

Open Street Map Project

Data Wrangling Process:

Does the project document the challenges encountered during the wrangling?

1. The code ran initially with a small sample of the dataset to identify initial problems. An audit was conducted for:
 - a. The list of highest level tags in the dataset
 - b. For the attribute, addr:street for street name
 - c. For the attributes, postal_code and addr:postcode for zip codes

Problems encountered in the dataset:

Street Names:

- a. Street names in the dataset are abbreviated such as St or St. for Street, Pl. for Place and Pkwy for Parkway etc.
- b. Some of the street names also have complete addresses such as “Bromfield Street #501” or “Franklin Street, Suite 1702”
- c. Audit result of street names is given below:

```
Cambridge : 2 Holland : 1
Center : 15 Hwy : 1
Circle : 7 Jamaicaaway : 2
Corner : 1 Lafayette : 1
#12 : 1 Court : 9 Lane : 27
#1302 : 1 Ct : 8 LEVEL : 1
#501 : 1 Dartmouth : 1 Longwood : 1
104 : 1 Dr : 1 Mall : 3 Square : 68
1100 : 1 Drive : 77 Market : 1 ST : 1
1702 : 1 Driveway : 1 Newbury : 2 St : 252
3 : 1 Elm : 1 Park : 52 st : 1
303 : 1 Ext : 1 Parkway : 27 St, : 1
6 : 1 Fellsway : 15 Pasteur : 3 St. : 42
846028 : 1 Fenway : 5 Pkwy : 10 street : 2
Albany : 3 floor : 2 Pl : 1 Street : 4139
Artery : 1 Floor : 1 place : 1 Street. : 1
Ave : 163 Garage : 1 Place : 83 Terrace : 17
Ave. : 21 Greenway : 2 Plaza : 2 Turnpike : 2
Avenue : 850 H : 1 Rd : 29 Way : 25
Boulevard : 12 Hall : 1 rd. : 1 Wharf : 5
Boylston : 2 Hampshire : 1 Road : 248 Windsor : 2
Broadway : 98 Highway : 13 Row : 6 Winsor : 1
Brook : 1 Highway : 1 South : 2 Yard : 1
Building : 1
```

Note: Some of the names of nodes such as train or bus stops have abbreviated street names such as “Harvard St @ Beacon St”. However, these are not street names and therefore, are not tagged by “addr:street”

Pin Code:

- a. Zip codes in Boston are 5-digit all numeric codes. However, some of the locations also had Zip+4 codes which are 5 digit codes followed by a “-” followed by 4 digits that determine a specific location within the zip code
- b. Some of the zip codes also have incorrect entries such as “MA” for Massachusetts
- c. Sample of the audit for pin codes is given below:

```
02128 : 29
02129 : 13
02130 : 185
02130-4803 : 1
02131 : 64
02131-3025 : 2
02131-4931 : 1
02132 : 137
02132-1239 : 1
02132-3226 : 1
02134 : 66
```

Is data cleaned programmatically?

2. The problems encountered in street names were corrected for the sample dataset as well as for the complete dataset. A mapping dictionary was created as shown below:

```
mapping = {'Ave': 'Avenue',
          'Ave.': 'Avenue',
          'Ct': 'Court',
          'Dr': 'Driveway',
          'Drive': 'Driveway',
          'floor': 'Floor',
          'H': 'Hall',
          'Hwy': 'Highway',
          'Hwy.': 'Highway',
          'Highway': 'Highway',
          'Park': 'Parkway',
          'Pkwy': 'Parkway',
          'Pkwy.': 'Parkway',
          'Pl': 'Place',
          'place': 'Place',
          'Rd': 'Road',
          'Rd.': 'Road',
          'rd.': 'Road',
          'Sq.': 'Square',
          'St': 'Street',
          'St.': 'Street',
          'ST': 'Street',
          'St.': 'Street',
          'st': 'Street',
          'street': 'Street',
          'Street.': 'Street',
          'Windsor': 'Windsor'
}
```

Based on the mapping dictionary, if the street name needs correction, the last part of the street name was replaced with the corresponding entry from the mapping dictionary. For those streets which do not belong to the mapping but still contain errors such as “Bromfield Street #501”, a separate set of commands were used to split the street name at the special character and just the street name was captured. Such cases were very few even in the whole dataset. The code for the same is given below:

```
def update_name(street_name, mapping):
    # Split the street name into parts and replace the last part based on the mapping
    x = street_name.split(" ")
    old = x[len(x)-1]
    try:
        street_name = street_name.replace(old, mapping[old])
        return street_name
    # The below case is used to fix errors in street names
    # where they're of the form ABC Street #501 or ABC Street, 501
    except KeyError:
        y = re.split(',#', street_name)
        return y[0]
```

The problems encountered with the postal code were corrected as follows:

1. In case the postal code contained non numeric entries, then the code skipped that entry and moved on the next as shown below:


```
# If the postal code is not in Zip/Zip+4 format ignore the entry, else modify it
elif tag.attrib['k'] == 'addr:postcode':
    if tag.attrib['v'] not in postal_types:
        continue
    else:
        tags1['value'] = update_postal_code(tag.attrib['v'])
```

2. In case the postal code was of the zip+4 format, then the first 5 digits were captured as the pin code. This is shown below:

```
# To adjust the postal code into standard 5 digit format
def update_postal_code(postal_code):
    postal_code = postal_code.split("-")[0]
    return postal_code
```

Overview of the data

Is the OSM XML large enough?

