Project 2: Data Wrangling with Mongo DB – OpenStreetMap Project

Map Area: Kaohsiung, Taiwan
Data: https://mapzen.com/metro-extracts
Kaohsiung -- https://s3.amazonaws.com/metro-extracts.mapzen.com/kaohsiung_taiwan.osm.bz2
Test Data:
KaoHsiung -- https://www.openstreetmap.org/export#map=14/22.6203/120.3120
Position Coordinates:
Kaohsiung -- North – South: .22.6302 – 22.6104; West – East: 120.2984 – 120.3255

1. Project Summary

a. What is your name?
   Wanda Chen

b. What E-mail address do you use to sign in to Udacity?
   wanda.chen@gmail.com

c. What area of the world you used for your project? Post a link to the map position and write a short description. Note that the osm file of the map should be at least 50MB.
   i. URL: https://www.openstreetmap.org/export#map=14/22.6203/120.3120
   ii. The data file I selected is from Map Zen – kaohsiung_taiwan.osm (the uncompressed size is 59.9MB)
   iii. I chose this particular place because it is the largest city in the southern part of Taiwan and it is about 1.5 hours drive from where I grew up. Although I lived in a smaller city which is not available in Map Zen, I do visit Kaohsiung whenever I can when I go back home.

d. Is there a list of Web sites, books, forums, blog posts, github repositories etc that you referred to or used in this submission (Add N/A if you did not use such resources)?

   i. Use this place to list the citations.
      1) Website to unzip file: http://www.7-zip.org/download.html
      2) mongoimport - http://docs.mongodb.org/manual/reference/program/mongoimport/
      3) mongo Shell Quick Reference - http://docs.mongodb.org/manual/reference/mongo-shell/
      4) Map features - http://wiki.openstreetmap.org/wiki/Map_Features
      5) Key:cuisine - http://wiki.openstreetmap.org/wiki/Key:cuisine
      6) Romanization of Chinese - http://en.wikipedia.org/wiki/Romanization_of_Chinese

   ii. Please carefully read the following statement and include it in your email:
   iii. *"I hereby confirm that this submission is my work. I have cited above the origins of* ***any*** *parts of the submission that were taken from Websites, books, forums, blog posts, github repositories, etc. By including this in my email, I understand that I will be expected to explain my work in a video call with a Udacity coach before I can receive my verified certificate."*

e. Is there any other important information that you would want your project evaluator to know?

   i. Basic setup rules for cleaning data file: See the following table (2-a-ii) for the method I used in cleaning the "address:street" name field. I left everything else (city, shop-name, etc.) alone. Also, from my understanding, in Taiwan, the address in English is separated by comma (","). In the data file, some addresses have "," separator, and some do not. To be uniform, I changed "," to space instead due to the way that addresses translated into English is not always number first.
   ii. Originally I selected "Taipei_taiwan.osm" and the uncompressed size is about 187MB. But when I tried to dump the file to "Taipei_taiwan.osm.json" file, I kept getting "Memory Error". I tried *json.dumps()* but got a memory error; I tried *json.dump()* which ran overnight but I still got a memory error or it (seemed like it) ran forever. So I changed my project file to "Kaohsiung_taiwan.osm", which is a smaller file. So, in the next section about address problems, it may not occur on "Kaohsiung_taiwan.osm" or "Kaohsiung_map.osm" (sample test file). It was selected when I test the "Taipei" file. Also, *audit.py* was modified according to the "Taipei" file.

2. Project Rubric Questions

a. Problems encountered in your map

   i. Unfamiliar Download File Type - After I downloaded the xml osm file (kaohsiung_taiwan.osm.bz2) from Map Zen, I realized it was a zip file that with the extension (.bz2) that I am not familiar with. After I did some research and consulted with someone with lots of programming experience, I used an open source program called 7-Zip to extract the data and get the data file that is just plain osm file.
   ii. Inconsistent Street/Road Names – In the dataset, I found some street name/address entered in many different formats:

