OpenStreetMap Project
[Data Wrangling with MongoDB]

Aishwarya Venketeswaran

Map Area: San Jose, CA, United States

Table of Contents

1. Problems Encountered in the Map

After downloading data from San Jose's dataset from Map Zen's Metro Extracts and running several tests on it, I found three main problems:

- Several problems with street names
- Postal Codes
- Multiple inaccuracies with cities

Over the next few sections, I will be discussing the issues I faced as I cleaned and audited the data.

Several problems with street names

Over-abbreviated Street Names

As I examined the dataset by running some queries, I found that several street names were over abbreviated. I am providing two examples to demonstrate the changes:
Example:
S. Bascom => South Bascom [In this case, only the prefix is abbreviated]
Blake Ave => Blake Avenue [In this case, only the suffix is abbreviated]
S De Anza Blvd. => South De Anza Boulevard [In this case, both the prefix and the suffix are abbreviated]

Extended Addresses in Street Name

In some cases, although the field was supposed to have only the street name, the whole address seemed to be there. In these case, my code cleaned the data so that only the street's name is present [i.e information till the first ","].
Example: Zanker Rd,San Jose, CA => Zanker Road

Spanish Street Names and Special Prefixes

As I was analyzing the unexpected street names, I realized that some of the street names were not really wrong and they were just unexpected. One major change in code was to recognize that street names in the Spanish language seem to have Spanish words for road/street as a prefix. Hence, in my code, I added another set of expected prefixes (including directions as well as Spanish words for road/street/walk,etc.) and checked street names for these prefixes.

```
expectedPrefixStreetTypes = ["Camina", "Calle",
"Paseo","Rio","West","South","North","East"]
```

Examples of street names that were initially shown in unexpected/wrong street names:
Calle De Luna, Rio Robles , Paseo Presada,etc.

Postal Codes

Zip codes were prefixed with “CA”

I had to fix several zip codes in which the state - CA was included. There were two main cases:

1. CA “a 5 digit code” : In this case, my code audits and cleans the zip code to just the 5 digit code.
   Example from dataset: CA 94035 => 94035
2. CA “a 5 digit code - a 4 digit code”: In this case, as expected, my code cleans the zip code to the 9 digit zip code [with the dash in between].
   Example from dataset: CA 94088-3453 => 94088-3453

Multiple inaccuracies with cities

Inclusion of counties not in Santa Clara

As I informally skimmed through the dataset, I saw one or two cities listed which were not really acceptable. Hence, I decided to programmatically search for cities that were actually not in the Santa Clara county. After doing running my program, I found these cities [that are not acceptable]:

```
{'1060': set(['1060']),
 'Alviso': set(['Alviso']),
 'Campbelll': set(['Campbelll']),
 'Felton': set(['Felton']),
 'Los Gato': set(['Los Gato']),
 'Moffett Field': set(['Moffett Field']),
 'Mt Hamilton': set(['Mt Hamilton']),
 'Redwood Estates': set(['Redwood Estates']),
 'Saj Jose': set(['Saj Jose']),
 u'San Jos\xe9': set([u'San Jos\xe9'])}
```

Incorrectly spelled cities

After observing the results after correcting the previous problem (Inclusion of counties not in Santa Clara), it became obvious that several of these entries contain city names which are actually part of San Jose(Santa Clara county) but are incorrectly spelled.This is what motivated me to make a mini spell checker [very basic]. My spell checking function compares the given city name to expected city names and computes a similarity index. If the similarity index is pretty high, the program guesses that the user might have spelled the city’s name wrong and replaces the incorrect name with the actual name (guess). After running the new code, this is the result:
San José => San Jose [0.875], Campbelll => Campbell [0.889], Los Gato => Los Gatos [0.889], Saj Jose => San Jose [0.875]
The calculated similarity index is recorded inside brackets.

2. Data Overview

In this section, I have some basic information regarding the dataset.

File sizes

```
san-jose_california.osm .... 163 MB
test.json .................. 271 MB
```

Number of documents

```
> db.city.find().count()
849314
```

Number of nodes

```
> db.city.find({ "type": { "$in":['node']} }).count()
756698
```

Number of ways

```
> db.city.find({ "type": { "$in":['way']} }).count()
92616
```

Sample document

```
> db.city.find_one()
{u'created': {u'changeset': u'11686320', u'version': u'10', u'uid':
u'14293', u'timestamp': u'2012-05-24T03:24:59Z', u'user':
u'KindredCoda'}, u'pos': [37.1582245, -121.6574737], u'visible': None,
u'_id': ObjectId('5574ddc80150061cc054d1d8'), u'type': u'node', u'id':
u'25457954'}
```

Number of unique users

```
> len(db.final.distinct("created.user"))
937
```

Top contributors

```
> db.final.aggregate([
{"$group":{"_id":"$created.user", "count":{"$sum":1}}},
            {"$sort": {"count":-1}},
            {"$limit":5}])
{u'count': 174826, u'_id': u'nmixter'}
{u'count': 168993, u'_id': u'mk408'}
{u'count': 83150, u'_id': u'Bike Mapper'}
{u'count': 60109, u'_id': u'n76'}
{u'count': 29489, u'_id': u'matthieun'}
```

