P3 - Wrangling Open Street Map Data

Data Analyst Nanodegree

Section 1: Problems encountered in the map

1 Problems encountered in my map

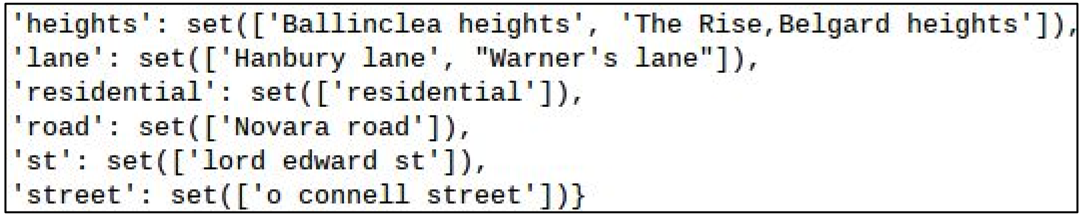
After investigating the data set for the Dublin area with some adjusted code from lesson 6, I noticed several problems with the data. I will discuss the following 3 problems and my steps to address them:

1) Lower cap street names (“o connell street”)

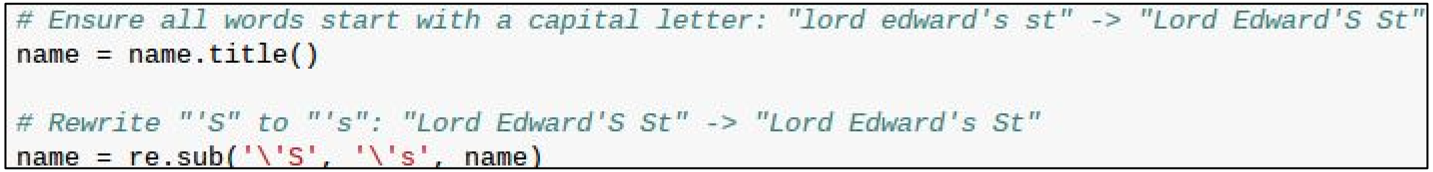
2) Typos, Abbrevations and American English spelling (“Parliament Sreet”, “Griffith Ave”, “Omni Center”)

3) Inconsistent postal codes (“D15”, “15”, “Dublin 15”, “D15 T1FD”)

1.1 Lower cap street names (“lord edward lane”)



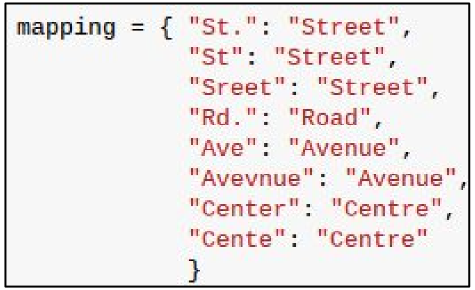
```
'heights': set(['Ballinclea heights', 'The Rise,Belgard heights']),
'lane': set(['Hanbury lane', "Warner's lane"]),
'residential': set(['residential']),
'road': set(['Novara road']),
'st': set(['lord edward st']),
'street': set(['o connell street'])}
```



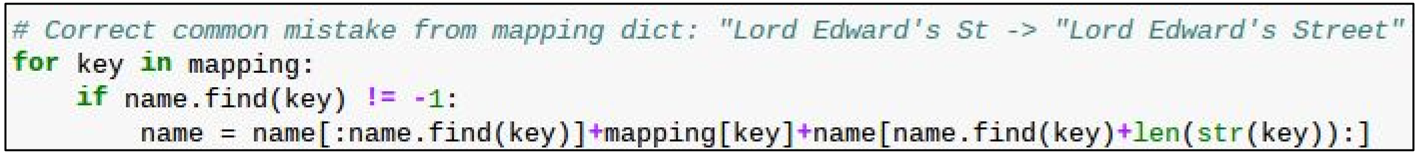
```
# Ensure all words start with a capital letter: "lord edward's st" -> "Lord Edward'S St"
name = name.title()

# Rewrite "'S" to "'s": "Lord Edward'S St" -> "Lord Edward's St"
name = re.sub('\'S', '\'s', name)
```

Several street addresses were in lower cap, including O’ Connell Street, one of the main shopping streets in Dublin. I added the *string.title()* method to the *update_streets* function to correct this.