#	Problem	Audit/Cleaning Solution
1	Some address coded with Chinese only - *博愛二路*	Leave it alone
2	Some address code with both Chinese and English -*九如四路 (Jiouru 4th Road)*	Chinese portion – leave it alone; English portion – expand abbreviation
3	Some address coded with only English - 3F, 293-1, Sec. 1, Fuxing S. Rd	Expand abbreviation only; leave the Floor with expansion
4	Some address coded with English, but with accent (special characters) or lower case only - *6, yùzhú 1st st, xinxing district*	Expand abbreviation only
5	Some address coded with English, but without accent (special characters) - *Dingjinghou Rd.*	Expand abbreviation only
6	Some address has ['] to indicate separation of Chinese characters - Alley 10, Long'an Ln.;	Expand abbreviation; Remove special character [']
7	Some address do not have ['] to indicate separation of Chinese characters -*九如四路 (Jiouru 4th Road)*	Leave it alone

8	Some street name in second Chinese character are lower case - *九如四路 (Jiouru 4th Road)*	Change "(" to "=="; ")" to "="; leave the name alone; expand the abbreviation
9	Some street name in second Chinese character are separated by "-"- *博愛一路 (Bo-ai 1st Road)*	Remove"-" between the name; expand the abbreviation

I feel the most difficult challenge for this issue is to make the whole address to have a consistent format: one needs to know Chinese to be able to transform the street name correctly. It will be difficult to change the second Chinese character of the address/street name to upper case or separated by space.

iii. Many street names are under *<tag k="name">,* not under *<tag k="addr:street">.* So I also did the street audit check on the name. Because of language issues – many coded with in Chinese only, I did not change the *<tag k>* value.

iv. Sometimes *<tag k="name">* has multi-language versions, such as: *<tag k="name:zh">*, *<tag k="name:en">*, *<tag k="name:jp">*, etc. I changed this into a long string to include all the language translations for the "special name".

b. Overview of the Data

This section contains basic information about the dataset and MongoDB queries that response to the Project Rubric Questions.

i. Size of the file

1) kaohsiung_taiwan.osm -- Before unzip – 4.38MB; After unzip – 59.9MB
   kao_hsiung_taiwan.osm.json – 66.3MB

ii. Number of documents

> db.kaohsiung.find().count()
641170

iii. Number of unique users

> db.kaohsiung.distinct("created.user").length
296

iv. Number of nodes and ways

1) Number of nodes
   > db.kaohsiung.find({"type":"node"}).count()
   563152
2) Number of ways:
   > db.kaohsiung.find({"type":"way"}).count()
   78016

v. Number of chosen type of nodes, like cafes, shops, etc.

1) Top 5 contributing user:

```
> db.kaohsiung.aggregate([
        {"$group":{"_id":"$created.user", "count":{"$sum":1}}},{"$sort":{"count":-1}},
{"$limit":5}])
{ "_id" : "Kagami", "count" : 188193 }
{ "_id" : "mcdlee", "count" : 93523 }
{ "_id" : "ezjerry", "count" : 79443 }
{ "_id" : "plotch", "count" : 74204 }
{ "_id" : "LouisLiu", "count" : 67363 }
```

2) Number of users appearing only once (1 post)

```
> db.kaohsiung.aggregate([
        {"$group":{"_id":"$created.user", "count":{"$sum":1}}},
{"$group":{"_id":"$count", "num_users":{"$sum":1}}},{"$sort":{"_id":1}},
{"$limit":1}])
{ "_id" : 2, "num_users" : 38 }
```

3) Top 10 appearing amenities

```
> db.kaohsiung.aggregate([
    {"$match":{"amenity":{"$exists":1}}}, {"$group":{"_id":"$amenity",
    "count":{"$sum":1}}},{"$sort":{"count":-1}}, {"$limit":10}])
{ "_id" : "school", "count" : 782 }
{ "_id" : "place_of_worship", "count" : 760 }
{ "_id" : "restaurant", "count" : 691 }
{ "_id" : "parking", "count" : 607 }
{ "_id" : "grave_yard", "count" : 492 }
{ "_id" : "bicycle_rental", "count" : 374 }
{ "_id" : "police", "count" : 308 }
{ "_id" : "bank", "count" : 170 }
{ "_id" : "fuel", "count" : 160 }
{ "_id" : "shelter", "count" : 136 }
```

4) Top 5 popular cuisines

```
>db.kaohsiung.aggregate([
        {"$match":{"amenity":{"$exists":1},"amenity":"restaurant"}},{"$group":{"_id
":"$cuisine", "count":{"$sum":1}}},{"$sort":{"count":-1}},{"$limit":5}])
{ "_id" : null, "count" : 363 }
{ "_id" : "chinese", "count" : 128 }
{ "_id" : "regional", "count" : 26 }
{ "_id" : "japanese", "count" : 25 }
{ "_id" : "burger", "count" : 18 }
```

