OpenStreetMap Project - Data Wrangling with SQL

Openstreet map (https://wiki.openstreetmap.org/wiki/Main_Page) is a free world map where anyone can contribute geographic data. Because anyone can contribute it stands to reason that the information contained may not always be accurate.

The data set used for this project is located at https://www.openstreetmap.org/export#map=12/32.7831/-96.8067 (https://www.openstreetmap.org/export#map=12/32.7831/-96.8067)

I picked this location because it is of interest to me. I recently relocated to the Dallas metro area and getting around and discovering things in a new city can be challenging. We will explore the accuracy of the information in this area and correct crucial data.

Overview of the Data

The dataset chosen is a rather large one: Dallas.osm - 692MB Dallas_metro.db - 392MB Nodes.csv - 291MB Nodes_tags.csv - 1.26MB Ways.csv - 23.0MB Ways_Nodes.csv - 85.5MB Ways_tags.csv - 37.2MB

When taking a look at the tags from Dallas.osm we can see there are over 3 million node and ways datapoints.

Mapparser.py

```
{'bounds': 1, 'member': 28666, 'meta': 1, 'nd': 3594762, 'node': 3200502, 'note': 1, 'osm': 1, 'relation': 1602, 'tag': 1125912, 'way': 339226}
```

So we started by looking at the elements and categorizing them according to data patterns. It is rather surpising to see that there are no problem characters associate with the data.

Tags.py

```
{'lower': 483367, 'lower_colon': 633552, 'other': 8993, 'problemchars': 0}
```

So next we took a look at unique user contributions. We can see that 841 unique users contributed to the dataset.

Users.py

```
841
```

With a little bit better view of the data we start but taking a look at what can be one of the most important areas of the data, streets. It won't matter what cool things we find to do in the city if we can't figure out how to get there. As we can see there are numerous problems with the data.

