**Project 3: OpenStreetMap Data Wrangling (SQL)**

By Wenny Wu

**Map Area**

**Richmond, Virginia, United States**

* <https://www.openstreetmap.org/relation/206815>
* <https://mapzen.com/data/metro-extracts/>

I grew up in Richmond, and my family is still there, so I still consider it home in many ways. I would like to contribute to RVA’s improvement on OpenStreetMap.org.

**Problems Encountered in the Map**

After downloading the dataset for the Richmond, VA, I noticed a few main problems with the data:

* Over­-abbreviated and inconsistent street name designations *(i.e. street, St, Street)*
* Inconsistent naming and capitalization of city names (*i.e. richmond, Richmond, Richmond City and glen Allen, Glen Allen)*
* Inconsistent state name abbreviation (*i.e. VA and Virginia)*
* Incorrect postal codes (Richmond area zip codes all begin with “23”, but the dataset was showing three zip codes outside of this region – 19335, 842029, 843050)

**Over-abbreviated Street Names**

Once the data was imported to SQL, I queried for street name abbreviations and postal code inconsistencies. To correct street names, I iterated over each word in an address, correcting them to their respective mappings in audit.py using the following function:

def update\_street(name, mapping):

words = name.split(" ")

for w in range(len(words)):

if words[w] in mapping:

words[w] = mapping[words[w]]

name = " ".join(words)

return name

Streets like “Midlothian Tnpk” become “Midlothian Turnpike”.

Updating city and state names were more straightforward since I could correct the names directly without having to iterate over each word. I created another function in audit.py to correct the city and state names to their respective mappings:

def update\_name(name, mapping):

if name in mapping.keys:

name = mapping[name]

return name

City names like “glen Allen” were corrected to “Glen Allen” and state name “Virginia” was abbreviated to “VA”.

**Postal Codes**

Postal codes were interesting because all Richmond postal codes begin with “23”, but three postal codes in the dataset did not make any sense. Here are the top ten postal codes in the map region, beginning with the highest count:

sqlite> SELECT value, COUNT(\*) as count

FROM (SELECT \* FROM nodes\_tags UNION ALL

SELECT \* FROM ways\_tags)

WHERE key = 'postcode'

GROUP BY value

ORDER BY count DESC

LIMIT 10;

value | count

23223 | 116

23220 | 96

23219 | 50

23059 | 42

23060 | 35

23230 | 32

23221 | 16

23235 | 14

23238 | 14

23112 | 10

I wanted to investigate those postal codes that did not belong in this map region:

sqlite> SELECT \* FROM

(SELECT \* FROM nodes\_tags UNION ALL

SELECT \* FROM ways\_tags)

WHERE key = 'postcode' AND value NOT LIKE ‘23%’;

045271535 | postcode | 19335 | addr

29722624 | postcode | 843050 | addr

29722628 | postcode | 842029 | addr

Zip code 19335 looks like it may be from another state:

sqlite> SELECT \* FROM

(SELECT \* FROM nodes\_tags UNION ALL

SELECT \* FROM ways\_tags)

WHERE id LIKE '%045271535';

045271535 | name | Dad | regular

045271535 | city | Downingtown | addr

045271535 | state | Pa | addr

045271535 | street | Braceland Dr | addr

045271535 | postcode | 19335 | addr

045271535 | housenumber | 480 | addr

And sure enough, someone incorrectly entered an address from Downingtown, PA.

Now, let’s look at the other two unexpected postal codes:

sqlite> SELECT \* FROM

(SELECT \* FROM nodes\_tags UNION ALL

SELECT \* FROM ways\_tags)

WHERE id LIKE '%29722624%';

29722624 | name | "Eugene P. and Louis E. Trani Center for Life Sciences” | regular

29722624 | website | <http://www.vcu.edu/lifesci/facilities/fac_eugene.html> | regular

