OpenStreetMap Sample Project
Data Wrangling with MongoDB

Stephan Ketterer

Map Area: Las Vegas, Nevada, United States
https://s3.amazonaws.com/metro-extracts.mapzen.com/las-vegas_nevada.osm.bz2

1.Problems Encountered in the Map

Even though I created a snippet file of only the first 100 entries, it was quite hard for me to find a special pattern that needed a different cleanup than the one used in the last programming assignment. Since the chosen map was also a part of the United States, I figured I will most likely encounter the same problems, especially since Las Vegas is a well known city, with many sights and a lot of conferences. This should mean that there exists where solid open street map data on it.So I went with the established clean up procedure that was used in the Case Study in Lesson 6.
Subsequently I tackled the problem with the zip codes sometimes containing "NV". As shown in my provided .py file I simply replaced the part of the string in question.
The addresses were formatted as described in the programming assignment and different kind of attributes were saved properly for later access with mongodb queries.

Zip code problem:
zip=db2.col2.aggregate([{"$match":{"address.postcode":{"$exists":1}}}, {"$group":{"_id":"$address.postcode", "count":{"$sum":1}}}, {"$sort":{"count":-1}}])

[{u'count': 46, u'_id': u'89122'}, {u'count': 40, u'_id': u'89109'}, {u'count': 35, u'_id': u'89015'}, {u'count': 25, u'_id': u'89101'},

{u'count': 24, u'_id': u'89005'}, {u'count': 20, u'_id': u'89121'}, {u'count': 19, u'_id': u'89119'}, {u'count': 16, u'_id': u'89142'}, {u'count': 14, u'_id': u'89002'}, {u'count': 14, u'_id': u'89118'}, {u'count': 11, u'_id': u'89103'}, {u'count': 11, u'_id': u'89139'}, {u'count': 10, u'_id': u'89169'}, {u'count': 9, u'_id': u'89104'}, {u'count': 8, u'_id': u'89113'}, {u'count': 8, u'_id': u'89014'}, {u'count': 8, u'_id': u'89123'}, {u'count': 7, u'_id': u'89183'}, {u'count': 7, u'_id': u'89052'}, {u'count': 7, u'_id': u'89107'}, {u'count': 7, u'_id': u'89117'}, {u'count': 7, u'_id': u'89146'}, {u'count': 6, u'_id': u'89128'}, {u'count': 6, u'_id': u'89135'}, {u'count': 6, u'_id': u'89149'}, {u'count': 6, u'_id': u'89102'}, {u'count': 6, u'_id': u'89012'}, {u'count': 5, u'_id': u'89074'}, {u'count': 5, u'_id': u'89108'}, {u'count': 4, u'_id': u'89081'}, {u'count': 4, u'_id': u'89115'}, {u'count': 4, u'_id': u'89129'}, {u'count': 3, u'_id': u'89131'}, {u'count': 3, u'_id': u'89130'}, {u'count': 3, u'_id': u'NV 89109'}, {u'count': 3, u'_id': u'89144'}, {u'count': 3, u'_id': u'89145'}, {u'count': 2, u'_id': u'89040'}, {u'count': 2, u'_id': u'89120'}, {u'count': 2, u'_id': u'NV 89052'}, {u'count': 2, u'_id': u'89134'}, {u'count': 2, u'_id': u'NV 89119'}, {u'count': 2, u'_id': u'89147'}, {u'count': 2, u'_id': u'89106'}, {u'count': 1, u'_id': u'NV 89142'}, {u'count': 1, u'_id': u'89154'}, {u'count': 1, u'_id': u'89148'}, {u'count': 1, u'_id': u'NV 89129'}, {u'count': 1, u'_id': u'NV 89145'}, {u'count': 1, u'_id': u'NV 89124'}, {u'count': 1, u'_id': u'NV 89107'}, {u'count': 1, u'_id': u'Nevada 89113'}, {u'count': 1, u'_id': u'NV 89014'}, {u'count': 1, u'_id': u'NV 89134'}, {u'count': 1, u'_id': u'89147-4111'}, {u'count': 1, u'_id': u'89011'}, {u'count': 1, u'_id': u'89025'}, {u'count': 1, u'_id': u'NV 89117'}, {u'count': 1, u'_id': u'89179'}, {u'count': 1, u'_id': u'NV 89123'}, {u'count': 1, u'_id': u'89178'}, {u'count': 1, u'_id': u'89030'}, {u'count': 1, u'_id': u'89191'}, {u'count': 1, u'_id': u'NV 89030'}, {u'count': 1, u'_id': u'89109-1907'}, {u'count': 1, u'_id': u'89044'}, {u'count': 1, u'_id': u'89032'}, {u'count': 1, u'_id': u'89156'}, {u'count': 1, u'_id': u'89161'}, {u'count': 1, u'_id': u'NV 89191'}, {u'count': 1, u'_id': u'NV 89031'}, {u'count': 1, u'_id': u'89105'}, {u'count': 1, u'_id': u'NV 89101'}]

