John Gritch

Data Wranging with MongoDB, Project:

OpenStreetMap Sample Project

Map Area: Tarrant County, Texas, United States

Section 1. Problems encountered in the map.

After downloading the dataset I audited the XML for validity, uniformity, completeness and consistency. I did not audit for accuracy. Attachment B at the end of this report has a list of the data issues that were examined. Overall the dataset was pretty clean for the issues I screened for. This may be due in part to the fact that an OSM board member and former mapquest employee Thea Clay maps and lives in Tarrant County. Thea Clay was the dataset's 2nd most active contibutor behind Frederik Ramm's fixbot with 30,429 edits.

However, there were still a few areas that needed to be addressed before shaping and exporting the data into JSON. One set of issues were several child `<tag>` tags for “maxspeed” that were missing mph unit designations. I added the unit designation programatically before importing into MongoDB.

I also cleaned up the place and street names by expanding out suffix abbreviations and cardinal prefixes into a common set of full words with the update_suffix and update_prefix functions in the cleaning, shaping and exporting script.

Another problem was with child `<tag k='addr:postcode'...>`. The post codes were not in a consistent format across the dataset and with some values having a TX prefix and some values having the additional 4 number suffix. I transformed these to be a standard 5 digit zip before importing.

A major problem that I encountered in the dataset that must be kept in mind when interpreting the database queries concerns the latitude and longitude values for many of the nodes. There are 496,688 nodes in the dataset and 15,657 have either a latitude or longitude (or both) value that is outside of the export boundary by 0.01 degree or more. At Tarrant County's latitude a hundreth of a degree is no less than 900 meters measured east to west or 1100 meters measured north to south. I ultimately decided not to remove the nodes that fell outside of the export boundary, because I concluded that this is quite possibly an expected behavior for the export tool and secondly, that for our purposes, there's no disadvantage in the tradeoff of more data in exchange for less geographical precision on the outside edges.

Section 2. Overview of the data

Student provides a statistical overview about their chosen dataset, like:

```
tarrant_county.osm ......... 120.8 MB
tarrant_county.json .... 128.7 MB
```

Number of documents

```
>db.tarrant.find( {} ).count()
```

551,978

Number of nodes

```
> db.tarrant.find( { 'type' : 'node' } ).count()
```

496,681

Number of ways

```
> db.tarrant.find( { 'type' : 'way' } ).count()
```

55,290

Number of unique contributing users

```
> db.tarrant.distinct('created.uid').length
```

596

Count and ratio of documents of version Y to the total count of documents. (var tot_docs == 551,978)

```
> db.tarrant.aggregate([ { $group: { _id: "$created.version", count:
{$sum : 1} } },  { $project: { count : 1, ratio : {$divide :
["$count" , tot_docs]} } }, {$sort: {count : -1} }, {$limit: 8} ])
```

```
{ "_id" : "2", "count" : 263243, "ratio" : 0.47690849997644835 }
{ "_id" : "1", "count" : 219477, "ratio" : 0.3976191080079279 }
{ "_id" : "3", "count" : 45035, "ratio" : 0.08158839663899649 } ...
```

Total number of amenities

```
> db.tarrant.aggregate([ { $match: { amenity : { $exists: true } } },
{ $group: { _id: null, count : { $sum : 1} } } ])
```

{ "_id" : null, "count" : 3670 }

“Non-standard OSM Amenities”
Or amenities not in the list of amenities listed on the OSM wiki*

```
> db.tarrant.aggregate([ { $match: { amenity: { $exists: true } } },
{ $match: { amenity:  { $nin: std_am } } }, { $group: {_id:
"$amenity", count :{ $sum: 1 } } }, { $sort: { count : -1 } } ])
```

{ "_id" : "boat_storage", "count" : 63 }

{ "_id" : "swimming_pool", "count" : 26 }

{ "_id" : "power", "count" : 6 }

{ "_id" : "public_building", "count" : 5 } ...