29722624 | building | yes | regular

29722624 | operator | “Virginia Commonwealth University" | regular

29722624 | city | Richmond | addr

29722624 | wheelchair | yes | regular

29722624 | street | West Cary Street | addr

29722624 | postcode | 843050 | addr

29722624 | levels | "3 with a basement floor" | building

29722624 | housenumber | 1000 | addr

sqlite> SELECT \* FROM

(SELECT \* FROM nodes\_tags UNION ALL

SELECT \* FROM ways\_tags)

WHERE id LIKE '%29722628%';

29722628 | name | “Cary Street Gym" | regular

29722628 | phone | "(804) 827-1100" | regular

29722628 | website | <http://www.recsports.vcu.edu/facilities/cary-street-gym/> | regular

29722628 | building | university | regular

29722628 | city | Richmond | addr

29722628 | wheelchair | yes | regular

29722628 | street | "south linden street" | addr

29722628 | postcode | 842029 | addr

29722628 | levels | “2 with basement floor" | building

29722628 | housenumber | 101 | addr  
  
Both of these addresses are indeed in Richmond and associated with Virginia Commonwealth University (VCU). However, it appears someone inputted a P.O. Box number in place of the postcode.

**Data Overview**

This section contains a basic summary about the dataset, the SQL queries used, and some additional information from the data.

**File sizes**

richmond\_virginia.osm..........123 MB

nodes.csv………………........48.2 MB

nodes\_tags.csv………………1.16 MB

ways.csv…………………….3.5 MB

ways\_nodes.csv…………….15.9 MB

ways\_tags.csv………………9.66 MB

**Number of nodes**

sqlite> SELECT COUNT(\*) FROM nodes;

591767

**Number of ways**

sqlite> SELECT COUNT(\*) FROM ways;

60844

**Number of unique users**

sqlite> SELECT COUNT(DISTINCT(uid))

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

408

**Top 10 contributing users**

sqlite> SELECT user, COUNT(\*) as num

FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways)

GROUP BY user

ORDER BY num DESC

LIMIT 10;

woodpeck\_fixbot 255236

RVA\_101 114162

CynicalDooDad 64188

Omnific 51530

gpstrails 39629

42429 20224

TIGERcnl 12156

bot-mode 11052

taber 10643

daddyklee 9519

**Number of users appearing only once (having 1 post)**

sqlite> SELECT COUNT(\*)

FROM (SELECT user, COUNT(\*) as num

FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways)

GROUP BY user

HAVING num = 1);

86

**Additional Ideas**

Despite being a small city, Richmond seems to have a large number of unique users and contributors on OpenStreetMap.Org. There is opportunity to incentivize users to continue contributing (86 users have only posted once) and to refer other new users, perhaps through gamification methods or offering of rewards/prizes. Prizes can also be awarded to users who make the most changes in cleaning the data, promoting a tidier and more integral dataset.

The dataset also provides a lot of detailed information about demographics, location of schools, amenities, etc. Richmond and its surrounding counties have been continually developing over the past few years. This data can be used to help with city planning and development efforts to build a city that better serves its local communities.

Of course, there are always pros and cons with regards to any implementation of changes/improvements. Getting more users onboard to contribute will indeed generate more data, but as was shown in this exercise, errors do occur and can lead to some very confusing and misleading conclusions. Communities also change frequently, and it is uncertain if those changes are reflected adequately in the dataset.

**Further 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;

restaurant 430

place\_of\_worship 326

school 223

fast\_food 197

fuel 107

bank 83

cafe 64

grave\_yard 62

fire\_station 54

pharmacy 46

**Most popular religions**

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 320  
muslim 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 10;

chinese 12

italian 11

pizza 11

mexican 10

sushi 8

american 7

burger 5

indian 4

thai 4

regional 3

**Conclusion**

Although Richmond is not a large city, this exercise has shown that a lot of work is left to be done in better mapping the area. I’m impressed to see how much user activity and effort goes into inputting this data into OpenStreetMap.org and look forward to how the community will continually contribute to these types of open-source projects in the near future.

**References**

<https://gist.github.com/carlward/54ec1c91b62a5f911c42#file-sample_project-md>

Udacity Forums: <https://discussions.udacity.com/c/nd002-p3-data-wrangling>