Wrangling with OpenStreetMap Data

Contents

[About 3](#_Toc423371359)

[Problems Encountered in the Map 3](#_Toc423371360)

[Street Names 3](#_Toc423371361)

[Postal Codes 3](#_Toc423371362)

[Data Overview 4](#_Toc423371363)

[File Sizes: 4](#_Toc423371364)

[Number of documents 4](#_Toc423371365)

[Number of nodes 4](#_Toc423371366)

[Number of ways 4](#_Toc423371367)

[Number of unique users 4](#_Toc423371368)

[Top 5 contributing users and their number of documents 4](#_Toc423371369)

[Number of users appearing only once 4](#_Toc423371370)

[Number of nodes without any address 5](#_Toc423371371)

[Top 10 amenities 5](#_Toc423371372)

[Top 10 cuisines 5](#_Toc423371373)

[Additional Ideas 5](#_Toc423371374)

[Out of range postal codes 5](#_Toc423371375)

[Nodes without Addresses 6](#_Toc423371376)

[Skewed Contributor statistics and Social Media 6](#_Toc423371377)

[Additional Data Exploration Observations 6](#_Toc423371378)

[Conclusion 6](#_Toc423371379)

# About

The aim of this project is to read the data downloaded from openstreetmap.com stored in an XML format, audit the data for cleanliness and consistency, convert the data into JSON, load it into a MongoDB database, and draw a few basic statistics.

* Orlando, Florida
* [OpenStreetMap URL](http://www.openstreetmap.org/relation/1128379#map=11/28.4813/-81.3676)
* [Mapzen URL](https://s3.amazonaws.com/metro-extracts.mapzen.com/orlando_florida.osm.bz2)

I choose Orlando because I recently moved here from Chicago and I wanted to explore the area around me as a Data Scientist.

# Problems Encountered in the Map

After downloading the dataset for Orlando, I did a number of iterations of running the audit.py, analyzing the problematic strings, modifying the audit.py and re-running the audit.py to make sure it’s working. I, mainly, focused on Street Names and postcodes.

## Street Names

The code we developed during the case study was to clean only the suffixes. So, I made a number of changes in the code to clean up the Street Names properly. Below is a summary of the observations. For the entire code, please refer to audit.py:

* Upon analyzing the valid suffixes for the Street names I found out that there were a number of suffixes need to be added in the expected tags such as Trail, Turnpike etc.
* Then, to clean the abbreviations I added St., St, Rd, Ave and the rest, which were applicable to Orlando data, to mapping variable
* Then, I analyzed the entire street names and saw a few occurrences of abbreviated street names starting with S and N. So, using REGEX, I added helper functions/variables and cleaned out those in a similar way to suffixes.
* Then, there were a few instances where the abbreviations were in the middle of the street names. Thankfully, those were only a handful, so I was able to clean them up in no time.
* Finally, for making the Street Names look more elegant, I added a code snippet to make the first letter capital of each word.

## Postal Codes

In the audit.py, I added one function to clean the postal codes in the Orlando data. Though, there were only a few codes which needed to be cleaned out. Below is a summary of my observations:

* Although, most of the codes were 5 digit strings, a few were starting with FL. So, I cleaned those out using simple sub function from re package.
* Then, there were a few postal codes which had 4 additional digits after a hyphen which is perfectly valid. However, to make the codes consistent, I stored only first 5 digits from the all the codes.

# Data Overview

Using Python scripts and Mongo shell, I came up with the following statistics:

## File Sizes:

* orlando\_Florida.osm 86.7 MB
* Orlando\_Florida.osm.json 94.2 MB

## Number of documents

> db.orlando.find().count()

412983

## Number of nodes

> db.orlando.find({'type':'node'}).count()

352621

## Number of ways

> db.orlando.find({"type":'way'}).count()

60355

## Number of unique users

> db.orlando.distinct('created.user').length

322

## Top 5 contributing users and their number of documents

> db.orlando.aggregate([{'$group': {'\_id': '$created.user','count': {'$sum': 1}}}, {'$sort': {'count': -1}}, {'$limit': 5}])

{ "\_id" : "NE2", "count" : 234609 }

{ "\_id" : "crystalwalrein", "count" : 30270 }

{ "\_id" : "epcotfan", "count" : 20884 }

{ "\_id" : "3yoda", "count" : 19144 }

{ "\_id" : "dale\_p", "count" : 12442 }

## Number of users appearing only once

>db.orlando.aggregate([{'$group':{'\_id':'$created.user','count':{'$sum':1}}},{'$group':{'\_id':'$count','num\_users':{'$sum':1}}},{'$sort':{'\_id':1}},{'$limit':1}])

