1. **Problems with the data**
   After downloading the dataset for Mumbai, India I used the data.py script to create a json file. After that I used the command

   mongoimport --collection osm --file mumbai_india.osm.json --jsonArray

   to upload the JSON data into MongoDB. Then I ran several queries on MongoDB which highlighted various issues with the data, namely:

   a. I noticed a new address type, namely “Sector” in certain addresses. The naming of sectors was not consistent, for example there were sectors such as “Sector 50(E)" and there were others such as “Sector 35D”. To make this consistent I chose the syntax “Sector 35D” and hence Sector 50(E) became Sector 50E
   b. The other problem was related to how areas were marked as East or West. For e.g. "Walawalkar Marg, Andheri(West)," has “West” in parenthesis and was changed to "Walawalkar Marg, Andheri West",
   c. Another problem was revealed by scrutinizing the postcodes via the query ***db.osm.distinct('address.postcode')***. Many of the postal codes contain a either a trailing space or a space in the middle. This problem was fixed by replacing the spaces with an empty string.

2. General Comments on the Data

a. In general the data looks very fragmented and incomplete. For example 582,830 records don't have an amenity associated and 585,083 don't have a religion field associated.

b. The top 5 amenities as revealed by the query

db.osm.aggregate({'$match': {}}, {'$group': {'_id': {'amenity': '$amenity'}, 'count': {'$sum': 1}}}, {'$sort': {'count': -1}})

are as follows:

i. { "_id" : { "amenity" : "place_of_worship" }, "count" : 354 }
ii. { "_id" : { "amenity" : "restaurant" }, "count" : 251 }
iii. { "_id" : { "amenity" : "school" }, "count" : 235 }
iv. { "_id" : { "amenity" : "bank" }, "count" : 204 }
v. { "_id" : { "amenity" : "hospital" }, "count" : 155 }

c. The top 5 religions as revealed by the query

db.osm.aggregate({'$match': {}}, {'$group': {'_id': {religion: '$religion}, 'count': {'$sum': 1}}}, {'$sort': {'count': -1}})
are as follows

i. *{ "_id" : { "religion" : "hindu" }, "count" : 128 }*
ii. *{ "_id" : { "religion" : "muslim" }, "count" : 95 }*
iii. *{ "_id" : { "religion" : "christian" }, "count" : 55 }*
iv. *{ "_id" : { "religion" : "zoroastrian" }, "count" : 9 }*
v. *{ "_id" : { "religion" : "jain" }, "count" : 8 }*

d. It was interesting to find that the data went back to 2008 and has been last updated as recently as Feb of 2015. The query used to determine this is:

db.osm.aggregate({'$match': {}}, {'$project': {'created.timestamp': 1}}, {'$sort': {'created.timestamp': 1}}, {'$limit': 100}).pretty()

3. **Additional Ideas**

    a. The stats command was run on the osm collection:

```
> db.osm.stats()
{
        "ns" : "test.osm",
        "count" : 585402,
        "size" : 165099360,
        "avgObjSize" : 282,
        "storageSize" : 243314688,
        "numExtents" : 13,
        "nindexes" : 1,
        "lastExtentSize" : 68579328,
        "paddingFactor" : 1,
        "systemFlags" : 1,
        "userFlags" : 1,
        "totalIndexSize" : 19009200,
        "indexSizes" : {
                "_id_" : 19009200
        },
        "ok" : 1
}
```

The results revealed that the size of the data in the collection is 165 MB. The raw JSON file on the file system is 160 MB:

-rw-r--r--+ 1 sidazad staff 160354660 Mar 9 23:46 mumbai_india.osm.json

Hence the raw JSON size is pretty close to the size of the data inside MongoDB, which makes a lot of sense since MongoDB stores BSON.

The total number of records are 585,402 with an average size of 282 Bytes. A default index exists on the _id field but there are no other indexes, since we haven't made any.

b. The query *db.osm.distinct('created.user')* revealed that there are 864 distinct users who have made contributions to the data. This doesn't sound like a lot of users.
The query

db.osm.aggregate({'$match': {}}, {'$group': {'_id': {'user': '$created.user'}, 'count': {'$sum': 1}}}, {'$sort': {'count': -1}})

also reveals that the top 12-15 users account for most of the records (which was surprising to me).

References

1. http://docs.mongodb.org/manual/
2. http://docs.mongodb.org/manual/reference/method/cursor.sort/
3. http://stackoverflow.com/questions/14109474/mongodb-mongoimport-too-large-failure-parsing-errors
4. https://github.com/sidazad/ezmongo