Data_Wrangling

January 26, 2018

1 Project 3: OpenStreetMap Data Case Study

1.1 Map Area

San Jose CA, United States

https://www.openstreetmap.org/relation/112143

https://mapzen.com/data/metro-extracts/metro/san-jose_california/

San Jose is the first city that I visited in USA and I have a wonderful memories with my friends, so I'm interested to explore the area virtually and be ready for the next visit.

1.2 Problems Encountered in the Map

After downloading and decompressing the XML file of San Jose area and running it against a provisional data.py file, I noticed the following main problems with the dataset: 1- Abbreviated and non-standardized street types ('St.', 'St', 'ST', 'STREET', 'street','Ave.', 'Ave', 'AVENUE', 'avenue','Pkwy', etc). 2- Inconsistent postal codes ('CA 94085', '95134-1358')

1.2.1 1- Abbreviated street names

After auditing the data set, I attempted to clean the data and uniform it by replacing the abbreviated street types 'St.', 'St', 'ST', 'STREET', 'street' with "Street", and 'Ave.', 'Ave', 'AVENUE', 'avenue' with "Avenue" and so on. I created a list of expected street types that match the standards. The function "audit_street_type" collects the last words in the "street_name" strings, and stored in the dictionary if they are not within the expected list. The function "street_types" return me the no uniform and abbreviated street types being used. The "is_street_name" function looks for tags that specify street names (k="addr:street"). The "audit" function returns a dictionary that match the above function conditions. The update_name function takes an old name to mapping dictionary, and update to a new one.

```
In [ ]: def update_name(name, mapping):
            """takes an old name to mapping dictionary, and update to a new one"""
            m = street_type_re.search(name)
            if m not in expected:
                if m.group() in mapping.keys():
                    name = re.sub(m.group(), mapping[m.group()], name)

            return name
```

This updated all substrings in problematic address strings, such that: “Cabrillo Ave” becomes: “Cabrillo Avenue”

1.2.2 2-Abbreviated street names

The postal codes are showing on the data set in different ways: five-digits, nine-digits, or start with the State code. I decided to uniform the zip codes to five main digit, so I dropped leading and trailing characters before and after the main 5 digit post code.

```
In [ ]: def update_postcode(postcode):
            if re.findall(r'^\d{5}$', postcode): # 5 digits 02118
                valid_postcode = postcode
                return valid_postcode
            elif re.findall(r'(^\d{5})-\d{4}$', postcode): # 9 digits 02118-0239
                valid_postcode = re.findall(r'(^\d{5})-\d{4}$', postcode)[0]
                return valid_postcode
            elif re.findall(r'MA\s*\d{5}', postcode): # with state code MA 02118
                valid_postcode =re.findall(r'\d{5}', postcode)[0]
                return valid_postcode
            else:
                return None
```

1.3 SQL Database Exploration

To prepare and populate the SQL database, I parsed the XML file and changed the shape and the format to a tabular format by generating CSV files and I developed a python script (DB_Pop.py) to populate the database.

1.3.1 The files size:

```
In [4]: import os
        print('The san-jose_california.osm file is {} MB'.format(os.path.getsize('san-jose_cali
        print('The sanjose.db file is {} MB'.format(os.path.getsize('sanjose.db')/1.0e6))
        print('The nodes.csv file is {} MB'.format(os.path.getsize('nodes.csv')/1.0e6))
        print('The nodes_tags.csv file is {} MB'.format(os.path.getsize('nodes.csv')/1.0e6))
        print('The ways.csv file is {} MB'.format(os.path.getsize('ways.csv')/1.0e6))
        print('The ways_tags.csv is {} MB'.format(os.path.getsize('ways_tags.csv')/1.0e6))
        print('The ways_nodes.csv is {} MB'.format(os.path.getsize('ways_nodes.csv')/1.0e6))
```

```
The san-jose_california.osm file is 411.442361 MB
The sanjose.db file is 227.483648 MB
The nodes.csv file is 161.478084 MB
The nodes_tags.csv file is 161.478084 MB
The ways.csv file is 15.236259 MB
The ways_tags.csv is 24.230945 MB
The ways_nodes.csv is 54.906156 MB
```

1.3.2 Number of nodes

```
In [ ]: SELECT COUNT(*) FROM nodes;
```

1891418

1.3.3 Number of ways

```
In [ ]: SELECT COUNT(*) FROM ways;
```

249292

1.3.4 Number of unique users

```
In [ ]: SELECT COUNT(DISTINCT(e.uid))
        FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) e;
```

