OpenStreetMap Project for Data Wrangling with MongoDB

Roger Jennings

Udacity Data Analyst Nanodegree

Open Street Map Area

Data Overview[1]

The area chosen for this project is Austin, TX. The OpenStreetMap link is: https://www.openstreetmap.org/relation/113314. I chose this area because it is my home town.

File sizes:

OSM File: austin_texas.osm: 175,250KB

JSON File: austin_texas.osm.json: 259,927KB

Number of nodes: 782417

Number of ways: 81195

Number of unique users: 907

Problems Encountered with the Map Data

An audit showed there were several types of problems with data. First, there are many variations of street names. The main ones were updated in the mapping list to account for them. Here are the updated expected names after auditing data for most common ones:

expected = ["Street", "Avenue", "Boulevard", "Drive", "Court", "Place", "Square", "Lane", "Road", "Trail", "Parkway", "Commons", "Crossing", "Highway", "Expressway", "Way", "Pass", "Speedway", "IH-35", "Loop", "Circle", "Cove", "620", "IH-35", "183"]

620, 183, and IH-35 were included because they are major highways with many addresses on these. This could cause some errors as there could be valid numbers at the end that are not related to highway numbers.

Here is the mapping that was used to update the addresses of other names before including in the database.

```
mapping = { "St": "Street",
        "St.": "Street",
        "street": "Street",
        "Ave": "Avenue",
        "Ave.": "Avenue",
        "Avene": "Avenue",
        "Rd.": "Road",
        "Rd": "Road",
        "RD": "Road",
        "Blvd": "Boulevard",
        "Blvd.": "Boulevard",
        "Cir": "Circle",
        "Ct": "Court",
```

```
"Cv": "Cove",
"Dr": "Drive",
"Dr.": "Drive",
"Expy": "Expressway",
"Expwy": "Expressway",
"Hwy": "Highway",
"I35": "IH-35",
"IH35": "IH-35",
"Ln": "Lane",
"Pkwy": "Parkway",
"lane": "Lane"
}
```

Also, directions (W, West, N, North, etc.) were many times appended to the end of addresses. This requires looking before the designation to get street identity.

Also, suite numbers at the address location (123 Main St. Suite 400) were found to be quite common. Another variation is No. at the end of the street name.

Basic MongoDB Queries

Node Count

```
print "\nnode count"
    nd_cnt = db.austin.find({"type":"node"}).count()
    print "\nNode count ", nd_cnt
```

Way Count

```
    print "\nway count"
    wy_cnt = db.austin.find({"type":"way"}).count()
    print "\nWay count", wy_cnt
```

Sample Street Addresses

This query displayed a sample of street addresses in the database.

```
pipeline = [{'$match': {'address.street':{'$exists':1}}}, {'$limit' : 5}]
sample_addresses  = db.austin.aggregate(pipeline)['result']
pprint.pprint(sample_addresses)
```

