Data Overview:

I downloaded the Denver-Boulder OSM file from the Map Zen. The size of the file was pretty big; below is the screenshot for both OSM and JSON file. This met the sizing requirements. I chose Denver because it is my favorite city in the United States.

```
711M Jun 16 22:57 denver.osm
841M Jun 20 20:40 denver.osm.json
```

The file mapparser.py was created to find the count of each tag in the OSM file. Below is the result of the following:

```
Rahuls-MacBook-Pro:P3 Baid$ python mapparser.py
{'bounds': 1,
 'member': 29384,
 'nd': 3926660,
 'node': 3364999,
 'osm': 1,
 'relation': 1639,
 'tag': 1869505,
 'way': 371406}
```

Number of documents:

```
> db.denver.count()
3420287
```

Number of Nodes:

```
  db.denver.find({'type':'node'}).count()
3364968
```

Number of Ways:

```
  db.denver.find({'type':'way'}).count()
55288
```

Number of Unique Users:

```
  db.denver.distinct('created.user').length
1684
```

Top contributing User:

```
  db.denver.aggregate([{'$group': {'_id': '$created.user','count':
 {'$sum': 1}}}, {'$sort': {'count': -1}}, {'$limit': 1}])
{ "_id" : "Your Village Maps", "count" : 590273 }
```

Number of cafes and shops:

```
  db.denver.find({"amenity":"cafe"}).count()
403
> db.denver.find({"amenity":"shop"}).count()
7
```

Problems Encountered:

The most common problem that was addressed in this data file was of the abbreviated street names. Things like Ave., St, Blvd etc. appeared constantly in the file. I changed that by using the audit.py file. Below are snapshots of the data before and after audit

```
        'W 64th Ave',
        'W 72nd Ave',
        'W 84th Ave',
        'W Ken Caryl Ave',
        'W Plymouth Ave',
        'W. 10th Ave',
        'W. 25th Ave',
        'Washington Ave',
        'West 136th Ave',
        'Yarmouth Ave',
        'w. 10th Ave'},
'Ave.': {'11200 W. 64th Ave.',
         'E 6th Ave.',
         'E. College Ave.',
         'W. Alameda Ave.'},
'Avenue)': {'East Bromley Lane (152nd Avenue)'},
'Baselin': {'Baselin'},
'Baseline': {'Baseline'},
'Blvd': {'745 Colorado Blvd',
         'Airport Blvd',
         'Colorado Blvd',
         'East Academy Blvd',
         'Eldorado Blvd',
         'Federal Blvd',
         'Green Valley Ranch Blvd',
         'Martin Luther King Blvd',
         'Martin Luther King Jr Blvd',
         'McCaslin Blvd',
         'N Sheridan Blvd',
         'S Lowell Blvd',
         'S University  Blvd',
         'S University Blvd',
         'Sheridan Blvd',
         'South Colorado Blvd',
         'South University Blvd',
         'Wadsworth Blvd'},
```

After:

```
Hwy 73 => Highway 73
DTC Pky => DTC Parkway
Martin Luther King Blvd => Martin Luther King Boulevard
Eldorado Blvd => Eldorado Boulevard
Sheridan Blvd => Sheridan Boulevard
Airport Blvd => Airport Boulevard
Martin Luther King Jr Blvd => Martin Luther King Jr Boulevard
S University Blvd => S University Boulevard
Federal Blvd => Federal Boulevard
Colorado Blvd => Colorado Boulevard
```

1. The other problem I encountered was with the abbreviation St. It was used for both churches and streets. This forced me to use the mapping only for the last abbreviation.
2. Some cities that are mapped not a part of Denver. For example cities like Aurora, Boulder etc. make up the top 5 cities.

```
  db.denver.aggregate([{"$match":{"address.city":{"$exists":1}}},{
"$group":{"_id":"$address.city","count":{"$sum":1}}},{"$sort":{"co
unt":-1}},{"$limit":5}])
{ "_id" : "Denver", "count" : 2676 }
{ "_id" : "Aurora", "count" : 474 }
{ "_id" : "Boulder", "count" : 397 }
{ "_id" : "Broomfield", "count" : 260 }
{ "_id" : "Parker", "count" : 245 }
```

This problem is relative. If we wanted to use only 'Denver' s our city for data analysis we can remove other cities from the mogodb dataset by the following command:
>db.denver.remove({"city":"XXX"})

3. Some Zip codes do not match the criteria:

```
{ "_id" : "80227", "count" : 2 }
{ "_id" : "80236", "count" : 1 }
{ "_id" : "80455", "count" : 1 }
{ "_id" : "80217", "count" : 1 }
{ "_id" : "80184", "count" : 1 }
{ "_id" : "CO 80247", "count" : 1 }
{ "_id" : "CO 80211", "count" : 1 }
{ "_id" : "80309", "count" : 1 }
{ "_id" : "CO 80439", "count" : 1 }
{ "_id" : "CO 80305", "count" : 1 }
{ "_id" : "Highlands Ranch,", "count" : 1 }
{ "_id" : "CO 80401", "count" : 1 }
{ "_id" : "CO80219", "count" : 1 }
```

This was resolved by the audit_zipcode method in audit.py. This method reduces postcodes to 5 digit strings.

Additional Ideas:

We can use our data to answer some basic/complex queries like:

1. Top buildings in Denver

```
  db.denver.aggregate([{'$match': {'building': {'$exists': 1}}}, {
'$group': {'_id': '$building','count': {'$sum': 1}}}, {'$sort': {'
count': -1}}, {'$limit': 10}])
{ "_id" : "yes", "count" : 119 }
{ "_id" : "residential", "count" : 29 }
{ "_id" : "school", "count" : 17 }
{ "_id" : "house", "count" : 12 }
{ "_id" : "dormitory", "count" : 11 }
{ "_id" : "commercial", "count" : 11 }
{ "_id" : "entrance", "count" : 9 }
{ "_id" : "retail", "count" : 8 }
{ "_id" : "apartments", "count" : 8 }
{ "_id" : "hut", "count" : 7 }
```

2. Top cuisines served in restaurants in Denver

```
  db.denver.aggregate([{"$match":{"amenity":{"$exists":1},"amenity
":"restaurant", "cuisine":{"$exists":1}}},{"$group":{"_id":"$cuisi
ne","count":{"$sum":1}}},{"$sort":{"count":-1}},{"$limit":5}])
{ "_id" : "american", "count" : 141 }
{ "_id" : "mexican", "count" : 137 }
{ "_id" : "pizza", "count" : 98 }
{ "_id" : "sandwich", "count" : 52 }
{ "_id" : "chinese", "count" : 44 }
```

Conclusion:
After the primary data exploration, I have found that the data is outdated, not serialized and needs a lot of work. But this report was a start in data wrangling and has provided an insight for what to expect out of a Data Scientist who is involved in wrangling.

For ideas to improve the data in the map, there can be a validation done by contacting the user who has added the information. Creating bots to write data can also be a valid suggestion. Machine Learning techniques can be used to identify zip codes/data that should not be a part of the dataset. The con to the above technique is that it would be difficult to validate every data point in the map. Creating bots can also lead to malicious data by fraudulent users. The challenge in Machine Learning would be the size of the dataset.