

Open Street Map Project

Area Chosen: Gloucester County, NJ and surrounding areas

I chose Gloucester County, NJ and the surrounding areas for the open map street project because I'm from this area originally. My great grandmother lived to be one hundred years old and during her life she impacted the town in many ways.

<https://www.openstreetmap.org/relation/2560068#map=10/39.7635/-75.1630>

Project Steps:

1. Notes from the Audit

The audit.py file was used to audit the data. The biggest issue for consistency and uniformity was street abbreviations. The code to audit the street names was provided in the course. This code was altered to find issues in the state as well as the telephone numbers.

```
In [ ]: #Sample of function to audit phone numbers.
def is_website(elem):
    return(elem.tag == "tag") and (elem.attrib['k'] == "phone")
```

2. Notes from Parse and Gather Data

The street map was an XML file. It was parsed in a Jupyter notebook iteratively due to the large size. The shape function was updated to pull the data into dictionaries and lists that populated csv files after separating it based on nodes, ways and tags. They were imported into SQLITE3. Finally, the database was imported into Jupyter notebook for analysis.

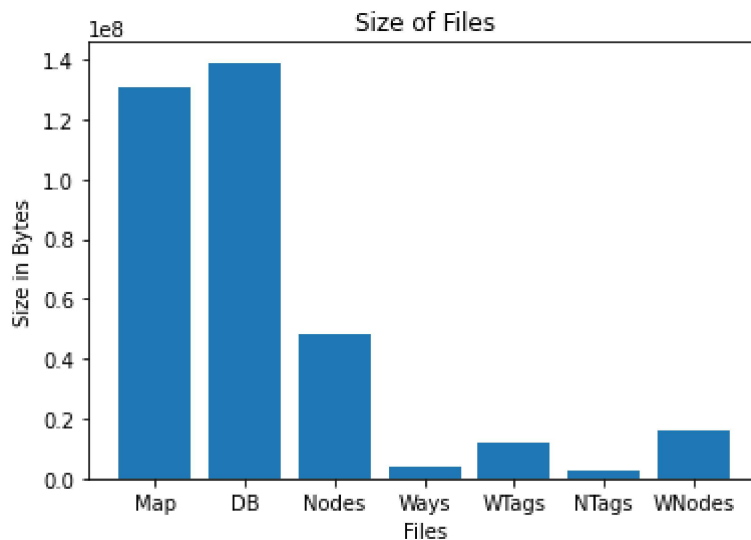
```
In [11]: print('Sizes of Files Used in Project')
print('Size of original file:', os.path.getsize('map_GC_NJ.xml'))
print('Size of database:', os.path.getsize('OPENSTREETMAP.db'))
print('Size of nodes csv file:', os.path.getsize('nodes.csv'))
print('Size of ways csv file:', os.path.getsize('ways.csv'))
print('Size of ways_tags csv file:', os.path.getsize('ways_tags.csv'))
print('Size of nodes_tags csv file:', os.path.getsize('nodes_tags.csv'))
print('Size of ways_nodes csv file:', os.path.getsize('ways_nodes.csv'))
```

```
Sizes of Files Used in Project
Size of original file: 130588843
Size of database: 138903552
Size of nodes csv file: 48020519
Size of ways csv file: 3802758
Size of ways_tags csv file: 11662143
Size of nodes_tags csv file: 2581592
Size of ways_nodes csv file: 16209415
```

```
In [57]: import matplotlib.pyplot as plt
import numpy as np
```

```
x = np.array(['Map', 'DB', 'Nodes', 'Ways', 'WTags', 'NTags', 'WNodes'])
y = np.array([130588843, 138903552, 48020519, 3802758, 11662143, 2581592, 16209415])

plt.title("Size of Files")
plt.xlabel('Files')
plt.ylabel('Size in Bytes')
plt.bar(x,y)
plt.show()
```



```
In [ ]: ##This is a sample of separating the tags at the colon. Such as addr:street will become
## Reference the main project to run the code.
        if LOWER_COLON.search(child.attrib['k']):
            fix_key = child.attrib['k']
            col_index = fix_key.find(':') + 1
            new_key = fix_key[col_index:].strip()
            node_dict['key'] = new_key
            node_dict['value'] = child.attrib['v']
            type_index = fix_key.find(':')
            new_type = fix_key[:type_index].strip()
            node_dict['type'] = new_type
```

