P3: Wrangle OpenStreetMap Data

Cincinnati, Ohio

1.Problems Encountered in Map

After examining the data file I found 3 main Problems with the data

1. Zip codes that weren't in the 452 range for Cincinnati.
2. Phone number formatting
3. Street types being abbreviated (Ave, St, Rd, etc.)

Zip Codes

I gave it the range of Zip codes in Cincinnati. Then I created a mapped Zip for any zipcode that doesn't fall in the range.

```
expectedZip = [45201,45999]
mappedZip = 45201
```

```
def update_postal_code(postal_code):
    try:
        if not (expectedZip[0] <= int(postal_code) <= expectedZip[1]):
            raise ValueError
        else:
            return int(postal_code)
    except ValueError:
        return mappedZip
    return name
```

Phone Numbers

Phone numbers were all over the place with formatting. I made it so each one was uniform and easy to read. The format was (513) – XXX -XXXX.

```
def audit_phone_number(invalid_phone, phone_number):
    try:
        if len(phone_number) != 12 or phone_number[:3] != '+513':
            raise ValueError
    except ValueError:
        invalid_phone[phone_number] += 1
```

```python
def update_phone_number(phone_number):
    phone_number = phone_number.translate(None, ' ()-')
    phone_number = '513' + phone_number[-6:]
```

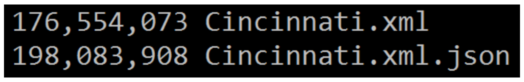
Street Names

Some street names were over abbreviated. To make them all uniform I gave expected Street types and from there I saw what abbreviations were problem. Then I mapped the problem abbreviations to the Correct format of each.

```python
def audit_street_type(street_types, street_name):
    m = street_type_re.search(street_name)
    if m:
        street_type = m.group()
        if street_type not in expected:
            street_types[street_type].add(street_name)
```

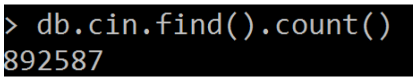
2.Data Overview

File Sizes:



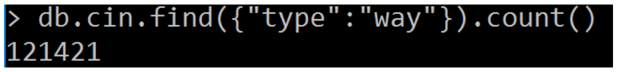
```
176,554,073 Cincinnati.xml
198,083,908 Cincinnati.xml.json
```

Number of Documents:



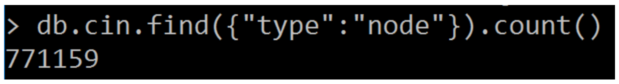
```
> db.cin.find().count()
892587
```

Number of Ways:



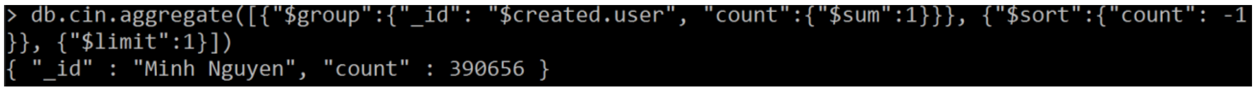
```
> db.cin.find({"type":"way"}).count()
121421
```

Number of Nodes:



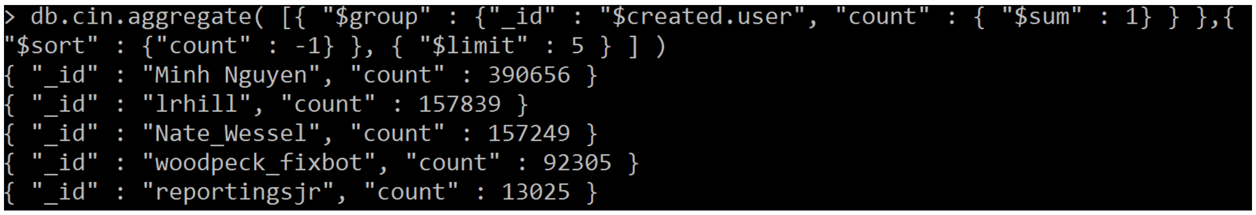
```
> db.cin.find({"type":"node"}).count()
771159
```

Highest Contributing User:



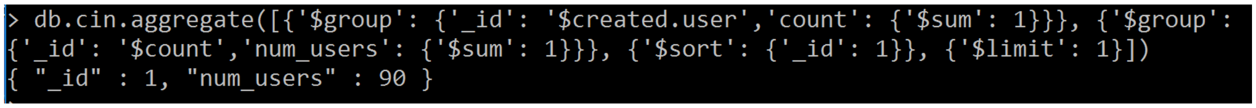
```
> db.cin.aggregate([{"$group":{"_id": "$created.user", "count":{"$sum":1}}}, {"$sort":{"count": -1
}}, {"$limit":1}])
{ "_id" : "Minh Nguyen", "count" : 390656 }
```

