Project OpenStreetMap: Submission

1 Overview of the Data

Map area: Kolkata, India

https://mapzen.com/data/metro-extracts/your-extracts/bfc06038eae9

I have decided to work on the OSM data for the metro city 'Kolkata' (India) since I grew up in the vicinity of this city and I am intimately familiar with the geography and culture of this area.

- Downloaded OSM data for Kolkata:
  `Kolkata.osm: 592 MB`
- Extracted sample OSM data out of the original data set and used for the development of the scripts/infrastructure:
  `Kolkata_sample.osm: 10 MB`
- Some statistics about the original dataset (Kolkata.osm) (extracted later from the SQL tables):
  - `Number of unique users: 321`
  - `Number of nodes: 2745253`
  - `Number of ways: 594989`
  - `Number of unique postcodes: 61`

Other interesting information about this city extracted from the dataset will be discussed later in this document.

2 Problems Encountered in the Map

2.1 Inconsistent street types

- Many street elemenets ('k' attribute value of tag element "addr:street") have value that does not represent only street name, but contains the full address or additional information. E.g. the value indicated the entire address, instead of only the street, like:

```
    Ward no 32, Near Mandirtala Bus Stop, Shibpur Rd,
Shibpur, Howrah, West Bengal 711102
    Ballygunge Gardens
    Street Number 70
```

- Found typographical mistakes in the street name:
  `D.r A.k paul raod`

- There are also non-english characters in the street name.
  `Dharmatala lane (ধর্মতলা লেন)`

2.2 Inconsistent postcode

'postcodes' are usually 6-digit numbers. But some of them are entered as a 5-digit or 7-digit numbers. Some of them have spaces or non-printable characters in them.

```
u'711106\u200e'
70014
7000026
700 027
```

2.3 Non-uniform City Names

City names are not uniformly entered. E.g.

```
Kolkata
Kolkata
KOLKATA
```

3 Audit the data

Refer to the function: '`audit_osm`' (File: `audit_osm.py`).

The street types are audited. The 'unexpected' street type are printed. Similarly, the postal codes and the City names are also printed to find out the inconsistencies.

4 Process the data

Refer to the function: '`process_map`' (File: `process_map.py`)

The 'process_map' function looks for the 'nodes' and 'way' top level elements and subsequently processes their sub-elements, to create the necessary data structures. During this process some parts of the data have been programmatically cleaned. Finally five .csv files have been produced to create a SQL database.

4.1 Programatically Cleaning up the data

I do not have a solution about how to fix the street name where the entire address (with street name, city and postcode have been entered). Maybe, this element can be further split with the knowledge of the city, postcode in this region. However I removed the inconsistencies in the street types. The approach I adopted to update the street types is as follows:

```
mapped_name = {'street':'Street', 'st.':'Street', 'st':'Street',
    'road':'Road', 'rd':'Road', 'rd.':'Road', 'raod':'Road',
    'sarani':'Sarani', 'pally':'Pally',
    'avenue':'Avenue', 'ave':'Avenue', 'ave.':'Avenue',
    'av':'Avenue', 'av.':'Avenue',
    'lane':'Lane', 'ln':'Lane', 'ln.':'Lane',
    'park':'Park', 'pk':'Park', 'pk.':'Park','row':'Row'}

def update_street_name(street_name):
    words = street_name.split(" ")
    for i in range(len(words)):
        if words[i].lower() in mapped_name:
            words[i] = mapped_name[words[i].lower()]
    updated_street_name = " ".join(words)
    return updated_street_name
```

For making the city names uniform I adopted an approach based on regular expression.

```
re_kolkata = re.compile(r'[kK][oO][lL][kK][aA][tT][aA]')

m = re_kolkata.search(tag_of_node_attrib_val)
if m is not None:
    tags_dict['value'] = 'Kolkata'
else:
    tags_dict['value'] = tag_of_node_attrib_val
```

The postcodes containing space and un-printable characters have been fixed through regular expression substitution to make sure they are integer values.

```
raw_postcode = tag_elem.attrib['v']
postcode = re.sub(r'[^0-9]', '', raw_postcode)
```

5 SQL Database Creation and Analysis

The five .csv files generated in the previous step are imported in sqlite3 to create the SQL database. At this point we are ready for some analysis.