Process and Clean

The cleaning function looked for certain tags and then updated the value if it found errors. The biggest opportunity was the street abbreviations. This was done by setting up a dictionary with the abbreviation as the key and the corrected state as a value. Then an exact match was found and updated the value turning it to the new dictionary. This corrected the data so that it was clean on the csv making our sql queries much easier and more accurate.

```
In [ ]: #Cleaning function. Please run data.py to see full function. This has been altered to s

def clean_up(k,v):
    if k == 'street':

        for abbv, name in {"Ave":"Avenue", "Rd":"Road", "Dr":"Drive", "Ter":"Terrace",
```

```

        v = re.sub(r"\b" + abbv + r"\b", name , v)
    return k, v

```

Learn how to Story, Query and Aggragate Data with SQL

The five csv files were imported into sqllite creating five tables in the database and some sample queries are below.

```

In [15]: import sqlite3
import pprint
import pandas as pd
import os

```

```

In [22]: # Create the connection
cnx = sqlite3.connect('C:/Users/knico/OneDrive/Documents/Python Scripts/Mongo/OPENSTREE

```

```

In [17]: cur = cnx.cursor()

```

```

In [18]: #Sample of the ways tags table
cur.execute(''Select * from ways_tags limit 5;'')
pprint.pprint(cur.fetchall())

```

```

[('11589484', 'zip_right', '08026', 'tiger'),
 ('11589484', 'zip_right', '08026', 'tiger'),
 ('11589484', 'zip_right', '08026', 'tiger'),
 ('11589484', 'zip_right', '08026', 'tiger'),
 ('11589484', 'zip_right', '08026', 'tiger')]

```

```

In [6]: ##All types from both tables.
cur.execute(''Select type
from nodes_tags union select type from ways_tags limit 5;'')
pprint.pprint(cur.fetchall())

```

```

[('addr',), ('building',), ('disused',), ('fire_hydrant',), ('name',)]

```

```

In [7]: #List the disctinct keys from the ways_tags and the count of each. Limited to 20 due to
cur.execute(''Select key, count(value) from ways_tags group by key order by count(val
print("Keys(Ways Tags), Count")
pprint.pprint(cur.fetchall())

```

```

Keys(Ways Tags), Count
[('source', 116138),
 ('zip_right', 44359),
 ('building', 23650),
 ('waterway', 18493),
 ('reviewed', 14350)]

```

Top Users

This section will show the top users in descending order with their changes.

```

In [8]: cur.execute(''Select user, count(changeset) from snodes group by uid having count(chan

```

```
pprint.pprint('User Name , Count of Changes')
pprint.pprint(cur.fetchall())
```

```
'User Name , Count of Changes'
[('AdamP511', 130889),
 ('dchiles', 88010),
 ('kylegiusti', 52876),
 ('woodpeck_fixbot', 32008),
 ('NJDataUploads', 29376),
 ('MSoslow', 25852)]
```

Problems Encountered in Map

I ran into multiple problems while working with data.

- Street Names: Some streets were routes, or uncommon names like Salina or Ferenzi or Corte di Verona. Those would need further research.
College Ave was corrected to College Avenue.
- Phone numbers had many formatting issues. Some started with +1, others had no area code. These were formatted to (xxx)xxx-xxxx. For example +1-856-881-6656 becomes (856)-881-6656. Some were much shorter than this so there was not a way to clean them without validation.
- Some of the websites were no longer valid. These could be audited with code but would need research to update.