Top 5 Contributing Users:



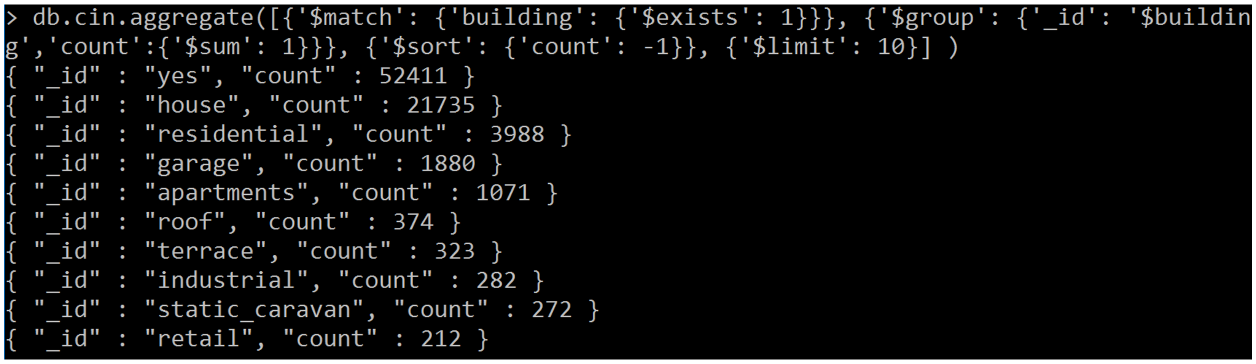
```
> db.cin.aggregate( [{ "$group" : {"_id" : "$created.user", "count" : { "$sum" : 1} } },{
"$sort" : {"count" : -1} }, { "$limit" : 5 } ] )
{ "_id" : "Minh Nguyen", "count" : 390656 }
{ "_id" : "lrhill", "count" : 157839 }
{ "_id" : "Nate_Wessel", "count" : 157249 }
{ "_id" : "woodpeck_fixbot", "count" : 92305 }
{ "_id" : "reportingsjr", "count" : 13025 }
```

Users with 1 Post:



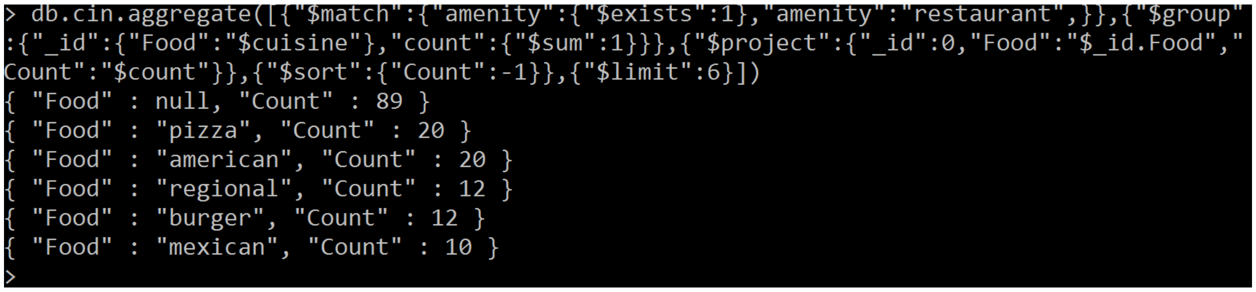
```
> db.cin.aggregate([{'$group': {'_id': '$created.user','count': {'$sum': 1}}}, {'$group':
{'_id': '$count','num_users': {'$sum': 1}}}, {'$sort': {'_id': 1}}, {'$limit': 1}])
{ "_id" : 1, "num_users" : 90 }
```

Top 10 Building Types:



