# Data Wrangling with MongoDB


## Thuy Quach


Map area: Seattle, WA, United States

http://www.openstreetmap.org/relation/237385 (http://www.openstreetmap.org/relation/237385)

https://mapzen.com/data/metro-extracts/#seattle (https://mapzen.com/data/metro-extracts/#seattle)


Abstract:

Open Street Map is a collaborative project to create a free editable map of the world. It is a great source of data. However as any open source data, it contains many inconsistent or incorrect information such as street types, postcodes and city names. The tasks of the project was to audit, fixing and processing the dataset of Seattle. MongoDB queries were used to obtain the overview of the data is and also other investigations.


## 1. Parse the download oms file and take a sample part of the map

The size of original seattle.osm was 1.54GB. To make it run with my PC, I parsed the file and took a sample of 1/20th of it.

## 2. Problems encountered in the map

#### Check problematic characters

Before process the data and add it into MongoDB, we should check the "k" value for each "tag" and see if they can be valid keys in MongoDB, as well as see if there are any other potential problems.

```
{'lower': 102074, 'lower_colon': 114390, 'other': 3692, 'problemch
ars': 0}
```

- There are no tags with problematic characters.
- There are 102074 tags that contain only lowercase letters and are valid.
- There are 114390 for otherwise valid tags with a colon in their names.
- There are 3692 tags that do not fall into the other three categories.

**Problems with street types**

We need to audit the OSMFILE to see the street types are appropriate ones.

```
Here is the list of street types:
['Glen', 'Fir', 'Northeast', 'Ridge', 'West', 'St.', 'Heights', 'R
d', 'Speedway', 'street', 'Blackburn', 'Northwest', 'Way', '1st',
'Gate', 'Circle', 'East', 'avenue', 'Meadows', 'Highway', 'Southwe
st', '104', 'Cleveland', 'North', 'west', 'Rise', 'Reach', 'NE',
'Meridian', 'Southeast', '18th', 'Loop', 'Hwy', '9', 'Snoqualmie',
'25th', 'Section', 'NW', 'Laventure', 'E', 'Center', 'Sandalwood',
'36th', 'Plaza', 'Alley', 'St', '99', 'Division', 'S', 'Green',
'W', 'Close', '20', 'N', 'Estates', 'Gardens', 'South', 'Point',
'S.E.', 'Terrace', 'SW', 'Esplanade', 'r', 'Crescent', 'Waugh', 'B
roadway', 'southwest', 'SE']
```

We could see that the street types were over-abbreviated such as "S.E", "N", "S", etc. We need to fix the unexpected street names to the appropriate ones by using mapping dictionary and then updated the street name. We will later use it to update the street names in json data.

**Problems with postcode**

Postcode is also important data. All the postcodes in Seattle start with 981. Let's check to see all the postcode are correct.

```
128
```

- There are 128 unmatched postcode.

```
Some of ummacthed postcode:
['98070', 'V8X 4V1', '98004', '98005', '98273', '98503', '98011',
'98057', '98002', '98052']
```

- Quick checking some of the postcodes are in other cities next to Seattle such as Kirkland and Bellevue. It would be interesting to find out later how many of the data are from those cities using MongoDB.

#### Problem with city name

From the postcode data, there was some data in different cities than Seattle in the dataset. I will later check with MongoDB queries to see more details.

## 3. Prepare the data set for MongoDB

#### Process the oms to json

The oms file was parsed. The output data was shaped by its elements and then wrote on json file. The json file then was imported to MongoDB for analyzing.

#### Auditting data with MongoDB

##### *Check the postcode*

MongoDB queries were used to analyze top postcodes.

```
Top 10 postcode:
[{u'count': 1147, u'_id': u'98034'}, {u'count': 968, u'_id': u'980
33'}, {u'count': 905, u'_id': u'98115'}, {u'count': 838, u'_id':
u'98103'}, {u'count': 716, u'_id': u'98118'}, {u'count': 692, u'_i
d': u'98117'}, {u'count': 610, u'_id': u'98125'}, {u'count': 492,
u'_id': u'98105'}, {u'count': 470, u'_id': u'98108'}, {u'count': 4
63, u'_id': u'98144'}]
```

Out of top 10 postcodes, two are in Kirkland (start with 980) and the rest are in Seattle (start with 981).

```
Total data with postcode is 13219 out of 378925 data.
```

```
Total data with Seattle postcode is 10162
```

There are many data don't have postcode. Among them, most are Seattle zipcode. Therefore we can remove the data that has postcode unmatched '981'.

### *Check the city name*

MongoDB queries were used to analyze all city names.

```
Top 10 cities: [{u'count': 10143, u'_id': u'Seattle'}, {u'count':
2113, u'_id': u'Kirkland'}, {u'count': 606, u'_id': u'Saanich'},
{u'count': 559, u'_id': u'Mount Vernon'}, {u'count': 138, u'_id':
u'Langford'}, {u'count': 115, u'_id': u'Oak Bay'}, {u'count': 92,
u'_id': u'Colwood'}, {u'count': 79, u'_id': u'Esquimalt'}, {u'coun
t': 78, u'_id': u'Sooke'}, {u'count': 49, u'_id': u'Metchosin'}]

Number of cities: 82
```

Some cites are not from Seattle such as Kirkland, Saanich, Langford, Oak Bay etc. Kirkland is a neighbor city of Seattle. Saanich and Oak Bay are two cities in Victoria, Canada. They are both nearby cities to Seattle. There are total of 81 cities are not Seattle.

