Data Wrangling with MongoDB - Open Street Maps Final Project

Data Summary:

The data used for this project was from the Tampa, Florida metropolitan area. A simple audit of the data showed that there were the following top-level tags:

```
 'bounds': 1,
 'member': 29974,
 'nd': 1663301,
 'node': 1405157,
 'relation': 1067,
 'tag': 1042342,
 'way': 156590
```

Within each of these tags there were a total of 938 unique child tags. When sorted by number, the 'highway' tag was the most populous (637186). Within the data, the 'addr:' child tags were in the top 100 most populous tags.

I focused on the "addr:" tag in the data. This tag, asopposed to the "tiger:" tag, is one that can be entered into the data manually. Since most of the data was manually entered, I thought cleaning the address data would have the greatest impact on the overall quality of the data. Within the address code I focused on three areas:

1. Street names
2. City names
3. Postal codes

Problems encountered in the map:

Street Names:

Within the "addr:street" tag there were three main problems: over-abbreviation of some street names, inconsistencies in whether the street direction (North, South, Northeast, etc.) was at the beginning or end of the street name, and inconsistencies in naming US and Florida highways. I addressed each of these problems in the final project code. I updated the substrings in the "addr:street" string such that all street names were fully spelled out, the street direction was fully spelled out and located at the beginning of the string, and all highways were designated as either "State Road" (for a Florida highway - according to local convention) or "US" for US highways. See update_street_name and update_direction in the final project code.

City Names:

Within the “addr:city” tag, there were some inconsistencies in the spelling and abbreviation of Saint Petersburg, Saint Petersburg Beach, and a few other cities (e.g.: St. Petersburg, St Petersburg, St Pete Beach). I updated all substrings such that each city name was capitalized and fully spelled out. See update_city in the final code.

Postal Codes:

Some of the postal codes entered in the “addr:postcode” tag contained the 5-digit zip code followed by a 4-digit unique identifier. In order to make all the data consistent, I removed the four-digit unique identifier from the postal codes. See update_postcode in the final project code.

Data Overview:

Size of file:

.osm : 310.842 MB
.json : 362.678 MB

Number of Documents:

```
db.tampa.count()    1561747
```

Number of Unique Users:

```
db.tampa.distinct(“created.user”).length    957
```

Number of Nodes:

```
db.tampa.find({type:"nodes"}).count()      1405053
```

Number of Ways:

```
db.tampa.find({type:"way"}).count()     156572
```

Top 10 Users:

```
db.tampa.aggregate({"$group": {
        "_id": "$created.user",
        "count": {"$sum": 1}}},
        {"$sort": {"count": -1}},
        {"$limit": 10})
```

```
user: number of documents
woodpeck_fixbot: 246061
coleman: 241804
grouper: 161539
EdHillsman: 111855
```

```
NE2: 78233
David Hey: 62511
LnxNoob: 60670
westampa: 42692
bot-mode: 41339
Chris Lawrence: 28464
```

Number/type of places of worship:

```
db.tampa.aggregate([{“$match”:{“amenity”:{“$exists”:true}, “amenity”:”place_of_worship”,
                              “religion”:{“$exists”:true}}},
                             {“$group”:{
                                    “_id”:”$religion”,
                                    “count”:{“$sum”:1}}},
                             {“$sort”:{“count”:-1}}])

{ "_id" : "christian", "count" : 784 }
{ "_id" : "jewish", "count" : 6 }
{ "_id" : "unitarian_universalist", "count" : 4 }
{ "_id" : "muslim", "count" : 3 }
{ "_id" : "buddhist", "count" : 3 }
{ "_id" : "scientologist", "count" : 3 }
{ "_id" : "bahai", "count" : 3 }
{ "_id" : "eckankar", "count" : 1 }
{ "_id" : "spiritual_living", "count" : 1 }
```

Most popular cuisines:

```
db.tampa.aggregate([{"$match":{"amenity":{"$exists":true},
                             "amenity":"restaurant","cuisine":{"$exists":true}}},
                             {"$group":{
                                    "_id":"$cuisine",
                                    "count":{"$sum":1}}},
                             { "$sort":{"count":-1}}])

{ "_id" : "american", "count" : 132 }
{ "_id" : "pizza", "count" : 81 }
{ "_id" : "seafood", "count" : 38 }
{ "_id" : "mexican", "count" : 38 }
{ "_id" : "italian", "count" : 37 }
{ "_id" : "chinese", "count" : 29 }
{ "_id" : "diner", "count" : 22 }
{ "_id" : "thai", "count" : 21 }
{ "_id" : "chicken", "count" : 21 }
{ "_id" : "burger", "count" : 20 }
{ "_id" : "barbecue", "count" : 19 }
{ "_id" : "sushi", "count" : 18 }
{ "_id" : "greek", "count" : 18 }
{ "_id" : "sandwich", "count" : 16 }
{ "_id" : "steak_house", "count" : 16 }
{ "_id" : "asian", "count" : 14 }
{ "_id" : "japanese", "count" : 12 }
{ "_id" : "latin_american", "count" : 11 }
{ "_id" : "vietnamese", "count" : 11 }
{ "_id" : "indian", "count" : 8 }
```

Other ideas about the data:

Incomplete changest logs:

I wanted to explore the “changeset” tag. According to the OpenStreetMap wiki, the changesets can be tagged in ways that identify where the data came from. In particular, the “source” tag and “bot” tag can help identify whether particular data was input by a person or by a bot. Given that 2 of the top 10 users have “bot” in their username, it is reasonable to suspect that much of this data was generated by a bot. However, when I queried the changset tags,I found very few tags for “source” (22435 tags - less than1.5% of the total number of documents) and no tags for “bot”. For the top 10 users, the number source tags used are as follows:

```
woodpeck_fixbot: 157
coleman: 712
grouper: 409
EdHillsman: 4052
NE2: 1149
David Hey: 69
LnxNoob: 74
westampa: 28
bot-mode: 64
Chris Lawrence: 0
```

The MongoDB query to find source and bot tags is:

```
db.tampa.aggregate({“$match”:{“source”:{“$exists”:true}}},
                   {“$group”:{“_id”:”$created.user”,
                             ”count”:{“$sum”:1}}},
                   {“$sort”:{“count”:-1}})

db.tampa.aggregate({“$match”:{“bot”:{“$exists”:true}}},
                   {“$group”:{“_id”:”$created.user”,
                             ”count”:{“$sum”:1}}},
                   {“$sort”:{“count”:-1}})
```

Analysis of the data could be much improved by mandating the addition of the source and bot tags in the changeset tag. I imagine that bot entered data would be more regularized than human entered data, and so further data cleaning resources could be focused on the regularizing the human entered data first by simply filtering the data using the bot and source tags. However, retroactive incorporation of these tags is difficult/impossible, so this strategy will not work for existing data.

Mapping latitude and longitude to focus resources on under-annotated areas

This dataset is comprised of map data from urban, suburban, and rural areas. I imagine that the number of documents is larger for the urban areas than for the suburban areas. It would be interesting to map the latitude and longitude of each tag onto a geographic map. I realize this is exactly what OSM does, but I’m suggesting a simple scatter plot of the pos tag that will give users a quick overview of under-annotated portions of the OSM data. This simple visualization would allow users to focus their efforts on annotating areas of the map where little information exists.