OpenStreetMap Project
Data Wrangling with MongoDB

Nicole Mister

Map Area: State College, PA, United States
https://s3.amazonaws.com/metro-extracts.mapzen.com/state-college_pennsylvania.osm.bz2

Problems Encountered in the Map

After downloading the OSM file for State College, Pennsylvania, I create a sample for testing the python scripts. However, I had difficulty capturing the specific data that needed to be cleaned in the sample. I used the OSM file in its entirety to audit and clean the data.

The following are the problems encountered during auditing:

- Inconsistent street names ['South Butz STreet')
- Incorrect postal codes

Inconsistent Street Names

I ran an audit script to test for street name consistency. The results returned with several street names that were not in the expected street name list. A few were added to the expected street name list since I found these names to be acceptable ("Alley", "Box", "Building", "Center", "Circle", "Pike", "Way", "522"). Others, I added to the mapping dictionary to change the street name as the data was changed into a JSON file ("Aly": "Alley", "Blvd": "Boulevard", "Dr": "Drive", "Ln": "Lane", "Rd": "Road", "STreet": "Street").

```
# Find Street Names Not in Expected List
```

```
audit('state-college-pa.osm')
{'522': set(['US 522']),
 'Alley': set(['Miller Alley']),
 'Aly': set(['McAllister Aly']),
 'Ave': set(['Delaware Ave',
             'E College Ave',
             'East Beaver Ave',
             'W College Ave',
             'W Freedom Ave',
             'West Freedom Ave']),
 'Blvd': set(['Colonnade Blvd']),
 'Box': set(['Post Office Box']),
 'Building': set(['Food Science Building',
                  'Rider Building',
                  'The 300 Building']),
 'Center': set(['Northland Center']),
 'Circle': set(['Montauk Circle']),
 'Dr': set(['Premiere Dr']),
 'Ln': set(['Sandy Ln']),
 'Pike': set(['Benner Pike', 'Boalsburg Pike']),
```

```
 'Rd': set(["McAlevy's Fort Rd"]),
 'STreet': set(['South Butz STreet']),
 'St': set(['4th St',
            'E Main St',
            'Hiester St',
            'N Juniata St',
            'N Patterson St',
            'S Fraser St',
            'S Garner St',
            'S Hiester St',
            'S Sparks St']),
 'St.': set(['North Atherton St.', 'S. Fraser St.']),
 'Way': set(['East Calder Way', 'West Calder Way'])}
```

Postal Codes

After running the script to update the street names, I imported the data into MongoDB. I ran queries to find all distinct postal codes. Because some post codes had 5 digits, while some were in the five digit hyphen four digit format, I queried on the first 5 characters. From the Wikipedia page, I know that the zip codes for State College are 16801, 16803, 16804 and 16805. However, I found that postal codes outside of the State College region were included in the data (16828, 16870, 16669, 16802, 16823, 17004, 17841, 16868, 16827, 17044, 17009 and 17747). In addition there was one instance of 'PA' entered as a postal code.

```
# Find distinct 5 digit post codes
cursor = db.pa.aggregate([
        {'$group': {'_id': {'$substr': ['$address.postcode', 0, 5]},
                            'count': {'$sum': 1}}}])
for doc in cursor:
    pprint.pprint(doc)
```

```
{u'_id': u'16801', u'count': 273}
{u'_id': u'', u'count': 347772}
{u'_id': u'16828', u'count': 1}
{u'_id': u'16870', u'count': 1}
{u'_id': u'16804', u'count': 1}
{u'_id': u'16803', u'count': 26}
{u'_id': u'16669', u'count': 1}
{u'_id': u'16802', u'count': 7}
{u'_id': u'16823', u'count': 3}
{u'_id': u'17004', u'count': 1}
{u'_id': u'17841', u'count': 1}
{u'_id': u'16858', u'count': 5}
{u'_id': u'16827', u'count': 2}
{u'_id': u'17044', u'count': 2}
{u'_id': u'17009', u'count': 2}
{u'_id': u'PA', u'count': 1}
{u'_id': u'17747', u'count': 1}
```

Data Overview

Below are the dataset statistics and MongoDB queries.

```
# Number of documents
> coll.find().count()
348100

# Filesize
> coll.dataSize()
82564498

# Number of nodes
db.pa.find({'type': 'node'}).count()
327544

# Number of ways
db.pa.find({'type': 'ways'}).count()
0

#Number of unique users
print len(db.pa.distinct("created.user"))
229

# Top 5 contributing users
import pprint
cursor = db.pa.aggregate([
      {'$group': {'_id': "$created.user", 'count': {'$sum': 1}}},
      {'$sort': {'count': -1}},
      {'$limit': 5}
   ])
for doc in cursor:
   pprint.pprint(doc)
{u'_id': u'woodpeck_fixbot', u'count': 206010}
{u'_id': u'Sven L', u'count': 82402}
{u'_id': u'TIGERcnl', u'count': 8170}
{u'_id': u'bot-mode', u'count': 7491}
{u'_id': u'DaveHansenTiger', u'count': 4903}

# Types and counts of buildings
import pprint
cursor = db.pa.aggregate([
      {'$group': {'_id': "$building", 'count': {'$sum': 1}}},
      {'$sort': {'count': -1}}
   ])
for doc in cursor:
   pprint.pprint(doc)
{u'_id': None, u'count': 347264}
{u'_id': u'yes', u'count': 716}
{u'_id': u'apartments', u'count': 57}
{u'_id': u'residential', u'count': 21}
{u'_id': u'entrance', u'count': 16}
{u'_id': u'house', u'count': 9}
{u'_id': u'commercial', u'count': 5}
{u'_id': u'retail', u'count': 4}
```

```
{u'_id': u'industrial', u'count': 2}
{u'_id': u'collapsed', u'count': 2}
{u'_id': u'ruins', u'count': 1}
{u'_id': u'public', u'count': 1}
{u'_id': u'office', u'count': 1}
{u'_id': u'Restaurant', u'count': 1}
```

Additional Ideas

The State College data could be improved by including only addresses within the city's designated postal codes. One of the postal codes was incorrect. The field for postal code could force 5 or 9 digit entries only. Additionally, street names should be changed to be consistent. The entry could be a drop down or an autofill to enforce consistency.