OpenStreetMap Data Case study

Map Area San Francisco, CA, United States

* <https://www.openstreetmap.org/export#map=11/8.4454/125.9525>
* <https://mapzen.com/data/metro-extracts/metro/san-francisco_california/>
* This is the map area of my favorite and neighbor city. So, am interested to see what database querying reveals, and I would like an opportunity contribute to its improvement on OpenStreetMap.org.

Problems Encountered in the Map

After initially downloading a small sample size of the San Francisco area and running it against a provisional data.py file, I noticed three main problems with the data, which I will discuss in the following order:

1. Inconsistent Postal codes(“CA94116”,”941164116”,”94116”)
2. “Incorrect” postal codes (San Francisco zip code all begins “941”, but a large number of zip codes were outside this region.)
3. Over abbreviated street names(“Lincoln Ave”)

Over abbreviated Street Names:

Once the data was imported to SQL, some basic querying revealed street name abbreviations and postal code inconsistencies. To deal with correcting street names, I opted not use regular expressions, and instead iterated over each word in an address, correcting them to their respective mappings in audit.py using the following function:

def update\_name(name, mapping):

#print name

''

name == "Lincon Ave"

m.group() = Ave

finnaly name will become Lincon Avenue

'''

m = street\_type\_re.search(name)

if m:

street\_type = m.group()

if street\_type not in expected and

street\_type in mapping.keys():

name = re.sub(street\_type\_re,

mapping[street\_type], name)

return name

Postal Codes

SELECT tags.value, COUNT(\*) as count

FROM (SELECT \* FROM nodes\_tags

UNION ALL

SELECT \* FROM ways\_tags) tags

WHERE tags.key='postcode'

GROUP BY tags.value

ORDER BY count DESC limit 10

Here are the top ten results, beginning with the highest count

Value | Count

94122 322

94611 194

94116 158

94117 93

94610 92

94118 77

94133 68

94103 50

94127 50

94109 35

Sort Cities by count, descending

SELECT tags.value, COUNT(\*) as count

FROM (SELECT \* FROM nodes\_tags UNION ALL

SELECT \* FROM ways\_tags) tags

WHERE tags.key LIKE '%city'

GROUP BY tags.value

ORDER BY count DESC limit 10;

And, the results, edited for readability:

Value Count

Redwood City 1564

San Francisco 1216

Berkeley 380

Piedmont 253

Palo Alto 111

Richmond 86

Oakland 85

Union City 20

Burlingame 19

Walnut Creek 17

SELECT \*

FROM nodes

WHERE id IN (SELECT DISTINCT(id) FROM nodes\_tags WHERE key='postcode' AND value='94611')

The result will be:

Id 1241641683 2301289858

lat 37.8304351 37.8253857

Ion -122.2472872 -122.2539761

User rabbitface cartobandit

Uid 321578 1425573

Version 4 1

Change set 21392096 16098254

Timestamp 2014-03-29T23:14:09Z 2013-05-12T16:26:31Z

Number of nodes:

SELECT COUNT(\*) FROM nodes;

882376

Number of ways:

SELECT COUNT(\*) FROM ways;

109782

Number of unique users:

SELECT COUNT(DISTINCT(e.uid))

FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) e;

1459

Top 10 Contributing users

User | num

Null 496082

andygol 99766

ediyes 59247

Luis36995 45295

dannykath 36397

RichRico 27739

Rub21 25550

calfarome 12689

oldtopos 11044

KindredCoda 9868

Number of users appearing only once (having 1 post)

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;

463

Additional Ideas

Contributor Statistics

Here are some user percentage statistics:

* Top user contribution percentage (“Null”) 60.22%
* Combined top 2 users' contribution (“Null” and “andygol”) 72.34%

Additional Data Exploration

Top 10 appearing amenities:

SELECT value, COUNT(\*) as num

FROM nodes\_tags

WHERE key='amenity'

GROUP BY value

ORDER BY num DESC

LIMIT 10;

Value | num

restaurant 167

bench 77

cafe 73

place\_of\_worship 54

bicycle\_parking 40

fast\_food 37

school 36

drinking\_water 34

post\_box 33

toilets 25

Biggest Religions (the first 5 in row)

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

LIMIT 5

The result should be:

Value | Num

christian 45

buddhist 2

jewish 2

muslim 2

Most Popular Cuisines:

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;

The result should be:

Value | num

mexican 16

pizza 12

chinese 8

american 7

Vietnamese 7

italian 6

thai 6

japanese 5

sandwich 4

asian 3

Conclusion:

In the review of this data it is obvious that the San Francisco area is incomplete, though I believe it has been well cleaned for the purposes of this project. I am interested to notice that a fair amount of GPS data makes it into OpenStreetMap.org on account of users, efforts, whether by coding a map editing both or otherwise. With a rough GPS data processor in place and working together with a more robust data processor similar to data.pyI think it would be possible to input a great amount of cleaned data to OpenStreetMap.org and import the data on SQLite studio to figure out the queries.

Anticipated Issues Portion

1) Solution

\* Anticipated Issues:

A) Issue #1: Completeness of the data: in the above data analysis the San Francisco city provide the data from the OpenStreetMap.org have not completed. The reason for this is the lack of necessary information provided by the city council to MapZen. For the future improvement download the metro extracts with the completeness of data.

B) Issue #2: Inconsistence of the data: in this data I see the inconsistence of street name abbreviations and postal code. The reason for this inconsistence problem is repeating the zip code and over abbreviated the street names. My suggestion for improving the inconsistence of data would be the data organized in the consistent format.

## References

* Udacity - <https://www.udacity.com/>
* Wikipedia - <https://www.wikipedia.org/>
* OpenStreetMap - [https://www.openstreetmap.org](https://www.openstreetmap.org/)
* Extract Maps:- <https://mapzen.com/data/metro-extracts/metro/san-francisco_california/>