

OpenStreetMap Sample Project

Data Wrangling with MongoDB

Ahmet Hamza EMRA

Map Area: Houston, TX, United States

<https://www.openstreetmap.org/relation/2688911#map=10/29.7721/-95.3229>

1. Problems Encountered in the Map

After initially downloading a small sample size of the Houston area and running it against a provisional data.py file, I noticed three main problems with the data, which I will discuss in the following order:

- Over-abbreviated street names or miss spellings corrected.
- “Incorrect” tag types corrected or removed
- “Incorrect” postal codes (*Houston area zip codes all begin with “77” however a large portion of all documented zip codes were outside this region.*)

Street Names

Before convert document to JSON format, I chance and correct miss spellings and other wrong things corrected with the python code. The code basically correct name for example “St. to Street. Other examples are:

```
"St": "Street",  
"St.": "Street",  
"Ave": "Avenue",  
"Rd.": "Road",  
'Ave.': 'Avenue',  
'Blvd': 'Boulevard',  
'Blvd.': 'Boulevard',  
'Dr': 'Drive',  
'Frwy': 'Freeway',  
'PkwY': 'Parkway',  
'Rd': 'Road',  
'Rd.': 'Road',  
'Stree': 'Street',  
'blvd': 'Boulevard'
```

“Incorrect” tag types corrected or removed

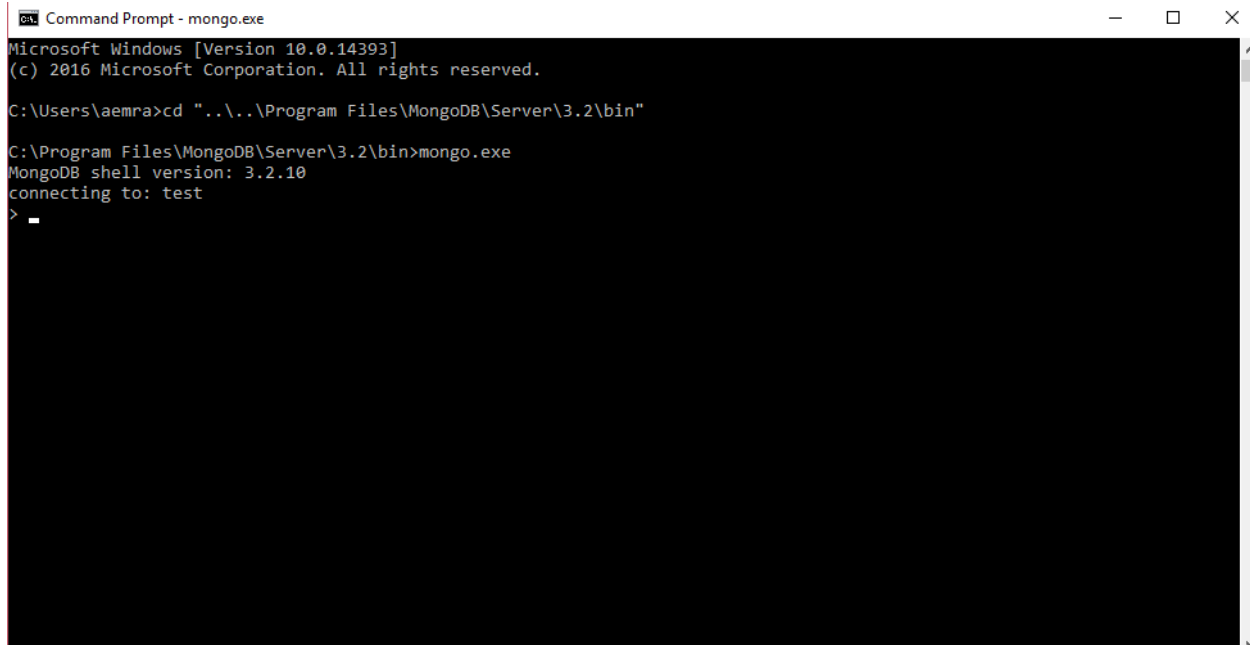
There were a lot of tag has value of “FIXME”, I checked them and most of them was questions or nonsense inputs from people. I remove them and then I convert XML document to JSON format document.

2. Uploading to MongoDB

For me it was the most complicated part. I prefer to upload with mongo shell.