5) Top 5 postal code in the data file

```
> db.kaohsiung.aggregate([
        {"$match":{"address.postcode":{"$exists":1}}},{"$group":{"_id":"$address.po
    stcode", "count":{"$sum":1}}},{"$sort":{"count":-1}}, {"$limit":5}])
{ "_id" : "813", "count" : 318 }
{ "_id" : "804", "count" : 170 }
{ "_id" : "803", "count" : 96 }
{ "_id" : "807", "count" : 46 }
{ "_id" : "913", "count" : 44 }
```

6) Top 3 Religion type

```
> db.kaohsiung.aggregate([
        {"$match":{"amenity":{"$exists":1},"amenity":"place_of_worship"}},
{"$group":{"_id":"$religion", "count":{"$sum":1}}},{"$sort":{"count":-1}},
{"$limit":3}])
```

```
{ "_id" : null, "count" : 452 }
{ "_id" : "taoist", "count" : 182 }
{ "_id" : "christian", "count" : 54 }
```

7) <u>Top 5 appearing shop</u>

```
>db.kaohsiung.aggregate([
        {"$match":{"shop":{"$exists":1}}}, {"$group":{"_id":"$shop",
    "count":{"$sum":1}}}, {"$sort":{"count":-1}}, {"$limit":5}])
{ "_id" : "convenience", "count" : 467 }
{ "_id" : "supermarket", "count" : 149 }
{ "_id" : "beverages", "count" : 78 }
{ "_id" : "yes", "count" : 77 }
{ "_id" : "motorcycle", "count" : 76 }
```

8) <u>Top 5 café</u>

```
> db.kaohsiung.aggregate([
        {"$match":{"amenity":{"$exists":1}, "amenity":"cafe"}},
    {"$group":{"_id":$cuisine", "count":{"$sum":1}}},{"$sort":{"count":-1}},
    {"$limit":5}])
{ "_id" : null, "count" : 83 }
{ "_id" : "coffee_shop", "count" : 30 }
{ "_id" : "regional", "count" : 2 }
{ "_id" : "tea", "count" : 2 }
{ "_id" : "咖啡豆專賣", "count" : 2 } (coffee bean dealer)
```

vi. Data information

- I ran all my files under a copy of "kaohsiung_taiwan.osm" (59.9MB)
- Sample test data is called – "kaohsiung_map.osm" (2.55MB)

vii. Extra information – sample output using the data file:

1) The output from mapparser.py –

Kaohsiung_map.osm	Kaohsiung_taiwan.osm
{'bounds': 1, 'member': 6473, 'nd': 12293, 'node': 8230, 'osm': 1, 'relation': 217, 'tag': 9040, 'way': 2158}	*{'bounds': 1, 'member': 22698, 'nd': 337626, 'node': 277461, 'osm': 1, 'relation': 1549, 'tag': 130300, 'way': 37930}*

2) The output from tags.py –

Kaohsiung_map.osm	*{'lower': 117341, 'lower_colon': 11653, 'other': 1020, 'problemchars': 286}*
Kaohsiung_taiwan.osm	*{'lower': 304944, 'lower_colon': 57904, 'other': 968, 'problemchars': 266}*

3) The output from users.py –

pprint.pprint(len(users))	
Kaohsiung_map.osm -- 58	*Kaohsiung_taiwan.osm -- 297*

4) The output from audit.py –
   <u>Sample output:</u>

pprint.pprint(len(st_types))	
Kaohsiung_map.osm -- 3	*Kaohsiung_taiwan.osm -- 42*

print name, "=>", better_name

Kaohsiung_map.osm –

```
光華二路 80 號 => 光華二路 80 號
林森三路 193 巷 => 林森三路 193 巷
6, yùzhú 1st st, xinxing district => 6 yùzhú 1st st xinxing district
```

Kaohsiung_taiwan.osm – (sample output)

```
Dachang 1st Rd. => Dachang 1st Road
前鎮區沱江街 200 號 => 前鎮區沱江街 200 號
博愛二路 (Bo-ai 2nd Road) => 博愛二路 == Boai 2nd Road =
6, yùzhú 1st st, xinxing district => 6 yùzhú 1st st xinxing district
```