1.2 Typos, Abbrevations and American English spelling



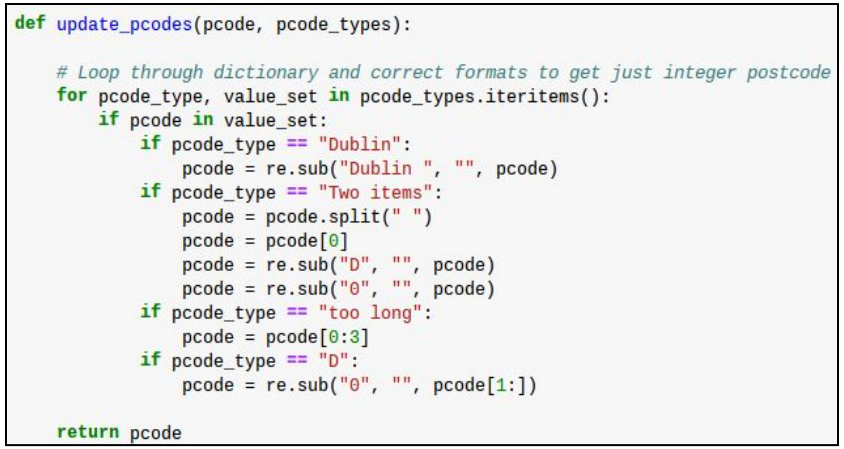
```
mapping = { "St.": "Street",
            "St": "Street",
            "Sreet": "Street",
            "Rd.": "Road",
            "Ave": "Avenue",
            "Avevnue": "Avenue",
            "Center": "Centre",
            "Cente": "Centre"
            }
```



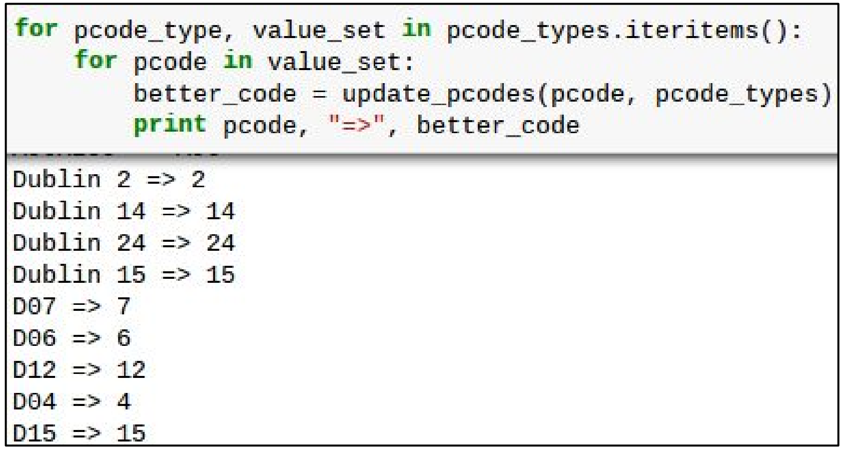
```
# Correct common mistake from mapping dict: "Lord Edward's St -> "Lord Edward's Street"
for key in mapping:
    if name.find(key) != -1:
        name = name[:name.find(key)]+mapping[key]+name[name.find(key)+len(str(key)):]
```

Some street addresses had typos or missing characters (i.e. ‘Parliament Sreet’), some abbreviations were used (i.e. ‘Griffith Ave’) and sometimes American English was used (i. e. ‘Omni Center’). I addressed this by including a mapping section in the *update_streets()* function.

1.3 Inconsistent postal codes



```python
def update_pcodes(pcode, pcode_types):

    # Loop through dictionary and correct formats to get just integer postcode
    for pcode_type, value_set in pcode_types.iteritems():
        if pcode in value_set:
            if pcode_type == "Dublin":
                pcode = re.sub("Dublin ", "", pcode)
            if pcode_type == "Two items":
                pcode = pcode.split(" ")
                pcode = pcode[0]
                pcode = re.sub("D", "", pcode)
                pcode = re.sub("0", "", pcode)
            if pcode_type == "too long":
                pcode = pcode[0:3]
            if pcode_type == "D":
                pcode = re.sub("0", "", pcode[1:])

    return pcode
```