{ "\_id" : 1, "num\_users" : 71 }

## Number of nodes without any address

>db.orlando.aggregate([{'$match':{'type':'node','address':{'$exists':0}}},{'$group':{'\_id':'NodesWithoutAddress','count':{'$sum':1}}}])

{ "\_id" : "NodesWithoutAddress", "count" : 351941 }

## Top 10 amenities

>db.orlando.aggregate([{'$match':{'amenity':{'$exists':1}}},{'$group':{'\_id':'$amenity','count':{'$sum':1}}},{'$sort':{'count':-1}},{'$limit':10}])

{ "\_id" : "parking", "count" : 553 }

{ "\_id" : "restaurant", "count" : 299 }

{ "\_id" : "fast\_food", "count" : 259 }

{ "\_id" : "place\_of\_worship", "count" : 240 }

{ "\_id" : "fuel", "count" : 108 }

{ "\_id" : "school", "count" : 89 }

{ "\_id" : "fountain", "count" : 72 }

{ "\_id" : "bank", "count" : 53 }

{ "\_id" : "parking\_space", "count" : 40 }

{ "\_id" : "swimming\_pool", "count" : 36 }

## Top 10 cuisines

> db.orlando.aggregate([{'$match':{'amenity':{'$exists':1}, 'amenity':'restaurant'}},{'$group':{'\_id':'$cuisine','count':{'$sum':1}}},{'$sort':{'count':-1}},{'$limit':10}])

{ "\_id" : null, "count" : 255 }

{ "\_id" : "american", "count" : 12 }

{ "\_id" : "italian", "count" : 6 }

{ "\_id" : "mexican", "count" : 5 }

{ "\_id" : "chinese", "count" : 3 }

{ "\_id" : "greek", "count" : 2 }

{ "\_id" : "sushi", "count" : 2 }

{ "\_id" : "steak\_house", "count" : 2 }

{ "\_id" : "asian", "count" : 2 }

{ "\_id" : "burger", "count" : 2 }

# Additional Ideas

Here are a few ideas/observations which I would like to mention:

## Out of range postal codes

I would like to mention is that Orlando’s airport is outside the city which makes it really hard to map the area since there are 2 bounded regions which are connected. Most of the Orlando postal codes begin with 328. However, I found these 2 sets of postal codes:

* 2 postal codes starting with 347 – 34786 (Windermere, FL) and 34761 (Ocoee, FL) which, in my opinion, shouldn’t have been a part of the Orlando Map.
* 3 postal codes starting with 327 which were very close to Orlando area and it would be up to reader’s interpretation how he or she would go about it.

## Nodes without Addresses

As we can see there are total number of nodes is 352621. However, out of these 351941 (99.81%) do not have any address. So, if we are looking at an application which provides you with point of interests, then it would be a good idea to maybe cleanse/suppress these nodes and then further process the data.

## Skewed Contributor statistics and Social Media

As we can see from the query results, NE2 alone made more than 56.80% contribution. And, top 5 users made more than 75% contribution to the map data. I googled and found out that NE2 is Russ Nelson who is founding board member of the Open Source Initiative. If such busy people can contribute so much to the free data, then imagine a world where people pledge to contribute/correct/review just one address be it their home/business/office. Eventually, this benefit would them as well, since they would be able to highlight their businesses as well. So to, further motivate people I guess, in the world of social media, it would be a good idea to keep Facebook page for OpenStreeMap Foundation more active and happening. This would raise awareness about their initiative and people would be able to comment if they found something odd about any of the addresses. Then can use google docs or any other forums where if people find anything missing/incorrect, they can submit their inputs easily.

## Additional Data Exploration Observations

From the queries I ran above it is clear that the data still lacks important information. For example if we consider cuisine, then 255 restaurants are missing this information. So, if we are considering using this data in an app which groups restaurants by their cuisine type then, it would miss quite a high number of restaurants.

Amenities also need to be cleaned because there were 2 amenities for parking – parking and parking\_space.

# Conclusion

The Orlando data is clearly messy though it has been updated by many users. While there are many additional opportunities for cleansing and validating the Orlando data, in my opinion the data set was well-cleaned for the purposes of this exercise.