Problems encountered in the map:

After initially downloading the map of the South-West Delhi, India area, I noticed three main problems with the data, which I will discuss in the following order:

- **Inconsistent postal codes**:- In India, we have postal codes with length 6. However, I found out some of the nodes don't have their postcodes of length 6. I simply removed them from the data.
- **Incorrect postal codes**:- Zip codes all begin with "11" however a large portion of all documented zip codes were outside this region. I simply removed them from the data.
- **Inconsistent telephone number**:- Phone numbers in the data had multiple forms.I corrected them using python phone library provided by google.
- Over-specified city names:- Some of the city names were not consistent. They had multiple values. I fixed them by just splitting them on a "," and then replacing all the city names with the first name in the split array.

You can find my code and more explanation in the ipython notebook provided.

Data Overview:

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

File Sizes:-
map.osm ……….. 292 mb
cleaned_phone.json ……….. 340.2 mb

No. of documents:- 1574787
query used = db.map.find().count()

Number of unique users:- 993
query used = db.map.distinct('created.uid')

Number of different node types:

- hotel : 6
- school: 30
- apartments: 56
- hospitals: 8

```
query used = db.map.aggregate([
        {“$match":{"building":{"$exists":1}}},
        {“$group":{"_id":"$building","total":{"$sum":1}}}
        ])
```

Number of nodes : - 1308525.0
Number of ways : - 266235.0

```
query used = db.map.aggregate([
                                        {"$match":{"type":{"$exists":1}}},
                                        {"$group":{"_id":"$type","total":{"$sum":1}}}
                                ])
```

Top 1 contributing user: saikumar with posts = 101768

```
query used = db.map.aggregate([
                                        {"$group":{"_id":"$created.user", "count":
{"$sum":1}}},
                                        {"$sort":{"count":-1}},
                                        {"$limit":1}
                                ])
```

Biggest religion:-

hindu with count 75

```
query used = db.char.aggregate([
                {"$match":{"amenity":{"$exists":1}, “amenity":"place_of_worship"}},
                {"$group":{"_id":"$religion", "count":{"$sum":1}}},
                {"$sort":{"count":1}}, {“$limit":1}
        ])
```

Additional Ideas:

Updating street names to a more manageable way: The one problem I found out was that in the dataset the street names don't have a particular order. They included city names in them. I couldn't fix it because almost every other street name had a different city name and it was feasible, at least for me to fix it. So, the dataset could from their end include only street names and not some other redundant information. Also, in datasets for cities of other countries for example U.S., contain abbreviations for streets with church road, school road etc etc. This was not present in the dataset I wrangled.

Conclusion:

After this review of the data it's obvious that the South West Delhi area is incomplete, though I believe it has been well cleaned for the purposes of this exercise. I noticed that the openstreetmap uses GPS coordinated to map area. Hence, a poor GPS tracker in place can lead to superfluous results from the user side.

References used:

- https://github.com/jdamiani27/Data-Wrangling-with-MongoDB
- https://github.com/ziyanfeng/udacity-data-wrangling-mongodb