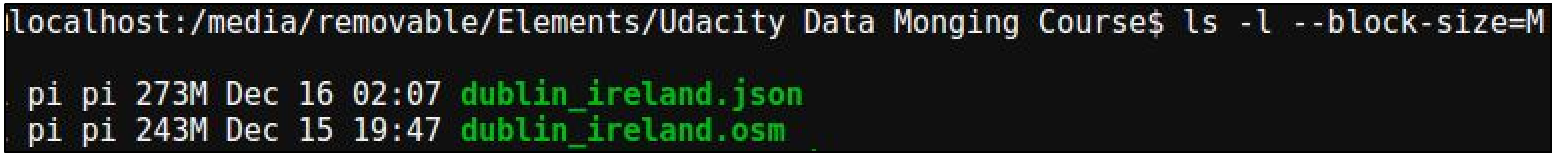
```python
for pcode_type, value_set in pcode_types.iteritems():
    for pcode in value_set:
        better_code = update_pcodes(pcode, pcode_types)
        print pcode, "=>", better_code
```

```
Dublin 2 => 2
Dublin 14 => 14
Dublin 24 => 24
Dublin 15 => 15
D07 => 7
D06 => 6
D12 => 12
D04 => 4
D15 => 15
```

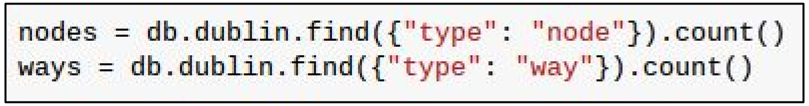
Postcodes in Dublin range from 1-24. Several different formats in the data set required adjustments, i.e. “Dxx” or “Dublin xx”. For the postcode 15 and 18 areas there were some entries with a new postcode system (i.e. ‘D15 YF53’). For consistency I simplified those to the old ‘15’ as well.

Section 2: Data Overview

2.1 Contributors



```
localhost:/media/removable/Elements/Udacity Data Monging Course$ ls -l --block-size=M

 pi pi 273M Dec 16 02:07 dublin_ireland.json
 pi pi 243M Dec 15 19:47 dublin_ireland.osm
```



```
nodes = db.dublin.find({"type": "node"}).count()
ways = db.dublin.find({"type": "way"}).count()
```

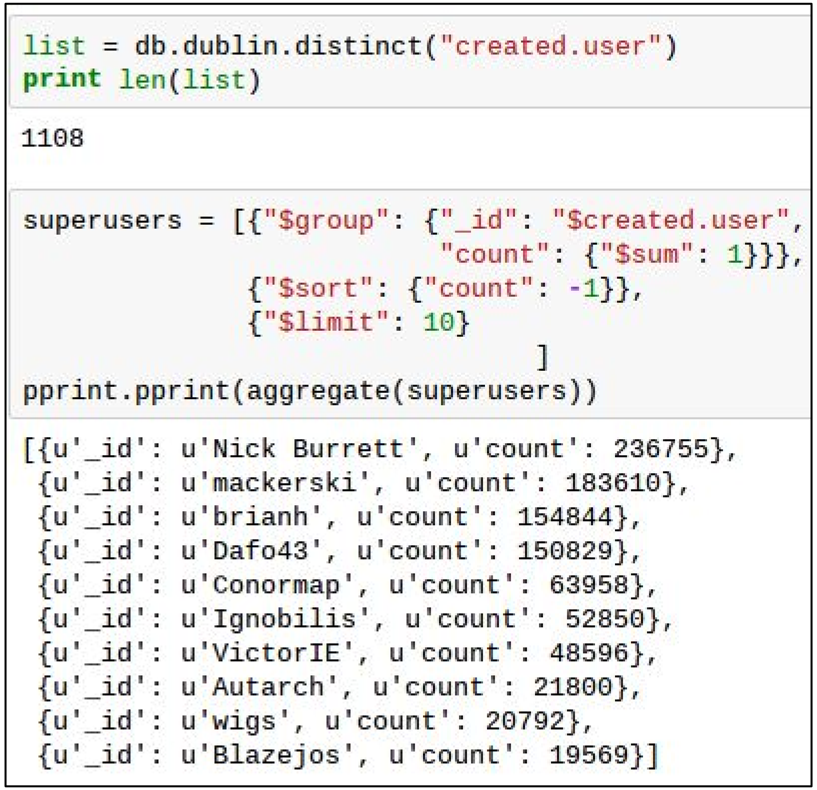
The OSM data for Dublin amounted to **273MB** in json format / **243MB** in XML format.

There are 1,068,553 nodes and 187,688 ways in the dataset.

2.2 User contributions

1,108 users contributed to the data set.

