OpenStreetMap Project for Udacity Nano degree Data Wrangling with MongoDB - *Mohammad Yaqoob*

1. Map Selection and Summary

I researched the cities on openstreetmap.org to find an interesting place to use as my data for this project. Most of the cities data was huge and my system was not able to run it. I found Tucson, AZ reasonable size that my system can handle. Hence I used Tucson, AZ for my project.

https://www.openstreetmap.org/export#map=11/32.1564/-110.8832

We learned the following munging techniques in the Data Wrangling with MongoDB class to assess the quality of data:

- Validity
- Completeness
- Accuracy
- Uniformity
- Consistency

I used some of these techniques to clean the selected data for this project. The following is the list of the files I used to perform Exploratory Data Analysis and perform data munging:

- users.py - used to explore users contributed and the total count.
- mapparser.py – used to explore tags and count of each tag.
- exppostcode.py – used to explore postal code.
- expphone.py – Used to explore Phone numbers.
- audit2.py – used to explore street names.
- tags.py – Used to explore tags for problems.
- data2.py – To shape data to import to MongoDB

I ran iterative parsing to process the map file to find out not only what tags are there, but also how many, to get the feeling on how much of which data I can expect to have in the map. Using mapparser.py found the following tags and there counts respectively.

{'bounds': 1, 'member': 6368, 'nd': 368017, 'node': 296924, 'osm': 1, 'relation': 231, 'tag': 285310, 'way': 36651}

2. Problems Encountered with Data

I ran the data against above mentioned scripts to explore different components of data to find out problems, inconsistencies, validations, uniformity, and completeness. I found the following main problems with the data:

- Found abbreviated street names ("E 6th St", "N. Arcadia Blvd.")
- Found inconsistent postal codes format ("85231", "85721-0078")
- Found inconsistent phone numbers format ("+1 520 621 1608", "520-879-2802", "(520) 638-6003", "5208792802")

Abbreviated Street Names

We have 309 unique users who have contributed to this map data. Hence, it is natural to find different styles and notations used for street names. Using audit2.py program I was able to explore different street names abbreviations used and updated all the abbreviations with detail names as follow:

E GRANT RD => East GRANT Road
W KING RD => West KING Road
W Old Magee Trl => West Old Magee Trail
E 6th St => East 6th Street
S Swan Rd => South Swan Road
N. Arcadia Blvd. => North Arcadia Boulevard

Postal Codes

EDA of postal code strings did not show any drastic inconsistencies in the map data. My EDA did not find any leading and trailing characters before and after the main 5-digit zip code. Only one zip code was found with 4-digit zip code extensions following a hyphen ("85721-0078"). I decided to accept both of the following formats since they are valid formats as follow:

85231 => valid
85721-0078 =>valid

If I observer issues with MongoDB aggregation calls on postal codes, I will update within MongoDB to constrict zip code to 5 digits for that one record.

Phone numbers

Phone number strings showed inconsistencies in data representation, using data munging techniques I cleaned the international code, spaces, and dashes in the phone numbers to make them consistent for local USA town.

+1 520 621 1608 => 5206211608
520-879-2802 => 5208792892

(520) 638-6003 => 5206386003

3. Overview of the Data

This section contains basic statistics about the dataset and the MongoDB queries used to gather them.

File sizes

```
tucson.osm ......... 70.1 MB     tucsonsample.osm ......... 7.11 MB
tucson.osm.json .... 75.9 MB     tucsonsample.osm.json .... 7.60 MB
```

Size

```
> db.map.dataSize()
105402000
```

Documents

```
>db.map.find().count()
333575
```

Nodes

```
> db.map.find({"type":"node"}).count()
296922
```

Ways

```
> db.map.find({"type":"way"}).count()
36650
```

Unique users

```
> db.map.distinct({"created.user"}).length
309
```

Cafe

```
> db.map.find( { amenity: "cafe" } ).count()
29
```

School

```
> db.map.find( { amenity: "school" } ).count()
259
```

Shop

```
> db.map.find( { shop: { $exists: true } } ).count()
241
```

Top 5 contributing user

```
> db.map.aggregate([{"$group":{"_id":"$created.user",
"count":{"$sum":1}}}, sort":{"count":1}}, {"$limit":5}])
```

```
{ "_id" : "GreggTownsend", "count" : 53764 }
{ "_id" : "ianmcorvidae", "count" : 46326 }
{ "_id" : "KristenK", "count" : 32136 }
{ "_id" : "Tom Layo", "count" : 31604 }
{ "_id" : "woodpeck_fixbot", "count" : 24762 }
```