	Name	Date modified	Type	Size
3.	 boston_machusetts.osm	7/18/2017 3:18 PM	OSM File	426,185 KB

Are overview statistics of the dataset computed?

4.
 - a) File Sizes

Name	Date modified	Type	Size
osmproject.db	7/27/2017 4:37 PM	Data Base File	249,773 KB
nodes.csv	7/27/2017 4:22 PM	CSV File	156,225 KB
ways_nodes.csv	7/27/2017 4:22 PM	CSV File	53,952 KB
ways_tags.csv	7/27/2017 4:22 PM	CSV File	22,303 KB
ways.csv	7/27/2017 4:22 PM	CSV File	20,518 KB
nodes_tags.csv	7/27/2017 4:22 PM	CSV File	17,147 KB

b) Number of nodes and ways

```
sqlite> select count(*) from nodes;
1940888
sqlite> select count(*) from ways;
310423
```

c) Number of unique users

```
sqlite> select sum(total) from (select count(distinct uid) as total from nodes union select count(distinct uid) from ways as total);
2091
```

d) Total number of libraries and number with wheelchairs

```
sqlite> select value, count(*) as total_lib from nodes_tags where key = "amenity" and value = "library" union
select a.value, count(*) as total_wheelchair from nodes_tags as a, (select id from nodes_tags where value='library') as b on a.id=b.id where a.key='wheelchair' group by a.value order by total_wheelchair desc;
library|276
yes|8
limited|2
```

e) Top Amenities in Boston

```
sqlite> select value, count(*) as total from nodes_tags where key = "amenity" group by value order by total desc limit 20;
bench|1071
restaurant|680
school|499
bicycle_parking|319
library|276
place_of_worship|275
cafe|271
fast_food|196
bicycle_rental|139
post_box|124
waste_basket|111
parking|91
fire_station|90
bank|80
bar|70
fountain|67
fuel|65
pub|63
atm|59
toilets|56
```

f) Popular Cuisines (Indian is quite popular)

```
sqlite> select value, count(*) as total from nodes_tags where key = "cuisine" group by value order by total desc limit 10;
coffee_shop|76
pizza|73
sandwich|51
mexican|46
american|45
italian|38
chinese|35
burger|30
indian|22
donut|21
```

Are ideas for additional improvements included?

5. Scope for improvement:

- a) One issue with the data is that there are multiple entries for a given data point oftentimes by the same user. For example, consider the query for Boston University below:

```
sqlite> select a.lat, a.lon, a.user, b.key, b.value from nodes as a, (select * from nodes_tags where key = "name" and value = "Boston University") as b where a.id = b.id;
42.3339865|-71.1047742|iandees|name|Boston University
42.3501469|-71.0911614|ryebread|name|Boston University
42.3503332|-71.0905927|ryebread|name|Boston University
42.3506625|-71.0910524|ryebread|name|Boston University
42.350867|-71.0903217|ryebread|name|Boston University
```

Furthermore, this data has no additional information beyond the name for multiple some node IDs. If such duplications can be accounted for by a combination of the latitude, longitude, name and key, the data may possibly be significantly reduced and speed up processing and further analysis.

- b) The number of users contributing to the dataset is very concentrated with 85% of nodes and ways being contributed only by 5 users. As a result of this, several data points may be incomplete, inaccurate or missing. Encouraging users to contribute more to the dataset would be helpful. This way, there is a greater possibility of identifying nodes on the map accurately as multiple users refer to it and there is also the possibility of capturing several more nodes that have not yet been identified. The query below gives an example of how concentrated the user base is.

```
sqlite> select count(*) as overall from ways union select sum(entries) as top5 from (select user, count(*) as entries from ways group by user order by count(*) desc limit 5);
265101
310423
sqlite> select count(*) as overall from nodes union select sum(entries) as top5 from (select user, count(*) as entries from nodes group by user order by count(*) desc limit 5);
1629659
1940888
```

Are benefits and problems with additional improvements discussed?

6. The benefit of more users contributing to the dataset is:

- a) More accurate data as multiple users refer to the same node
b) More detail in the data. For example, while earlier users may refer to a node simply by the name of the place and type, with multiple users contributing, additional details like address, additional details like whether it is wheelchair friendly etc. can be captured

- c) Missing data – Several data points that are missing right now can be captured with contributions from additional users

The problems with this data improvement:

- a) Cleaning of the data will become a challenge as highlighted earlier – additional steps need to be performed to capture the most details of any given location
- b) The size of the dataset will increase resulting in greater processing time