Map Area

Miami, Florida, USA
Link: https://mapzen.com/data/metro-extracts/metro/miami_florida/85933669/Miami/
I chose this map because I went to my graduate school in Miami and know this place pretty well. I was curious to learn more and possibly contribute to OpenStreetMap project by undertaking something I am familiar about.

Problems Encountered in the Map

A careful look at a small subset of a map data using a text editor and short python scripts reveals the following issues and comments.
1. An extensive variety of abbreviation used for street such as Trl, Ln, Pl, Rd, Ave, Ter, Dr, Ct, Cir. Although others were pretty staright forward, I looked up for the street "Sabal Trl" and found out that “Trl” actually refers to trail.

2. The name of the county was always accompanied by the state. For example "Broward, FL". This could possibly be a format from the TIGER data.

3. Inconsistent format issues with postal codes, example, FL-33140, FL 33134, 33029 etc.

4. Incorrect postal codes. All postal codes form Miami should begin with 3 but some entrees start with 8 and 9.

Fixing Abbreviated Street Names

Auditing street name was done using the python regular expression and the following mapping scheme.

```
mapping = { "St": "Street",  "St.": "Street",
        "Ave": "Avenue", "Ave.": "Avenue",
        "Rd.": "Road", "Rd": "Road",
        "Pl": "Place",  "Pl.": "Place",
        "Trl": "Trail", "Ln": "Lane",
        "Dr": "Drive", "Dr.": "Drive",
        "Ct": "Court", "Ct.": "Court",
        "Cir": "Circle", "Ter": "Terrace"
        }
```

Postal Codes

Postal codes mainly had formatting issues. Some of the inconsistent issues are FL 33487-3536, FL-33139, 33472. Since all postal codes in Miami should start with 3, I used a regular expression facility in python to force a 5-digit zip code format starting with 3.
Given below is the SQL query for extracting the postal codes.

```
Sqlite> SELECT tags.value, COUNT(*) as count
            FROM (SELECT * FROM nodes_tags UNION ALL SELECT * FROM ways_tags)
            tags WHERE tags.key='postcode'
            GROUP BY tags.value ORDER BY count DESC;
```

Given below are the top 10 zip codes with the highest count.

33327,6770
33326,6470
33331,3117
33135,2150
33332,1883
33125,1804
33142,1783
33133,1412
33145,1283
33126,1213

Also, two of the zip codes (82914, 91730) were completely out of place. After using the following queries to locate the latitude and longitude,

A) Query for zipcode 82941

```
sqlite> SELECT *
   FROM nodes
   WHERE id IN (SELECT DISTINCT(id) FROM nodes_tags WHERE key='postcode' AND
            value='82941');
```

3919421157,26.2350009,-80.2539469,MountainAddict,2679756,1,36272094,2015-12-30T21:48:47Z

The geographical coordinates (26.2350009,-80.2539469) actually point to a place in Coral Springs in Miami but the zipcode 82941 points to Pinedale, WY.

B) Query for zipcode 91730

```
sqlite> select * from nodes WHERE id IN (SELECT DISTINCT(id) FROM nodes_tags
          WHERE   key='postcode' AND value='91730');
```

4065776313,25.4898499,-80.4574628,iMarketSolutions,3692214,1,37906776,2016-03-17T20:59:28Z

The geographical coordinates (25.4898499,-80.4574628) points to a place in Homestead city that is in the neighborhood of Miami but the zipcode 91730 actually points to Rancho Cucamonga, CA. These entries are obviously wrong.

A few other wrong entries in place of zip code value include phone number ("(561) 795-4333"), a four digit number (3447), a text (FL), single digit (0) and a 3 digit code ("361-0529”). All of these have to be fixed in the database.

Data Overview and Exploration
This section covers overall data file sizes, SQL queries for interrogating various aspects of the data and additional analyses.

File Sizes

miami_florida.osm	611 MB
miami_florida.db	317 MB
nodes.csv	226 MB
nodes_tags.csv	11 MB
ways.csv	20 MB
ways_tags.csv	57 MB
ways_nodes.cv	72 MB

Number of Nodes

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

2618595

Number of Ways

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

329685

Number of Unique Users

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

1807

Total Number of Posts

```
sqlite> select count(*)
        FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) e;
```

2948280

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;
```

MiamiBuildingsImport	854087
grouper	352780
carciofo	265018
woodpeck_fixbot	226268
Latze	135924
freebeer	116207
bot-mode	54634
NE2	50612
Seandebasti	50383
westendguy	45003

Number of Users having single post

```
sqlite> SELECT COUNT(*)
        FROM
            (SELECT e.user, COUNT(*) as num
             FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) e
            GROUP BY e.user
             HAVING num=1)  u;
```

547

Diversity in User Contribution
The topmost contributor contributes to about 29 percent of the total posts. The top four contributors account for about 58 percent of the total posts. Total posts sum to 2948280.

Additional Statistical Analysis:

Top 10 appearing amenities

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

School	1492
Restaurant	567
place_of_worship	563
kindergarten	521
fast_food	304
fountain	297
fire_station	292
fuel	240

parking 232
police 186

Top 10 religions in terms of number of places of worship

```
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 10 ;
```

christian,522
jewish,16
buddhist,3
unitarian_universalist,2
muslim,1
scientologist,1

Most popular cuisines

```
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 10;
```

pizza,36
italian,32
american,27
mexican,15
seafood,12
chinese,11
burger,9
cuban,8
latin_american,8
sandwich,7

Conclusion

The open street map data on Miami, Florida includes all cities surrounding Miami. I noticed an extensive variety of abbreviations used for street. There are issues with some zip code entries which have to be fixed. One way to improve the data is to validate the match between the geographical coordinates, zip code and city name. This can be done using any web facility that maps the geographical coordinates to the place. The validation would straight away point out any discrepancies that exists, specially the blatant errors in zip codes can be fixed right away. Though less likely, if a given building or landmark is erroneously associated to specific geographical coordinates, we need a manual comparison to other maps such as google or TIGER data, or do a field trip.