OpenStreetMap Data for Denver – Boulder:

This project is a requirement for the Udacity's Data Wrangling with MongoDB course. For more information on OpenStreetMap, go to its website. After downloading the OSM XML file of a particular area of interest, data are extracted and cleaned and converted to a database. For this project, the database used is SQL using SQLite.

Map Area

The area i choose is Denver -Boulder, Colorado. I went to this city on trip and explored a lot in this area. As per the project requirement it has about 68 MB file size uncompressed.

https://mapzen.com/data/metro-extracts/metro/denver-boulder_colorado

Problems Encountered in the Map

After downloading the data from the above link and I observed the data carefully which helped me in finding the major issues in the data, which are the following:

1) Abbreviation of street names ("S boulder St 105"):

By observing the audit of street names Some of the street names are abbreviated inconsistently. To resolve this problem, I used update_name function. With this function and other sub functions we have successfully updated the street names. Example of updating street name is shown below.

E Arapahoe Rd' -> East Arapahoe Road

'S Golden Rd'-> South Golden Road.

2) Inconsistent postal codes:

By observing the Postal codes, we can clean them as we know the length of the postal codes so we can remove the junk of data by using the length of it and we can filter the postal codes by knowing the information that all the postal codes in the Denver starts with number 8.

3) Inconsistent phone number format (+1 -908-409-1294, 9084091294):

Having looked at the phone number list we can say there is inconsistency. As in order to maintain consistency I followed the below regular expression.
example:

+1 -908-409-1294 -> 908-409-1294

9084091294 -> 908-409-1294

```
phone_re = re. compile(r'^\d\d\d\-\d\d\d\-\d\d\d\d$')
```

4) City name for is inconsistent:

To was issue with the street names as many shortcuts were used for many cities. In order to maintain consistency, we have done mapping of cities. With the help of mapping we have updated the data.
example:

boulder -> Boulder

B'lder -> Boulder

Co -> Colorado

5) Handling more than one phone number:

For some of the records the data was holding more than one phone number. So, in order to maintain consistency, I have dropped extra phone numbers information. I know it was a data loss but maintaining consistency was more important in this scenario.

Extraction of Data from OSM File to CSV Files:

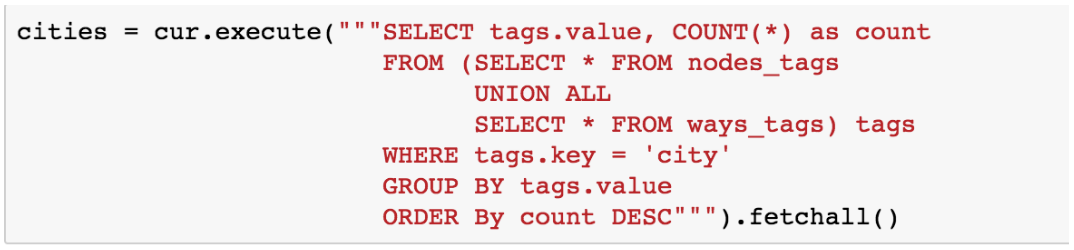
The general scheme for processing osm files start from creating csv files as output files then shaping the output, validating this output against a set schema and then writing the output onto the csv files.

Creation SQL Database:

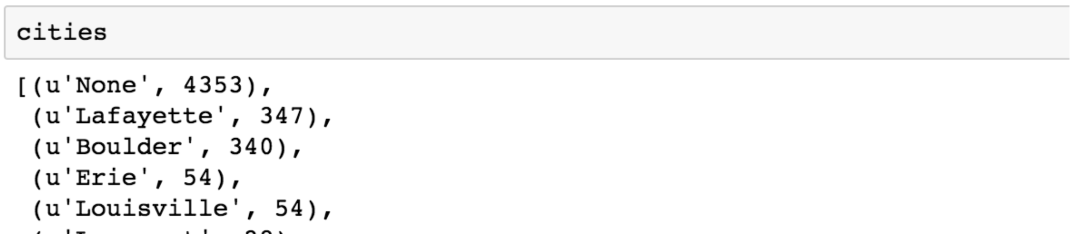
Using the schema specified in the following site https://gist.github.com/swwelch/f1144229848b407e0a5d13fcb7fbbd6f. Creating the SQL database (do_co.db) was done using Python according to the method outlined in the course forum (https://discussions.udacity.com/t/creating-db-file-from-csv-files-with-non-ascii-unicode-characters/174958/6)

Querying the SQL Database:

Querying the list of cities shown were cleaned:



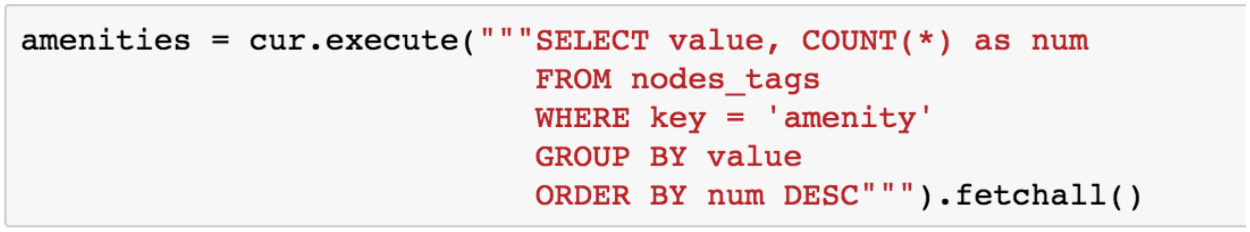
```
cities = cur.execute("""SELECT tags.value, COUNT(*) as count
                        FROM (SELECT * FROM nodes_tags
                              UNION ALL
                              SELECT * FROM ways_tags) tags
                        WHERE tags.key = 'city'
                        GROUP BY tags.value
                        ORDER By count DESC""").fetchall()
```



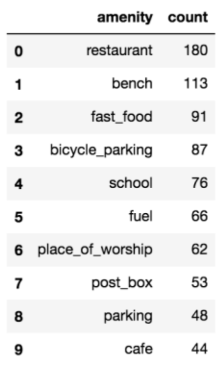
```
cities
```

```
[(u'None', 4353),
 (u'Lafayette', 347),
 (u'Boulder', 340),
 (u'Erie', 54),
 (u'Louisville', 54),
```

Querying the list of popular amenities:



```
amenities = cur.execute("""SELECT value, COUNT(*) as num
                           FROM nodes_tags
                           WHERE key = 'amenity'
                           GROUP BY value
                           ORDER BY num DESC""").fetchall()
```



	amenity	count
0	restaurant	180
1	bench	113
2	fast_food	91
3	bicycle_parking	87
4	school	76
5	fuel	66
6	place_of_worship	62
7	post_box	53
8	parking	48
9	cafe	44

Data Overview:

Files overview:

denver-boulder_colorado.osm   1.05 GB

sample44.osm	105.9 MB
do_co.db	56.7 MB
nodes.csv	41.1 MB
nodes_tags.csv	1.7 MB
ways.csv	3.3 MB
ways_tags.csv	6.5 MB
ways_nodes.csv	13.1 MB

Number of nodes:

```
[IN]
cur.execute("SELECT COUNT(*) FROM nodes")
nodes = cur.fetchall()
nodes

[OUT] [(477359,)]
```

Number of ways:

```
In [144]: cur.execute("SELECT COUNT(*) FROM ways")
        ways = cur.fetchall()
        ways
Out [144]:
        [(52847,)]
```

Number of Unique users

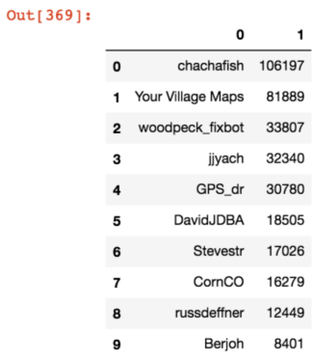
```
In [19]: cur.execute("""SELECT COUNT(DISTINCT(e.uid))
                FROM (SELECT uid from nodes UNION ALL SELECT uid FROM ways) e""")
        users = cur.fetchall()
        users
Out [19]:
        [(1625,)]
```

Top 10 Contributors:

```
top10contributors = cur.fetchall()
top10contributors

[(u'chachafish', 106197),
 (u'Your Village Maps', 81889),
 (u'woodpeck_fixbot', 33807),
 (u'jjyach', 32340),
 (u'GPS_dr', 30780),
 (u'DavidJDBA', 18505),
 (u'Stevestr', 17026),
 (u'CornCO', 16279),
 (u'russdeffner', 12449),
 (u'Berjoh', 8401)]
```

By using pandas, we can have a clear view of the results.



Out[369]:

	0	1
0	chachafish	106197
1	Your Village Maps	81889
2	woodpeck_fixbot	33807
3	jjyach	32340
4	GPS_dr	30780
5	DavidJDBA	18505
6	Stevestr	17026
7	CornCO	16279
8	russdeffner	12449
9	Berjoh	8401

Improvement of Data IN Areas:

A standardization of the k values should be instituted by OpenStreetMap. Anything that does not fit the list of these k values should create an error upon data entry for contributors. Also, the format for the values might also be standardized. A disadvantage of such rules however, might discourage contributors causing a slow development of OSM. However, if an automated cleaning program is instituted, it might be ok.

Conclusion:

In this OpenStreetMap project of Boulder-Colorado, we have cleaned a significant amount of data. But there are lot of areas improvement needed in the dataset. As by observing it contains less amount of information such as amenities, tourist attractions, and popular places and other useful interest. This database contains outdated information.

So, I think there are several opportunities for cleaning and validation of the data in the future.

Additional Suggestion and Ideas:

Control typo errors

1)With the help of parser we can parse every word input by the user.

2)We need to follow a pattern to input data which makes the user follow every time to input their data.

3)We can develop scrip or bot to clean the data on daily bases or certain period.

4)By providing the popular places and interests there is more chance to increase views on the map because many people enter the famous name on the map.