Area chosen for analysis: London,United Kingdom

I chose the city of London as I would like to go and settle down in London after my education.I would like to know what the city has to offer.

- https://www.openstreetmap.org/relation/65606#map=10/51.4904/-0.0879

The London osm dataset is around 2.5 gb therefore it is reduced to 20 mb by selecting every 125th top level element from the original dataset.All cleaning and analysis is run on this sample for ease of computation.

Initial view of the data:

Code is written to find out the different tags and their numbers.I found that node,way,nd and tags are the most used tags in the datset.It is clear that these are the tags which must be focused on so as to to clean the dataset and carry out analysis.

Problems with the data

1.Inconsistent street names
2.Abbreviated Street names
3.Spelling mistakes for street names
4.Unformatted phone numbers
5.Incosistent use of area and country code for phone numbers
6.Incorrect Postal codes

1.Street Names:

On auditing, Following are the issues found with the street names and actions taken to clean them

1.From the audit, the following inconsistencies have been found
a) Excessive use of 'The'
Action : All The's have been eliminated from street names.
Example : The Fairway **changed to** Fairway

b) Abbreviations for street as st, Ave for avenue
Action: All abbreviations found converted to full forms
Example: Great North Rd changed to Great North Road

c) Spelling mistakes Rioad for Road, Ridgway for Ridgeway
Action : All mistakes found have been corrected
Example: Framfield Rioad **changed to** Framfield Road

d) More than 1 names recorded with commas.
Action: First name before the comma has been selected.
Example : Theberton Street,Theberton Road **changed to** Theberton Street

e) xml format scripts such as `<val='Park Drive',<Priority; inDataSet: false, inStandard: false, selected: false>>`

Action: the name in the inverted commas has been extracted and returned as street name
Example- Park Drive was set as street name for the above example

2.Phone numbers

1.Many phone numbers starting with 020,20,+44. 020 is london area code and +44 is UK country code.
2.All numbers have been formatted to begin with +44.

3.If the number starts with london area code(landline phones), the numbers are formatted as shown below

02076039999 => +44-20-7603-9999

The eight digits after the area code are unique for every telephone number

4.The numbers starting without the area code(mobile numbers) are formatted as follows

01462455733 => +44-1462-455733

The 10 digits after the country code are unique

5.All invalid phonenumbers and invalid texts instead of phonenumbers is converted to NONE

00441322660497 => None

3.Postal Codes

1.The postal codes in london must have atleast 6 or a maximum of 8 characters including the white space
2.The format requires for the first character of the first part of the code to never equal V,Q,X and the second character of the first part to never equal I,J,Z
3.The first character of the second part must always be a number
4.There are a few postal codes with just the first part and the second part missing, these have been cleaned.

The postal codes have largely been found to be following the above conditions just for a few codes which are incomplete.
Action: All incomplete postal codes and postal codes not satisfying above conditions are changed to None.

File Update

After every audit and cleaning process, a writer function is used to rewrite the osm file with the new changes proposed in the cleaning phase.

Database

After auditing and cleaning the data, the data must now be converted to a database so it can be queried.For this purpose we first need to parse the elements in the osm xml file, transform them from document format to tables so that they can written into csv files.The csv files are then imported to tables created in a database using the sql manager.

The schema for the tables is provided in the schema.py file.

Data Overview

london_england.osm ------ 2549.937747 MB
london_sample_final.osm ------ 20.6758766174 MB
london.db ------ 14.96484375 MB
nodes.csv ------ 7.49239826202 MB
ways.csv ------ 0.785620689392 MB
nodes_tags.csv ------ 0.582620620728 MB
ways_tags.csv ------ 1.29504680634 MB
ways_nodes.csv ------ 2.77119064331 MB

Querying Database

1) Number of nodes

```
SELECT COUNT(*) FROM nodes;
```

Result- 94586

2) Number of ways

Query- `SELECT COUNT(*) FROM ways;`

Result- 13640

3) Number of unique users

```
Query-SELECT COUNT(DISTINCT(dis.uid))
   FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) dis ;
```

Result- 2671

4) Top user

```
Query-SELECT m.user,COUNT(*) as num
   ...> FROM( SELECT user FROM nodes UNION ALL SELECT user FROM ways)m
   ...> GROUP BY m.user
   ...> ORDER BY num DESC
   ...> LIMIT 1;
```