After cleaning up this problem and reuploading it to mongodb, the same query gave the desired result:

zipcleaned=db3.myc.aggregate([{"$match":{"address.postcode": {"$exists":1}}}, {"$group":{"_id":"$address.postcode", "count":{"$sum":1}}}, {"$sort":{"count":-1}},{"$limit": 20}])

[{u'count': 46, u'_id': u'89122'}, {u'count': 40, u'_id': u'89109'}, {u'count': 35, u'_id': u'89015'}, {u'count': 25, u'_id': u'89101'}, {u'count': 24, u'_id': u'89005'}, {u'count': 20, u'_id': u'89121'}, {u'count': 19, u'_id': u'89119'}, {u'count': 16, u'_id': u'89142'}, {u'count': 14, u'_id': u'89002'}, {u'count': 14, u'_id': u'89118'}, {u'count': 11, u'_id': u'89139'}, {u'count': 11, u'_id': u'89103'}, {u'count': 10, u'_id': u'89169'}, {u'count': 9, u'_id': u'89104'}, {u'count': 8, u'_id': u'89113'}, {u'count': 8, u'_id': u'89123'}, {u'count': 8, u'_id': u'89014'}, {u'count': 7, u'_id': u'89052'}, {u'count': 7, u'_id': u'89183'}, {u'count': 7, u'_id': u'89107'}]

As predicted, the faulty “NV” has been successfully cleaned from the data.

City names:

Looking at the city names:
cities=db2.col2.aggregate([{"$match":{"address.city":{"$exists": 1}}}, {"$group":{"_id":"$address.city", "count":{"$sum":1}}}, {"$sort":{"count":-1}}])

[{u'count': 248, u'_id': u'Las Vegas'}, {u'count': 79, u'_id': u'Henderson'}, {u'count': 5, u'_id': u'North Las Vegas'}, {u'count': 5, u'_id': u'Boulder City'}, {u'count': 2, u'_id': u'Spring Valley'}, {u'count': 2, u'_id': u'las vegas'}, {u'count': 2, u'_id': u'Boulder City NV'}, {u'count': 2, u'_id': u'Las vegas'}, {u'count': 2, u'_id': u'Boulder City, NV'}, {u'count': 2, u'_id': u'Las Vagas'},

{u'count': 2, u'_id': u'Overton'}, {u'count': 2, u'_id': u'Las Vegas, NV'}, {u'count': 1, u'_id': u'Nellis AFB'}, {u'count': 1, u'_id': u'Moapa'}, {u'count': 1, u'_id': u'Whitney'}]

It is observable that the data is still not perfectly formatted. There are some misspellings (Las Vagas), but also just different capitalization that leads to different values for the same city. Also looking at "Boulder City, NV" it seems that when submitting data points, it is not always clear where to actually put the state abbreviation in the data.
Cleaning could be achieved the same way described as in the zip code problem. Simply replace the "NV" with an empty character.