*created by building a list of the amenitiy Values from the table located on this page.
http://wiki.openstreetmap.org/wiki/Key:amenity

Section 3. Additional Ideas

Actions to improve the data set

After producing a summary of the amenities in the dataset that were not on the OSM wiki's list of standard amenities I noticed that many of the included non-standard amenities were items that had been deprecated like amenity=public_building (recommended building=public) and many values like amenity=lawyer, amenity=shop, amenity = shop should be reclassified with the office=* or shop=* key values. These corrections would reclassify almost all of the 35 non-standard amenity tags in the dataset.

This same technique of producing a standard list of offices or shops from the OSM wiki and then running an analogous query could be used to clean up and standardize the other major tag key's used in the OSM. Overall though I was pleased by the low number of mischaracterized tags.

Additional data analysis

```
# List of nodes referenced by each way, in descending order
db.tarrant.aggregate([ { $match: {type: 'way'} }, {$unwind:
"$node_refs"}, {$group : {_id : "$_id", count : {$sum : 1} } },
{$sort: {count: -1} } ])
```

{ "_id" : ObjectId("54f0c975c9833bdc1c30701c"), "count" : 2000 }
{ "_id" : ObjectId("54f0c975c9833bdc1c306daa"), "count" : 2000 }
{ "_id" : ObjectId("54f0c975c9833bdc1c306316"), "count" : 1857 }...

```
# Average number of nodes referenced by each way
db.tarrant.aggregate([ { $match: {type: 'way'} }, {$unwind:
"$node_refs"}, {$group : {_id : "$_id", count : {$sum : 1} } },
{ $group : {_id: null, avg_nodes_per_way : {$avg : "$count" } } } ])
```

{ "_id" : null, "avg_nodes_per_way" : 10.781696509314523 }

Possible future uses of the OpenStreetMap

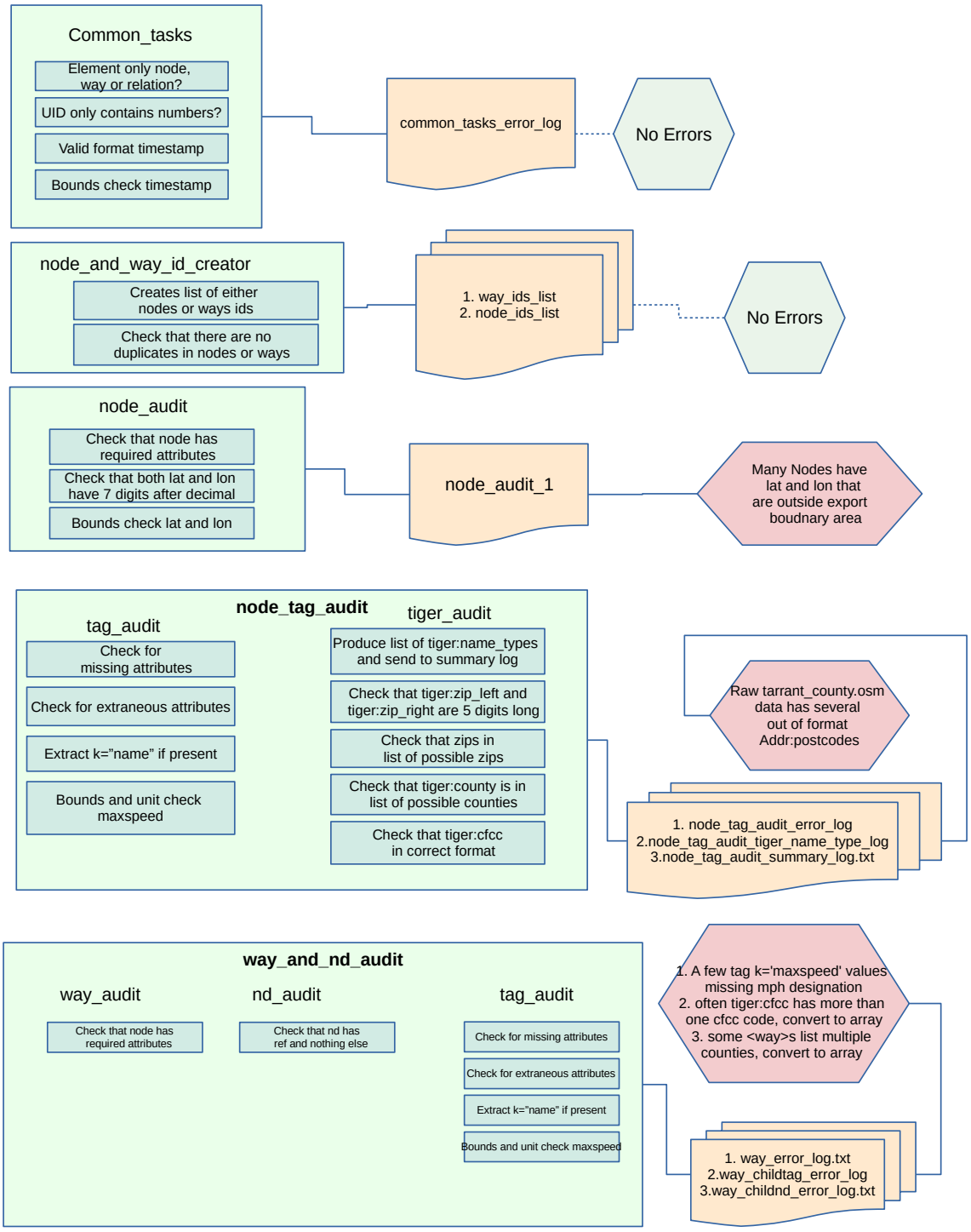
My experience with the OpenStreetMap is limited to this project, but it seems that the wiki nature of the tool lends itself to allowing large groups of people who share a common purpose, but not a common organizing body to organize themselves. By this I mean something like the 85,000 people who

