Project Data Wrangling MongoDB

Project Details:

1. Problems Encountered in the Map

I decided for a map area in France, especially in Aix-en-Provence, because I am going to participate in an 'Ironman Race' this year, so I wanted to get familiar with the area.

Characteristics and problems in the data set:

- When I started to analyse the street types I figured out that the French characters like (é,ê, etc) where encode with Hexadecimal Number in the UTF-8 form. I decided to keep this unicode character because when I export the data to MongoDB they literals should use the UTF-8 encoding. Moreover it looks like Python encodes the unicode literals which contain only 1 Byte codes automatically into Strings. So I converted the String values back into UTF-8 values.
- I encountered 3 unique Postal code entries. Because tags had more than one postcode, I generated a list for the postcode entry in the json file (set(['13090;13100', '13100', '13090'])
- The street names included different names for the same street type ('Rue', 'rue'), (Chemin, chemin). So I used the one with the capital letter in the beginning.

Additional Auditing:

- Checking for the City name I found some discrepancy. The city is written in 5 different ways :

```
 [{u'_id': u'aix-en-Provence', u'count': 1},
{u'_id': u'Aix-en-provence', u'count': 1},
{u'_id': u'Aix-en-Provence Cedex 2', u'count': 1},
{u'_id': u'AIx-en-Provence', u'count': 1},
{u'_id': u'Aix en Provence', u'count': 15},
{u'_id': u'Aix-en-Provence', u'count': 465}]
```

  In addition to the name I found the word Cedex included in city names. This is an addition which describes an external mail box. It is mentioned that the word must always be written in capital letters (CEDEX).

  I cleaned all the data in MongoDB, so the json sample file was not cleaned. For cleaning I used the RE library in Python and the find() and update() functions in MongoDB

  After cleaning the data I repeated the auditing to see whether my changes were effectively:

```
[{u'_id': u'Aix-en-Provence CEDEX 2', u'count': 1},
 {u'_id': u'Aix-en-Provence', u'count': 483}]
```

  The code snippets are induced because the data was only modified in MongoDB

- Auditing the Amenities

  Some amenities have a _ character as a split sign e.g. social_facility, arts_centre. Others are separated with a white space character e.g. pizza truck.
  I cleaned the data again with the regular expression library in Python and updated the data in MongoDB. This is only a part of the data. As one can see "fast_food" changed to "fast food"
  Before :

```
{u'_id': u'fast_food', u'count': 32},
{u'_id': u'atm', u'count': 32},
{u'_id': u'bench', u'count': 33},
{u'_id': u'pharmacy', u'count': 33},
{u'_id': u'fountain', u'count': 40},
{u'_id': u'parking_space', u'count': 46},
{u'_id': u'telephone', u'count': 51},
{u'_id': u'school', u'count': 63},
{u'_id': u'recycling', u'count': 73},
{u'_id': u'restaurant', u'count': 76},
{u'_id': u'parking', u'count': 289}]
```

After:

```
{u'_id': u'fast food', u'count': 32},
{u'_id': u'atm', u'count': 32},
{u'_id': u'bench', u'count': 33},
{u'_id': u'pharmacy', u'count': 33},
{u'_id': u'fountain', u'count': 40},
{u'_id': u'parking space', u'count': 46},
{u'_id': u'telephone', u'count': 51},
{u'_id': u'school', u'count': 63},
{u'_id': u'recycling', u'count': 73},
{u'_id': u'restaurant', u'count': 76},
{u'_id': u'parking', u'count': 289}]
```

2. Queries with MongoDB

File Sizes :

Aix_en_Provence.osm :	50.2 MB
Aix_en_Provence.osm.json	55.5 MB

Number of Collection Objects:

```
>db.Aix_en_Provence.find().count()
  242979
```

Number of 'Node' Types and 'Way' Types:

```
> db.Aix_en_Provence.find({“type":"node"}).count()
  206819
> db.Aix_en_Provence.find({"type":"way"}).count()
   34517
```

Number of Unique Users:

```
>db.Aix_en_Provence.distinct("created.uid").length
  325
```

Top 1 contributing user:

```
>db.Aix_en_Provence.aggregate([{"$group":{"_id":"created.uid", “count":
 {"$sum":1}}}, {"$sort":{"count":1}}, {"$limit":1}])
[{u'count': 166078, u'_id': u’16038'}]
```

Users with only one Entry/ Collection Object :

```
> db.Aix_en_Provence.aggregate([{"$group":{"_id":"$created.user", “count":
  {"$sum":1}}},{"$group":{"_id":"$count", "num_users":{"$sum":1}}}, {"$sort":{"_id":1}},
  {"$limit":1}])
  { "_id" : 1, "num_users" : 63 }
```

(The _id counts the number of Entries)

Analysing Amenities:

The Querie:

```
        db.Aix_en_Provence.aggregate([{'$match' : {'amenity' : {'$exists' : 1}}},{'$group' :
       {'_id' :'$amenity', 'count' : {'$sum' : 1}}},  {'$sort' : {'count' :-1}}, {'$limit' : 5}])
```

The result #Top 5 Amenities:

```
  [{u'_id': u'parking', u'count': 289},
   {u'_id': u'restaurant', u'count': 76},
   {u'_id': u'recycling', u'count': 73},
   {u'_id': u'school', u'count': 63},
   {u'_id': u'telephone', u'count': 51}]
```

I was wondering why there are still so much telephones around the area, so I took a closer look. The strings are all formatted in UTF-8.