As is common with open-source projects, the top 10 users are responsible for the vast majority of entries.



```
list = db.dublin.distinct("created.user")
print len(list)
```

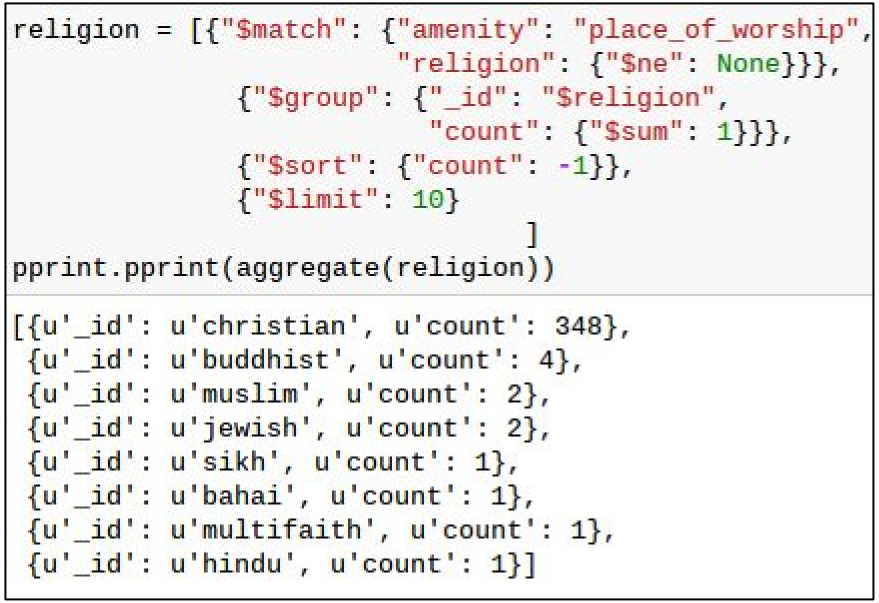
```
1108
```

```
superusers = [{"$group": {"_id": "$created.user",
                          "count": {"$sum": 1}}},
              {"$sort": {"count": -1}},
              {"$limit": 10}
                                  ]
pprint.pprint(aggregate(superusers))
```

```
[{u'_id': u'Nick Burrett', u'count': 236755},
 {u'_id': u'mackerski', u'count': 183610},
 {u'_id': u'brianh', u'count': 154844},
 {u'_id': u'Dafo43', u'count': 150829},
 {u'_id': u'Conormap', u'count': 63958},
 {u'_id': u'Ignobilis', u'count': 52850},
 {u'_id': u'VictorIE', u'count': 48596},
 {u'_id': u'Autarch', u'count': 21800},
 {u'_id': u'wigs', u'count': 20792},
 {u'_id': u'Blazejos', u'count': 19569}]
```

2.4 Dublin religions

The fact that 97% of *places of worship* are Christian (348 out of 360) confirms that Ireland is still heavily Catholic. No surprises here.

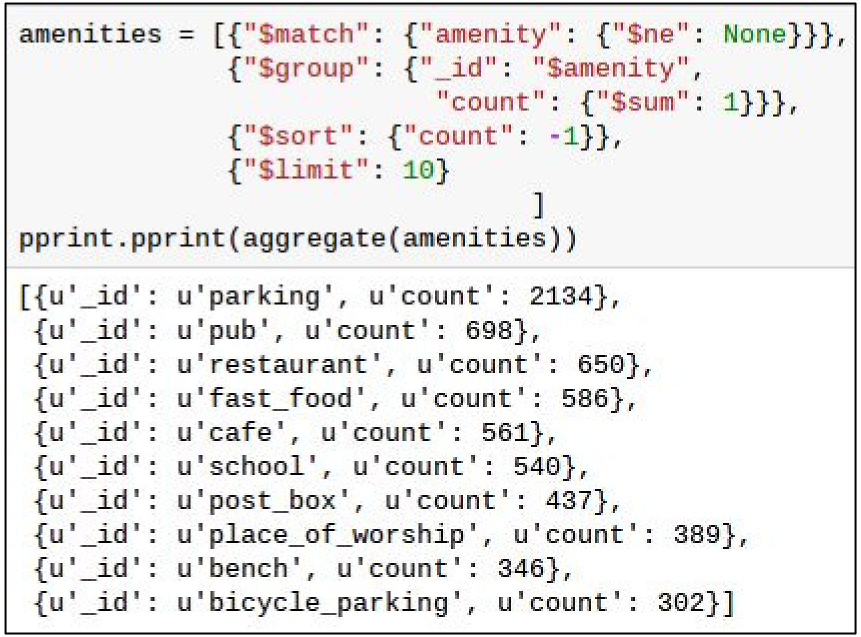


```
religion = [{"$match": {"amenity": "place_of_worship",
                        "religion": {"$ne": None}}},
            {"$group": {"_id": "$religion",
                        "count": {"$sum": 1}}},
            {"$sort": {"count": -1}},
            {"$limit": 10}
                                   ]
pprint.pprint(aggregate(religion))
```

```
[{u'_id': u'christian', u'count': 348},
 {u'_id': u'buddhist', u'count': 4},
 {u'_id': u'muslim', u'count': 2},
 {u'_id': u'jewish', u'count': 2},
 {u'_id': u'sikh', u'count': 1},
 {u'_id': u'bahai', u'count': 1},
 {u'_id': u'multifaith', u'count': 1},
 {u'_id': u'hindu', u'count': 1}]
```

