1. Problems Encountered in the map

I downloaded data from http://metro.teczno.com/ for Sydney, NSW, Australia. With data analysis done both before loading into the database and after there were several issues highlighted with the data.

1.1. Street Types

Initially I did some analysis on the street types in my python script. This was done by checking street types against an expected list within the 'audit()' module and updating the expected list for correct types. Then I was able to see what corrections needed to be made and updated a list of mappings to do this in the conversion to json. The module 'update_name()' when called within 'shape_element()' fixed the street names for the JSON import in MongoDB.

The expected list covered all but a handful of street types:

```
['Street', 'Avenue', 'Boulevard', 'Drive', 'Road', 'Court', 'Place',
'Circuit', 'Lane', 'Parade', 'Crescent', 'Highway', 'Way', 'Close']
```

The mapping list covered all but a small number of discrepancies:

```
{ "St": "Street", "St.": "Street","st": "Street", "street": "Street",
"Ave":"Avenue", "Av.":"Avenue", "Rd":"Road", "Rd.":"Road", "road":"Road",
"Hwy":"Highway", "place":"Place"}
```

Thus the json file and Mongo database contained the corrected street names.

1.2. Postcodes

Sorting the postcodes in descending and ascending order highlighted that some of the postcodes contained the state name 'NSW' as well as the standard Australian four digit postcode.

Sorting top 5 ascending:

```
pipeline = [{'$match':{'address.postcode':{'$exists':1}}},
{'$group':{'_id':'$address.postcode', 'count':{'$sum':1}}}, {'$sort':
{'_id':-1}}, {'$limit':5}]
```

Output:

```
{u'_id': u'Phillip Street', u'count': 1}
{u'_id': u'NSW 2567', u'count': 1}
{u'_id': u'NSW 2127', u'count': 3}
{u'_id': u'NSW 2120', u'count': 3}
{u'_id': u'NSW 2068', u'count': 1}
```

You can also see the first response does not even contain a postcode. Sorting descending showed that one postcode started with a '1' and so was not consistent with NSW postcodes starting with a '2'. Additionally one was only 3 digits long and one contained '?'s.

I subsequently fixed the postcodes that contained 'NSW' in the module 'fix_db()'.

```
result = db.openmaps.find({"address.postcode": {"$regex": "NSW "}})
```

```
for i in result:
  pprint.pprint(i['address']['postcode'][-4:])
  i['address']['postcode'] = i['address']['postcode'][-4:]
        db.openmaps.save(i)
```

1.3. Amenities

Analysing the amenities list showed some small issues when looking at those amenities with a count of 1:

```
pipeline = [{'$match':{'amenity':{'$exists':1}}}, {'$group':{'_id':'$amenity',
'count':{'$sum':1}}}, {'$sort': {'count':1}}, {'$match':{'count':1}}]
```

There was a misspelling of 'school' as 'scol' and several with multiple works joined by a '+' or '_' which means it would not match up with an otherwise identical description. Some identical things may be known by different name such as 'preschool' and 'kindergarten' or may be capitalized or not so there will be multiple instances of the same things each with their own count.

On a bespoke basis within module 'fix_db()' I also fixed the amenities names for 'scol' and replaced '+' with ' '.

```
result = db.openmaps.find({"amenity": {"$regex": "scol"}})
for i in result:
  pprint.pprint(i)
  i['amenity'] = 'school'
  db.openmaps.save(i)


result = db.openmaps.find({"amenity": {"$regex": "+"}})
for i in result:
  pprint.pprint(i)
  i['amenity'] = i['amenity'].replace('+', ' ')
  db.openmaps.save(i)
```

2. Overview of the data

This part comprises various statistics concerning the openstreetmaps data set for Sydney and the MongoDB commands used to calculate them.

Sydney_australia.osm = 236 MB
Sydney_australia.osm.json = 292 MB

Count tags - From python function count_tags()

```
{'bounds': 1,
 'member': 42175,
 'nd': 1303027,
 'node': 1085081,
 'osm': 1,
 'relation': 2779,
```

```
 'tag': 577692,
 'way': 148493}
```

Number of documents

```
db.openmaps.find().count()
1,233,574
```

Number of nodes

```
db.openmaps.find({'type':'node'}).count()
1,084,682
```

Number of ways

```
db.openmaps.find({'type':'way'}).count()
148,380
```

Number of unique users

```
len(db.openmaps.distinct('created.user'))
1,490
```

Most prolific users and number of edits

```
pipeline = [{'$group': {'_id':'$created.user', 'count':{'$sum':1}}}, {'$sort':
{'count':-1}}, {"$limit":5}]

{u'_id': u'behemoth14', u'count': 119721}
{u'_id': u'inas', u'count': 98201}
{u'_id': u'cleary', u'count': 51829}
{u'_id': u'OSMF Redaction Account', u'count': 42990}
{u'_id': u'Rhubarb', u'count': 41210}
```