Top 10 Amenities

```
> db.map.aggregate([{"$match":{"amenity":{"$exists":1}}},
{"$group":{"_id":"$amenity", "count":{"$sum":1}}}, {"$sort":{"count":-
1}}, {"$limit":10}])
{ "_id" : "parking", "count" : 671 }
{ "_id" : "university", "count" : 268 }
{ "_id" : "school", "count" : 259 }
{ "_id" : "restaurant", "count" : 113 }
{ "_id" : "fast_food", "count" : 90 }
{ "_id" : "place_of_worship", "count" : 44 }
{ "_id" : "hospital", "count" : 36 }
{ "_id" : "fuel", "count" : 33 }
{ "_id" : "cafe", "count" : 29 }
{ "_id" : "public_building", "count" : 28 }
```

4. Additional Ideas

I ran couple of additional MongoDB queries to find more about the data that can help in bringing out some additional ideas for this data. These queries pointed me to some of my additional ideas with respect to this data.

1: 55% of the contribution is from 5 contributors (manual and automated). Even though the contribution is not heavily skewed, it is still not completely uniform across all contributors. 55 contributors only contributed once (see additional queries) this might suggest lack of motivation to contribute for these and other low contributing users. Some sort of system is needed to motivate contributors e.g. display board with top contributors with stars (1 star = 5% contribution, 2 star = 10% contribution) next to their names to spur contribution from users. Some sort of reward system (gift certificates, coupons) from local business to 4 star contributors probably another idea that can be used by working closely with local businesses.

2: Queries about most popular cuisine and places of worship show significant number with null information. This indicates opportunity to get contributors involved in resolving the missing data. This also represents the opportunity to develop intelligent bots to crawl the web with current information and search for missing information regarding the establishment and perform automated updates.

.

Other Queries:

1: Top 5 User count with lowest contributions.

```
> db.map.aggregate([{"$group":{"_id":"$created.user",
"count":{"$sum":1}}}, {"$group":{"_id":"$count",
"numusers":{"$sum":1}}}, {"$sort":{"_id":1}}, {"$limit":5}])
{ "_id" : 1, "numusers" : 51 }
{ "_id" : 2, "numusers" : 24 }
{ "_id" : 3, "numusers" : 16 }
{ "_id" : 4, "numusers" : 12 }
{ "_id" : 5, "numusers" : 8 }
```

2: Top 5 Cuisines.

```
> db.map.aggregate([{"$match":{"amenity":{"$exists":1},
"amenity":"restaurant"}}, {"$group":{"_id":"$cuisine",
"count":{"$sum":1}}},{"$sort":{"count":-1}}, {"$limit":5}])
{ "_id" : null, "count" : 52 }
{ "_id" : "mexican", "count" : 15 }
{ "_id" : "italian", "count" : 6 }
{ "_id" : "chinese", "count" : 6 }
{ "_id" : "greek", "count" : 5 }
```

3: Top 5 User places for worship.

```
 > db.map.aggregate([{"$match":{"amenity":{"$exists":1},
"amenity":"place_of_worship"}},{"$group":{"_id":"$religion",
"count":{"$sum":1}}}, {"$sort":{"count":-1}
}, {"$limit":5}])
{ "_id" : "christian", "count" : 33 }
{ "_id" : null, "count" : 7 }
{ "_id" : "muslim", "count" : 2 }
{ "_id" : "buddhist", "count" : 1 }
{ "_id" : "jewish", "count" : 1 }
```

Conclusion

The data revealed some interesting facts about Tucson, AZ. It shows Tucson is vibrant multiethnic city (see Cuisine, places for worship). Although, I cleaned the data enough for the purpose of this exercise, but there is still need cleaning the data. It has rich tags creating many possibilities to perform analysis. It has all the basics labeled e.g. streets, official buildings, universities, places for worship, and commercial places. With top 4 contributors actual users and not automated bots indicate that OSM is known in the area.