Number of users who made only one post

```
> db.final.aggregate([
            {"$group":{"_id":"$created.user", "count":{"$sum":1}}},
            {"$group":{"_id":"$count", "num_users":{"$sum":1}}},
            {"$sort":{"_id":1}},
            {"$limit":1}])
{u'num_users': 185, u'_id': 1}
```

#Number of cafes in the area

```
> db.final.aggregate([
      {"$match":{"amenity":{'$exists':1},"amenity":"cafe"}},
       {"$group":{"_id":"cafe","count":{"$sum":1}}},
       {"$limit" : 1}])
{u'count': 193, u'_id': u'cafe'}
```

#Number of restaurants in the area

```
> db.final.aggregate([
       {"$match":{"amenity":{'$exists':1},"amenity":"restaurant"}},
       {"$group":{"_id":"restaurant","count":{"$sum":1}}},
       {"$limit" : 1}])
{u'count': 661, u'_id': u'restaurant'}
```

3. Additional Ideas

Additional Explorations of dataset

#Top religions practiced in this area

```
> db.final.aggregate([{"$match":{"amenity":{"$ne":None},"amenity":
"place_of_worship","religion":{ "$ne":None}}},
{"$group":{"_id":"$religion","count":{"$sum":1}}},
{"$sort" : {"count":-1}},
{"$limit" : 6}])
{u'count': 300, u'_id': u'christian'}
{u'count': 5, u'_id': u'jewish'}
{u'count': 3, u'_id': u'buddhist'}
{u'count': 2, u'_id': u'hindu'}
{u'count': 2, u'_id': u'muslim'}
{u'count': 2, u'_id': u'unitarian_universalist'}
```

#Most popular cuisines of this area

```
> db.final.aggregate([{"$match":{"amenity":{"$exists":1},"amenity":
"restaurant" ,"cuisine":{ "$ne":None}}},
{"$group":{"_id":"$cuisine","count":{"$sum":1}}},
{"$sort" : {"count":-1}},
{"$limit" : 6}])
{u'count': 51, u'_id': u'chinese'}
{u'count': 48, u'_id': u'mexican'}
{u'count': 37, u'_id': u'pizza'}
{u'count': 28, u'_id': u'american'}
{u'count': 28, u'_id': u'italian'}
{u'count': 26, u'_id': u'indian'}
```

#Total number of bus stations in this area

```
> db.final.aggregate(
[{"$match":{"amenity":{'$exists':1},"amenity":"bus_station"}},
            {"$group":{"_id":"bus_station","count":{"$sum":1}}},
            {"$limit" : 1}])
{u'count': 7, u'_id': u'bus_station'}
```

#Bank with the highest number of ATMs

```
> db.final.aggregate([{"$match":{"amenity":{'$exists':1},"amenity":
"atm","operator":{ "$ne":None}}},
{"$group":{"_id":"$operator","count":{"$sum":1}}},
{"$sort" : {"count":-1}},
{"$limit" : 1}])
{u'count': 9, u'_id': u'Wells Fargo'}
```

#Pharmacy with the highest number of outlets

```
> db.final.aggregate([{"$match":{"amenity":{'$exists':1},"amenity":
"pharmacy", "operator":{ "$ne":None}}},
{"$group":{"_id":"$operator","count":{"$sum":1}}},
{"$sort" : {"count":-1}},
{"$limit" : 1}])
{u'count': 4, u'_id': u'Walgreens'}
```

Conclusion

After considering the dataset of San Jose, I realize that I have done considerable amount of cleaning and auditing of the data [by recognizing the scope of this project]. However, I would not claim that the dataset is completely accurate and free of errors.

As I was working in this dataset, I realized the huge number of human errors that can be introduced into a dataset. Simple things (from the perspective of a non-technical person) such spelling mistakes can lead to great inaccuracies. Although I was happy to realize that several users are contributing to this project of building maps, I think a higher number of contributors will translate into higher number of errors. Therefore, my suggestion is that data can be entered into the dataset by using a computer program. Data about a city can be obtained by parsing web pages. I recognize that implementing such a program efficiently would be a hard task. However, programmatically obtaining data could reduce the errors in the dataset.

Improving the dataset will definitely help analysts in making better analysis about the data. Having data stored more uniformly will definitely be helpful. As I was analyzing the database and trying to make conclusion about city, one question I wanted to find an answer to was - “How many bicycles can be parked in San Jose? or How many bicycle parking spots are there in San Jose?”. As I gave some initial commands to answer this question, I found that although several parking spaces were recorded, the number of spots in that parking lot (capacity) was not recorded for all of them. Hence, answering this question accurately would be impossible.

References

https://www.sccgov.org/sites/sccphd/en-us/Partners/Data/Documents/popu%20by%20cities%202010.pdf
https://mapzen.com/data/metro-extracts
http://stackoverflow.com/ [Many discussions]
https://docs.python.org/3/