5.1 Find all the cities in the Kolkata map area:

It appears that the DB contains cities near Kolkata. However, there is a city which is far away from Kolkata:

```
select tags.value, count (*) as cnt from
      (select * from nodes_tags union all
       select * from ways_tags) tags
where tags.key='city' group by tags.value order by cnt desc;
```

```
Kolkata|604
Salt Lake (Bidhan Nagar)|373
Saltlake (Bidhannagar)|46
Howrah|12
Barasat|2
Baruipara|2
Rajarhat|2
Bidhannagar|1
City|1            <-------- This is not a city name
Murshidabad|1     <-------- Not in the neighborhood of Kolkata
Salt Lake|1
```

5.2 Universities in Kolkata:

```
select T1.value from
   (select * from nodes_tags union all select * from ways_tags) as T0
       left join
   (select * from nodes_tags union all select * from ways_tags) as T1
       where
           T0.id = T1.id and
           T0.key='amenity' and T0.value='university' and
           T1.key='name';
```

Shows the names of the universities:

```
West Bengal University of Technology
Maulana Abul Kalam Azad Institute of Asian Studies
International School of Business & Media
Rabindra Bharati University
North Bengal University
Burdwan University - Kolkata Office
NITTTR Residential Campus
Aliah University
Jadavpur Univeristy
Calcutta Institute of Engineering and Management
University of Calcutta, Ballygunge Science College Campus
Calcutta University Alipore Campus
```

```
Ramakrishna Mission Vivekananda university
Indian Institute of Science, Education and Research
Administrative Training Institute
IIT Kharagpur, Kolkata Campus
Calcutta University Technology Campus
All India Institute of Hygiene and Public Health
SN Bose National Centre for Basic Schiences
Jadavpur University
West Bengal National University of Juridical Sciences
IISD
University of Calcutta, Rajabazar Science College Campus
Regent
Jute Agricultural Research Centre
Serampore College
Rabindra Bharati University
University of Calcutta, Barrackpore Trunk Road Campus
University of Calcutta, College Street Campus
University of Calcutta, Law College Campus
```

5.3 Places of worship based on religion:

```
select T1.value, count(*) as cnt from
   (select * from nodes_tags union all select * from ways_tags) as T0
         left join
   (select * from nodes_tags union all select * from ways_tags) as T1
         where
             T0.id = T1.id and
             T0.key='amenity' and T0.value='place_of_worship' and
             T1.key='religion'
         group by T1.value order by cnt desc;
```

```
hindu|15
christian|14
muslim|6
jewish|2
Irrespective of religion|1
buddhist|1
sikh|1
```

Hindus are the majority population in Kolkata. Interestingly, the data shows that the number of hindu places of worship is not proportionally as high as the number of people living here. This is probably because the preferred way of practicing this religion is not through places of worship for this group of people.

5.4 List of Hindu temples:

```
select T1.value, T2.value from
    (select * from nodes_tags union all select * from ways_tags) as T0
```

```
        left join
    (select * from nodes_tags union all select * from ways_tags) as T1
        left join
    (select * from nodes_tags union all select * from ways_tags) as T2
        where
           T0.id = T1.id and T1.id = T2.id and
           T0.key='amenity' and T0.value='place_of_worship' and
           T1.key='religion' and T2.key='name' and T1.value = 'hindu';
```

```
hindu|Mahanirban Math
hindu|Kali Temple
hindu|Birla Mandir
hindu|Radha Govinda Mandir
hindu|23 Palli Durga Temple
hindu|Lokenath Temple
hindu|Ram Krishna Temple, Charigram
hindu|Shitala Kali Mandir
hindu|Dakshineswar kali Temple
hindu|Annapurna Mandir
hindu|Baro Mandir
hindu|Roy Bari Mandir
hindu|Dharmaraj Mandir
```

