Denver Open Street Map Munging

Map Area

Denver, CO, United States
* http://www.openstreetmap.org/relation/253750 * The sample file used can be found in the github notebook

I currently live and work in Denver but am new to the area so I thought I'd explore the OSM data and see what it contains

Problems found in data set

I first downloaded a much smaller sample set of Denver to start invesigating any problems. Below is what
I found and how I addressed the issues:

Most of the user entered data simply had inconsistent abbreviations which I decided to clean up:

- Ave changed to Avenue
- Dr changed to Drive
- Pky to Parkway
- ct to Court, as well as Ct to Court
- Rd and Rd. as well as Ave., these small changes were more difficult to catch

I kept some the entered data the same like North State Highway 83 and Park Avenue West even though they at first appeared to need cleaning. The vast majority of problems were fixed by auditing the data and then changing them to the updated names mapping that I continue updating as I saw more and more of my data set.

The sample that I used, being so small, didn't fully prepare me for the errors that existed in the truly OSM file. After loading that and beginning to clean it, I had to circle back and look at additional problems created like the abbreviations with a '.' at the end.

Overview of the data

Size of the file:

```
denver-boulder_colorado.osm.....850mb
sample.osm....4mb
nodes.csv...325mb
nodes_tags.csv...11mb
ways.csv...26mb
ways_tags.csv...108mb
way_nodes.csv...62mb
```

Number of nodes:

I decided to sum the number of breweries in the Denver area because CO has such a high profile for microbreweries. It did not disappoint as it appeared 95th on the list of most mentioned node keys

nodes:

```
"""select count(*) from nodes;"""
```

8757993

ways:

```
"""select count(*) from ways;"""
```

432284

users:

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

2050

breweries:

```
"""select count(nt.key) from nodes_tags as nt
            where nt.key = 'brewery';"""
```

85

node_keys:

```
"""select nt.key, count(nt.key) from nodes_tags as nt
            group by nt.key order by count(nt.key) desc limit 100;"""
```

List of node_keys by frequency:

```
[(u'natural', 50249),
 (u'highway', 46458),
 (u'source', 37580),
 (u'name', 21163),
 (u'power', 15389),
 (u'crossing', 12797),
```

```
  (u'amenity', 11326),
  (u'street', 7469),
  (u'housenumber', 7448),
  (u'postcode', 6798),
  (u'city', 5981),
  (u'ele', 5702),
  (u'feature_id', 5252),
  (u'created', 4794),
  (u'state', 4545),
  (u'shop', 3846),
  (u'barrier', 3177),
  (u'noexit', 2863),
  (u'bicycle', 2697),
  (u'county_id', 2652),
  (u'state_id', 2652),
  (u'landuse', 2586),
  (u'county_name', 2568),
  (u'local_ref', 2493),
  (u'phone', 2238),
  (u'website', 2238),
  (u'feature_type', 2143),
  (u'cuisine', 1677),
  (u'tourism', 1510),
  (u'railway', 1487),
  (u'operator', 1183),
  (u'leisure', 1182),
  (u'note', 1075),
  (u'traffic_calming', 959),
  (u'alt_name', 899),
  (u'emergency', 897),
  (u'office', 872),
  (u'opening_hours', 838),
  (u'ref', 800),
  (u'entrance', 793),
  (u'import_uuid', 785),
  (u'capacity', 735),
  (u'man_made', 670),
  (u'bicycle_parking', 624),
  (u'religion', 601),
  (u'access', 574),
  (u'unit', 559),
  (u'type', 507),
  (u'supervised', 480),
  (u'tmp', 457),
  (u'place', 450),
  (u'id', 368),
  (u'gnis:County_num', 360),
  (u'gnis:ST_alpha', 360),
  (u'gnis:ST_num', 360),
  (u'shelter', 360),
  (u'waterway', 360),
  (u'gnis:Class', 359),
  (u'gnis:County', 359),
  (u'is_in', 355),
  (u'covered', 334),
  (u'denomination', 331),
  (u'designation', 311),
```

```
 (u'direction', 311),
 (u'bench', 310),
 (u'design', 265),
 (u'aeroway', 257),
 (u'sport', 241),
 (u'building', 213),
 (u'floor', 208),
 (u'parking', 192),
 (u'information', 189),
 (u'craft', 186),
 (u'country', 175),
 (u'edited', 173),
 (u'atm', 165),
 (u'stop_id', 158),
 (u'historic', 154),
 (u'housename', 142),
 (u'brand', 133),
 (u'wheelchair', 130),
 (u'description', 123),
 (u'material', 122),
 (u'backrest', 121),
 (u'lamp_type', 113),
 (u'park_ride', 110),
 (u'fee', 102),
 (u'motor_vehicle', 100),
 (u'population', 99),
 (u'traffic_signals', 99),
 (u'internet_access', 92),
 (u'foot', 88),
 (u'support', 88),
 (u'source_ref', 86),
 (u'brewery', 85),
 (u'leaf_type', 85),
 (u'old_name', 83),
 (u'smoking', 83),
 (u'takeaway', 81),
 (u'shelter_type', 73)]
 ```
Additional query:
I wanted to see if there were certain users who went after finding breweries, and it appears there are!
```python
beer_finders = """select n.user, count(distinct n.id) from nodes_tags as nt
        join nodes as n on n.id = nt.id
        where nt.key = 'brewery'
        group by n.user
        order by count(n.id) desc;"""
```

```
[(u'ColoMapper', 70),
 (u'chachafish', 10),
 (u'Your Village Maps', 2),
 (u'Spanholz', 1),
 (u'mattchn', 1)]
 ```

Additional Ideas:

```
My hunch was that certain areas of Denver would be more inclined to add data to OSM due to topography, culture, income, etc so I went ahead and split up the number of data points into the 4 different quadrants of the city to see if there was an even or uneven spread

```python
QUERY = """SELECT n.lat, n.lon from nodes as n"""
q2 = """select nt.value, count(*) from nodes_tags as nt
        group by nt.value
        order by count(*) desc"""

hp = 39.75              # Meaning horizontal point of city
vp = -105.010845        # Meaning vertical point of city

q1 = 0 # different quadrants of city
q2 = 0
q3 = 0
q4 = 0

print len(rows)
for row in rows:
    try:
        lat = float(row[0].strip())
    except:
        continue
    try:
        lon = float(row[1])
    except:
        continue
    if lat > hp:
        if lon < vp:
            q1 += 1
        else:
            q2 += 1
    elif lat < hp:
        if lon < vp:
            q4 += 1
        else:
            q3 += 1

print q1, q2, q3, q4
```

length of rows: 8757993
Quadrant 1:3074419
Quadrant 2:1234075
Quadrant 3:2526140
Quadrant 4:1399067

I am not surprised that Quadrant 1 has the most due to its very active downtown area and that Quadrant 2 does not have many due to its much lower population and greater greenspace

Conclusion:

The data was cleaner than I expected coming from user generated input. What took a little bit of
```

time to contort was the string inputted lat and lons into usable integers for analysis.

I would like to do further work concerning other areas of the state seeing as Colorado is such a high percentage of leisure users especially in the mountains. Do these users take time to add data to OSM or do users tend to just do so when they are near their home areas?

My guess would be that users are actually more likely to do so when viewing a highly attractive place like Red Rocks Amphitheatre or Vail Ski Area. The complications would obviously be having to use a much larger dataset which can really slow down the system. Just auditing and analyzing Denver was quite cumbersome. An additional complication would be not knowing if the OSM user was a visitor or local at the attraction. Without knowing this, we obviously couldn't do much analysis on whether or not more people add OSM data near home or when traveling.