[{u'_id': ObjectId('552996448726e33844f210ef'),
u'address': {u'city': u'Austin',
u'housenumber': u'2911',
u'postcode': u'78705',
u'state': u'TX',
u'street': u'San Jacinto Boulevard'},
u'amenity': u'pub',
u'created': {u'changeset': u'11808243',

```
        u'timestamp': u'2012-06-05T17:27:27Z',
        u'uid': u'190869',
        u'user': u'elbatrop',
        u'version': u'2'},
 u'id': u'280232008',
 u'name': u'Crown and Anchor',
 u'phone': u'+1 512 322-9168',
 u'pos': [30.2925357, -97.735633],
 u'type': u'node',
 u'website': u'http://crownandanchorpub.com'},
{u'_id': ObjectId('552996448726e33844f21196'),
 u'address': {u'housenumber': u'1607',
        u'postcode': u'78701',
        u'street': u'San Jacinto Boulevard'},
 u'amenity': u'pub',
 u'created': {u'changeset': u'12282918',
        u'timestamp': u'2012-07-18T13:14:09Z',
        u'uid': u'722137',
        u'user': u'OSMF Redaction Account',
        u'version': u'5'},
 u'id': u'281362888',
 u'name': u"Schultz's Beer Garden",
 u'pos': [30.2778151, -97.7363223],
 u'type': u'node',
 u'wheelchair': u'limited'},
{u'_id': ObjectId('552996458726e33844f2d675'),
 u'address': {u'housename': u'4401',
        u'postcode': u'78664',
        u'street': u'North IH 35'},
 u'created': {u'changeset': u'22071440',
        u'timestamp': u'2014-05-01T19:48:28Z',
        u'uid': u'1554107',
        u'user': u'jgpacker',
        u'version': u'3'},
 u'id': u'345201951',
 u'name': u'Round Rock Premium Outlet',
 u'pos': [30.5664621, -97.6901121],
 u'shop': u'clothes',
 u'type': u'node',
 u'website': u'premiumoutlets.com'},
{u'_id': ObjectId('552996458726e33844f2fddf'),
 u'address': {u'city': u'Austin',
        u'housenumber': u'161',
        u'postcode': u'78748',
        u'state': u'TX',
        u'street': u'West Slaughter Lane'},
 u'amenity': u'fast_food',
 u'created': {u'changeset': u'28264414',
```

u'timestamp': u'2015-01-19T21:53:17Z',
u'uid': u'703517',
u'user': u'Iowa Kid',
u'version': u'7'},
u'id': u'354680878',
u'name': u'Chik-Fil-A',
u'pos': [30.1670069, -97.7926409],
u'type': u'node'},
{u'_id': ObjectId('552996458726e33844f2fdf4'),
u'address': {u'city': u'Austin',
u'housenumber': u'9300',
u'postcode': u'78748',
u'street': u'South I-35 Service SB'},
u'amenity': u'fast_food',
u'created': {u'changeset': u'20649142',
u'timestamp': u'2014-02-19T03:32:04Z',
u'uid': u'703517',
u'user': u'Iowa Kid',
u'version': u'5'},
u'cuisine': u'burger',
u'id': u'354684318',
u'name': u'Whataburger',
u'pos': [30.1661438, -97.7879823],
u'type': u'node'}]

Top Users

This query displayed the top 10 users in the database.

```
pipeline = [{'$match': {'created.user':{'$exists':1}}}, {'$group': {'_id':'$created.user', 'count':{'$sum':1}}},
{'$sort': {'count':-1}}, {'$limit' : 10}]
result  = db.austin.aggregate(pipeline)['result']
pprint.pprint(result)
```

[{u'_id': u'woodpeck_fixbot', u'count': 241846},
{u'_id': u'varmint', u'count': 38755},
{u'_id': u'richlv', u'count': 37661},
{u'_id': u'Iowa Kid', u'count': 35145},
{u'_id': u'Clorox', u'count': 34522},
{u'_id': u'HJD', u'count': 29081},
{u'_id': u'Cam4rd98', u'count': 27708},
{u'_id': u'afdreher', u'count': 25660},
{u'_id': u'Chris Lawrence', u'count': 18702},
{u'_id': u'TexasNHD', u'count': 18077}]

