osm_project_final

October 26, 2016

1 Map Area: Portland, OR United States

- https://mapzen.com/data/metro-extracts/metro/portland_oregon/

The area I chose to explore is Portland, Oregon in United States. Portland, Oregon's largest city, sits on the Columbia and Willamette rivers, in the shadow of snow-capped Mount Hood. It's known for its parks, bridges and bicycle paths, as well as for its eco-friendliness and its microbreweries and coffeehouses.

The reason why I chose Portland to explore is because I've visited once before and would like to live there one day. It is a lovely city with unique neighborhoods and this project presented an opportunity to explore it further.

2 Problems Encountered: Auditing & Cleaning process

To familiarize myself with the dataset, I first took a sample of the XML data and explored its main components: nodes, ways, and tags. As OSM is an open-source, volunteer-created project, it is especially prone to having "dirty" data. Before proceeding with data analysis, it'd be helpful to iteratively audit and clean the data.

2.1 Better street types

When traveling in Portland, I remember noticing unique street names. Thinking that there may be more incorrectly recorded street types than other cities, I decided to audit the street types. After writing validation rules and programatically checking the data, I encountered the following main problems with street types:

- over-abbreviated street type (e.g. Rd)
- lowercase street type (e.g. street)

To standardize the street names, I corrected the above street types using the following update_name function and mapping variable which specifies what the new name should be.

```
def update_name(name, mapping):
    #initialize m and search through argument name using regex
    m = street_type_re.search(name)
    if m:
        street_type = m.group()
```

```
        #if street_type is in mapping, then substituate the value
        if street_type in mapping.keys():
            name = re.sub(street_type, mapping[street_type], name)
    return name
```

2.2 Better zipcodes

After testing and confirming the street names were corrected, I took a look at zipcodes. In 1983, the U.S. Postal Service introduced an expanded ZIP Code system that it called ZIP+4. Though it is not widely used, some zipcodes continue to use these 4-digit zipcode extensions. When I explored the zipcodes, as expected, there were many 5-digit postal codes with 4-digit extensions (e.g. 91350 - 4923).

To allow for more consistent queries, I used the following audit_zipcode function to drop the trailing characters after the 5-digit zipcode.

```
def audit_zipcode(postcode):
    if "-" in postcode:
        return postcode [:5]
    else:
        return postcode
```

After the street types and postal codes were cleaned, data was prepared to be inserted into a SQL database. The elements in OSM XML file were parsed and transformed to tabular format, making it possible to write to .csv files. These .csv files were then imported to a SQL database as tables and readied for analysis!

3 Overview Statistics of Portland OSM

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

Size of the file

```
portland.osm ......... 1.5 GB
osm.db .......... 896 MB
nodes.csv ............. 583 MB
nodes_tags.csv ........ 9.7 MB
ways.csv .............. 56 MB
ways_tags.csv ......... 148 MB
ways_nodes.csv ......... 170 MB
```

Number of nodes and ways

```
sqlite> SELECT COUNT(*) FROM nodes
6488804

sqlite> SELECT COUNT(*) FROM ways
834262
```

Number of unique users

```
sqlite> SELECT COUNT(DISTINCT(u.uid))
FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) u

1175
```

Top 5 contributing users

```
sqlite> SELECT u.user, COUNT(*) as num
FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) u
GROUP BY u.user ORDER BY num DESC LIMIT 5;

Peter Dobratz_pdxbuildings|1965703
lyzidiamond_imports|1911157
Mele Sax-Barnett|590113
baradam|539851
Darrell_pdxbuildings|435863
```

4 Additional Statistics & Data Exploration

Total Number of Contributions & User Contribution Statistics

```
sqlite> SELECT COUNT(*) as num
FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) u
```

- Total Number of Contibutions: 7323066
- Top User (Peter Dobratz_pdxbuildings) Contribution %: 26.8
- Top 5 Users Contribution %: 74.3
- Top 10 Users Contribuation %: 88.4

Above statistics suggest that for Portland, majority (nearly 75%) of OSM data is contributed by top 5 users. Top contributing user (Peter Dobratz) has contributed more than a quarter of the available data. This skewed distribution may be due to these users building and using a bot that is automatically editing the map. Another possibility is that a group or an organization is sponsoring or working on the map together (both rank #1 and #5 user has _pdxbuildings following their usernames).

4.1 Data Exploration

Top 5 amenities

```
sqlite> SELECT value, COUNT(*) as num
FROM nodes_tags
WHERE key = "amenity"
GROUP BY value
ORDER BY num DESC
LIMIT 5;

bicycle_parking|2453
```

```
place_of_worship|880
bench|830
waste_basket|756
restaurant|657
```

Total capacity for bicycles

```
sqlite> SELECT COUNT(*) as num
FROM nodes_tags
    JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='bicycle_parking') u
    ON nodes_tags.id=u.id
WHERE nodes_tags.key='capacity';

2230
```

Top 5 religion

```
sqlite> SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
    JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='place_of_worship') u
    ON nodes_tags.id=u.id
WHERE nodes_tags.key='religion'
GROUP BY nodes_tags.value
ORDER BY num DESC
LIMIT 5;

christian|838
buddhist|7
jewish|3
muslim|3
unitarian|2
```

Most popular cuisine

```
sqlite> SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
    JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='restaurant') u
    ON nodes_tags.id=u.id
WHERE nodes_tags.key='cuisine'
GROUP BY nodes_tags.value;

pizza|65
```

5 Additional Ideas

As the metro extract is inclusive of the surrounding areas of Portland, one way we can improve the data is filtering out these areas before analysis. One benefit would be access to a more accurate depiction of Portland. For example, for Portland specific questions such as "how many coffeeshops are in Portland?" we won't be reporting inflated numbers.

To implement the improvement, the challenge would be defining where Portland's boundary actually lies. We can use the zipcodes to specify the areas that fall within city of Portland and remove the rest from the dataset. Zipcodes in Portland start with digits '972', so it would be less complex to isolate than other areas.

6 Conclusion

Although Portland's OSM data is in a good quality, we can make additional edits to improve the map. One way is to use a more powerful bot that can programmatically edit and help standardize the map.

As suggested earlier, the top contributing users may already be bot developers (given the skewed contributor distribution). If these users can share best practices to edit or utilize the same bot, this would also standardize the editing practices, resulting in a better OSM of Portland for everyone.

List of resources referred to: - https://www.google.co.uk/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=portland%20city - https://en.wikipedia.org/wiki/ZIP_Code#ZIP.2B4 - https://gist.github.com/carlward/54ec1c91b62a5f911c42#file-sample_project-md - https://gist.github.com/swwelch/f1144229848b407e0a5d13fcb7fbbd6f - https://discussions.udacity.com/t/creating-db-file-from-csv-files-with-non-ascii-unicode-characters/174958 - https://discussions.udacity.com/t/preparing-for-database-quiz-testing-env/173288

In [ ]: