

# OpenStreetMap Data Case Study

---

## Map Area

Syracuse, New York, United States

- [OpenStreetMap](#)
- [MapZen](#)

This map is of Syracuse New York, a town that I visited very often when attending RPI.

## Problems Encountered in the Map

---

After parsing and auditing dataset of Syracuse, New York, I have noticed couple problems with the data. I will discuss each problem in details below.

- Overabbreviated street name (*'James St'*)
- Inconsistant names for State Routes (*'New York 31'*, *'State Route 31'*, *'State Highway 31'*)
- Incorrect spelling of street name (*'Presidential Courts'*)
- Inconsistent zip codes (*"132059211"*, *"13206-2238"*)
- Incorrect zip codes (Syracuse's zip code starts with 132 but a large portion of the data is outside of the city boundaries)

## Street names and state highway names

I run an audit of the street names in the dataset and have found two main problems. There is one street named James Street that is overabbreviated as *James St.* And Presidential Court is misspelled as *Presidential Courts* Secondly, I have found state highway names to be quite inconsistent. After googling for the correct format of state highway name, I have decided to go with the google map way of naming them. So 'State Route 31' would be 'New York 31'. Below is a python function that I created to correct the problems.

```
def update_name(name):
    mapping = { "St": "Street",
                "Courts": "Court"
              }
    street_type_re = re.compile(r'\b\S+\'$', re.IGNORECASE)
    m = street_type_re.search(name)
    if m:
        street_name = mapping[m.group()]
        name = street_type_re.sub(street_name, name)
    elif name in ['New York 31', 'State Route 31', 'State Highway']:
        name = 'New York 31'
    elif name == 'State Route 298':
        name = 'New York 298'
    elif name == 'US Route 11':
        name = 'Route 11'
    return name
```

## Zip codes

I didn't find any zip code with letters or special characters. Although the data does include zip codes not necessarily in the city of Syracuse but all in close vicinity, I have decided to keep those zip-codes as they are in the metropolitan area of Syracuse. There are couple zip codes that are entered in ZIP+4 and I decided to covert them to standard 5 digits

format.

Here is a python function to clean up the zip codes.

```
def update_zip(zip_code):  
    if len(zip_code) == 5:  
        return zip_code  
    else:  
        return zip_code[0:5]
```

# Data Overview and Additional Ideas

---

## File sizes

syracuse_new-york.osm	.....	63.9 MB
syracuse_osm.db	.....	56.4 MB
nodes.csv	.....	23 MB
nodes_tags.csv	.....	1.4 MB
ways.csv	.....	2 MB
ways_tags.csv	.....	6 MB
ways_nodes.csv	.....	7.7 MB

## Number of nodes

```
SELECT COUNT(*)  
FROM nodes;
```

# Number of ways

```
SELECT COUNT(*)  
FROM ways;
```

35309

# Number of unique users

```
SELECT COUNT(DISTINCT(t.user))  
FROM (SELECT user FROM ways UNION ALL SELECT user FROM nodes) AS t
```

228

# Top 10 contributing users

```
SELECT t.user, COUNT(t.id) AS num  
FROM (SELECT user, id FROM ways UNION ALL SELECT user, id FROM nodes) AS t  
GROUP BY t.user  
ORDER BY num DESC  
LIMIT 10;
```


```
zeromap|158135  
woodpeck_fixbot|75599  
DTHG|27938  
yhahn|8130  
RussNelson|8077  
fx99|4496  
bot-mode|4417  
timr|2861
```

TIGERcn1|2077

Johnc|2035

## Top 10 zip-codes

```
SELECT t.value, count(*) AS num
FROM (SELECT value, key FROM ways_tags UNION ALL SELECT value, key
WHERE t.key = 'postcode'
GROUP BY t.value
ORDER BY num DESC
LIMIT 10;
```



```
13224,821
13214,513
13210,475
13205,282
13206,279
13108,171
13212,136
13057,114
13031,112
13066,110
```

## Number of users appearing only once

```
SELECT COUNT(*) FROM (SELECT t.user, COUNT(*) as num
FROM (SELECT user FROM ways UNION ALL SELECT user FROM nodes) AS t
GROUP BY t.user
HAVING num = 1) AS z;
```

# Additional Ideas

---

## Gas station brand with biggest presence

```
SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
      JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='fuel')
      ON nodes_tags.id=i.id
WHERE nodes_tags.key='brand'
GROUP BY nodes_tags.value
ORDER BY num DESS;
```

```
Sunoco,4
Mobil,3
Citgo,2
"Byrne Dairy",1
Costco,1
Gulf,1
"Kwik Fill",1
Speedway,1
Valero,1
```

## Top 5 most popular fast food brand

```
SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
      JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='fast_fo
      ON nodes_tags.id=i.id
WHERE nodes_tags.key='name'
GROUP BY nodes_tags.value
ORDER BY num DESC
```

```
LIMIT 5;
```

```
Subway,14  
McDonald's,5  
Dunkin' Donuts,4  
Pavone's Pizza,3  
Burger King,2
```

## Most popular artwork in Syracuse

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

```
sculpture,3  
statue,2
```

## Conclusion

Although this Syracuse dataset is relatively clean when compared to other metropolitan areas, I see many potential areas of improvement in terms of the consistence of tag names, tiger gps locations, as well as outdated data. For the purpose of this project, I think the dataset has been well cleaned.