2.Data Overview

File sizes

lasvegas.osm	175 MB
lasvegas.json	190 MB

```
# Number of documents
db2.col2.find().count()

885656


# Number of nodes

db2.col2.find({"type":"node"}).count()

799544
```

#Number of ways

db2.col2.find({"type":"way"}).count()

86112

#Number of unique users

db2.col2.distinct("created.user").__len__()

681

#Top contributing user

db2.col2.aggregate([{"$group":{"_id": "$created.user", "count":{"$sum":1}}},{"$sort":{"count":-1}},{"$limit":1}])

{u'count': 254460, u'_id': u'alimamo'}

3. Additional Ideas

Contributor statistics:

The amount each user contributed varied greatly.

Top user contribution percentage (‘alimamo’)=28.73%

Top 10 users contribution = 74.33%

Additional data exploration using MongoDB queries:

```
# Top 10 appearing amenities
db2.col2.aggregate([{"$match": {"amenity": {"$exists": 1}}},
                    {"$group": {"_id": "$amenity", "count":
                    {"$sum": 1}}},
                    {"$sort": {"count": -1}},
                    {"$limit": 10}
[{u'count': 797, u'_id': u'parking'},
{u'count': 531, u'_id': u'school'},
 {u'count': 364, u'_id': u'place_of_worship'},
 {u'count': 269, u'_id': u'fountain'},
 {u'count': 218, u'_id': u'restaurant'},
{u'count': 181, u'_id': u'fast_food'},
 {u'count': 143, u'_id': u'fuel'},
{u'count': 69, u'_id': u'fire_station'},
{u'count': 67, u'_id': u'hospital'},
{u'count': 63, u'_id': u'post_office'}]



# Most popular cuisines
db2.col2.aggregate([{"$match":{"amenity":{"$exists":1},
"amenity":"restaurant"}}, {"$group":{"_id":"$cuisine",
"count":{"$sum":1}}},{"$sort":{"count":-1}}, {"$limit":3}])
{u'count': 105, u'_id': None},
```

{u'count': 16, u'_id': u'pizza'},

{u'count': 15, u'_id': u'mexican'}

#Places of worship

db2.col2.aggregate([{"$match":{"amenity":{"$exists":1},
"amenity":"place_of_worship"}},

{"$group":{"_id":"$religion", "count":{"$sum":1}}},

{"$sort":{"count":-1}}, {"$limit":10}])

[{u'count': 339, u'_id': u'christian'},

{u'count': 10, u'_id': None},

{u'count': 4, u'_id': u'jewish'},

{u'count': 3, u'_id': u'bahai'},

{u'count': 2, u'_id': u'muslim'},

{u'count': 2, u'_id': u'buddhist'},

{u'count': 1, u'_id': u'scientologist'},

{u'count': 1, u'_id': u'sikh'},

{u'count': 1, u'_id': u'unitarian_universalist'}, {

u'count': 1, u'_id': u'hindu'}]

#Number of Casinos

casinos=db2.col2.aggregate([{"$match":{"amenity":'casino'}},{"
$group": {"_id":"casinos", "count": {"$sum":1}}}])

[{u'count': 30, u'_id': u'casinos'}]

#Number of Banks

banks=db2.col2.aggregate([{"$match":{"amenity":'bank'}},{"$group": {"_id":"null", "count": {"$sum":1}}}])

[{u'count': 53, u'_id': u'banks'}

Conclusion

Clearly the data for Las Vegas is quite extensive and someone can get a lot of information out of it. But at second glance, it is obvious that a lot of information is either missing, or not properly formatted yet. Taking a look at the top ten list of amenities, restaurant and fast_food are listed separately, where clearly there should be some connection or a common tag. Looking at the cuisine of restaurants, by far the biggest number is “None”. This shows that there is still a lot of work to be done. The same phenomenon can be observed when looking at houses of worship. Number two on the list is again identified as “None”, even though just by looking at the overall numbers, one can speculate that most of them are probably of Christian denomination. I think a possible approach to fill out the missing information and getting a better map file would be cross referencing the database with as many applicable databases as possible. I think vast improvements could be made, for looking for common tags and names and enriching the osm database with that new data. Different companies must have different databases covering the same area, for example a utility company might have very

detailed and specific data relevant to their needs that could be integrated into the osm database. Lastly I think the huge number of casinos in Las Vegas must have something to do with the in my opinion very large number of banks for a city of that size.