Data Wrangling with MongoDB

OpenStreetMap Project

Map Area: New York and Connecticut I-95 Corridor

<https://www.openstreetmap.org/export#map=12/41.0662/-73.5583>

For the project I decided to work with the neighborhood where I grew up. I still go back frequently and would like to improve the map there!

Web Sources- MongoDB documentation; learn python the hard way; Stack Overflow; Udacity Discussion Boards

1. Problems Encountered in the Map
   1. The first problem I discovered was a lack of user data in the file I downloaded from the Overpass API. The output for the following query was 0 when I ran it in the command prompt:

db.map.distinct("created.user").length

I then looked through several entries to confirm this, and saw that there were no “user” tags.

I notified the instructors who subsequently amended the Overpass API instructions so it would include user data.

* 1. The second problem I encountered was finding an appropriately sized data file to work with. Originally I had intended to audit and clean data for New York City, but RAM limitations on my PC prevented me from doing so. I instead settled on the I-95 corridor in New York and Connecticut. The original XML file is 59.1MB, which both satisfies the requirements for the project, and is small enough that it does not overwhelm my computer.

* 1. The main problem, however was establishing a naming convention for some unusual street types. I was a bit surprised, however, that very few roads needed to be changed. I had run this exercise previously using a different dataset and there were a number of streets that needed to be changed. For example, some roads only displayed a number. I had to do a Google search to find what the number was referring to. In most cases, it was a highway, and I added the highway to the “expected” list and the correct name to the mapping dictionary. The amended street list is below.

Valley Rd => Valley Road

Iroquois Rd => Iroquois Road

Courtland Ave => Courtland Avenue

Harbor Dr => Harbor Drive

Pine St => Pine Street

Park St => Park Street

North Water St => North Water Street

Grove St => Grove Street

Elm St => Elm Street

1. Data Overview
   1. Number of nodes: 838,920
      1. db.map.find( { type: "node" } ).length()
   2. Number of ways: 85,576
      1. db.map.find( { type: "way" } ).length()
   3. DataSize: 189,429,296
      1. db.map.dataSize()
   4. Unique users: 248
      1. db.map.distinct("created.user").length
   5. I then looked at the 10 most popular amenities using:
      1. db.map.aggregate([{ "$group" : { "\_id" : "$amenity",

"count" : { "$sum" : 1 }}}, { "$sort" : { "count" : -1 }}, {"$limit" : 5}])

The results were:

u'result': [{u'\_id': None, u'count': 920800},

{u'\_id': u'parking', u'count': 930},

{u'\_id': u'school', u'count': 840},

{u'\_id': u'place\_of\_worship', u'count': 728},

{u'\_id': u'fire\_station', u'count': 154},

{u'\_id': u'restaurant', u'count': 153},

{u'\_id': u'grave\_yard', u'count': 145},

{u'\_id': u'library', u'count': 108},

{u'\_id': u'fuel', u'count': 67},

{u'\_id': u'fast\_food', u'count': 59}]}

* 1. I then chose to dig a little bit deeper on the most popular restaurants. I first created a list of all the restaurants and their type of cuisine. Given that the list has 153 entries, I will limit it to the first 5:.
     1. db.map.aggregate([{"$match": {"amenity": "restaurant"}},

{"$project":{"\_id": "$name", "cuisine": "$cuisine"}}])

* + 1. u'result': [{u'\_id': u"Joe's Pizzeria"},

{u'\_id': u'Cove Pizza'},

{},

{u'\_id': u"Brennan's By The Beach"},

{u'\_id': u'Fat Cat Pie Company', u'cuisine': u'pizza'}]}

* 1. Clearly, the list is very incomplete. This may be an opportunity for further data cleaning, and more complete information on the restaurants can be inserted into the database.
  2. Lastly, I took a look at the 5 most popular cuisines. Again, the dataset is not complete so the most popular “cuisine” was “None”. Unsurprisingly though, pizza was the second most popular in the map area with 17 restaurants..
     1. db.map.aggregate([{"$match": {"amenity": "restaurant"}},

{ "$group" : { "\_id" : "$cuisine",

"count" : { "$sum" : 1 }}}, { "$sort" : { "count" : -1 }}, {"$limit" : 5}])

1. Additional Ideas
   1. In querying the data I noticed that it is woefully incomplete in listing the recreation stops within the map area. Given my interests in outdoor activities and ice hockey, I would like to add parks and ice rinks to the data set. Adding that data for the entire map area could prove to be very challenging, but focusing on the areas I am familiar with is a bit more manageable. I can use the insert function to add more documents to the OSM dataset. Hopefully other people will find this additional data useful!
2. Conclusion
   1. In the end, I was a bit surprised that the dataset was not more complete given that it is a well-populated area and includes several major transport hubs. The good news, however, is that it provides plenty of opportunity to improve my data auditing and cleaning skills!