Steps:

1. Open cmd and start mongod.exe application
2. Open another command prompt and start mongo.exe application

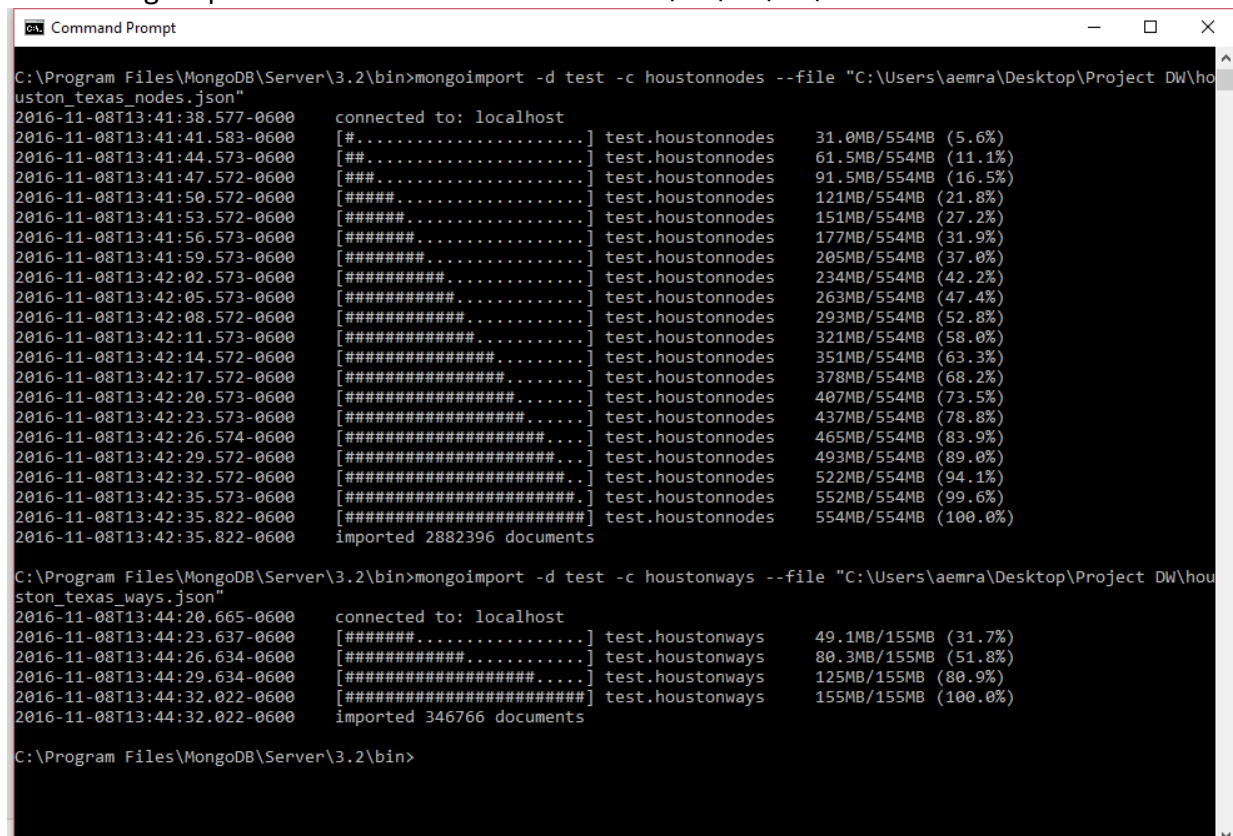


```
Command Prompt - mongo.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Aemra>cd "..\..\Program Files\MongoDB\Server\3.2\bin"

C:\Program Files\MongoDB\Server\3.2\bin>mongo.exe
MongoDB shell version: 3.2.10
connecting to: test
>
```

