OpenStreetMap Data Case Study

Map Area

Oahu Hawaii, USA

http://www.openstreetmap.org/relation/3489649#map=10/21.4869/-157.9655

The osm file (oahu_data.osm) used for this project has 51.3 MB.

Problems encountered in the Map

After downloading the map and running the file, I noticed the following problems with the data, which were corrected:

- Inconsistent state names: “Hi”, “hi”, “HI”
- Inconsistent postal codes: ‘HI 96819"
- Inconsistent city names: “honolulu”, “Honlulu”
- Abbreviated street names: 'Kalakaua Ave', 'Ala Moana Blvd’, 'Kipapa Dr'

Abbreviated Street Names:

The code:

```
expected = ["Street", "Avenue", "Boulevard", "Drive", "Court", "Place", "Square", "Lane", "Road",
        "Trail", "Parkway", "Commons", "Circle", "Highway", "Way", "Loop", "Walk", "King"]
mapping = { "St": "Street",
        "St.": "Street",
        "Ave" : "Avenue",
        "Ave." : "Avenue",
        "Rd" : "Road",
        "Rd." : "Road",
        "Blvd" : "Boulevard",
        "Blvd." : "Boulevard",
        "Cir" : "Circle",
        "Cir." : "Circle",
        "Ct" : "Court",
        "Ct." : "Court",
        "Dr" : "Drive",
        "Dr." : "Drive",
        "Pl" : "Place",
        "Pl." : "Place",
        "Hwy" : "Highway",
            "Pkwy" : "Parkway",
            "highway": "Highway",
            "king" : "King"
        }


def audit_street_type(street_types, street_name):
    m = street_type_re.search(street_name)
    if m:
        street_type = m.group()
        if street_type not in expected:
            street_types[street_type].add(street_name)

#finding the street name
def is_street_name(elem):
    return (elem.attrib['k'] == "addr:street")


#updating the name
def update_name(name, mapping):
    m = street_type_re.search(name)
    if m:
        street_type = m.group()
        if street_type in mapping.keys():
            if m not in expected:
                name = re.sub(m.group(), mapping[m.group()], name)
    return name
```

The results:

```
{'106': set(['Pualei Cir, Apt 106']),
 'Ave': set(['Kalakaua Ave']),
 'Blvd': set(['Ala Moana Blvd']),
 'Center': set(['Enchanted Lakes Shopping Center']),
 'Dr': set(['Kipapa Dr']),
 'Honolulu': set(['Moanalua, Honolulu']),
 'Hwy': set(['Kamehameha Hwy']),
 'Mall': set(['Fort Street Mall', 'McCarthy Mall']),
 'Momi': set(['Pali Momi']),
 'Pkwy': set(['Meheula Pkwy']),
 'St': set(['Ala Pumalu St', 'Lusitania St']),
 'St.': set(['Lusitania St.']),
 'Terrace': set(['Round Top Terrace']),
 'highway': set(['kanehameha highway']),
 'king': set(['king'])}
king => King
Enchanted Lakes Shopping Center => Enchanted Lakes Shopping Center
Pualei Cir, Apt 106 => Pualei Cir, Apt 106
Lusitania St. => Lusitania Street
Pali Momi => Pali Momi
Meheula Pkwy => Meheula Parkway
Fort Street Mall => Fort Street Mall
McCarthy Mall => McCarthy Mall
Round Top Terrace => Round Top Terrace
Moanalua, Honolulu => Moanalua, Honolulu
Kamehameha Hwy => Kamehameha Highway
Ala Moana Blvd => Ala Moana Boulevard
Kalakaua Ave => Kalakaua Avenue
Ala Pumalu St => Ala Pumalu Street
Lusitania St => Lusitania Street
Kipapa Dr => Kipapa Drive
kanehameha highway => kanehameha Highway
```

Inconsistent state names:

The code:

```
#updating the variable

mapping = {

            "HI": "Hawaii",
            "hi": "Hawaii",
            "Hi": "Hawaii"
        }




#finding the state name
def is_state_name(elem):
    return (elem.attrib['k'] == "addr:state")
```

The results:

```
{'HI': set(['HI']), 'Hi': set(['Hi']), 'hi': set(['hi'])}
hi => Hawaii
HI => Hawaii
Hi => Hawaii
```

Inconsistent postal codes:

The code:

```
# finding the postal codes
def get_postcode(element):
    if (element.attrib['k'] == "addr:postcode"):
        postcode = element.attrib['v']
        return postcode


# updating the postal codes
def update_postal(postcode):

    for v in mapping:
        if postcode is None:
            continue
        else:
            if re.match(v,postcode):
                postcode = postcode.replace(v,mapping[v])
    return postcode
```

The results:

```
{' 96819': 1,
 '96701': 4,
 '96707': 4,
 '96712': 32,
 '96712-9998': 1,
 '96717': 1,
 '96734': 10,
 '96734-9998': 1,
 '96744': 3,
 '96753': 1,
 '96762': 1,
 '96782': 2,
 '96786': 9,
 '96789': 4,
 '96791': 2,
 '96792': 2,
 '96795': 1,
 '96797': 9,
 '96813': 49,
 '96814': 78,
 '96815': 258,
 '96815-2518': 1,
 '96815-2830': 1,
 '96815-2834': 2,
 '96816': 8,
 '96817': 6,
 '96818': 10,
 '96819': 6,
 '96821': 1,
 '96822': 72,
 '96825': 3,
 '96825-9998': 1,
 '96826': 75,
 '96826-4427': 1}
```

Inconsistent city names:

The code

```
expected = ["Honolulu"]

#updating this variable
mapping = {

        "honolulu": "Honolulu",
        "Honlulu": "Honolulu"
     }
```

The results:

```
{'Aiea': set(['Aiea']),
 "Hale'iwa": set(["Hale'iwa"]),
 'Haleiwa': set(['Haleiwa']),
 u'Hale\u2018iwa': set([u'Hale\u2018iwa']),
 'Hauula': set(['Hauula']),
 'Honlulu': set(['Honlulu']),
 'Kailua': set(['Kailua']),
 'Kaneohe': set(['Kaneohe']),
 'Kapolei': set(['Kapolei']),
 'Laie': set(['Laie']),
 'Mililani': set(['Mililani']),
 'Wahiawa': set(['Wahiawa']),
 'Waialua': set(['Waialua']),
 'Waimanalo': set(['Waimanalo']),
 'Waipahu': set(['Waipahu']),
 'honolulu': set(['honolulu'])}
Hauula => Hauula
Hale‘iwa => Hale‘iwa
Waipahu => Waipahu
Kaneohe => Kaneohe
Kapolei => Kapolei
Waimanalo => Waimanalo
Hale'iwa => Hale'iwa
Wahiawa => Wahiawa
Haleiwa => Haleiwa
Laie => Laie
Aiea => Aiea
Waialua => Waialua
honolulu => Honolulu
Kailua => Kailua
Honlulu => Honolulu
Mililani => Mililani
```

Exploring the Dataset

In this section, after importing the data in SQL, I explore the dataset by running some queries:

Number of nodes

```
sqlite> SELECT count(*) FROM nodes;

count(*)
234730
```

Number of ways

```
sqlite> SELECT count(*) FROM ways;

count(*)
25061
```

Number of unique users

```
sqlite> SELECT COUNT(DISTINCT uid) AS Unique_users
FROM nodes;

Unique_users
392
```

Top 10 contributing users

```
sqlite> SELECT user, count(*) AS num
FROM ways
GROUP BY user
ORDER BY num DESC
LIMIT 10;

user,num
Tom_Holland,6295
bot-mode,2567
cbbaze,1816
neuhausr,1343
kr4z33,1292
ikiya,957
bdiscoe,836
aaront,627
beweta,542
tiger,490
```

Number of towers on the island:

```
SELECT value, count(*) AS num
FROM nodes_tags
WHERE type = 'tower'
GROUP BY value
ORDER BY num DESC;

communication|21
lighting|8
observation|1
radar|1
```

Top 10 postal codes:

```
SELECT tags.value, count(*) as count FROM (SELECT * FROM nodes_tags UNION
ALL SELECT * from ways_tags) tags WHERE tags.key = 'postcode' GROUP BY
tags.value ORDER BY count DESC LIMIT 10;

value,count
96815,265
96814,80
96826,79
96822,73
96813,52
```

```
96712,33
96818,12
96734,10
96786,9
96797,9
```

Number of amenities:

```
SELECT value, count(*) as num
FROM nodes_tags
WHERE key ='amenity'
GROUP BY value
ORDER BY num DESC;

cafe|1
```

By exploring the data, I observed that in order to find more local amenities is not possible by using “amenity” as the key. Instead I had to search using the key =’en’ and type = ‘name’:

```
sqlite> SELECT value FROM nodes_tags WHERE key = 'en' AND type = 'name';
Honolulu
Punaluu
Laie
Kaneohe
Hauula
Wheeler Army Airfield
Ko Olina
National Memorial Cemetery of the Pacific
Kaupo Cove
Kalaeloa Airport
World Mission Society Church of God (WMC) english.watv.org (Ahnsahnghong,
Passover, Elohim)
Rommy's Kafuku Prawns and Shrimp
Waikiki Brewing Company
Barefoot beach cafe
Teddy's Bigger Burger
Sea life Park Hawaii
Coconut Cafe
Rumfire
Alamo car rental
Dean & Deluca’s
Helena's Hawaiian Food
```

Conclusion

After reviewing of the data, I see that the Oahu area is incomplete. I cannot extract a list of landmarks, parks or museums, and the list of amenities is lacking. There is missing information, which can be fixed by contributors who can add data to the map to cover the missing information and to improve the existing one. Finding ways of attracting the local contributors or tourists, to easy add new data or to modify the existing one through their mobile devices, can do this. One way to contribute is using the OpenStreetView application.

I've tested the OpenStreetView iphone application with the purpose to add data by uploading photos taken in my local city Copenhagen, Denmark. For unknown reasons, the upload could not be completed although I've tried several times at different intervals of time with different photos sizes. This is a problem for potential contributors as they might lose confidence and interest. Having an application that work is essential for keeping the OpenStreetView map updated with enough data to make an accurate analysis.