Audit.py

```
{'Verdes': set(['Palos Verdes']), 'Rd': set(['E Grauwyler Rd', 'S. Hampton Rd', 'Interstate 30 Frontage Rd', 'I-20 Frontage Rd', 'Midway Rd', 'E Overton Rd', 'Stone Canyon Rd', 'N Support Rd', 'US-175 Frontage Rd', 'Hall Rd', 'Cedar Springs Rd', 'Garland Rd']), 'Elias': set(['Via Jesse Elias']), 'Gate': set(['Vista Gate']), 'Inwood': set(['Inwood']), 'Deseo': set(['Deseo']), '20': set(['East
Interstate Highway 20', 'West I 20', 'I 20', 'West Interstate 20', 'Interstate Highway 20']), 'Paraiso': set(['Paseo Paraiso']), 'Verde': set(['Verde']), 'Jacob': set(['Via James Jacob']), 'Redondo': set(['Redondo']), '408': set(['Spur 408']), 'Cima': set(['La Cima']), 'Clemente': set(['San Clemente']), 'D': set(['Avenue D']), 'H': set(['Avenue H']), 'L': set(['Avenue L']), 'Nile': set(['Nile']), 'Pkwy': set(['International Pkwy']), '12': set(['South Loop 12', 'North Loop 12', 'Loop 12']), 'Wandt': set(['Wandt']), 'Kearney': set(['E Kearney']), 'Wren': set(['Wren']), 'Rio': set(['Camino Rio']), 'Halsey': set(['Halsey']), 'Chase': set(['Slick Rock Chase']), '56th': set(['56th']), 'Fontana': set(['La Fontana']), 'West': set(['Plymouth Drive West', 'Lago Vista West', 'East Technology Boulevard;Technology Boulevard West', 'Story Road West']), 'Expy': set(['N Central Expy']), 'Lago': set(['Camino Lago']), 'Dr.': set(['Rugged Dr.']), '80': set(['West US Highway 80']), 'Rd.': set(['Hillcrest Rd.']), '360': set(['North State Highway 360', 'State Highway 360']), 'C': set(['Avenue C']), 'Central': set(['Empire Central']), 'G': set(['Avenue G']), '300': set(['Forest Central Drive, Suite 300']), 'K': set(['East Avenue K']), 'Kjo': set(['7815 McCallum Blvd 14203 Dallas TX 75252 Kjo']), '306': set(['W Illinois Ave #306']), 'Antonio': set(['Via San Antonio']), '102': set(['North Market Street #102']), 'Millmar': set(['Millmar']), '30': set(['West Interstate 30', 'Interstate 30']), 'Catherine': set(['Via Saint Catherine']), 'Lynnacre': set(['Lynnacre']), 'Avenida': set(['Via Avenida']), 'road': set(['Innwood road']), 'St.': set(['Dyer St.']), '175': set(['North Highway 175']), 'Thrush': set(['Wood Thrush']), 'Eduardo': set(['Via San Eduardo']), 'Downs': set(['Churchill Downs']), 'Tranquilo': set(['Tranquilo']), 'Vernon': set(['Mt Vernon']), 'Sheree': set(['Sheree']), 'Willowood': set(['Willowood']), 'B': set(['Avenue B', 'West Main Street #B']), '635': set(['N Central Expressway Ste 635', 'West Interstate 635']), 'F': set(['Avenue F']), 'J': set(['Avenue J']), 'St': set(['Farrington St', 'Knox St', 'Live Oak St']), '75062': set(['75062']), 'Mound': set(['Pleasant Mound']), '161': set(['State Highway 161', 'TX 161', 'South Highway 161']), 'Sonoma': set(['Sonoma']), 'Terraza': set(['Terraza']), '2010': set(['North Saint Paul Street, Suite 2010']), 'Ave': set(['Lemmon Ave', 'Henderson Ave', 'Oak Lawn Ave', 'Greenville Ave']), 'Gillette': set(['Gillette']), 'Ln': set(['Valley View Ln', 'Livingston Ln']), 'Jolla': set(['La Jolla']), 'Camilla': set(['Camilla']), 'blvd': set(['trinity blvd']), 'East': set(['East Technology Boulevard;Technology Boulevard East', 'Reunion Boulevard East', 'Road to Six Flags East', 'Griffin Street East', 'Technology Boulevard East', 'Us Highway 80 East']), 'LN': set(['CARUTH HAVEN LN']), 'Estrella': set(['Via Estrella']), 'Stonecourt': set(['Stonecourt']), 'Glenwood': set(['Glenwood']), 'Tierra': set(['Nueva Tierra']), 'Dr': set(['W Ledbetter Dr', 'Lone Star Dr', 'W Pioneer Dr']), 'A': set(['Avenue A']), '200': set(['Webb Chapel Rd 200']), 'Pierce': set(['Jo Pierce']), '202': set(['Canton Street, Suite 202']), 'E': set(['Las Colinas Blvd E', 'Avenue E', 'Reunion Blvd E']), 'I-30': set(['West I-30', 'East I-30']), 'I': set(['Avenue I']), 'Av': set(['Lemmon Av']), 'Fernando': set(['San Fernando']), 'Barcelona': set(['Barcelona']), '1150': set(['North Pearl Street, Suite 1150']), 'Q': set(['Avenue Q']), 'Cardiff': set(['Cardiff']), 'Saba': set(['North San Saba', 'South San Saba']), 'South': set(['South Parkway Boulevard South']), 'Birchbrook': set(['Birchbrook']), '356': set(['E State Highway 356']), 'Haskell': set(['Haskell']), 'Hunterwood': set(['Hunterwood']), 'Blvd': set(['North Macarthur Blvd'])})
```

Because this is a very large set of possible problems we need to approach this carefully. We discovered that Dallas metro streets can be particularly unique. So we start with the clearer entries.

The entries for Avenue[A-Q] are correct and will not be altered. Those ending with abbreviations like Rd. or Blvd. are also straight-forward to clean so we will add those to the mapping.

Now we will look at the less clear entries. One entry that is unique to the area is Via Estrella. This is accurate and is not abbreviated and the order is correct, as are all associated listings. https://goo.gl/maps/ZvZEQu72Nos (https://goo.gl/maps/ZvZEQu72Nos). Also in this area are a number of streets that do not have a suffix associated with them. Those have been ignored.

Nile is another oddity, and here in Dallas it can cause a barrage of bug repellent to be aimed in your direction. In this case, Nile should be Nile Drive.

Pleasant Mound is a cemetary in Dallas. The entry was correct to reflect its location at South Buckner Boulevard.

Of the remaining 20 entries I was unable to determine the correct suffix associated with the following 8 entries:

- Halsey
- Slick Rock Chase
- Millmar
- Lynnacre
- Wood Thrush
- Mt Vernon
- Stonecourt
- Hunterwood

These entries are unique in that they are within a short distance of each other and are either correct in one city or have a suffix in the other city.

update_street_name.py

All corrections were added to mappings. This decision was made as it was more straight-forward to correct the data properly than by approaching it with additional scripting.

data_wrangling_schema.sql

Creating the database with data_wrangling_schema.sql resulted in an extra entry in each table. Also an error of datatype mismatch returned for the nodes and ways imports. I was only able to create the tables when 'PRIMARY KEY' was removed.