1476

1.3.5 Top 10 contributing users

```
In [ ]: SELECT e.user, COUNT(*) as num
        FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) e
        GROUP BY e.user
        ORDER BY num DESC
        LIMIT 10;
```

andygol 295703 nmixter 281980 mk408 134683 Bike Mapper 94905 samely 80730 3vivekb_sjsidewalks_import 77994 RichRico 75583 dannykath 73911 MustangBuyer 64652 karitotp 62074

1.3.6 Number of Taco

```
In [ ]: SELECT COUNT(*) FROM nodes_tags WHERE value LIKE '%taco%';
```

28

1.4 Additional Database Exploration

1.4.1 Count Tourism Related Categories

```
In [ ]: SELECT tags.value, COUNT(*) as count FROM (SELECT * FROM nodes_tags UNION ALL SELECT *
        GROUP BY tags.value
        ORDER BY count DESC;
```

picnic_site 207 hotel 111 information 77 motel 66 attraction 39 artwork 32 museum 31 viewpoint 20 gallery 7 guest_house 5 camp_site 4 theme_park 4 caravan_site 2 zoo 2 bus_stop 1 construction 1 scenic_view 1

1.4.2 Top 10 appearing amenities

```
In [ ]: SELECT value, COUNT(*) as num
        FROM nodes_tags
        WHERE key='amenity'
        GROUP BY value
        ORDER BY num DESC
        LIMIT 10;
```

restaurant 898 fast_food 429 bench 319 cafe 264 bicycle_parking 208 place_of_worship 170 toilets 163 school 138 fuel 130 bank 128

1.4.3 Top 10 popular cuisines

```
In [ ]: SELECT nodes_tags.value, COUNT(*) as num
        FROM nodes_tags
            JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='restaurant') i
            ON nodes_tags.id=i.id
        WHERE nodes_tags.key='cuisine'
        GROUP BY nodes_tags.value
        ORDER BY num DESC
        LIMIT 10;
```

vietnamese 81 chinese 69 mexican 65 pizza 58 japanese 44 indian 35 italian 31 american 28 thai 28 sushi 23

1.4.4 Top 10 Fast Food Chain cuisines

```
In [ ]: SELECT nodes_tags.value, COUNT(*) as num FROM nodes_tags JOIN (SELECT DISTINCT(id)
        FROM nodes_tags WHERE value='fast_food') i ON nodes_tags.id=i.id WHERE nodes_tags.key=
         GROUP BY nodes_tags.value ORDER BY num DESC LIMIT 10;
```

Subway 41 Jamba Juice 13 Panda Express 13 Taco Bell 12 Chipotle 11 McDonald's 9 Togo's 9 KFC 8 Burger King 7 Baskin-Robbins 5

1.4.5 Top 10 cities

```
In [ ]: SELECT tags.value, COUNT(*) as count
        FROM (SELECT * FROM nodes_tags UNION ALL
              SELECT * FROM ways_tags) tags
        WHERE tags.key LIKE '%city'
        GROUP BY tags.value
        ORDER BY count DESC
        LIMIT 10;
```

Sunnyvale 3437 San Jose 1068 Morgan Hill 404 Santa Clara 340 Saratoga 234 San José 176 Los Gatos 146 Milpitas 113 Campbell 93 1 85

1.5 Additional Ideas

It is obvious that the San Jose area data quality is poor, many incomplete street names or postal codes, and the data is not uniform. For example from the query below we can notice that we have San Jose and San José for the same city which lead to a misleading statistics also we can see the name of city as a number and we can't do anything about it for now.

I suggest that the data entry should be more restricted and following a standard, also having an instantaneous data validator for volunteers will help to improve the quality of the open street map project.

1.6 Benefits and Anticipated Problems in Implementing the Improvement

1.6.1 Benefits

To get more benefits from the OpenStreetMap, it will be nice to add reviews from public, and add place to visit suggestions for tourist and visitors. In this way local business will be involved and pushed to add more detailed and accurate to OpenStreetMap project to look for potential customers.

1.6.2 Anticipated problems

OpenStreetMap is an open source project, so there is no paid employees on it which explain the low quality of data, so users may not relies on it, and there is alternatives that users may use to look for address and amenities such as google map, so that may not encourage the business to participate and fund it.