Result- The Maarssen Mapper|3804

5) Top amenities

```
Query- SELECT value, COUNT(*) as num
   ...> FROM nodes_tags
   ...> WHERE key='amenity'
   ...> GROUP BY value
   ...> order by num desc
   ...> limit 10;
```

Result-

```
post_box|114
bench|82
bicycle_parking|57
pub|38
cafe|36
waste_basket|36
fast_food|34
restaurant|31
telephone|30
parking|28
```

6) Number of Hospitals in the area

```
Query-  SELECT value, COUNT (*) as num
```

```
   ...> FROM nodes_tags
   ...> WHERE key='amenity' AND value='hospital'
   ...> GROUP BY value;
```

Result - hospital|2

ADDITIONAL QUERIES

7) Popular religion

Query-
```
SELECT nodes_tags.value, COUNT(*) as rel
   ...> FROM nodes_tags
   ...>     JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='place_of_worship') m
   ...>     ON nodes_tags.id=m.id
   ...> WHERE nodes_tags.key='religion'
   ...> GROUP BY nodes_tags.value
   ...> ORDER BY rel DESC
```

Result - christian|14

8) Number of historic memorials

Query-
```
SELECT value,COUNT(*) as num
   ...> FROM NODES_TAGS
   ...> WHERE key='historic' AND value='memorial'
   ...> GROUP BY value;
```

Result

memorial|14

9) Number of fueling stations

Query-
```
SELECT value,Count(*) as num
   ...> FROM ways_tags
   ...> WHERE key='amenity' and value='fuel'
   ...> GROUP BY value
```

Result- fuel|5

12) Number of electrical charging stations

Query-
```
select value,count(*) as num
   ...> from nodes_tags
   ...> where key='amenity' and value='charging_station'
   ...> group by value;
```

Result- charging_station|1

13)Top 5 cities

```
SELECT tags.value, COUNT(*) as count
   ...> FROM (SELECT * FROM nodes_tags UNION ALL SELECT * FROM ways_tags) tags
   ...> WHERE tags.key LIKE '%city'
   ...> GROUP BY tags.value
   ...> ORDER BY count DESC
   ...> LIMIT 5;
```

Result

```
Reading|380
London|203
Swanley|47
Maldon|45
Horsham|43
```

Conclusions from analysis

1) The Maarssen Mapper user has contributed the most to the dataset.
2)London has more christians compared to anyother religion
3)There are 14 World War 2 memorials in London
4)There are 5 fuel stations and 1 Electric charging station.This implies that though London is moving towards clean energy it is proceeding very slowly
5)Reading is the biggest city around London.
6)There are 2 hospitals in London.

CONCLUSION

The dataset is large enough and captures a lot of date with respect to the city but the data provided by the users generates a lot of issues for cleaning.The user is at his discretion to provide input to the openmap which is a great advantage as collective data can result in data sharing but this is also a bane as users can input erroneous data and minute mistakes in the input can result in lots of time wasted in cleaning the data,lets take an example of a simple street name,users can upload names such as road,Road,ROAD,RD,R# for the same tag,this becomes very complex while cleaning huge datasets as the original dataset in my example.

A viable solution would be to create predictive algorithms at the input which would provide the user suggestions while inputting data largely reducing trivial mistakes during input.

Suggestions for improving analysis

Improvement 1

The incorrect postal codes can be corrected and updated using other applications like google maps. The latitude and longitude coordinates with respect to those tags can be uploaded to an application like google maps which would return postal code for that point and the postal code can be updated.

Benefits would be that more postal codes would be corrected and a lot more valid data can be collected to be used for analysis.
One issue would be the accuracy of external application,if the external application returns inaccurate data it would lead to faulty results.

Additional Ideas

Idea 1

The openstreetmap data could contain user reviews about a certain place and this data can be used to help other users in making choices.For example users could add data about traffic on roads,when the data is collected the data can be used by other users to know about rush hour times and choose alternate routes in case of emergencies.

Benefits are it would help other users in making decisions faster.The downside is that the data must be updated constantly which is difficult as new reviews keep getting generated every minute.

A viable solution would be to update the files every week or month so that to some extent latest information is avaialable.

Idea 2

The data can be validated using external sources so that no data is recorded inaccurately which would help further analysis.