Jean Phillips
5/1/2021

OPENSTREETMAP DATA WRANGLING PROJECT

Introduction

This project is part of the Udacity Data Analyst Nanodegree. The project stipulates that students perform data munging and cleaning prior to analyzing the data. Data is downloaded from the OpenStreetMap project, which is an open-source repository of user-maintained global map data.

Map Area

The map area I chose to explore for this project is Germantown, TN. Germantown is a town located adjacent to Memphis, TN, and I bought a home there two years ago. Since moving to Germantown, I have really enjoyed exploring the area and seeing the various amenities (parks, trails, restaurants) the town has to offer. For that reason, I thought it would be fun to use the town for this project.

Problems Encountered in the Map

After initially downloading the osm data for Germantown, TN, I decided to take a smaller sample of the data to work with my scripts to determine what issues were present in the data prior to working with the substantially larger dataset.

One of the main issues with the dataset is inconsistency in the street address data. Many street names are abbreviated, such as “Ave” for “Avenue.” Additionally, there are inconsistencies in the punctuation of the abbreviated street names leading to further potential confusion, as with “St.” versus “St” versus “Street.” Some of the streets were missing the street designation, as with “Fletcher” versus “Fletcher Trace.” This inconsistency is an issue of both validity and consistency: the data should conform to a standard format to be valid, and the data should be in logical agreement for consistency.

Furthermore, there were inconsistencies noted in the zip code data. Germantown, as a small town, only encompasses 3 zip codes (38138, 38139, and 38133). There were, however, many more zip codes than those 3 represented in the data set. Many of these other zip codes were identifiable as zip codes in neighboring communities such as Memphis and Cordova. Thus, this inconsistency is a concern of validity and consistency, as well as accuracy. Ultimately, the decision was made not to make changes to the zip code data in the dataset because the change would not have a significant impact on the strength of our analysis.

Data Cleaning Plan

An iterative cleaning plan was implemented for the streets in the map. The street endings in the data set were compared to a standardized list of expected street types using a provisional street name auditing script, which produced a dictionary of the non-standard street names. From there, the street names were standardized during the conversion from XML to CSV files by applying a mapping to the data.

The data from the osm file was then used to create the csv files, which were subsequently used to populate the database with tables against which our queries would be run.

Jean Phillips
5/1/2021

Data Overview

A review of the file sizes for the datasets and files used in this project:

Germantown.osm: 149.3MB
Germantown_ten.db: 130MB
Nodes.csv: 79.1MB
Nodes_tags.csv: 939KB
Ways.csv: 4.77MB
Ways_tags.csv: 10.1MB
Ways_nodes.csv: 25 MB

Next, I ran the below code to find the number of unique users in the dataset, which was found to be 621 with the query below:

In [13]:
```
# number of unique users
query = "SELECT COUNT(DISTINCT(e.\"b'uid'\"))FROM (SELECT \"b'uid'\" FROM Nodes UNION ALL SELECT \"b'uid'\" FROM Ways) as e;"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
[(621,)]
```

Then, I was interested in finding the number of unique nodes, which was found to be 730313 by using the query below:

```
# number of nodes
query = "SELECT count(DISTINCT(\"b'id'\")) FROM nodes;"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
[(730313,)]
```

Next, I was interested in finding the number of unique ways, which was found to be 60538 by using the query below:

```
# number of ways
query = "SELECT count(DISTINCT(\"b'id'\")) FROM ways;"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
[(60538,)]
```

Then, I was curious about the top ten users who had contributed/edited data in the Germantown, TN OpenStreetMap file, so I ran the following query which returned:

```
#Top 10 contributing users

query = "select e.\"b'user'\", count(*) as num from (select \"b'user'\" from nodes UNION ALL select\
  \"b'user'\" from ways) as e group by \"b'user'\" order by num desc limit 10;"
c.execute(query)
rows=c.fetchall()
print ('Top 10 contributing users and their contribution:\n')
pprint.pprint(rows)



#Total users
print ('\n')
print ('Total appearances of the users:\n')
query = "select count(e.\"b'user'\") from (select \"b'user'\" from nodes UNION ALL select \"b'user'\" from ways) as e;"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
Top 10 contributing users and their contribution:

[("b'OSM901'", 536638),
 ("b'kj81xa96e9'", 98948),
 ("b'Kh3fBq29cVa21'", 68837),
 ("b'Rub21'", 7263),
 ("b'Chris Lawrence'", 4661),
 ("b'DeVietor'", 3399),
 ("b'sboxer22'", 2879),
 ("b'TIGERcnl'", 2840),
 ("b'HazyArc14'", 2612),
 ("b'PhishNoC'", 2577)]


Total appearances of the users:

[(790851,)]
```

The top contributor, OSM901, is likely a bot given how much data they have contributed to the set. Given that our local area code is 901, the name made me smile at least!