CREATE TABLE nodes (id INTEGER NOT NULL, lat REAL, lon REAL, user TEXT, uid INTEGER, version INTEGER, changeset INTEGER, timestamp TEXT); CREATE TABLE nodes_tags (id INTEGER, key TEXT, value TEXT, type TEXT, FOREIGN KEY (id) REFERENCES nodes(id)); CREATE TABLE ways (id INTEGER NOT NULL, user TEXT, uid INTEGER, version TEXT, changeset INTEGER, timestamp TEXT); CREATE TABLE ways_tags (id INTEGER NOT NULL, key TEXT NOT NULL, value TEXT NOT NULL, type TEXT, FOREIGN KEY (id) REFERENCES ways(id)); CREATE TABLE ways_nodes (id INTEGER NOT NULL, node_id INTEGER NOT NULL, position INTEGER NOT NULL, FOREIGN KEY (id) REFERENCES ways(id), FOREIGN KEY (node_id) REFERENCES nodes(id));SQLite version 3.9.2 2015-11-02 18:31:45 Enter ".help" for usage hints. sqlite> .read data_wrangling_schema.sql sqlite> .mode csv sqlite> .import nodes.csv nodes sqlite> .import nodes_tags.csv nodes_tags sqlite> .import ways.csv ways sqlite> .import ways_tags.csv ways_tags sqlite> .import ways_nodes.csv ways_nodessqlite> select count(*) from nodes; 3200503 sqlite> select count(*) from ways; 339227 sqlite>

db_create.py

When we created the database with the python script it was successful. When the database was created with the python script db_create.py the queries returned the expected result.

SQLite version 3.9.2 2015-11-02 18:31:45 Enter ".help" for usage hints. Connected to a transient in-memory database. Use ".open FILENAME" to reopen on a persistent database. sqlite> .open dallas_metro.db sqlite> .schema CREATE TABLE nodes (id INTEGER PRIMARY KEY NOT NULL, lat REAL, lon REAL, user TEXT, uid INTEGER, version TEXT, changeset INTEGER, timestamp DATE); CREATE TABLE ways (id INTEGER PRIMARY KEY NOT NULL, user TEXT, uid INTEGER, version TEXT, changeset INTEGER, timestamp DATE); CREATE TABLE nodes_tags (id INTEGER, key TEXT, value TEXT, type TEXT, FOREIGN KEY (id) REFERENCES nodes(id)); CREATE TABLE ways_tags (id INTEGER NOT NULL,key TEXT NOT NULL,value TEXT NOT NULL,type TEXT,FOREIGN KEY (id) REFERENCES ways(id)); CREATE TABLE ways_nodes (id INTEGER NOT NULL,node_id INTEGER NOT NULL, position INTEGER NOT NULL,FOREIGN KEY (id) REFERENCES ways(id),FOREIGN KEY (node_id) REFERENCES nodes(id)); sqlite> .tables nodes nodes_tags ways ways_nodes ways_tags sqlite> select count(*) from nodes; 3200502 sqlite> select count(*) from ways; 339226 sqlite>

Top 10 data creators

We first take a look at which user made the most contributions to the dataset. Here we can see that **Andrew Matheny** contributed the most by a very large margin. It is rather interesting that the bot was behind him by a margin of **4-to-1**.

```
sqlite> select a.user, count(*) as total from (select user from nodes union all select user from ways) a group by a.user order by
total desc limit 10; Andrew Matheny_import|2768934 Andrew Matheny|169403 woodpeck_fixbot|153431 Stephen214|61608
fmmute|24731 brianboru|20865 Zachy_P|18399 dwh1985|16801 AgentBlue|15905 Mara|14914 sqlite>
```

Dallas Metro top 5

To give us a sense of the area we take a look at the top 5 cities in this dataset. Dallas has grown exponentially since I arrived so it is no surprise to see that this portion of the metro is overlapping many cities.

```
sqlite> select city.value, count(*) as total from (select * from nodes_tags union all select * from ways_tags) city where
city.key='city' group by city.value order by total desc limit 5; Dallas|792 Grand Prairie|248 Irving|165 Garland|114 Mesquite|66
sqlite>
```