### Data cleaning

So, after running the analysis, we need to clean the street types, postcode and city names.

### *Update street name*

```
Total data after updating street types: 378925
```

### *Drop data with postcode unmatch '981'*

```
Total data after updating street types and postcode: 375868
```

### *Drop city name unmatched 'Seattle'*

```
Total data after updating street types, postcode and city names :
374540
```

### Write clean data to json

**Load the data into Mongo DB**

## 4. Data Overview

This section contains basic statistics about the dataset. MongoDB queries are used to get the data.

```
File name : clean_seattle3.json
File size: 91MB

Number of documents: 374540

Number of nodes: 343065

Number of ways: 94413

Number of unique users: 1447

Top 1st user: {u'count': 61392, u'_id': u'Glassman'}

Top 10th user:
[{u'count': 61392, u'_id': u'Glassman'}, {u'count': 37383, u'_id':
u'SeattleImport'}, {u'count': 33100, u'_id': u'tylerritchie'},
{u'count': 31165, u'_id': u'woodpeck_fixbot'}, {u'count': 16124,
u'_id': u'alester'}, {u'count': 11278, u'_id': u'STBrenden'}, {u'c
ount': 10735, u'_id': u'Glassman_Import'}, {u'count': 8995, u'_i
d': u'Brad Meteor'}, {u'count': 8441, u'_id': u'Amoebabadass'},
{u'count': 6041, u'_id': u'zephyr'}]
```

## 5. Additional analysis

**Improving OMS data quality suggestions**

***Address with over-abbreviated and inconsistent street types***

The street types data could be improved by let the contributing users enter an auto-corrected street types. Each countries and/or cities may have different street types. If possible, the street type data of interested area could be collected via cross-reference sources such as local transit goverment, postmail company or Google Maps. In case there is no standard existing street types, the local open street map community could make one data processor similar to my auditing and updating street types functions.

***Unmatched postcode and city name***

Similar to street types data, postcode data could be much improved by having an standard postcode. If the new data's postcode does not in the appropriate list, it should be warned. Same with city name, it should be same with city of interest. A data processor similar to my auditing and fixing postcode and city name could work well.

***Encourage user parcipation and collaboration***

From the user data, there are 1447 users in which top 10 user contribution is 60%. In top 10 user there is only one name appears with the word 'bot'. It looks like it could have a great chance of collaboration to make and edit beter map. Certain gamification elements, such as badges or leaderboard would strive to leverage user's desires to create better data and community collaboration.

**Additional data exploration using MongoDB queries**

***Top 10 amenities***

```
[{u'_id': u'parking', u'count': 383},
 {u'_id': u'bicycle_parking', u'count': 149},
 {u'_id': u'school', u'count': 123},
 {u'_id': u'restaurant', u'count': 117},
 {u'_id': u'bench', u'count': 94},
 {u'_id': u'place_of_worship', u'count': 69},
 {u'_id': u'cafe', u'count': 53},
 {u'_id': u'waste_basket', u'count': 48},
 {u'_id': u'fuel', u'count': 43},
 {u'_id': u'fast_food', u'count': 43}]
```

We could see that 'parking' is the top amenity in Seattle. It is not a surprise given Seattle is a home city of many companies. 'Bicycle_parking', 'bench' and 'waste_basket' are also in the top 10 amenity. There was also a lot of cafe. It could be Seattle is home of Starbucks.

***Top 10 cuisine***

```
[{u'_id': None, u'count': 49},
 {u'_id': u'pizza', u'count': 9},
 {u'_id': u'american', u'count': 7},
 {u'_id': u'mexican', u'count': 6},
 {u'_id': u'chinese', u'count': 5},
 {u'_id': u'thai', u'count': 4},
 {u'_id': u'sandwich', u'count': 4},
 {u'_id': u'italian', u'count': 4},
 {u'_id': u'japanese', u'count': 3},
 {u'_id': u'regional', u'count': 2}]
```

The top 1st restaurant had 'None' name. It should be included since it accounts to about half of the total cuisines. Seattle seems to have a good variety of cuisines from 'pizza' to 'mexcian', 'italian' and other asian ones.

## 6. Conclusion

Seattle open street map dataset is a good data source for data wrangling purpose. It contains inconsistant street types; postcodes that are not in Seattle; and cities are in nearby cities of Seattle or even in Canada. They are all cleaned. Over view of the cleaned data was reported. Some interesting data such as top amenities and cuisine were analyzed.

To improve the OMS data quality, I suggested to have a cross-reference with other data sources such as local goverment data, post mail company or Google Maps to ensure the correctness and consistance of the OMS data. If those cross-reference data are not available, the OMS community could collaborate and come up with their own standard data and clean it as my way in this project.