Postcode Count

This query displayed the number of postal codes for each one in the area.

```
pipeline = [{"$match":{"address.postcode":{"$exists":1}}},
{"$group":{"_id":"$address.postcode","count":{"$sum":1}}}, {"$sort":{"count":-1}}]
postcode = db.austin.aggregate(pipeline)['result']
pprint.pprint(postcode)
```

```
[{u'_id': u'78704', u'count': 62},
 {u'_id': u'78705', u'count': 54},
 {u'_id': u'78757', u'count': 48},
 {u'_id': u'78759', u'count': 46},
 {u'_id': u'78681', u'count': 45},
 {u'_id': u'78640', u'count': 43},
 {u'_id': u'78702', u'count': 39},
 {u'_id': u'78746', u'count': 38},
 {u'_id': u'78701', u'count': 37},
 {u'_id': u'78745', u'count': 32},
 {u'_id': u'78751', u'count': 28},
 {u'_id': u'78664', u'count': 26},
 {u'_id': u'78750', u'count': 25},
 {u'_id': u'78758', u'count': 24},
 {u'_id': u'78613', u'count': 23},
 {u'_id': u'78703', u'count': 23},
 {u'_id': u'78712', u'count': 22},
 {u'_id': u'78748', u'count': 21},
 {u'_id': u'78741', u'count': 21},
 {u'_id': u'78753', u'count': 19},
 {u'_id': u'78723', u'count': 19},
 {u'_id': u'78731', u'count': 18},
 {u'_id': u'78735', u'count': 18},
 {u'_id': u'78660', u'count': 17},
 {u'_id': u'78749', u'count': 17},
 {u'_id': u'78610', u'count': 16},
 {u'_id': u'78620', u'count': 15},
 {u'_id': u'78744', u'count': 14},
 {u'_id': u'78722', u'count': 14},
 {u'_id': u'78737', u'count': 12},
 {u'_id': u'78727', u'count': 12},
 {u'_id': u'78756', u'count': 11},
 {u'_id': u'78752', u'count': 11},
 {u'_id': u'78717', u'count': 10},
 {u'_id': u'78738', u'count': 10},
 {u'_id': u'78729', u'count': 9},
 {u'_id': u'78621', u'count': 9},
 {u'_id': u'78726', u'count': 8},
 {u'_id': u'78602', u'count': 8},
 {u'_id': u'76574', u'count': 8},
 {u'_id': u'78734', u'count': 8},
```

{u'_id': u'78665', u'count': 7},
{u'_id': u'78641', u'count': 6},
{u'_id': u'78652', u'count': 6},
{u'_id': u'78617', u'count': 6},
{u'_id': u'78721', u'count': 6},
{u'_id': u'78728', u'count': 5},
{u'_id': u'78739', u'count': 5},
{u'_id': u'78724-1199', u'count': 5},
{u'_id': u'78732', u'count': 5},
{u'_id': u'78669', u'count': 5},
{u'_id': u'78724', u'count': 4},
{u'_id': u'78626', u'count': 4},
{u'_id': u'78645', u'count': 4},
{u'_id': u'78628', u'count': 3},
{u'_id': u'78736', u'count': 2},
{u'_id': u'78640-6137', u'count': 2},
{u'_id': u'78705-5609', u'count': 2},
{u'_id': u'78733', u'count': 2},
{u'_id': u'78612', u'count': 2},
{u'_id': u'TX 78758', u'count': 2},
{u'_id': u'78666', u'count': 1},
{u'_id': u'Texas', u'count': 1},
{u'_id': u'78634', u'count': 1},
{u'_id': u'78619', u'count': 1},
{u'_id': u'TX 78759-3504', u'count': 1},
{u'_id': u'TX 78724', u'count': 1},
{u'_id': u'78680', u'count': 1},
{u'_id': u'TX 78745', u'count': 1},
{u'_id': u'78640-4520', u'count': 1},
{u'_id': u'78656', u'count': 1},
{u'_id': u'TX 78613', u'count': 1},
{u'_id': u'76574-4649', u'count': 1},
{u'_id': u'78646', u'count': 1},
{u'_id': u'14150', u'count': 1},
{u'_id': u'78682', u'count': 1},
{u'_id': u'78691', u'count': 1},
{u'_id': u'78653', u'count': 1},
{u'_id': u'78728-1275', u'count': 1},
{u'_id': u'78747', u'count': 1},
{u'_id': u'78753-4150', u'count': 1},
{u'_id': u'78758-7013', u'count': 1},
{u'_id': u'78704-5639', u'count': 1},
{u'_id': u'78676', u'count': 1},
{u'_id': u'78957', u'count': 1},
{u'_id': u'78754', u'count': 1},
{u'_id': u'78758-7008', u'count': 1},
{u'_id': u'78704-7205', u'count': 1},
{u'_id': u'TX 78728', u'count': 1},