It's all about those Amenities

Then we took a look at what amenities the area offers. This being Texas, no one should be surprised that religion has the most entries.

```
sqlite> select value, count(*) as total from nodes_tags where key='amenity' group by value order by total desc limit 10;
place_of_worship|917 restaurant|246 fast_food|162 school|103 cafe|52 fuel|44 fire_station|40 post_box|39 toilets|28 parking|24
sqlite>
```

If religion is your thing

When we take a closer look we can see that the majority of entries are categorized as christian.

```
sqlite> select nodes_tags.value, count(*) as total from nodes_tags join(select distinct(id) from nodes_tags where
value='place_of_worship') w on nodes_tags.id=w.id where nodes_tags.key='religion' group by nodes_tags.value order by total
desc limit 1; christian|883 sqlite>
```

Out of curiosity we take an even deeper look to see all entries.

```
sqlite> select nodes_tags.value, count(*) as total from nodes_tags join(select distinct(id) from nodes_tags where
value='place_of_worship') w on nodes_tags.id=w.id where nodes_tags.key='religion' group by nodes_tags.value order by total
desc; christian|883 bahai|2 muslim|2 buddhist|1 hindu|1 jewish|1 scientologist|1 unitarian_universalist|1 sqlite>
```

Park it!

We then take a wider look at amenities and see that parking is a big one. A common problem in any major retro is finding and place to park and Dallas is noexception. Food overall is a close second. Personally, food and coffee are my favorite topics.

```
sqlite> select value, count(*) as total from ways_tags where key='amenity' group by value order by total desc limit 10;
parking|1657 school|444 restaurant|392 fast_food|381 fuel|289 bank|145 place_of_worship|111 car_wash|109 bar|86
post_office|38 sqlite>
```

Food!

Speaking of food we dive in and discover that Mexican cuisine followed closely by pizza and american are popular in the metro. One of the benefits of living in a metro area is the diversity of the food.

sqlite> select nodes_tags.value, count(*) as total from nodes_tags join(select distinct(id) from nodes_tags where value='restaurant') r on nodes_tags.id=r.id where nodes_tags.key='cuisine' group by nodes_tags.value order by total desc; mexican|18 pizza|14 american|12 italian|10 burger|8 chinese|8 asian|5 regional|5 vietnamese|5

More about FOOD!

Finally, we take a slightly different view of food to see that Mexican still is top choice and then the lines begin to blur a big in favor of american and regional cuisines.

sqlite> select ways_tags.value, count(*) as total from ways_tags join(select distinct(id) from ways_tags where value='restaurant') r on ways_tags.id=r.id where ways_tags.key='cuisine' group by ways_tags.value order by total desc; mexican|9 american|6 regional|5 chicken|4 chinese|3 seafood|3 italian|2 japanese|2 pizza|2 sandwich|2 German,_market|1 american;breakfast;fried_chicken|1 american;burger|1 american;steak_house|1 barbecue|1 breakfast|1 burger|1 fish|1 mexican;Street_Tacos|1 texmex|1

Other Thoughts and Ideas

As you can see the possibility for discovery is endless the Dallas metro dataset. One possible approach is investigating how restaurants are broken down categorically. Texas is all about barbeque and it is really surprising that for this area there is only one entry. It could be that this area doesn't have a good representation of barbeque restaurants but I doubt that. A quick look at Yelp tells me that this is indeed incorrect.

One of the most popular apps for getting places in the city is Waze. Google maps is too inconsistent and often does not take into account construction and other possible delay factors that is necessary to get around efficiently. The potential for better information, quicker with accurate updates is avaiable through OpenStreetMap, but this dataset shows that not many people are aware that it exists. 841 unique contributors in a large portion of the metro is essentially demonstrating almost no one knows. Perhaps if more people became aware of the rich data points available and knew that they could play an active role in the discovery of this amazing city we could see greater accuracy. Who knows what treasures may be discovered!

Sources

https://gist.github.com/carlward/54ec1c91b62a5f911c42#file-sample_project-md (https://gist.github.com/carlward/54ec1c91b62a5f911c42#file-sample_project-md) https://www.w3schools.com/sql/default.asp (https://www.w3schools.com/sql/default.asp)
https://www.sqlite.org/docs.html (https://www.sqlite.org/docs.html)
https://discussions.udacity.com/t/are-the-csvs-supposed-to-be-double-spaced/285305 (https://discussions.udacity.com/t/are-the-csvs-supposed-to-be-double-spaced/285305)
https://discussions.udacity.com/t/nodes-csv-1-insert-failed-datatype-mismatch/239638 (https://discussions.udacity.com/t/nodes-csv-1-insert-failed-datatype-mismatch/239638)