Most prolific users and number of edits by fraction of total

```
pipeline = [{'$group': {'_id':'$created.user', 'count':{'$sum':1}}}, {'$project':
{'_id':'$_id', 'percent':{'$divide':['$count', total]}}}, {'$sort': {'percent':-
1}}, {"$limit":5}]
{u'_id': u'behemoth14', u'percent': 0.09705214279808103}
{u'_id': u'inas', u'percent': 0.0796068983295692}
{u'_id': u'cleary', u'percent': 0.04201531484937263}
{u'_id': u'OSMF Redaction Account', u'percent': 0.03484995630582357}
{u'_id': u'Rhubarb', u'percent': 0.03340699463510093}
```

Number of amenities entries

```
pipeline = [{'$match':{'amenity':{'$exists':1}}}, {'$group':{'_id':'$amenity',
'count':{'$sum':1}}}, {'$group':{'_id':'total amenity',
'count':{'$sum':'$count'}}}]
12,112
```

Number of unique amenities

```
len(db.openmaps.distinct('amenity'))
139
```

Most common amenities

```
pipeline = [{'$match':{'amenity':{'$exists':1}}}, {'$group':{'_id':'$amenity',
'count':{'$sum':1}}}, {'$sort': {'count':-1}}, {'$limit':10}]
{u'_id': u'parking', u'count': 2951}
{u'_id': u'school', u'count': 889}
{u'_id': u'bench', u'count': 789}
{u'_id': u'restaurant', u'count': 617}
{u'_id': u'cafe', u'count': 616}
{u'_id': u'toilets', u'count': 556}
{u'_id': u'fast_food', u'count': 479}
{u'_id': u'pub', u'count': 420}
{u'_id': u'drinking_water', u'count': 417}
{u'_id': u'place_of_worship', u'count': 415}
```

3. Additional Ideas

I thought it would be interesting to see what postcodes contained the most pubs and schools. The MongoDB commands and results were as follows, first for pubs, then schools.

```
pipeline = [{'$match':{'amenity':{'$exists':1}, 'amenity':'pub'}},
{'$group':{'_id':'$address.postcode', 'count':{'$sum':1}}}, {'$sort': {'count':-
1}}, {'$limit':10}]
{u'_id': None, u'count': 333}
{u'_id' u'2010', u'count': 7}
{u'_id': u'2000', u'count': 5}
{u'_id': u'2011', u'count': 5}
{u'_id': u'2037', u'count': 4}
{u'_id': u'2008', u'count': 4}
{u'_id': u'2040', u'count': 4}
{u'_id': u'2016', u'count': 4}
```

```
pipeline = [{'$match':{'amenity':{'$exists':1}, 'amenity':'scool'}},
{'$group':{'_id':'$address.postcode', 'count':{'$sum':1}}}, {'$sort': {'count':-
1}}, {'$limit':10}]
```

```
{u'_id': None, u'count': 790}
{u'_id': u'2570', u'count': 7}
{u'_id': u'2155', u'count': 5}
{u'_id': u'2567', u'count': 5}
{u'_id': u'2122', u'count': 4}
{u'_id': u'2067', u'count': 4}
{u'_id': u'2763', u'count': 4}
{u'_id': u'2061', u'count': 3}
{u'_id': u'2768', u'count': 3}
{u'_id': u'2153', u'count': 3}
```

Clearly there is an issue in that the majority of amenities don't have a postcode assigned making this analysis largely useless. This issue leads to one potential improvement in the data and that is to try and ascertain the postcode or suburb from the latitude and longitude details.

An alternate project could be to work out where the highest density or a particular amenity is given the position coordinates and plot them on a map. This would be especially interesting if we were also able to plot population densities to see if amenity densities are distributed relative to the population and see if school or pub densities matched population densities. It could highlight if a particular should be serviced by more schools. However to make valid conclusions you would also want to know the population demographics, such as children per captia, as well as size of schools in the area in terms of number of pupils.

Conclusion

Although the Openstreetmap data is not complete the amount of cleaning identified was fairly minimal. It appears there is some interesting work that could be done by relying on the data coordinates to do more analysis on where the majority of amenities are especially in relation to where people live.