"Problems encountered in your map"

Problem I was trying to solve in my code is to unify names of the streets in particular area in East London. In general, data was quite clean but still I was able to find nonstandard values. In "audit" code I unified street names. Here is the whole amount of nonstandard street names which were found in East London map file and their unified versions.

```
mapping = { "St": "Street",
            "St.": "Street",
            "Rd.": "Road",
            "Rd" : "Road",
            "ROAD," : "Road",
            "Ave": "Avenue",
            "st" : "Street",
            "Peninsular" : "Peninsula"}
```

I have found specific for London names of streets like Circus, Passage, Wharf, Close, Crescent, Yard etc. which are represented quite clear in my piece of map but in a bigger map you can find Circus as Cs, Passage as Ps.

First I used `street_name.title()` to initially convert street names into correct format, for example from "canary wharf" to "Canary Wharf" or from "BOROUGH HIGH STREET" to "Borough High Street".

"Overview of the data"

Before working on data I made a data summary for myself: what does the data consist of?

First code "mapparser" helped me to count tags in the whole 94MB file.

My output is:

```
{'bounds': 1, 'member': 66390, 'meta': 1, 'nd': 462814, 'node': 359721, 'note': 1,
'osm': 1, 'relation': 2066, 'tag': 408196, 'way': 64109}
```

Code in "tags" allows me to check the format of tags in the document. My output is:

```
{'lower': 345926, 'lower_colon': 52079, 'other': 10187, 'problemchars': 4}
```