5) The output from data.py –

pprint.pprint(len(data))	
Kaohsiung_map.osm -- 10388	*Kaohsiung_taiwan.osm -- 315391*

6) Code to import json to mongodb

```
C:\Program Files\MongoDB 2.6 Standard\bin>mongoimport -host localhost:27017 -db
mydb -collection kaohsiung -file "c:\udacity\data analysis\project 2\FinalProjec
t\Final\kaohsiung_taiwan.osm.json" –jsonArray
```

c. Other ideas about the database

1) <u>Statistics:</u>
   - The number 1 user ("Kagami") contribution percentage – 29.35%
   - Combined top 3 users ("Kagami", "mcdlee","ezjerry") contribution percentage – 56.33%
   - Combined top 25 users contribution percentage – 96.79%
   - For place_of_worship and restaurant cuisine the number 1 for both lists are "Null". It can mean it either is a combination of different cuisines, or small temples that do not have a category. Since I grew up in Taiwan, my best guess for the null value in the cuisine category is for street food, or a small restaurant that is a combination of local food. There are many restaurants, cafés, or eateries in Taiwan that are a combination of different cuisines.
2) <u>Ideas for improving dataset:</u> It will be a great for people to setup rules for people who want to contribute to the dataset, especially for international countries that contain Unicode, ASCII, or other characters. When I checked the data file, I found several problems. (a) Some postal_code have 3 digits, 5 digits, or more digits. When I was growing up, it was 3 digits; I do not have knowledge of the current rules. It would be nice if it had some documents for each countries that specified the certain rules for that

country. (b) Because some fields (or field values) may not suitable for one country, the best choice is “NULL”. Also, it is possible that some groups of people do not have their own place to worship, and they are leasing/renting other places as their worship place. Instead of counting them in one place, it may be counted them as “Null” or not counting them. (c) It should set a uniform rule on how to translate Unicode/ASCII code or the case for the alphabetic. In the cuisine field, some people typed in “Japanese” or “Korean” instead of “japanese” or “korean” in the Key:cuisine page. (d) There are a few ways of encoding other languages (such as Unicode, ASCII) that many of people know, but there are also multiple Romanization methods, such as Yale Romanization, Wade-Giles, Hanyu Pinyin, etc. The data file should have a guideline on how to encode other languages’ characters. It takes time to input uniformly and correctly, but it will save more time and energy to clean/audit good data file.

3) Additional idea in other project:
After cleaning the data file, people can use it on their GPS, Maps, or OpenStreetMap to update the data as they go and provide more information on the fields that are given. For example, if they stop by some shops and find out they have website or updated information, they can upload it to their smartphones as soon as possible instead of waiting someone else to updated and go through the cleaning/audit process again!

3. Lesson 6 exercise code

Note: I did not remove all the instruction that was related to Lesson 6 exercise. I only modified the code to make sure work it with the data file I tested.

4. Conclusion

After checking the data, 1) it is difficult to convert or clean the data for someone who does not know the language, or to clean the area of map that they do not have knowledge or are not familiar. For example: the following examples were coded with Chinese language that it has similar values to what was given – in parentheses.

{ "_id" : "麵食", "count" : 4 } (cuisine - noodle/pasta)
{ "_id" : "霜淇淋", "count" : 2 } (cuisine - ice cream)
{ "_id" : "日本料理", "count" : 2 } (cuisine – japanese)
{ "_id" : "廣式燒臘烤鴨飯", "count" : 2 } (cuisine – Cantonese BBQ meal)
{ "_id" : "雪花冰、布丁豆花", "count" : 2 } (cuisine – Shaved_iced, pudding tofu dessert)
{ "_id" : "道教", "count" : 30 } (Religion - Taoism)
{ "_id" : "基督教", "count" : 1 } (Religion – Christian)

2) It is also difficult to determine when I am done with data cleaning. After I thought I cleaned most of major fields, I found out there were more fields that needed to be cleaned because I missed some fields the first few times. It took many repetitive steps to get the data file clean. If we can have consistent input methods for other languages, that will help us to avoid a large amount of manual updating of the data file. If the data is really clean, then when we upload it back to the GPS, we can avoid invalid records that lead drivers to nowhere or roads/places that do not exist, which can be a life saver in real life. I heard some people got lost or lost their lives because the GPS led them nowhere or the road ended or did not exist anymore because of unexpected causes.