoyangbyeongwon
47|Janghangseongnugabyeongwonaptaeksihochuljeonhwa
45|Malgeunsingyeongjeongsingwachoemyeonkeullinik
45|Uiryobeopinsilloamuiryojaedansilloambyeongwon
45|Sunchanggurimchodeunghakgyobyeongseolyuchiwon
44|Buanchangbukchodeunghakgyobyeongseolyuchiwon
44|Imsildeokchichodeunghakgyobyeongseolyuchiwon
43|Gyeongcheonchodeunghakgyobyeongseolyuchiwon
43|Byeongnyangchodeunghakgyobyeongseolyuchiwon
43|Imsilsamgyechodeunghakgyobyeongseolyuchiwon
```

Data Overview

The Kind of the Tag and Number of Occurance

```
$ python mapparser.py
{'member': 9904,
 'meta': 1,
 'nd': 993827,
 'node': 843894,
 'note': 1,
 'osm': 1,
 'relation': 424,
 'tag': 238463,
 'way': 83851}
```

File sizes

```
$ ls -lh *.db *.osm *.csv
-rw-r--r--  1 kwchun  staff    89M  5 22 22:52 gunsan.db
-rw-r--r--  1 kwchun  staff   172M  5 21 23:12 gunsan.osm
-rw-r--r--  1 kwchun  staff   8.7M  5 21 23:36 gunsan_sample.osm
-rw-r--r--  1 kwchun  staff    68M  5 22 21:22 nodes.csv
-rw-r--r--  1 kwchun  staff   2.7M  5 22 21:22 nodes_tags.csv
-rw-r--r--  1 kwchun  staff   4.9M  5 22 21:33 ways.csv
-rw-r--r--  1 kwchun  staff    23M  5 22 21:33 ways_nodes.csv
-rw-r--r--  1 kwchun  staff   5.5M  5 22 21:33 ways_tags.csv
```

Number of nodes

```
sqlite> select count(*) from nodes;

843894
```

Number of ways

```
sqlite> select count(*) from ways;

83851
```

Number of unique users

```
sqlite> select count(distinct(e.uid))
from (select uid from nodes union all select uid from ways) e;

537
```

Top 10 contributing users

```
sqlite> select e.user, count(*) as num
from (select user from nodes union all select user from ways) e
group by e.user
order by num desc
limit 10;

generalred|325870
maphunter36|210195
alimamo|131724
lorenzo23622|24598
Ataur Rahman Shaheen|22563
cyana|19948
Jockhyeng1|11454
octel|9301
KLon12|9212
沈偉 (Wei-shen)|6969
```

Number of users posted only once

```
sqlite> select count(*)
from (select e.user, count(*) as num
from (select user from nodes union all select user from ways) e
group by e.user
having num=1)  u;

70
```

Other Ideas about the Dataset

According to Ko:Map Features (http://wiki.openstreetmap.org/wiki/Ko:Map_Features#.ED.91.9C.EA.B8.B0.EB.B2.95), Korean name should be indicated in 'name' and 'name: ko', and English name in 'name: en'. However, out of the 230,000 tags, over 10,000 tags are written in the form of *Korean name (English name).* This is probably the easiest error to find in Gunsan map.

```
sqlite> select count(*)
from (select * from nodes_tags union all select * from ways_tags) tags;"

236809

sqlite> select count(*)
from (select * from nodes_tags union all select * from ways_tags) tags
where tags.value like '%(%';"

12727
```

Additional Data Exploration

Top 10 appearing amenities

```
sqlite> select value, count(*) as num
from nodes_tags
where key='amenity'
group by value
order by num desc
limit 10;

hospital|1363
school|389
fuel|343
restaurant|276
dentist|218
bank|213
kindergarten|203
clinic|199
doctors|183
place_of_worship|156
```

Biggest religion

```
sqlite> select nodes_tags.value, count(*) as num
from nodes_tags
join (select distinct(id) from nodes_tags where value='place_of_worship')
i
on nodes_tags.id=i.id
where nodes_tags.key='religion'
group by nodes_tags.value
order by num desc;

christian|90
buddhist|40
jeungsan|1
```

Most popular cuisines

```
sqlite> 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 3;

korean|7
chicken|6
asian|5
```

Conclusion

As an example of Korean map, I looked at a map of Gunsan area and looked for problems. Modification of data that does not follow the new postal code and address scheme requires manual operation using external data. Another common problem is the *Korean (English)* format in 'name' field which violates the tag recommendations. I suspect that this problem can be fixed in a programmatic way.