After processing data I studied it by writing a few queries. Below are my queries and the output.

```
# check if data is inserted into MongoDB
pipeline = [
          {'$limit' : 6}
]
```

```
[{u'_id': ObjectId('549741e303895f141015f0f1'),
 u'created': {u'changeset': u'13416558',
              u'timestamp': u'2012-10-08T18:36:25Z',
              u'uid': u'900987',
              u'user': u'Metanautics fixes',
              u'version': u'5'},
 u'id': u'104306',
 u'pos': [51.5232881, -0.1205737],
 u'type': u'node'},
{u'_id': ObjectId('549741e303895f141015f0f2'),
 u'created': {u'changeset': u'24275401',
              u'timestamp': u'2014-07-21T15:08:27Z',
              u'uid': u'481116',
              u'user': u'bigalxyz123',
              u'version': u'5'},
```

```
 u'id': u'104309',
 u'pos': [51.5234897, -0.1149117],
 u'type': u'node'},
{u'_id': ObjectId('549741e303895f141015f0f3'),
 u'created': {u'changeset': u'13419900',
              u'timestamp': u'2012-10-08T22:22:19Z',
              u'uid': u'900987',
              u'user': u'Metanautics fixes',
              u'version': u'4'},
 u'id': u'104310',
 u'pos': [51.5229923, -0.1144836],
 u'type': u'node'},
{u'_id': ObjectId('549741e303895f141015f0f4'),
 u'created': {u'changeset': u'2511284',
              u'timestamp': u'2009-09-17T12:28:02Z',
              u'uid': u'20163',
              u'user': u'bri g',
              u'version': u'3'},
 u'highway': u'traffic_signals',
 u'id': u'104312',
 u'pos': [51.5215036, -0.1133081],
 u'type': u'node'},
{u'_id': ObjectId('549741e303895f141015f0f5'),
 u'created': {u'changeset': u'2554926',
              u'timestamp': u'2009-09-21T09:11:46Z',
              u'uid': u'20163',
              u'user': u'bri g',
              u'version': u'5'},
 u'id': u'104313',
 u'pos': [51.5202454, -0.1125015],
 u'type': u'node'},
{u'_id': ObjectId('549741e303895f141015f0f6'),
 u'created': {u'changeset': u'2511284',
              u'timestamp': u'2009-09-17T12:28:03Z',
              u'uid': u'20163',
              u'user': u'bri g',
              u'version': u'4'},
 u'id': u'104314',
 u'pos': [51.5190718, -0.1118119],
 u'type': u'node'}]
```

```
# find 30 most frequent type of shops in the area
pipeline = [
           {"$group" : {"_id" : "$shop",
                        'count' : { "$sum" : 1} } },
            { "$sort" : { "count" : -1 } },
            { "$skip" : 1},
            { "$limit" : 30 }
]
```

```
[{u'_id': u'convenience', u'count': 450},
{u'_id': u'clothes', u'count': 300},
{u'_id': u'supermarket', u'count': 195},
{u'_id': u'hairdresser', u'count': 142},
{u'_id': u'estate_agent', u'count': 74},
{u'_id': u'bookmaker', u'count': 73},
{u'_id': u'newsagent', u'count': 69},
{u'_id': u'jewelry', u'count': 63},
{u'_id': u'dry_cleaning', u'count': 61},
{u'_id': u'bicycle', u'count': 57},
{u'_id': u'vacant', u'count': 53},
{u'_id': u'furniture', u'count': 51},
{u'_id': u'bakery', u'count': 47},
{u'_id': u'mobile_phone', u'count': 45},
{u'_id': u'gift', u'count': 44},
{u'_id': u'books', u'count': 41},
{u'_id': u'shoes', u'count': 40},
```

```
{u'_id': u'laundry', u'count': 37},
{u'_id': u'alcohol', u'count': 35},
{u'_id': u'beauty', u'count': 33},
{u'_id': u'optician', u'count': 30},
{u'_id': u'car_repair', u'count': 28},
{u'_id': u'electronics', u'count': 26},
{u'_id': u'stationery', u'count': 24},
{u'_id': u'yes', u'count': 23},
{u'_id': u'florist', u'count': 23},
{u'_id': u'sports', u'count': 23},
{u'_id': u'art', u'count': 22},
{u'_id': u'butcher', u'count': 22},
{u'_id': u'deli', u'count': 21}]

# find top 10 contributing users
pipeline = [
           {'$match': {'created.user':{'$exists':1}}},
           {'$group': {'_id':'$created.user',
                       'count':{'$sum':1}}},
           {'$sort': {'count':-1}},
           {'$limit' : 10}
]

[{u'_id': u'Amaroussi', u'count': 85661},
{u'_id': u'Paul The Archivist', u'count': 70300},
{u'_id': u'Tom Chance', u'count': 62752},
{u'_id': u'Blumpsy', u'count': 46836},
{u'_id': u'Ed Avis', u'count': 45634},
{u'_id': u'Mauls', u'count': 36853},
{u'_id': u'Harry Wood', u'count': 33152},
{u'_id': u'DLMatthews', u'count': 31197},
{u'_id': u'TimSC_Data_CC0_To_Andy_Allan', u'count': 27347},
{u'_id': u'mapper999', u'count': 21350}]


# find top 10 types of places which have an article in wikipedia
pipeline = [
           {'$match': {'wikipedia':{'$exists':1}}},
           {'$group': {'_id':'$amenity',
                       'count':{'$sum':1}}},
           {'$sort': {'count':-1}},
           {'$skip': 1},
           {'$limit' : 10}
]

[{u'_id': u'place_of_worship', u'count': 247},
{u'_id': u'theatre', u'count': 136},
{u'_id': u'restaurant', u'count': 110},
{u'_id': u'university', u'count': 84},
{u'_id': u'pub', u'count': 83},
{u'_id': u'school', u'count': 64},
{u'_id': u'hospital', u'count': 36},
{u'_id': u'courthouse', u'count': 24},
{u'_id': u'cinema', u'count': 19},
{u'_id': u'bar', u'count': 18}]


# find names of articles in wikipedia for cinemas
   pipeline = [
           {'$match': {'amenity':'cinema',
                       'wikipedia':{'$exists':1}}},
           {'$project':{'_id': '$name',
                        'wikipedia':'$wikipedia'}},
           {'$limit':3}
]
```

```
[{u'_id': u'Screen On The Green', u'wikipedia': u'en:The Screen On The Green'},
{u'_id': u'Cineworld', u'wikipedia': u'en:The O2'},
{u'_id': u'BFI IMAX', u'wikipedia': u'en:London_IMAX'}]
```

```
# find name of the restaurant and its cuisine
pipeline = [
           {'$match': {'amenity':'restaurant',
                       'name':{'$exists':1}}},
           {'$project':{'_id':'$name',
                        'cuisine':'$cuisine'}}
]
```

```
. . .
{u'_id': u'Shandis', u'cuisine': u'persian'},
{u'_id': u'Rabieng', u'cuisine': u'thai'},
{u'_id': u'Rajmoni', u'cuisine': u'indian'},
{u'_id': u'La Petit Auberge', u'cuisine': u'french'},
{u'_id': u'Masala Zone', u'cuisine': u'indian'},
{u'_id': u'Sangria', u'cuisine': u'tapas'},
{u'_id': u'Strada', u'cuisine': u'pizza'},
{u'_id': u'Caf\xe9 Gallipoli Again', u'cuisine': u'turkish'},
{u'_id': u'Desperados', u'cuisine': u'mexican'},
{u'_id': u'ThaiVeg', u'cuisine': u'vegan'},
{u'_id': u'Tortilla', u'cuisine': u'mexican'},
. . .
```

”Other ideas about the datasets”

After exploring this dataset I may conclude that it is not complete so I would keep working on this dataset.

For example, I noticed in ‘shop’ tags:

```
{u'_id': u'yes', u'count': 23}
```

Obviously there is no type of shop ‘yes’ so I’d find out what does that mean and change it.

I would also add more cuisine types to the restaurants as just a few of them have it.

If I look beyond this exercise and even beyond my current knowledge I would add information about street traffic to OpenStreetMap. I would analyze traffic data from other sources like Google or Yandex and add them to OSM as an average level of traffic in rush hours (morning 7-10 am and evening 5-8 pm) in particular street, for example classifying from 1 to 10: `{‘traffic’: ‘5’}`. When we get this information about the area or the whole city, we can decide where mostly traffic is happening and new metro station or new road is needed.

The other idea is about classifying buildings as residential, commercial or state ones. We can also add number of floors in the buildings and approximate population of the area. Knowing these data and analysing it we can find out needs of districts in schools, hospitals, shops, cinemas etc.