Project Report: Data Wrangling with MongoDB

Sumit Srivastava

Map Area: New Delhi, India

1. Problems Encountered in the Map

I downloaded the new_delhi.osm form the MapZen website. Parsing the data showed me the following problems in the dataset.

a) **Some address names were over abbreviated** -
   Ex: "h/no 1/55 sadar bazar delhi cantt" should be "House Number 1/55 sadar bazaar delhi cantonment"
   "Sec - 19, Poket 3, Dwarka" should be "Sector – 19 Pocket 3, Dwarka"
   Such abbreviations were removed programmatically to ensure that address names have consistency for entire dataset

b) **Unexpected address in "ways" and "node"**
   Although the dataset is for New Delhi, the capital city of India, some of the addresses belonged to the neighboring states.
   Ex: 1) "Village Chhalera & Sadarpur, Sadarpur, Sector 45, Noida". This address is in the state of Uttar Pradesh and not in New Delhi
   2) "Old Faridabad-Jasana Road". This address is in the state of Haryana and not in New delhi
   Considering the presence of such addresses, I think it is more appropriate to call the dataset as 'National Capital Territory' instead of New_Delhi. Since these addresses belong to actual places in vicinity of New Delhi I did not remove them from the dataset.

The task of reshaping the element present in 'new_delhi.osm' file and changing the abbreviations in address name was performed using ***'insert_new_delhi.py'*** program. The code for this program is present in the project submission. The input to this program was 'new_delhi.osm' file and output was 'new_delhi.osm.json'. This file was inserted into mongoDB to study the data.

2. Data Overview

I decided to use the openstreet data related to New Delhi, India.
Following are some of the basic statistics associated with the data

File sizes

1) new_delhi.osm ..... 106 MB
2) new_delhi.osm.json .... 122 MB

Tag Types present in the data

1) Number of nodes: 502930
2) Number of ways: 87470
3) 'bounds': 1
4) 'member': 13577,
5) 'nd': 650899,
6) 'osm': 1,
7) 'relation': 2389,
8) 'tag': 194313,

a) These tag comments were generated by parsing the 'new_delhi.osm' file using an iterative parser.
b) **'count_tags.py'** was used to generate the tags. The code is present in project submission

Some Basic queries

a) 'new_delhi.osm.json' was imported into mongoDB using the following directive

➔ *mongoimport --db examples –c new_delhi --file new_delhi.osm.json*

b) Once the database was imported into mongoDB, I ran some queries against it using the **'basic_queries.py'** program. The code for 'basic_queries.py' is present in project submission.

Number of Documents
Query - *db.new_delhi.find().count()*
Result – 590400

Number of Nodes
Query - *db.new_delhi.find({"type":"node"}).count()*
Result – 502930

Number of Ways
Query - *db.new_delhi.find({"type":"way"}).count()*
Result – 87470

c) An aggregation based query was also created to get the above result
Query - *result = db.new_delhi.aggregate([{"$group":{"_id":"$type",*
"count":{"$sum":1}}}])

Result - {u'count': 87470, u'_id': u'way'}

{u'count': 502930, u'_id': u'node'}

 # Number of distinct users

```
disctinct_users = db.new_delhi.distinct("created.user")
print "Number of distinct user is ", len(disctinct_users)
```

Result - **Number of distinct user is 687**

 # Number 1 contributing user

```
Aggregation pipeline = [{"$group":{"_id":"$created.user",
                                   "count":{"$sum":1}}},
                        {"$sort":{"count":-1}},
                        {"$limit":1}]
```

Result - {u'count': 269215, u'_id': u'Oberaffe'}

So the top contributing user is – *Oberaffe*

 # Number of users who have only 1 post

```
Aggregation pipeline = [{"$group":{"_id":"$created.user",
                                   "count":{"$sum":1}}},
                        {"$group":{"_id":"$count",
                                   "user_number":{"$sum":1}}},
                        {"$sort":{"_id":1}},
                        {"$limit":1}]
```

Result - *{u'_id': 1, u'user_number': 135}*

So the number of users who have contributed only once is 135

3. Additional Ideas

An analysis of the amenities field (explored further in section 4) shows that many useful information is currently missing in the data. For example

1) The total number of places of worship is 299, however for 59 of them, the religion field is kept empty i.e 20% of total
2) Similarly out of 133 restaurants, 87 of them do not have information about the cuisines field, i.e 65% of total

This indicates that the data is still incomplete to a large extent. For capital city of India, the number of restaurants must be certainly higher than 133. Even for restaurants that are present, several important fields are missing. Increasing awareness among public amenities like restaurants and schools about the existence of this database and how it can help them in increasing their business will go a long way in building a more comprehensive database for New Delhi.

4. Additional data exploration using MongoDB queries

#**Top 10 appearing amenities**

```
Aggregation  pipeline = [{"$match":{"amenity":{"$exists":1}}},
                         {"$group":{"_id":"$amenity",
                         "count":{"$sum":1}}},
```

```
{"$sort":{"count":-1}},
{"$limit":10}]
```

Result

Amenity	**Count**
school	**880**
place_of_worship	**299**
parking	**289**
fuel	**197**
hospital	**179**
restaurant	**133**
atm	**130**
college	**124**
bank	**102**
police	**85**

#**Top religion** (i.e. religion having maximum number of place of worship)

```
Aggregation  pipeline = [{"$match":{"amenity":{"$exists":1},
                                    "amenity":"place_of_worship"}},
                         {"$group":{ "_id":"$religion"
                                    "count":{"$sum":1}}},
                         {"$sort":{"count":-1}},
                                {"$limit":1}]
```

Result - {u'count': 122, u'_id': u'hindu'}

#**Number of places of worship for different religion**

```
Aggregation  pipeline = [{"$match":{"amenity":{"$exists":1},
                                    "amenity":"place_of_worship"}},
                         {"$group":{ "_id":"$religion"
                                    "count":{"$sum":1}}},
                         {"$sort":{"count":-1}},
                        ]
```

Result:

Religion	Count
Hindu	122
None	59
Muslim	47
Christian	33
Sikh	28
Jain	5
Buddhist	3
Zoroastrian	1
Bahai	1

Important: As we can see for around 59 places of worship, the religion field is empty. This indicates that the dataset is still incomplete. The top religion is Hindu, while muslims, Christianity and Sikhism occupy the top 5 slot.

Top 5 Different cuisines in restaurant

```
Aggregation_pipeline = [{"$match":{"amenity":{"$exists":1},
                                   "amenity":"restaurant"}},
                        {"$group":{"_id":"$cuisine",
                                   "count":{"$sum":1}}},
                        {"$sort":{"count":-1}},
                         {"$limit":5} ]
```

Result :

Cuisine	**Count**
None	**87**
Indian	**9**
Chinese	**5**
Regional	**4**
Vegetarian	**3**

Important: As we can see for around 87 restaurants, the cuisine field is empty. This indicates that the dataset is still incomplete. After 'None', the top cuisine is 'Indian' which is not a surprise.

Conclusion

1) In this project, we first downloaded openstreet data about new delhi and then cleaned it using the 'insert_new_delhi.py' program. While cleaning we discovered that it would be more appropriate to call this data as 'national_capital_territory.osm' since a lot of neighboring cities are also present in this data.
2) Considering the number of empty fields that are present in the amenities section, it is clear that the dataset is incomplete. All the statistics shown in this report were generated using the 'basic_queries.py' program.