2.4 Dublin amenities

...that said, the true place of worship in Ireland is of course the pub, the top amenity aside from parking.



```
amenities = [{"$match": {"amenity": {"$ne": None}}},
             {"$group": {"_id": "$amenity",
                         "count": {"$sum": 1}}},
             {"$sort": {"count": -1}},
             {"$limit": 10}
                                   ]
pprint.pprint(aggregate(amenities))
```

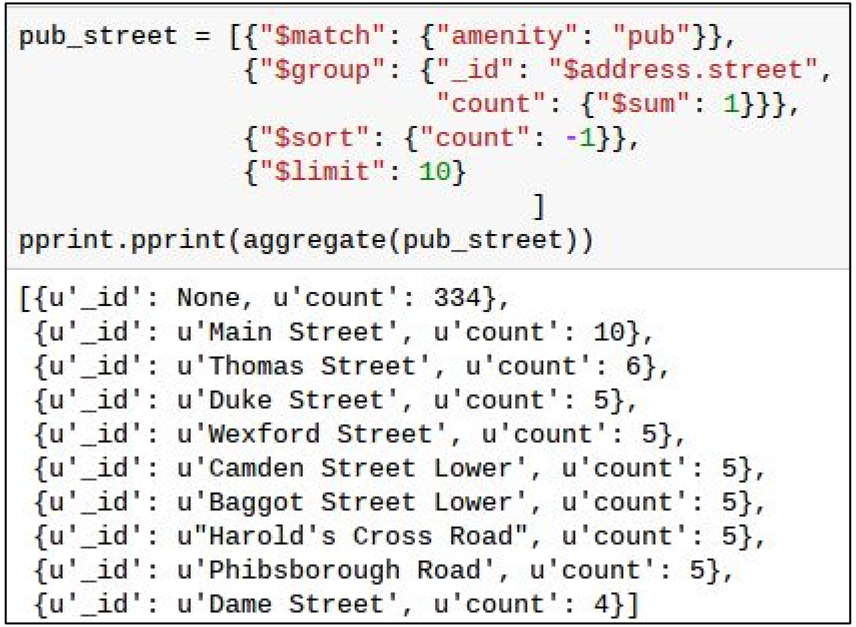
```
[{u'_id': u'parking', u'count': 2134},
 {u'_id': u'pub', u'count': 698},
 {u'_id': u'restaurant', u'count': 650},
 {u'_id': u'fast_food', u'count': 586},
 {u'_id': u'cafe', u'count': 561},
 {u'_id': u'school', u'count': 540},
 {u'_id': u'post_box', u'count': 437},
 {u'_id': u'place_of_worship', u'count': 389},
 {u'_id': u'bench', u'count': 346},
 {u'_id': u'bicycle_parking', u'count': 302}]
```

2.5 Dublin pubs

Admittedly it’s hard *not* to find a pub in Dublin, however it seems Main Street is the street with most pubs (10).

Of course this is not very reliable information, as most pubs don’t have a street address listed at all (334).

On the one hand this is a weakness of the dataset. On the other hand, when giving directions the Irish are more likely to refer to pubs than to street names anyways.