{u'_id': u'78730', u'count': 1}]

Additional MongDB Queries

Amenities

I thought it might be interesting to see what amenities are listed and the number of each type. The following amenity query did that.

```
print "\namenities"
pipeline = [{"$match":{"amenity":{"$exists":1}}}, {"$group":{"_id" : "$amenity","count":{"$sum":1}}}, {"$sort":{"count":-1}}]
amenities = db.austin.aggregate(pipeline)['result']
pprint.pprint(amenities)
```

The results are quite interesting. Some are expected like 689 restaurants and 487 places of worship. But there are many that were not expected. There are 591 waste baskets, 349 benches, and 1 diving board for example.

```
[{u'_id': u'parking', u'count': 1858},
 {u'_id': u'restaurant', u'count': 689},
 {u'_id': u'waste_basket', u'count': 591},
 {u'_id': u'school', u'count': 574},
 {u'_id': u'fast_food', u'count': 510},
 {u'_id': u'place_of_worship', u'count': 487},
 {u'_id': u'fuel', u'count': 373},
 {u'_id': u'bench', u'count': 349},
 {u'_id': u'shelter', u'count': 232},
 {u'_id': u'bank', u'count': 153},
 {u'_id': u'cafe', u'count': 126},
 {u'_id': u'pharmacy', u'count': 120},
 {u'_id': u'grave_yard', u'count': 119},
 {u'_id': u'bar', u'count': 118},
 {u'_id': u'public_building', u'count': 72},
 {u'_id': u'fire_station', u'count': 62},
 {u'_id': u'swimming_pool', u'count': 62},
 {u'_id': u'car_wash', u'count': 55},
 {u'_id': u'toilets', u'count': 50},
 {u'_id': u'bicycle_parking', u'count': 44},
 {u'_id': u'post_office', u'count': 42},
 {u'_id': u'library', u'count': 41},
 {u'_id': u'atm', u'count': 40},
 {u'_id': u'hospital', u'count': 38},
 {u'_id': u'pub', u'count': 35},
 {u'_id': u'post_box', u'count': 28},
 {u'_id': u'dentist', u'count': 21},
 {u'_id': u'cinema', u'count': 20},
```

{u'_id': u'fountain', u'count': 15},
{u'_id': u'doctors', u'count': 13},
{u'_id': u'community_centre', u'count': 12},
{u'_id': u'veterinary', u'count': 12},
{u'_id': u'townhall', u'count': 12},
{u'_id': u'nightclub', u'count': 10},
{u'_id': u'car_rental', u'count': 10},
{u'_id': u'drinking_water', u'count': 9},
{u'_id': u'kindergarten', u'count': 9},
{u'_id': u'police', u'count': 9},
{u'_id': u'theatre', u'count': 9},
{u'_id': u'parking_space', u'count': 8},
{u'_id': u'college', u'count': 8},
{u'_id': u'bicycle_repair_station', u'count': 7},
{u'_id': u'parking_entrance', u'count': 6},
{u'_id': u'university', u'count': 6},
{u'_id': u'bus_station', u'count': 6},
{u'_id': u'shop', u'count': 5},
{u'_id': u'clinic', u'count': 4},
{u'_id': u'arts_centre', u'count': 4},
{u'_id': u'community_center', u'count': 3},
{u'_id': u'bicycle_rental', u'count': 3},
{u'_id': u'marketplace', u'count': 3},
{u'_id': u'childcare', u'count': 2},
{u'_id': u'studio', u'count': 2},
{u'_id': u'exercise_point', u'count': 2},
{u'_id': u'food_court', u'count': 2},
{u'_id': u'telephone', u'count': 2},
{u'_id': u'charging_station', u'count': 2},
{u'_id': u'recycling', u'count': 2},
{u'_id': u'ice_cream', u'count': 2},
{u'_id': u'prison', u'count': 2},
{u'_id': u'optician', u'count': 1},
{u'_id': u'diving_board', u'count': 1},
{u'_id': u'auditorium', u'count': 1},
{u'_id': u'social_facility', u'count': 1},
{u'_id': u'fire', u'count': 1},
{u'_id': u'Covered Pavillion', u'count': 1},
{u'_id': u'whirlpool', u'count': 1},
{u'_id': u'sporting goods', u'count': 1},
{u'_id': u'compressed_air', u'count': 1},
{u'_id': u'boat_rental', u'count': 1},
{u'_id': u'animal_shelter', u'count': 1},
{u'_id': u'gym', u'count': 1},

