DATA OVERVIEW

I have analyzed the city of Richmond, Virginia. I have lived in Virginia for some time and though this would be an interesting project.. I also looked at the number and types of tags:

File size:

richmond_virginia.osm -> 126 MB

Richmond_virginia.osm.json -> 161 MB

Number of Nodes:

db.richmond_virginiaC.find({"type":"node"}).count()

node: 1156838

db.richmond_virginiaC.find({"type":"way"}).count()

way: 115006

db.richmond_virginiaC.find().count()

Documents: 1271860

Number of Edits/Updates:

Edited/updates- 1271860

Number of Unique users:

db.richmond_virginiaC.distinct("created.user").length

Unique users- 364

Top five contributors:

```
result=db.richmond_virginiaC.aggregate([
        #{"$match": {"a":"restaurant"}},
        {"$group" :{"_id": "$created.user",
                "count": {"$sum":1}}},
        {"$sort":{"count":-1}},
        {"$limit":5}])
[{u'_id': u'woodpeck_fixbot', u'count': 521186},
 {u'_id': u'RVA_101', u'count': 223942},
 {u'_id': u'CynicalDooDad', u'count': 127132},
 {u'_id': u'Omnific', u'count': 103208},
 {u'_id': u'gpstrails', u'count': 49330}]
```

81% of the changes/edits were done by those 5 people.

Problems encountered in the map

While the data seemed relatively clean, the following issues or problems were noticed from the outset.

a) Abbreviated street names including Rd., St., Ct...
b) Parkway was shortened to Pkwy and Pky
c) Over abbreviated street names, N. 24th St
d) Postal codes that were not 5 digits e.g. 23233 \u200e and 23236-3103

To clean those problems, the following techniques was used

Street Names

To deal with abbreviated street names, a mapping was developed with all possible street name abbreviations. In the mapping, Pkwy and Pky were included for Parkway. As we also had an over abbreviated street names i.e. street names abbreviated in both the prefix and suffix, the code was modified to meet these requirements.

```
b=[]
i=0 # used in case a street name matches more than one mapping eg S. Addison St
for x in mapping.keys():
    if name.find(x)==(len(name)-len(x)):  # check for suffix e.g Rd.,St.
      i=i+1
      if i==1:
         b=name.replace(x,mapping.get(x))
      else:
          b=b.replace(x,mapping.get(x))

    if name.find(x)==0 and (x=="N." or x=="S."): # check for prefix e.g N.,S.
       i=i+1
       if i==1:
          b=name.replace(x,mapping.get(x))
       else:
          b=b.replace(x,mapping.get(x))
```

Postal code

As with postcodes, the only problem identified was a suffix issue, where the zip code was more than 5 digits. To resolve the issue, we had the postcode truncated i.e.

```
if tag.attrib["k"]== "addr:postcode":

   node["address"].update({tag.attrib["k"].replace("addr:",""):tag.attrib["v"][0:5]})
```

1. More stats on the data

a) There are a total of 563 restaurants with *Panera Bread* being the most common restaurant with 12 locations. Interestingly, I found out that there is one restaurant in Richmond from Ethiopia my home town.

```
result=db.richmond_virginiaC.aggregate([
        {"$match": {"amenity":"restaurant"}},
        {"$group" :{"_id": "$name",
                    "count": {"$sum":1}}},
        {"$sort":{"count":-1}}])
```

b) The most common cuisine is Chinese followed by Mexican and pizza.

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

c) McDonald's is the most common fast food chain with 33 stores

d) Christianity is the most prevalent religion with Baptist being the most common denomination.

```
result=db.richmond_virginiaC.aggregate([
        {"$match": {"amenity":"place_of_worship"}},
        {"$group" :{"_id": "$denomination",
                    "count": {"$sum":1}}},
        {"$sort":{"count":-1}}])
```

2. Idea about the data set

The data set used is mostly edited or updated by humans. This makes it prone to errors. I think for accuracy purposes, we should be able to collect and update the data set from everyday gadgets like our cell phones and wifi hotspots.

This data set, on the condition that its accuracy is maintained, can be used for driverless cars. For this purposes, the data set need to include more information including location of stop signs and zebra crosses to mention some.

For the data to be precisely accurate, I think we need to collect information from vehicles and infrastructures. Using building as beacons and measuring the location of other vehicles and infrastructures with relative the beacon and so on. This is more accurate than GPS systems whose accuracy is +- 10m.

The challenges with these suggestions are the current availability of the technologies. V2V and Vehicle to infrastructure communication is at its early stage. Moreover, as of today we have the technology to finding the location of each cell phone and reporting. As this technology uses base station than satellite signals it has a better accuracy. But for application suggested, I think we would need more accuracy and this is going to come with the new generation of technology i.e. 5G, where each cellphone could act as a base station or for this purposes a beacon.

Conclusion

The data set is relatively clean for our purposes. Throughout this project, the street names were updated and postcodes were cleaned up. Using MongoDB, we tried to collect a high level information from the data set.

Things that I found interesting from the data set:

1. While 382 people participated in updating ir editing the data, 80% percent of it was done by just 5 people
2. In the city of Richmond, there are nine prisons. I think that is a lot for just one city. But a simple googling shows that Richmond has more than average rate of crimes.
3. There are more than 400 churches but only two Jew's centers and only one Muslim centers