In [10]:

```
# Corte di Verona is an example of one of the street names that didn't have an abbrevia
cur.execute('''Select * from nodes_tags where key= 'street' and value = 'Corte di Veron
pprint.pprint('Id, Key, Value, Type')
pprint.pprint(cur.fetchall())
```

```
'Id, Key, Value, Type'
[('7366606614', 'street', 'Corte di Verona', 'addr')]
```

In [38]:

```
cur.execute('''Select * from nodes_tags where (value like ('%Greenview%')) or value lik
pprint.pprint('Id, Key, Value, Type')
pprint.pprint(cur.fetchall())
```

```
'Id, Key, Value, Type'
[('5933348073', 'street', 'College Avenue', 'addr'),
 ('5933348073', 'phone', '(856)881-6656', 'regular')]
```

In [44]:

```
#ALL streets have been corrected to street.
cur.execute('''Select * from nodes_tags where value like ('% St%') and key='street' '''
pprint.pprint('Id, Key, Value, Type')
pprint.pprint(cur.fetchall())
```

```
'Id, Key, Value, Type'
[('314922316', 'street', 'Cooper Street', 'addr'),
 ('1589823283', 'street', 'Cooper Street', 'addr'),
 ('1750178361', 'street', 'Center Street', 'addr'),
 ('4453238172', 'street', 'South Main Street', 'addr'),
 ('5065480125', 'street', '2nd Street', 'addr'),
 ('5276461827', 'street', 'Sourth Main Street', 'addr'),
 ('8569384286', 'street', 'Poplar Street', 'addr'),
 ('8690697181', 'street', 'Main Street', 'addr')]
```

Suggestions for Improvement

1. Validate the websites. Reference code in top section where one website was used 9 times, and in block above where some websites did not work at all. Now that people are using the internet for everything, having a good website and a working link is extremely important.
2. The sports could be normalized once it is in SQL tables. In the site, they should have LOV so that things like karate fall under martial arts so it is more standardized. My guess is that the business is trying to increase sales by hitting more key words so they may want to leave these as is.
3. Possibly have requirements such as forcing user to enter an area code for phone numbers.

In [20]: *#List the distinct keys from the ways_tags and the count of each. This is a List of dis*
`cur.execute(''Select distinct(value) from ways_tags where key='website' limit 5 '')`
`print("Websites")`
`pprint.pprint(cur.fetchall())`

Websites

```
[('https://www.clementonpark.com/',),
 ('https://www.kennedyhealth.org/',),
 ('http://www.mcdonalds.com/',),
 ('https://www.walmart.com/store/1742/turnersville-nj/whats-new',),
 ('https://www.walgreens.com/locator/walgreens-180-bridgeton-pike-mantua-nj-08051/id=6741',)]
```

In [21]: `cur.execute(''Select distinct(value) from ways_tags where key='sport' and value like`
`print("Websites")`
`pprint.pprint(cur.fetchall())`

Websites

```
[('martial_arts',), ('martial_arts;karate',)]
```

By converting to Pandas dataframes, we can do a quick analysis.

In [26]: `df_nodes = pd.read_sql_query("SELECT * FROM snodes", cnx)`
`df_nodes.describe()`

Out[26]:

	id	lat	lon	user	uid	version	changeset	timestamp
count	568948	568948	568948	568948	568948	568948	568948	568948
unique	568948	429682	475017	851	851	19	8852	47813
top	103323783	39.8125000	-75.0625000	AdamP511	10756340	1	8699085	2020-04-20T20:09:59Z
freq	1	25	25	130889	130889	472080	25012	1980

Sites and References Used


W3 Schools

Udacity Data Nano Degree Mentor and Peer Boards WGU Chatter Python and XML by Christopher A. Jones and Fred L. Drake Jr.

<https://360techexplorer.com/tools/python-2-to-3-converter/>


<https://www.mygreatlearning.com/blog/regular-expression-in-python/>

<https://stackoverflow.com/questions/31697043/replace-exact-substring-in-python>

<https://www.kite.com/python/answers/how-to-test-a-url-in-python>!  (attachment:image.png)

<https://amiradata.com/python-get-file-size-in-kb-mb-or-gb/>

<https://www.markdownguide.org/basic-syntax/#ordered-lists>

Linked in Learning Joe Marini, Python JSON XML and the Web <https://www.linkedin.com/learning/python-xml-json-and-the-web/overview-of-the-requests-library?u=2045532>!  (attachment:image-2.png)