1. Problems Encountered in the Map:

After browsing through the xml version of the downloaded map, I found the following issues:

- Abbreviations and inconsistency in the street names
- Incorrect/Absence of actual city name in the address
- Inconsistent postal codes
- Inconsistent phone numbers

Abbreviations and Inconsistencies in the street name

A look at a few samples of elements in the xml data revealed that abbreviations like "rd" was being used for Road. Also, some of the words were in lower case. I updated all such strings to appropriate words.

Incorrect/Absence of actual city name in the address

I found that the city name was incorrectly tagged in some of the elements. For instance in one case the city name was "Marathalli" which is actually the name of an area in Bengaluru. Since, the map data downloaded is for Bengaluru I updated the city name to "Bengaluru" wherever it was missing.

Also, since the name of the city has been changed from "Bangalore" to "Bengaluru" recently I observed that both versions of the name have been used through out the dataset. Hence, I changed the city name from Bangalore to Bengaluru everywhere.

Inconsistent postal codes

Few of the postal codes had spaces in between. I stripped the spaces from such postal codes.

Inconsistent phone numbers

Few of the phone numbers had spaces or hyphens in between. I stripped the spaces and hyphens from such phone numbers. Also, in case of multiple phone numbers a few of them were seperated by comma where as the others were seperated by a semi colon. I replaced all the seperators with colon.

2. Data Overview

City – Bangalore
Name – bengaluru_india.osm
Xml uncompressed size: 678 MB
Note – I used only 25% of the above dataset as I was unable to run my code on the entire dataset
Final Dataset size – 171 MB
Final Json size – 198 MB
Number of documents - 952984 documents
Code:

```
mongoimport.exe --db test --collection bengaluru --file bengaluru_india.osm.json
```

Numbers and Types of elements:

```
pipeline = [
{"$group":{"_id":"$type","count":{"$sum":1}}}
]
result = db.bengaluru.aggregate(pipeline)
print "Types of elements:"
print list(result)
```

[{u'count': 180397, u'_id': u'way'}, {u'count': 772587, u'_id': u'node'}]

Way – 180397
Node - 772587

Number of unique users:
```
num_users = len(db.bengaluru.distinct("created.user"))
print "Number of unique users: ", num_users
```

Number of unique users: 857

Top 10 users:
```
pipeline = [
{"$group":{"_id":"$created.user","count":{"$sum":1}}},
{"$sort":{"count":-1}},
{"$limit":10}
]
result = db.bengaluru.aggregate(pipeline)
print "Top 10 users:"
print list(result)
```

Top 10 users:
[{u'count': 37987, u'_id': u'premkumar'}, {u'count': 32112, u'_id': u'saikumar'}, {u'count': 31319, u'_id': u'akhilsai'}, {u'count': 28698, u'_id': u'vamshikrishna'}, {u'count': 26216, u'_id': u'jasvinderkaur'}, {u'count': 25796, u'_id': u'shekarn'}, {u'count': 25277, u'_id': u'hareesh11'}, {u'count': 25150, u'_id': u'thrinath'}, {u'count': 24731, u'_id': u'praveeng'}, {u'count': 24220, u'_id': u'masthanvali'}]
Number of users who have update only once:
[{u'_id': 18906, u'num': 1}]

3. Additional Ideas

- Amenity based tagging: It can be observed from the data that out of the total set of documents around 951250 have not been assigned any amenity tag.
  Out of the rest of the nodes/ways which have assigned tags the number of essential services like fire station (4 tags) and police station (24 tags) have very few tags. Since these are essential services, a more essential amenities based approach has to be put in place.
- Many of the amenity tags are different though they form a similar group. For example, there are different tags like restaurants, cafe, confectionaries, etc which are actually subsets of a general tag "food". It would be very helpful if we add a generic amenity tag to the nodes which form a similar group.
- Postal codes: The postal codes are not available for all the elements. We can extract postal codes for a particular element based on its area by searching through the other elements which are from the same area.

Additional data exploration

Top 5 postal codes that have highest tags
```
pipeline = [
{"$match":{"address.postcode":{"$exists":1}}},
```

{"$group":{"_id":"$address.postcode","count":{"$sum":1}}},
{"$sort":{"count":-1}},
{"$limit": 5}
]
result = db.bengaluru.aggregate(pipeline)

[{u'count': 23, u'_id': u'560066'}, {u'count': 20, u'_id': u'560040'}, {u'count': 12, u'_id': u'560095'}, {u'count': 12, u'_id': u'560048'}, {u'count': 11, u'_id': u'560079'}]

Top 5 amenities

pipeline = [

{"$match":{"amenity":{"$exists":1}}},
{"$group":{"_id":"$amenity","count":{"$sum":1}}},
{"$sort":{"count":-1}} ,
{"$limit": 5}
]
result = db.bengaluru.aggregate(pipeline)
print "Types of elements:"
print list(result)

[{u'count': 257, u'_id': u'restaurant'}, {u'count': 184, u'_id': u'place_of_worship'}, {u'count': 144, u'_id': u'school'}, {u'count': 133, u'_id': u'bank'}, {u'count': 127, u'_id': u'atm'}]

Conclusion:

It can be observed that the data in the map for Bengaluru is incomplete as many of the elements have not been tagged with appropriate name/description. Though the data has been cleaned for inconsistencies and other inaccuracies there still remain other inaccuracies and inconsistencies like street name containing entire addressses or house number containing entire addresses. These can be fixed using NLP techniques.