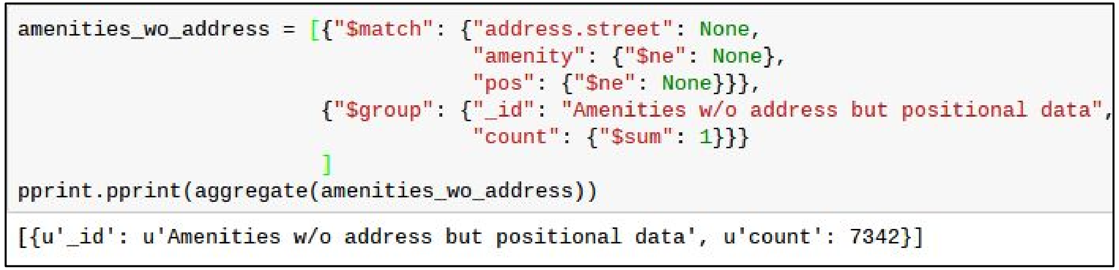


```
pub_street = [{"$match": {"amenity": "pub"}},
              {"$group": {"_id": "$address.street",
                          "count": {"$sum": 1}}},
              {"$sort": {"count": -1}},
              {"$limit": 10}
                                  ]
pprint.pprint(aggregate(pub_street))
```

```
[{u'_id': None, u'count': 334},
 {u'_id': u'Main Street', u'count': 10},
 {u'_id': u'Thomas Street', u'count': 6},
 {u'_id': u'Duke Street', u'count': 5},
 {u'_id': u'Wexford Street', u'count': 5},
 {u'_id': u'Camden Street Lower', u'count': 5},
 {u'_id': u'Baggot Street Lower', u'count': 5},
 {u'_id': u"Harold's Cross Road", u'count': 5},
 {u'_id': u'Phibsborough Road', u'count': 5},
 {u'_id': u'Dame Street', u'count': 4}]
```

Section 3: Additional Ideas & Conclusion

3.1 Idea: Estimate addresses from positional data



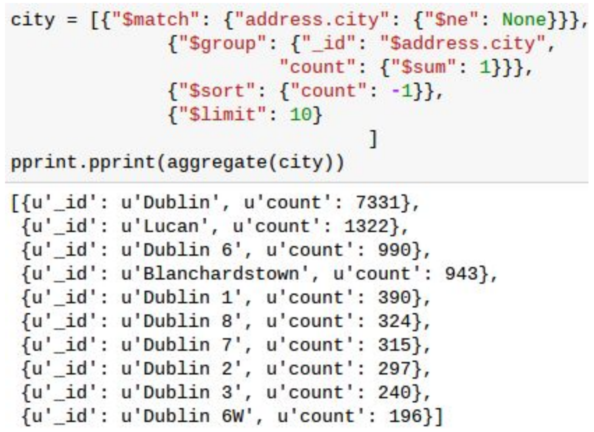
```
amenities_wo_address = [{"$match": {"address.street": None,
                                    "amenity": {"$ne": None},
                                    "pos": {"$ne": None}}},
                        {"$group": {"_id": "Amenities w/o address but positional data",
                                    "count": {"$sum": 1}}}
                        ]
pprint.pprint(aggregate(amenities_wo_address))
```

```
[{u'_id': u'Amenities w/o address but positional data', u'count': 7342}]
```

As we saw in the pub query, many amenities don’t have an address. 7,342 of those amenities however do have positional data. We could cross-relate this positional data to the streets listed as ways and thereby estimate the street address of the amenities.

Potentially this could work not only for amenities but for many types of nodes without an address. A possible risk is that the waypoints of a street are far apart or that positional data of nodes is not accurate enough. I would try with a sample first for which street addresses are available to see how well the matching could work.

3.1 Conclusion



```
city = [{"$match": {"address.city": {"$ne": None}}},
              {"$group": {"_id": "$address.city",
                          "count": {"$sum": 1}}},
              {"$sort": {"count": -1}},
              {"$limit": 10}
                                      ]
pprint.pprint(aggregate(city))
```

```
[{u'_id': u'Dublin', u'count': 7331},
 {u'_id': u'Lucan', u'count': 1322},
 {u'_id': u'Dublin 6', u'count': 990},
 {u'_id': u'Blanchardstown', u'count': 943},
 {u'_id': u'Dublin 1', u'count': 390},
 {u'_id': u'Dublin 8', u'count': 324},
 {u'_id': u'Dublin 7', u'count': 315},
 {u'_id': u'Dublin 2', u'count': 297},
 {u'_id': u'Dublin 3', u'count': 240},
 {u'_id': u'Dublin 6W', u'count': 196}]
```

There are of course more things that can be cleaned up for the Dublin area. For example, the city tag has been left untouched so far and includes some postcode data that could be reassigned. Regarding streets and postcodes however, I do believe it has been cleand up pretty well for the purpose of this exercise.