This pipeline helped me to analyse this telephone station, because comparing to my hometown in Germany, I was confused because I didn’t see any of these in a while.

```
[{'$match' : {'amenity' : ‘telephone’}, {'$limit' : 7}]
```

Part of the result:

```
{u'_id': ObjectId('568823f76e8db9d7da1f7ed9'),
 u'amenity': u'telephone',
 u'created': {u'changeset': u'10833761',
              u'timestamp': u'2012-03-01T00:20:52Z',
              u'uid': u'210173',
              u'user': u'osmmaker',
              u'version': u'4'},
 u'id': u'300667811',
 u'operator': u'France T\xe9l\xe9com',
 u'payment:telephone_cards': u'yes',
 u'pos': [43.5286147, 5.4508934],
 u'type': u'node'}]
```

It looks like the telephone amenity is a normal telephone station.

Analysing the postal code:

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

result:

```
[{u'_id': [13090], u'count': 325},
 {u'_id': [13100], u'count': 142},
 {u'_id': [13097], u'count': 2},
 {u'_id': [13100, 13090], u'count': 1},
 {u'_id': [13626], u'count': 1},
{u'_id': [13090, 13100], u'count': 1}]
```

Only one postcode has a huge difference compared to the others. After comparing the original address, this postal code was the mailbox number (Boîte aux lettres) and the normal postal code was set to 13090. These information were updated with the following command.

```
db.Aix_en_Provence.update({'address.postcode' : 13626},
           {'$set':{'address.postcode': 13090,
           'address.BauxL' : 13626}})
```

part of the **result:**

```
{u'_id': ObjectId('568823fa6e8db9d7da22cad4'),
 u'address': {u'BauxL': 13626,
          u'city': u'Aix-en-Provence',
          u'housenumber': u'9',
          u'postcode': 13090,
          u'street': u'Avenue Jules Isaac'},
 u'building': u'yes',
 u'created': {u'changeset': u'33025734',
          u'timestamp': u'2015-08-01T08:06:11Z',
          u'uid': u'337393',
          u'user': u'lecafou',
          u'version': u'5'},
 u'id': u'87163529',
 u'node_refs': [u'1013501831',
           u'1013529408',
         …}
```

Street names :

On the street names only a few corrections had to be done. As mentioned above, some street types started with small letters. After running a query on the updated dataset, the new list of all kind of street names:

Query/ Pipeline:

```
[{"$match":{"address.street":{"$exists":1}}}, {"$group":{"_id":"$address.street",
"count":{"$sum":1}}}, {"$sort":{'count' :-1}}]
```

Result of all different street types:

```
BD
Traverse
Boulevard
Peyssonnel
Centre
Route
Rue
Impasse
Cours
Chemin
4
Square
Place
Avenue
Allée
```

Most popular Cuisines (Top 3):

```
> db.Aix_en_Provence.aggregate([{"$match":{"amenity":{"$exists":1},
"amenity":"restaurant", "cuisine" : {"$exists":1}}}, {"$group":{"_id":"$cuisine", "count":
{"$sum":1}}},{'$sort' : {'count' :-1}}, {"$limit":3}])

{ "_id" : "french", "count" : 10 }
{ "_id" : "regional", "count" : 4 }
{ "_id" : "pizza", "count" : 4 }
```

3. Additional Ideas

Contributor Statistic:

The database has 325 contributors and 242979 collection (Json)objects. The top 10 contributors were the following users with the related counts of collections nodes:

```
{ "_id" : "krysst", "count" : 166078 }    68.3 % of the collection
{ "_id" : "PierenBot", "count" : 19386 } 8.0 % of the collection
{ "_id" : "hromain", "count" : 15448 }    6.4 % of the collection
{ "_id" : "SylvainM", "count" : 6752 }
{ "_id" : "tuxayo", "count" : 6626 }
{ "_id" : "RatZillaS", "count" : 5567 }
{ "_id" : "lecafou", "count" : 3094 }
{ "_id" : "botdidier2020", "count" : 2460 }
{ "_id" : "Med", "count" : 1408 }
{ "_id" : "Pierre Mauduit", "count" : 1280 }
```

The Top 5 users account for more than 88 % of all collection nodes. So there should be definitely a gamification structure to motivate more users to participate in the editing of these charts. I looked up some information about the leading user “krysst” and found out that he is actually a professional geographer. The more people participate in the editing of the map the more up to date the chart becomes. Moreover the OSM dataset does have very few information about private houses of citizens, which makes it hard to use this data set for mail delivery services, which depend on that data

An information which could be separately added beside the geographical data (long, lat) could be the altitude. I checked the XML file for ‘ele’ attribute but couldn’t find any (with the python root iteration). It is described on the OSM wiki that normally very few people add height information to their notes. Problem for a general use of altitude information are the reference heights and the conversion to the WSG-84. So I think it is very hard to check whether a user converted the altitude correctly and used data with correct reference points.
A solution to this problem could be a Cross-referencing with other databases which include altitude heights.

While checking the most popular restaurant cuisines, I thought about models of reviewing the restaurant and showing this information in the node set (As a tag for example). Thinking of a review implemented with a scale. This review architecture should be a part of the OSM GUI interface. Because if a user has to create a node for this kind of review, very view people would do so (as seen in the contributor list above).

4. Conclusion:

The data set was good to handle. There were only some problems concerning the structure of the data set. Because the French language uses letters like é,ê the Strings were always converted into the original UTF-8 format.
Moreover some street names began with small letters which were changed to capital ones.

The Queries in MongoDB revealed some important information. The French people like there own food the most, French restaurants appear the most.
325 unique users helped to create the OSM file. The user with the most nodes accounted for more than 68% of all nodes.
Around 19% of all users accounted for only 1 collection node.

5. References :

- map area : https://www.openstreetmap.org/note/479765#map=10/43.5057/5.5124&layers=QN
- http://mongodb.com
- Python Documentation for various libraries : XML, Json, Codecs, Pymongo
- Of course Google/stackoverflow for programming issues