3. Form another command prompt go to bin directory and use mongo import code
Code: "mongoimport -d test -c new -file C:\..\..\..\foldername"



```
Command Prompt

C:\Program Files\MongoDB\Server\3.2\bin>mongoimport -d test -c houstonnodes --file "C:\Users\Aemra\Desktop\Project DW\houston_texas_nodes.json"
connected to: localhost
2016-11-08T13:41:38.577-0600 [#.....] test.houstonnodes 31.0MB/554MB (5.6%)
2016-11-08T13:41:41.583-0600 [##.....] test.houstonnodes 61.5MB/554MB (11.1%)
2016-11-08T13:41:44.573-0600 [###.....] test.houstonnodes 91.5MB/554MB (16.5%)
2016-11-08T13:41:47.572-0600 [####.....] test.houstonnodes 121MB/554MB (21.8%)
2016-11-08T13:41:53.572-0600 [#####.....] test.houstonnodes 151MB/554MB (27.2%)
2016-11-08T13:41:56.573-0600 [#####.....] test.houstonnodes 177MB/554MB (31.9%)
2016-11-08T13:41:59.573-0600 [#####.....] test.houstonnodes 205MB/554MB (37.0%)
2016-11-08T13:42:02.573-0600 [#####.....] test.houstonnodes 234MB/554MB (42.2%)
2016-11-08T13:42:05.573-0600 [#####.....] test.houstonnodes 263MB/554MB (47.4%)
2016-11-08T13:42:08.572-0600 [#####.....] test.houstonnodes 293MB/554MB (52.8%)
2016-11-08T13:42:11.573-0600 [#####.....] test.houstonnodes 321MB/554MB (58.0%)
2016-11-08T13:42:14.572-0600 [#####.....] test.houstonnodes 351MB/554MB (63.3%)
2016-11-08T13:42:17.572-0600 [#####.....] test.houstonnodes 378MB/554MB (68.2%)
2016-11-08T13:42:20.573-0600 [#####.....] test.houstonnodes 407MB/554MB (73.5%)
2016-11-08T13:42:23.573-0600 [#####.....] test.houstonnodes 437MB/554MB (78.8%)
2016-11-08T13:42:26.574-0600 [#####.....] test.houstonnodes 465MB/554MB (83.9%)
2016-11-08T13:42:29.572-0600 [#####.....] test.houstonnodes 493MB/554MB (89.0%)
2016-11-08T13:42:32.572-0600 [#####.....] test.houstonnodes 522MB/554MB (94.1%)
2016-11-08T13:42:35.573-0600 [#####.....] test.houstonnodes 552MB/554MB (99.6%)
2016-11-08T13:42:35.822-0600 [#####.....] test.houstonnodes 554MB/554MB (100.0%)
2016-11-08T13:42:35.822-0600 imported 2882396 documents

C:\Program Files\MongoDB\Server\3.2\bin>mongoimport -d test -c houstonways --file "C:\Users\Aemra\Desktop\Project DW\houston_texas_ways.json"
connected to: localhost
2016-11-08T13:44:20.665-0600 [#####.....] test.houstonways 49.1MB/155MB (31.7%)
2016-11-08T13:44:23.637-0600 [#####.....] test.houstonways 80.3MB/155MB (51.8%)
2016-11-08T13:44:26.634-0600 [#####.....] test.houstonways 125MB/155MB (80.9%)
2016-11-08T13:44:32.022-0600 [#####.....] test.houstonways 155MB/155MB (100.0%)
2016-11-08T13:44:32.022-0600 imported 346766 documents

C:\Program Files\MongoDB\Server\3.2\bin>
```

3. Data Overview

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

File sizes

Houston_texas.osm	643 MB
Houston_texas_ways.json	159 MB
Houston_texas_nodes.json	567 MB

#Number of Documents

```
> db.houstonnodes.find().count()
```

```
2882396
```

```
> db.houstonways.find().count()
```

```
365469
```

#Number of unique users input (limited to 10):

Nodes:

```
In [39]: q6=[doc for doc in db.houstonnodes.aggregate([{"$group":{"_id":"$created.user", "count":{"$sum":1}}}, {"$sort":{"count":-1}}, {"$limit":10}])]
pprint.pprint(q6)

[{'_id': 'woodpeck_fixbot', 'count': 588574},
 {'_id': 'TexasNHD', 'count': 543274},
 {'_id': 'afdreher', 'count': 376692},
 {'_id': 'scottyc', 'count': 172487},
 {'_id': 'cammace', 'count': 165707},
 {'_id': 'brianboru', 'count': 114565},
 {'_id': 'claysmalley', 'count': 93469},
 {'_id': 'RoadGeek_MD99', 'count': 69860},
 {'_id': 'skquinn', 'count': 64037},
 {'_id': 'xsintrik', 'count': 48721}]
```

