OpenStreetMap Project

Map Area

Oxford, Ohio and the surrounding area.

To get the raw data I used, go to the Overpass API and pass the Query Form:

```
(node(39.3587, -84.8419, 39.6961, -84.2033);<;);out meta;
```

This is where Miami University is located which is where I went to college, so I am excited to see what I can uncover!

Reproducing

1. Get the map data stated above (place in data folder and rename to full_map.osm)
2. Navigate to Project directory
3. Run `python clean_open_map.py`
4. Run `sqlite3 open_map.db`
5. From sqlite shell run `.read build_database.sql`

You should now have a cleaned database to work with!

Issues Encountered in My Map

- Abbreviated street names (ex. ST, Ct, Dr instead of Street, Court, Drive)
- Names being all uppercased (ex. "ELM STREET BUILDING" "HERITAGE COMMONS CENTER" "AIRPORT RADIO BUILDING")
- Alternate names being listed after a semicolon instead of as an alt_name tag (ex. "Mount Pleasant Cemetery;Monroe Cemetery" , "Little Miami National and State Scenic River; Little Miami River")
- "YAGER STADIUM - EAST (CONCESSIONS," seems to have had a parsing error (or possibly human)

Abbreviated Street Names

These are easy enough to clean up. Using the REGEX provided in the case study I found the street names and then checked if they were in the list of expected names ("Street", "Avenue", "Boulevard", "Drive", "Court", "Place", "Square", "Lane", "Road", "Trail", "Parkway", "Commons", "Pike", "Highway", "Way") if not then I try to convert it to an expected name with the following function:

```
def update_name(name, mapping):
    '''Convert abbreviations to a preferred name'''
    road_type = name.split()[-1:][0]
    cleaned = mapping[road_type]
    name = name.replace(road_type, cleaned)
    return name

mapping = { "Blvd." : "Boulevard",
            "Ct" : "Court",
            "Dr" : "Drive",
            "St" : "Street",
            "St." : "Street",
            "ST" : "Street",
            "Ave" : "Avenue",
            "Ave." : "Avenue",
            "Rd." : "Road",
            "Rd" : "Road"
            }
```

Uppercase Names

Getting just the names that should not be uppercase was a little tricky because there are abbreviations, special codes, and other various situations where it was inappropriate. However I found if I first checked if a name was uppercased and had a space in it, then that it didn't contain any numbers, it would only be ones that should be fixed. After I could just use the title method on the string. Relevant part of name cleanup code:

```
    if value == value.upper() and ' ' in value:
        if hasNumbers(value) == False:
            value = value.title()
```

Alternate Names

Finding the instances of multiple names listed with semicolons was straightforward, with the only exception need being to ignore them if they were a flag (the naming convention of United States; Ohio seems consistent and correct for them). After identifying these I passed them to function that creates an 'alt_name' tag (to match the correctly entered ones) with the same element id and type as the name we are splitting them from. The relevant name clean up section:

```
    elif tag_key == 'name' and ';' in value and tag_type != 'flag':
        values = value.split(';')
        value = values[0]
        add_alt_name(elem, values[1], element_id, tag_list)
```

And the function it calls:

```
def add_alt_name(elem, alt_name, element_id, tag_list):
    '''Add a new tag for alternate names'''
    type_key = k_prep(elem)
    tag_type = type_key[0]
    tag_list.append({'id': element_id,
            'key': 'alt_name',
            'value': alt_name,
            'type': tag_type})
```

Potential parsing error

While checking into the uppercased names I came across "YAGER STADIUM - EAST (CONCESSIONS,", which made me concerned there may be some sort of parsing error with one of the data sources. However after checking for tags that end in commas and for any open parenthesis I wasn't able to find similar issues. I decided to rule this as user error and did not write a programatic fix in the ingestion script (seems like over engineering). The following sql can be used to fix this after building the database though (now title cased from data cleaning).

```
UPDATE way_tags
SET value = 'Yager Stadium - East (Concessions)'
WHERE value = 'Yager Stadium - East (Concessions,';
```

Data Overview

File Sizes

File	Size
full_map.osm	60.6 MB
open_map.db	32.4 MB
nodes.csv	22.2 MB
nodes_tags.csv	519 KB
ways.csv	1.9 MB
ways_tags.csv	4.7 MB
ways_nodes.csv	7.6 MB

Suggested Overviews

Number of nodes

```
SELECT COUNT(*) FROM node;
```

260,897

Number of ways

```
SELECT COUNT(*) FROM way;
```

31,518

Unique users

```
SELECT COUNT(DISTINCT(users.uid))
FROM (SELECT uid
        FROM node
        UNION ALL SELECT uid
        FROM way) users;
```

255

Number of landfills

```
SELECT count(*) FROM way_tags WHERE value = 'landfill';
```

15

I always thought it was odd how many landfills you go past driving south to Cincinnati, but this is WAY more than I expected!