{u'_id': u'vending_machine', u'count': 1},
{u'_id': u'Flag', u'count': 1},
{u'_id': u'waste_disposal', u'count': 1},
{u'_id': u'Condominium complex', u'count': 1},
{u'_id': u'courthouse', u'count': 1},
{u'_id': u'yes', u'count': 1},
{u'_id': u'nursing home', u'count': 1},
{u'_id': u'hotel', u'count': 1},
{u'_id': u'Fine Wine & Liquor', u'count': 1},
{u'_id': u'Drainage', u'count': 1},
{u'_id': u'amusement_park', u'count': 1}]

Cuisine Types

Seeing the number of restaurants, I was curious about the types of cuisine offered in Austin. So, I ran a query to see what comes up.

```
pipeline = [{"$match":{"cuisine":{"$exists":1}}}, {"$group":{"_id" : "$cuisine","count":{"$sum":1}}},
{"$sort":{"count":-1}}]
    cuisine = db.austin.aggregate(pipeline)['result']
    pprint.pprint(cuisine)
```

As expected, burgers, Mexican (this is Texas!), sandwiches, and pizza top the list, in that order. I was surprised to see 76 different types of cuisine. However, some of them are not accurate. In the data there are cuisines called “mexican” as well as “Mexican”, Bar-B-Q and bbq and barbecue and BBQ and Texas Style BBQ. There are several other problems that compound the accuracy. Much like I did to clean up road names, this would be a good area to consolidate.

[{u'_id': u'burger', u'count': 112},
{u'_id': u'mexican', u'count': 88},
{u'_id': u'sandwich', u'count': 48},
{u'_id': u'pizza', u'count': 44},
{u'_id': u'chicken', u'count': 32},
{u'_id': u'american', u'count': 27},
{u'_id': u'chinese', u'count': 23},
{u'_id': u'coffee_shop', u'count': 21},
{u'_id': u'italian', u'count': 18},
{u'_id': u'indian', u'count': 14},
{u'_id': u'thai', u'count': 14},
{u'_id': u'asian', u'count': 12},
{u'_id': u'sushi', u'count': 12},
{u'_id': u'regional', u'count': 10},
{u'_id': u'japanese', u'count': 10},
{u'_id': u'barbecue', u'count': 9},
{u'_id': u'ice_cream', u'count': 7},