I was curious then about the numbers of different types of nodes. First, I investigated the number of cafes and shops, as suggested in the original assignment instructions by running the following code:

```
 #number of chosen type of nodes, like cafes, shops etc.

query = "SELECT \"b'value'\" , count(*) FROM (select \"b'key'\" , \"b'value'\" from \
nodes_tags UNION ALL select \"b'key'\", \"b'value'\" from ways_tags) as e where \"b'value'\" like '%cafe%';"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)


print ('\n')

query = "SELECT \"b'value'\", count(*)  FROM (select \"b'key'\" , \"b'value'\" from \
nodes_tags UNION ALL select \"b'key'\" , \"b'value'\" from ways_tags) as e where \"b'value'\" like '%shop%';"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
[("b'cafe'", 12)]


[("b'coffee_shop'", 110)]
```

This shows there are 12 cafes and 110 shops in the OSM data for Germantown, TN. Given that Germantown is a small area that is largely residential, this number feels about right.
Following that exercise, I was curious about the top 20 types of cuisine for restaurants in Germantown, TN, so I ran the following query and subsequently got the below results:

```
query="select \"b'value'\",count(*) as num from (select \"b'key'\",\"b'value'\" from \
nodes_tags UNION ALL select \"b'key'\",\"b'value'\" from ways_tags) as e where e.\"b'key'\" \
like '%cuisine%' group by \"b'value'\" order by num desc limit 20;"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
[("b'burger'", 14),
 ("b'sandwich'", 7),
 ("b'coffee_shop'", 7),
 ("b'pizza'", 6),
 ("b'mexican'", 6),
 ("b'chicken'", 6),
 ("b'tex-mex'", 4),
 ("b'seafood'", 4),
 ("b'american'", 4),
 ("b'italian'", 3),
 ("b'wings'", 2),
 ("b'steak_house'", 2),
 ("b'japanese'", 2),
 ("b'indian'", 2),
 ("b'chinese'", 2),
 ("b'barbecue'", 2),
 ("b'asian'", 2),
 ("b'korean'", 1),
 ("b'frozen_yogurt'", 1),
 ("b'donut;coffee_shop'", 1)]
```

Surprisingly, burger restaurants were the most common type of cuisine in town. I would have predicted that barbecue joints would have been much higher up the list, since Germantown is just outside Memphis which is world-famous for its barbecue.
Next, I was curious about the top 20 amenities for my town so I performed the following search with the following results:

```
query="select \"b'value'\", count(*) as num from nodes_tags where \"b'key'\"=\"b'amenity'\" \
group by \"b'value'\" order by num desc limit 20;"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
[("b'place_of_worship'", 85),
 ("b'school'", 51),
 ("b'restaurant'", 36),
 ("b'fast_food'", 18),
 ("b'fountain'", 14),
 ("b'clinic'", 8),
 ("b'cafe'", 6),
 ("b'pharmacy'", 5),
 ("b'post_office'", 4),
 ("b'library'", 4),
 ("b'grave_yard'", 4),
 ("b'dentist'", 4),
 ("b'veterinary'", 3),
 ("b'fuel'", 3),
 ("b'bank'", 3),
 ("b'townhall'", 2),
 ("b'ice_cream'", 2),
 ("b'fire_station'", 2),
 ("b'doctors'", 2),
 ("b'bench'", 2)]
```

That's a whole lot of places of worship for one small town, but this is the South, after all! It's almost funny that there are equal numbers of benches and doctors represented as amenities in this dataset.

I then thought I might explore the various leisure activities available in Germantown, and thus ran the below query:

```
query="select a.\"b'value'\" as leisure, count(*) as num from (select \"b'key'\",\"b'value'\" \
from nodes_tags UNION ALL select \"b'key'\",\"b'value'\" from ways_tags) as a \
where a.\"b'key'\"=\"b'leisure'\" group by a.\"b'value'\" order by num desc;"
c.execute(query)
rows=c.fetchall()

pprint.pprint(rows)
```

```
[("b'swimming_pool'", 983),
 ("b'pitch'", 255),
 ("b'park'", 41),
 ("b'playground'", 36),
 ("b'golf_course'", 5),
 ("b'track'", 3),
 ("b'nature_reserve'", 2),
 ("b'garden'", 2),
 ("b'fitness_centre'", 2),
 ("b'sports_centre'", 1),
 ("b'fishing'", 1),
 ("b'dance'", 1)]
```

The town is a very beautiful natural area, with lots of trees and green spaces. It is not surprising to me to see so many parks, playgrounds, nature reserves, etc. That does seem like a lot of swimming pools though; the data must be including backyard pools.

Other Ideas

There were a few surprises I found in the dataset, and those are the basis for the following additional suggestions for improvement to the dataset. There is a certain incompleteness to the tag information for nodes and ways. The dataset just does not have robust and complete information. One glaring example is the number of doctor's offices listed as amenities, that number being two. Germantown houses a medical district; there are far more than two doctor's offices in the town. Another example is the number of barbecue restaurants: although it may not be the most popular cuisine in the town, there are definitely more than two such restaurants in Germantown. Increased user engagement with OpenStreetMap would increase the quantity data available and thereby improve the robustness of our analysis. Of course, additional data comes with risks and problems of its own. Just because there is additional data, does not mean that it is good data. Therefore, an additional suggestion would be for OpenStreetMap to more strictly enforce the standards for the quality and veracity of the data submitted by its users. Of course, more stringent standards may subsequently reduce user engagement which may negatively impact the OpenStreetMap project.