Additional Exploration

University buildings added by user

```
SELECT way.user, count(*)
FROM way_tags, way
WHERE way_tags.key = 'building'
AND way_tags.value = 'university'
AND way_tags.id = way.id
GROUP BY way.user;
```

user	count
DSMcGregor	84
coopermj	59

I was wondering if Miami would add their buildings themselves but it looks like these are normal users contributing them opposed to an official sounding user name.

Restaurants and Bars

```
SELECT value, COUNT(*)
FROM node_tags
WHERE value = 'restaurant'
OR value = 'bar'
GROUP BY value;
```

value	count
bar	2
restaurant	13

Sadly restaurants and bars are not tagged well at all in this area, with it being a college town there are a lot of bars and it would have been interesting to look more into these.

```
SELECT way_tags.key, COUNT(*) as num
FROM way_tags
    JOIN (SELECT DISTINCT(id)
            FROM way_tags
            WHERE way_tags.key = 'building'
            AND way_tags.value = 'university') i
            ON way_tags.id = i.id
GROUP BY way_tags.key
ORDER BY num DESC
LIMIT 20;
```

tag	count
building	143
name	139
state	80
county_name	78
feature_id	78
import_uuid	78
source	77
ele	74
name_1	24
amenity	6
housenumber	5
postcode	5
street	5
city	4
leisure	4
name_2	4
operator	4
website	4
levels	3
phone	3

I was also a bit disappointed that there is not more interesting information tagged to the buildings since it would have been interesting to look at things like percent of different types of school buildings (dorms, administrative, dining halls, etc.).

On the bright side we can get the names of the building and could potentially infer the types of buildings, with the exception of being able to separate dorms from lecture halls since the majority of both are called "(Person's Name) Hall".

```
SELECT way_tags.value
FROM way_tags
    JOIN (SELECT DISTINCT(id)
            FROM way_tags
            WHERE way_tags.key = 'building'
            AND way_tags.value = 'university') i
    ON way_tags.id = i.id
WHERE way_tags.key = 'name'
LIMIT 10;
```

"Harry T. Wilks Conference Center" "Schwarm Hall" "Phelps Hall" "Mosler Hall" "Rentschler Hall" Gymnasium "Yager Stadium - West (Main Bldg)" "Yager Stadium - East (Concessions)" "Walter L. Gross Center" "Harris Dining Hall"

tiger entries that are reviewed

```
SELECT tags.value, count(*)
FROM (SELECT *
        FROM node_tags
        UNION ALL SELECT *
        FROM way_tags) tags
WHERE type = 'tiger'
AND key = 'reviewed'
GROUP BY value;
```

value	count
no	10947
yes	103

After seeing how many things are entered from the tiger system, I was curious how many get reviewed. It would be interesting to see how the one percent here compares to areas with different demographics such as a large city and rural farm land.

Ideas for improve the data set

It would be awesome to get a lot of the bars and restaurants in Oxford tagged. I would also love to see at least the primary purpose of each campus building added to the data set. With it being a college campus, it seems like a straight forward way to address these issues would be to get a professor interested in using this data set for a class. Then having an assignment where students submit additional data points. I am also curious if you could use the Yelp API to label restaurants and bars along with what type of cuisine they are (from a technical standpoint I think yes, from a ToS standpoint... probably unlikely).

With either of these ideas, I think the biggest challenge would be keeping all of the information up to date. New bars and restaurants open and shut down at an alarming rate in Oxford so it may be hard to ever have all the data correct. Miami has also been building a lot of new building lately.

If we were able to pull it off though it would be a lot of interesting things we could look at like what type of restaurants are most common, breakdown of campus buildings by type, locations tagged as both a bar and as a restaurant, etc.

Conclusion

The Miami University campus seems to be tagged pretty well, even if they do not have quite as much information as I would like. Oxford seems to be really missing a lot of information though, especially when it comes to buildings that are of public interest.

It would be really nice to see a company like Google or Yelp give back to the OpenStreetMap community by feeding in at least the building/company names which they have, since you see the OpenStreetMap copy write in the corner of all their maps and they clearly benefit from it.