converge on Cowboys Statium for a football game, or a group of people engaged in a political protest, like the TEA party or Occupy Wall Street. By mirroring the data for the place of interest and then simplifying the process of adding tags (to a process like dropping a pin in apple maps) you could set up a transient map that showed graphically where certain activities were taking place. Using the football (or sporting event in general) one example might be that people who speak a foreign language, or are fans of the away-team, or are college students of a particular college, or whatever it is could find each other during the “tailgate” before the game. After the event is over you reset the map.

Attachment A (on the next page).

This attachment is a diagram of the different scripts that I used to audit the data. In the first draft I was using a single script, but my processing times were extending past two hours. Because of these delays I moved to a series of scripts so that I could iterate faster. I ended up re-writing many of the functions to be more efficient, and in retrospect probably moved too far in the direction of fragmentation.

Attachment A.



Attachment B.

1. Top level element is only node, way or relation
2. UID only contains numbers
3. Timestamp in valid YYYY-MM-DDThh:mm:ssTZ format
4. Timestamp occurred after OSM start date of July 2004
5. Node has minimum required attributes ('id', 'visible', 'version', 'changeset', 'timestamp', 'user','uid', 'lat', 'lon')
6. Node id is unique among the observed nodes
7. Node latitude and longitude both have 7 digits after the decimal place
8. Node lat latitude is south of boundary area by X.XX degrees
9. Node lat latitude is north of boundary area by X.XX degrees
10. Node longitude is west of boundary area by X.XX degrees
11. Node longitude is east of boundary area by X.XX degrees
12. Way id is unique among the observed ways
13. Way has required attributes ('id', 'visible', 'version', 'changeset', 'timestamp', 'user')
14. tiger:zip_left and tiger:zip_right are correct 5 digit length
15. tiger:zip_left and tiger:zip_right are in list of possible zip codes
16. tiger:county is Tarrant county or is in list of possible surrounding counties (caused by taking a square bounding area that circumscribed the county)
17. tiger:cfcc is in correct letter-number-number format
18. child <tag> has two attributes with key's k and v
19. child <tag> has no other extraneous attributes
20. child <tag> listed maxspeed if present, does not exceed 85 mph
21. child <tag> listed maxspeed if present, is in mph
22. child <tag k='addr:postcode'> is in list of possible zipcodes
23. child <nd> has ref attributed
24. child <nd> has no other extraneous attributes