Ways:

```
In [40]: q6=[doc for doc in db.houstonways.aggregate([{"$group":{"_id":"$created.user", "count":{"$sum":1}}}, {"$sort":{"count":-1}}, {"$limit":10}])]
pprint.pprint(q6)

[{'_id': 'afdreher', 'count': 73034},
 {'_id': 'balrog-kun', 'count': 33825},
 {'_id': 'scottyc', 'count': 32250},
 {'_id': 'DaveHansenTiger', 'count': 29565},
 {'_id': 'cammace', 'count': 28135},
 {'_id': 'skquinn', 'count': 16956},
 {'_id': 'RoadGeek_MD99', 'count': 13553},
 {'_id': 'brianboru', 'count': 12761},
 {'_id': 'claysmalley', 'count': 12364},
 {'_id': 'Memoire', 'count': 7632}]
```

Number of users appearing only once (having 1 post)

```
In [42]: q10=[doc for doc in db.houstonnodes.aggregate([{"$group":{"_id":"$created.user", "count":{"$sum":1}}},
{"$group":{"_id":"$count", "num_users":{"$sum":1}}},
{"$sort":{"_id":1}}, {"$limit":1}])]
pprint.pprint(q10)
[{'_id': 1, 'num_users': 237}]
```

3. Additional Ideas

Top 10 appearing amenities

```
In [45]: q11=[doc for doc in db.houstonnodes.aggregate([{"$match":{"amenity":{"$exists":1}}}, {"$group":{"_id":"$amenity",
"count":{"$sum":1}}}, {"$sort":{"count":-1}}, {"$limit":10}])]
pprint.pprint(q11)
[{'_id': 'place_of_worship', 'count': 2211},
{'_id': 'school', 'count': 1033},
{'_id': 'fountain', 'count': 657},
{'_id': 'restaurant', 'count': 636},
{'_id': 'fast_food', 'count': 586},
{'_id': 'fire_station', 'count': 355},
{'_id': 'fuel', 'count': 253},
{'_id': 'pharmacy', 'count': 171},
{'_id': 'bank', 'count': 165},
{'_id': 'police', 'count': 162}]
```

Biggest religion

```
In [11]: q7=[doc for doc in db.houstonnodes.aggregate([{"$match":{"amenity":{"$exists":1}, "amenity":"place_of_worship"}},
{"$group":{"_id":"$religion", "count":{"$sum":1}}},
{"$sort":{"count":-1}}])]
pprint.pprint(q7)
[{'_id': 'christian', 'count': 2150},
{'_id': 'None', 'count': 23},
{'_id': 'buddhist', 'count': 14},
{'_id': 'jewish', 'count': 12},
{'_id': 'muslim', 'count': 8},
{'_id': 'unitarian_universalist', 'count': 4}]
```

Names of Muslim places

```
In [28]: q8=[doc for doc in db.houstonnodes.aggregate([{"$match":{"amenity":{"$exists":1}},
{"$match":{"religion":"muslim"}},
{"$group":{"_id":"$name", "count":{"$sum":1}}}]])
pprint.pprint(q8)
[{'_id': 'Masjid As Sabireen', 'count': 1},
{'_id': 'Makkah Masjid of Greater Houston', 'count': 1},
{'_id': 'Isgh Islamic Society of Greater Houston', 'count': 1},
{'_id': 'Islamic Society of Greater Houston', 'count': 5}]
```

Names of Christian places

```
In [46]: q8=[doc for doc in db.houstonnodes.aggregate([{"$match":{"amenity":{"$exists":1}},
{"$match":{"religion":"christian"}},
{"$group":{"_id":"$name", "count":{"$sum":1}}},
{"$limit":5}])]
pprint.pprint(q8)
[{'_id': 'Pine Island Church', 'count': 1},
{'_id': 'Advent Lutheran Church', 'count': 1},
{'_id': 'Church of the Good Shepherd', 'count': 1},
{'_id': 'Iglesia Bautista Houston', 'count': 1},
{'_id': 'First Baptist Church of Patton Lakes', 'count': 1}]
```

Other queries can be found in [Project_queries.ipynb](#)

Conclusion

After this review of the data it's obvious that the Houston area is incomplete, though I believe it has been well cleaned for the purposes of this exercise. It interests me to notice a fair amount of GPS data makes it into OpenStreetMap.org on account of users' efforts, whether by scripting a map editing bot or otherwise. With a

rough GPS data processor in place and working together with a more robust data processor similar to data.py I think it would be possible to input a great amount of cleaned data to OpenStreetMap.org.