OpenStreetMap Data Case Study

Map Area

Houston, TX, US

This area is where I'm now living, so I'm more interested to see what database querying reveals, and I'd like an opportunity to contribute to its improvement on OpenStreetMap.org.

1.General Impression for the data

The custom area I chose mainly include the city area for Houston. The dataset is about 200MB. First I used get_sample.py to extract 5% of the 200 MB dataset as sample data, and run the iterateparsing.py, I found the tags in the sample include:
{'node': 38998, 'nd': 51553, 'member': 890, 'tag': 38291, 'relation': 55, 'way': 7096, 'osm': 1}

2.Problems Encountered in the Map

I used the audit.py script to view unusual data.

Problem 1: abbreviation issue

```
def update_name(name, mapping):
    #print name
    #print mapping
    if num_line_street_re.match(name):
        nth = nth_re.search(name)
        name = num_line_mapping[nth.group(0)] + " Line"
        print name
        return name
    else:
        original_name = name
        for key in mapping.keys():
            # Only replace when mapping key match (e.g. "St.") is
found at end of name
            type_fix_name = re.sub(r'\s' + re.escape(key) + r'$',
' ' + mapping[key], original_name)
            nesw = nesw_re.search(type_fix_name)
            if nesw is not None:
                for key in street_mapping.keys():
                ……
                ……
          if type_fix_name != original_name:
                # print original_name + "=>" + type_fix_name
                return type_fix_name
```

Old Katy Rd => Old Katy Road
East Sam Houston Pkwy N => East Sam Houston Parkway North
Milby St => Milby Street
Post Oak Blvd. => Post Oak Boulevard

Problem 2: postcode inconsistency, some have 5 digits while others have 9digits.

```
POSTCODE = re.compile(r'[A-z]\d[A-z]\s?\d[A-z]\d')
def audit_postcode(osmfile):
    post_file = open(osmfile, "r")
    for event, elem in ET.iterparse(post_file, events=("start",)):
        if elem.tag == "node" or elem.tag == "way":
            for tag in elem.iter("tag"):
                if tag.attrib['k'] == 'addr:postcode':
                    post_code = re.sub(" ", "",
tag.attrib['v'].strip())
                    #print post_code
                    m = POSTCODE.match(post_code)
                    if m is None:
                        print post_code
                     if len(post_code) > 5:
                        print post_code
                        post_code = post_code[0:5]
    post_file.close()
```

77009
77042-9998
73032
Some postcodes are 5 digits, while some are 9 digits, it's better to restrict them to 5 digits.

3. Data Overview

Houton.osm : 450 MB
Houston.db: 339 MB
nodes.csv: 159MB
nodes_tags.csv: 4.6 MB
ways.csv: 17.3 MB
ways_tags.csv: 54 MB
ways_nodes.csv: 57.8 MB

Number of nodes

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

1978133

Number of ways

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

301230

Number of unique users

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

1269

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;
```

afdreher 437900
woodpeck_fixbot 372710
cammace 193489
scottyc 181363
brianboru 114638
claysmalley 109969
RoadGeek_MD99 75512
skquinn 74396
Memoire 53616
TexasNHD 47201

4. Additional ideas

No.1 appearing amenities

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

place_of_worship|1879
fountain|682
restaurant|619
school|585
fast_food|547
fire_station|265
fuel|233
pharmacy|160
bank|152

bench|149

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
LIMIT 1;
```

christian 571

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 1;
```

mexican|68
pizza|33
american|32
italian|32
chinese|23
burger|18
barbecue|15
seafood|14
sandwich|11
mediterranean|9

```
sqlite> SELECT nodesTags.value, COUNT(*) as num        FROM
nodesTags            JOIN (SELECT DISTINCT(id) FROM nodesTags
WHERE value='bank') i            ON nodesTags.id=i.id
WHERE nodesTags.key='name'         GROUP BY nodesTags.value
ORDER BY num DESC        LIMIT 5;
```

Chase|36
Wells Fargo|20
Bank of America|10

BBVA Compass|7
Capital One|7
Prosperity Bank|5
Amegy Bank|3
Capital One Bank|3
Frost Bank|3
BBVA|2

How many krogers and fiesta (popular supermarket in Houston)

```
sqlite> select count(*) from nodes_tags where value like
'%Kroger%';
```

30

```
sqlite> select count(*) from nodes_tags where value like
'%fiesta%';
```

9

Conclusion:

Houston is a really big city. I just chose the part of it (main city area). The data quality is good, I did not find big problem except for abbreviation issue and postcode issues. The database exploration is interesting. Over 1200 users have contributed to the map. The most common amenity in place for worship, that's a surprise, however, Christian is the biggest religion is as expected. No surprise that Chase and Wells Fargo are most common banks. And Kroger, a popular supermarket in Houston, is everywhere. I suspected that the restaurant information is very limited, since most types of restaurant are Mexican, which does not quite make sense since fast food like burger and pizza should be much more than Mexican food. So, I think there are several opportunities for cleaning and validation of the data in the future.

Additional Suggestion and Ideas

We can build parser which parse every word input by the users to avoid common typos. Also we can put some restriction to inputs so that the formatted input can avoid many errors and inconsistencies. And it will be great if the public security information can be integrated into the map, like in which area the criminal are more frequency, so that the tourists' safety can be improved.