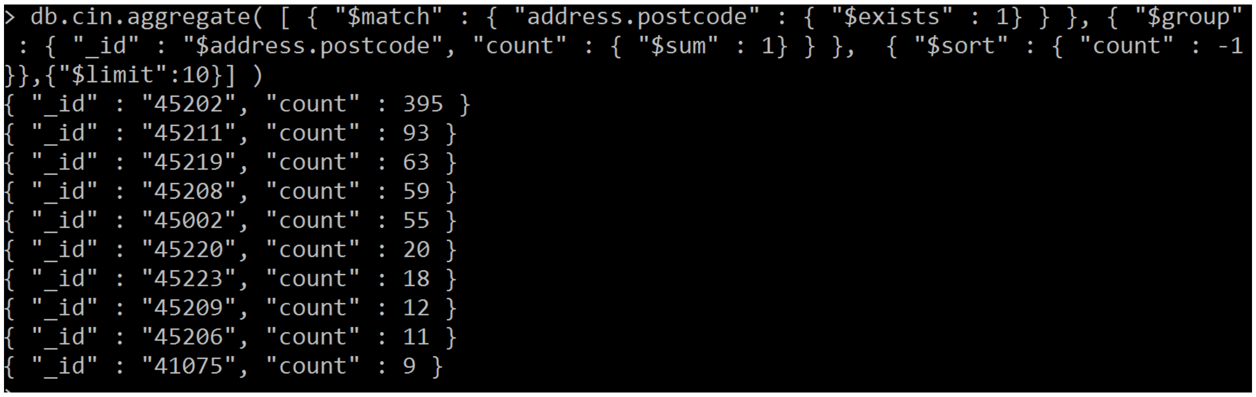
```
> db.cin.aggregate([{'$match': {'building': {'$exists': 1}}}, {'$group': {'_id': '$buildin
g','count':{'$sum': 1}}}, {'$sort': {'count': -1}}, {'$limit': 10}] )
{ "_id" : "yes", "count" : 52411 }
{ "_id" : "house", "count" : 21735 }
{ "_id" : "residential", "count" : 3988 }
{ "_id" : "garage", "count" : 1880 }
{ "_id" : "apartments", "count" : 1071 }
{ "_id" : "roof", "count" : 374 }
{ "_id" : "terrace", "count" : 323 }
{ "_id" : "industrial", "count" : 282 }
{ "_id" : "static_caravan", "count" : 272 }
{ "_id" : "retail", "count" : 212 }
```

Top 5 Food Types:



```
> db.cin.aggregate([{"$match":{"amenity":{"$exists":1},"amenity":"restaurant",}},{"$group"
:{"_id":{"Food":"$cuisine"},"count":{"$sum":1}}},{"$project":{"_id":0,"Food":"$_id.Food","
Count":"$count"}},{"$sort":{"Count":-1}},{"$limit":6}])
{ "Food" : null, "Count" : 89 }
{ "Food" : "pizza", "Count" : 20 }
{ "Food" : "american", "Count" : 20 }
{ "Food" : "regional", "Count" : 12 }
{ "Food" : "burger", "Count" : 12 }
{ "Food" : "mexican", "Count" : 10 }
>
```

Top 5 ZipCodes:



```
> db.cin.aggregate( [ { "$match" : { "address.postcode" : { "$exists" : 1} } }, { "$group"
 : { "_id" : "$address.postcode", "count" : { "$sum" : 1} } },  { "$sort" : { "count" : -1
}},{"$limit":10}] )
{ "_id" : "45202", "count" : 395 }
{ "_id" : "45211", "count" : 93 }
{ "_id" : "45219", "count" : 63 }
{ "_id" : "45208", "count" : 59 }
{ "_id" : "45002", "count" : 55 }
{ "_id" : "45220", "count" : 20 }
{ "_id" : "45223", "count" : 18 }
{ "_id" : "45209", "count" : 12 }
{ "_id" : "45206", "count" : 11 }
{ "_id" : "41075", "count" : 9 }
```

3.Other Ideas about the Datasets

While examining the data set I think it would be beneficial to have a standardization method for certain data. Making it easier to read and uniform provides for clean analyzing. As for Data like Null or Yes, I think that these values should not be an option, because it helps no one out in the long run. A system of checks and balances could be put into place like Wikipedia. If something makes no sense the community should be able to edit it when they see it to prevent skewed data. To provide for more user participation I think participation should be rewarded with a custom cryptocurrency that can be traded for Bitcoin. It will boost incentive and may even take off. With this improvement, there will be a spike in submissions. Like with many things most might be inaccurate and flawed because people are trying to collect the coins. Bots might even be made to continuously submit data to get coins. If this idea is implemented a system of checks and balances is a must. Banning IPs and Limiting Daily submissions are just two of many ways to prevent such abuse.

4.Conclusion

In my Conclusion, I found there to be many buildings described as “yes”. Also, type of Food served as “null”. With these being top in their respective lists it makes the data very skewed. If these were filled in accurately data would be much more accurate. Also, the distribution of users contributing is very uneven. The first user has 2 times as many as the second. With everything you don’t want data to be biased and you run the chance of that with uneven user distribution.