5.5 Unique postcodes

```
select count (*) from (select distinct(value) from
         (select * from nodes_tags union all select * from ways_tags)
       where key = 'postcode');
```

```
61
```

5.6 Top 10 postcodes available in the database

```
select value, count (*) as cnt from
    (select value from
         (select * from nodes_tags union all select * from ways_tags)
     where key = 'postcode')
group by value order by cnt desc limit 10;
```

```
700064,911
700107,20
700019,15
711102,12
700027,10
700020,8
700015,7
700016,7
700156,7
700071,5
```

5.7 Number of unique contributors to the map

```
select count(*) from (select distinct(user) from nodes);
```

```
321
```

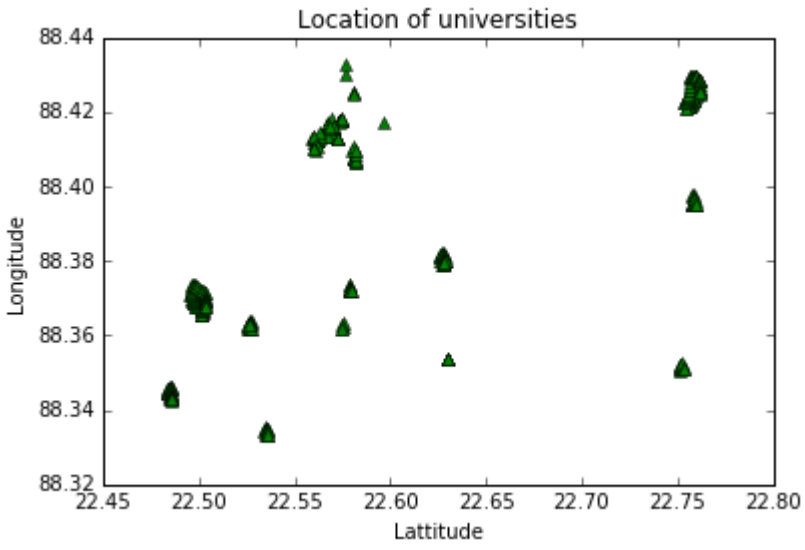
5.8 Top 10 contributors

```
select user, count(*) as cnt from nodes group by user order by cnt desc limit 10;
```

```
Rondon237,144302
saikumar,103565
samuelmj,100049
Apreethi,95584
harisha,73832
venugopal009,73169
shiva05,70687
ravikumar1,69893
hareesh11,68166
harishvarma,62376
```

6 Data Visulization

The python script '`visualize_Kolkata_universities.py`' has been written to plot the location of the universities in Kolkata.



7 Data Size

```
Kolkata.osm: 592 MB
Kolkata.db: 324 MB
nodes.csv: 227 MB
nodes_tags.csv: 259 KB
ways.csv: 36 MB
ways_nodes.csv: 82 MB
ways_tags.csv: 20 MB
```

8 Conclusion and Idea about Data Quality Improvement

This is a very large database with a very large number of contributor posts. Several discrepencies and inconsistencies have been noticed in the data. It may be helpful to have an automated audit program to review the code before committing in the OSM repository. The automated checker program can not only point out the wrong entry, but can also give suggestion of the right usage. I would use an audit program like 'audit_osm'. Besides street type, an enhanced version of the program would flag useful statistics like:

- Number of nodes
- Number of ways
- Number of unique postcodes
- Number of unique cities

A GUI or OSM editor can also be created to initially show all the data in excel. This could highlight the obvious outliers allowing the data entry person to edit and save the data in a modified OSM file. In order to highlight the outliers, the tool would accept a list of acceptable values from the user. The GUI could also provide tool tips to indicate the correct format for all the entries.

Although the improvement achieved by the above-mentioned tool is a welcome idea for flagging outliers, it will require context -sensitive input from the user. E.g. in the case of Kolkata metro, the user is expected to be familiar with this place and should be able to say that the expected city names are 'Kolkata', 'Salt Lake, 'Howrah', 'Barasat', etc.

Revision History

Revision	***Date***	***Description***	
1.0	10/24/2016	Initial Submission report	AD
2.0	11/05/2016	Incorporated review feedback	