{u'_id': u'vietnamese', u'count': 6},
{u'_id': u'steak_house', u'count': 4},
{u'_id': u'vegetarian', u'count': 3},
{u'_id': u'greek', u'count': 3},
{u'_id': u'korean', u'count': 3},
{u'_id': u'Mexican', u'count': 2},
{u'_id': u'Bar-B-Q', u'count': 2},
{u'_id': u'seafood', u'count': 2},
{u'_id': u'kebab', u'count': 2},
{u'_id': u'mediterranean', u'count': 2},
{u'_id': u'American', u'count': 2},
{u'_id': u'international', u'count': 2},
{u'_id': u'tex-mex', u'count': 2},
{u'_id': u'tacos', u'count': 1},
{u'_id': u'Japanese', u'count': 1},
{u'_id': u'fish', u'count': 1},
{u'_id': u'Sandwich', u'count': 1},
{u'_id': u'Chicken', u'count': 1},
{u'_id': u'american,_wings', u'count': 1},
{u'_id': u'bbq', u'count': 1},
{u'_id': u'salad', u'count': 1},
{u'_id': u'Jamaican,_Cuban', u'count': 1},
{u'_id': u'Coffee shop', u'count': 1},
{u'_id': u'Argentinian', u'count': 1},
{u'_id': u'Sandwhich shop', u'count': 1},
{u'_id': u'Haute_Cuisine', u'count': 1},
{u'_id': u'Frozen_Yogurt', u'count': 1},
{u'_id': u'tavern', u'count': 1},
{u'_id': u'Mexican_Korean', u'count': 1},
{u'_id': u'Coffee_shop,_Bar', u'count': 1},
{u'_id': u'peruvian', u'count': 1},
{u'_id': u'el_salvadorian', u'count': 1},
{u'_id': u'Indonesian', u'count': 1},
{u'_id': u'donuts', u'count': 1},
{u'_id': u'Tacos,_coffee', u'count': 1},
{u'_id': u'Moroccan', u'count': 1},
{u'_id': u'Cuban', u'count': 1},
{u'_id': u'ethiopian', u'count': 1},
{u'_id': u'colombian', u'count': 1},
{u'_id': u'BBQ', u'count': 1},
{u'_id': u'Donuts,_Breakfast,_Ice_Cream,_Treats', u'count': 1},
{u'_id': u'Greek/Mediteranian', u'count': 1},
{u'_id': u'cajun', u'count': 1},
{u'_id': u'french', u'count': 1},

{u'_id': u'Texas Style BBQ', u'count': 1},
{u'_id': u'Sushi', u'count': 1},
{u'_id': u'Hamburgers and fires', u'count': 1},
{u'_id': u'kids', u'count': 1},
{u'_id': u'pollo_asado', u'count': 1},
{u'_id': u'Asian_Fusion', u'count': 1},
{u'_id': u'Indian/Chinese_Fusion', u'count': 1},
{u'_id': u'ICe_cream', u'count': 1},
{u'_id': u'spanish', u'count': 1},
{u'_id': u'Chinese', u'count': 1},
{u'_id': u'American, home cooking', u'count': 1},
{u'_id': u'Breakfast', u'count': 1},
{u'_id': u'Tex-Mex', u'count': 1},
{u'_id': u'*', u'count': 1},
{u'_id': u'brea', u'count': 1}]

Worship Denominations

In addition, I was curious about the religious affiliations in Austin. Some, but not all, of the places of worship have a denomination field. The results of the following query are shown below.

```
    print "\n Worship Denominations"
    pipeline = [{"$match":{"denomination":{"$exists":1}}}, {"$group":{"_id" :
"$denomination","count":{"$sum":1}}}, {"$sort":{"count":-1}}]
    denom = db.austin.aggregate(pipeline)['result']
    pprint.pprint(denom)
```

[{u'_id': u'baptist', u'count': 109},
{u'_id': u'lutheran', u'count': 29},
{u'_id': u'methodist', u'count': 29},
{u'_id': u'presbyterian', u'count': 21},
{u'_id': u'catholic', u'count': 21},
{u'_id': u'pentecostal', u'count': 13},
{u'_id': u'mormon', u'count': 6},
{u'_id': u'roman_catholic', u'count': 5},
{u'_id': u'seventh_day_adventist', u'count': 4},
{u'_id': u'episcopal', u'count': 2},
{u'_id': u'jehovahs_witness', u'count': 2},
{u'_id': u'anglican', u'count': 2},
{u'_id': u'mormon3', u'count': 1},
{u'_id': u'nondenominational', u'count': 1},
{u'_id': u'ahmadiyya', u'count': 1},
{u'_id': u'latter_day_saints', u'count': 1},
{u'_id': u'quaker', u'count': 1},

{u'_id': u'protestant', u'count': 1},
{u'_id': u'nazarene', u'count': 1}]

Summary

There are still many more opportunities to clean and update the data than undertaken in this project. However, the data included in the Open Street Map database for the Austin area is quite sufficient for the objective. It does seem that Open Street Map could be made much more robust programmatically.