Map Area

Southampton, England

https://mapzen.com/data/metro-extracts/metro/southampton_england/

The map is of Southampton, England. I had visited a friend who lives there some years ago and really enjoyed exploring the place. I would be happy to contribute to improving its map data to help future travellers who find themselves in this port city !

I. Problems encountered with the data

After downloading the data, I extracted a sample which had every tenth data point. After running the sample against an interim data validation file, I encountered the following errors

1. Data in the format addr:street, addr:city, addr:housenumber as 'k' values

2. Empty 'k' data

3. Garage and Garages listed as seperate 'v' values where 'k' = building in ways_tags

4. Removing problematic characters (e.g. =\+/) which might cause errors when inserted into SQL

1. Data in the format addr:street, addr:city, addr:housenumber as 'k' values

After importing data to SQL, querying revealed Data in the format addr:street, addr:city, addr:housenumber.

To correct it, i made a function which searches for ':' in each 'k' value. If it finds ':', then it puts the text before ':' as 'type' and text after as key

#elem is the data element being checked and 'out' is the dictionary storing the cleaned values

```
if elem.attrib['k'].find(':') == -1: #if there is no ':'
    out['type']='regular'#regular type
    out['key']=elem.attrib['k']#key is k vale


if not elem.attrib['k'].find(':') == -1: #if there is ':' present
    pos=w.attrib['k'].find(':')#get the position index of ':'

    out['type']=elem.attrib['k'][:pos]#type is the k value before ':'
    out['key']=elem.attrib['k'][pos+1:]#key is the k value after ':'
```

2. Empty 'k' data

Some data elements had an empty 'k' data. Not only does this fail validation of the schema, but also creates a problem later when trying to analyze the data. Hence, I made a function which checks the 'k' data, and if it is empty, puts in the text 'undefined'.

```
#elem is the data element being checked and out is the dictionary storing the cleaned values
if elem.attrib['k']=='': #check if the k value is empty
    w.attrib['k']=='undefined'
```

3. Garage and Garages listed as seperate kinds of 'v' values where 'k' = building in ways_tags

Garage and Garages are listed as seperate values in ways_tags with 'k' key as building. As they have the same implication, I will group them together under 'Garage'for standardization.

The code below checks if the 'K' value is building and 'v' value is garages, and converts the output value into 'garage'

#elem is the data element being checked and out is the dictionary storing the cleaned values

```
if  elem.attrib['k'] == 'building' and  elem.attrib['v'] =='garages':
   out['value']='garage'
```

4. Removing problematic characters (e.g. =\+/) which might cause errors when inserted into SQL

Certain characters (e.g. =\+/) can cause SQL to misinterpret them as functions. Hence, checking if the 'k' key value has them, and clustering them under the kay name 'ProblemChars'

```
PROBLEMCHARS = re.compile(r'[=\+/&<>;\'"\?%#$@\,\. \t\r\n]')
if PROBLEMCHARS.search(elem.attrib['k']):
            out['key']='ProblemChars'
        else:
            out['key']=elem.attrib['k']
```

II. Data Overview

File Sizes

southampton_england.osm ..... 66 MB

southampton.db ..... 49 MB

nodes.csv ..... 23 MB

nodes_tags.csv ..... 0.78 MB

ways.csv ..... 3.1 MB

ways_tags.csv ..... 6 MB

ways_nodes.cv ..... 8.8 MB

Number of nodes

```
sqlite> SELECT COUNT(*) FROM nodes;
273569
```

Number of ways

```
sqlite> SELECT COUNT(*) FROM ways;
51517
```

Number of unique users

```
sqlite> SELECT COUNT(DISTINCT(u.uid))
FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) u;
513
```

Top 10 contributing users

```
sqlite> SELECT u.user, COUNT(*) as num
FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) u
GROUP BY u.user
ORDER BY num DESC
LIMIT 10;
```

```
Chris Baines       107076
Harjit (CabMyRide)   24966
0123456789         23686
Nick Austin        17618
```

pcman1985 14135

Deanna Earley 13399

Arjan Sahota 12932

Kuldip (CabMyRide) 9605

Andy Street 9171

Harry Cutts 6674

Number of users having a single post

```
sqlite> SELECT COUNT(*)
FROM
   (SELECT u.user, COUNT(*) as num
    FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) u
    GROUP BY u.user
    HAVING num=1)  f;
101
```

III. Additional Data Statistics

Top Amenities across Node Tags and Way Tags

```
SELECT value,count(*) AS num
from
(SELECT value FROM nodes_tags
```

```
WHERE key='amenity'
UNION ALL
SELECT value FROM ways_tags
WHERE key='amenity')e
GROUP BY value
ORDER BY num desc
LIMIT 20;
```

```
parking|705
bicycle_parking|300
post_box|284
fast_food|160
restaurant|150
pub|142
telephone|118
place_of_worship|112
bench|104
school|92
cafe|78
toilets|56
atm|46
waste_basket|37
bar|36
pharmacy|35
```

bank|34

recycling|28

post_office|21

doctors|20

Comparison of fast food VS Restauraunts (Is it truly the land of fish & Chips?)

Fast Food

```
SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
  JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='fast_food') q
  ON nodes_tags.id=q.id
WHERE nodes_tags.key='cuisine'
GROUP BY nodes_tags.value
ORDER BY num DESC;
```

chinese|13

fish_and_chips|11

sandwich|6

indian|4

pizza|4

burger|2

chicken|2

italian|2

kebab|2

pie|2

chinese_food_and_fish_and_chips|1

Restauraunts

```
SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
  JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='restaurant') q
  ON nodes_tags.id=q.id
WHERE nodes_tags.key='cuisine'
GROUP BY nodes_tags.value
ORDER BY num DESC;
```

chinese|5

italian|5

indian|4

pizza|2

thai|2

British|1

american|1

chicken|1

greek|1

scandinavian|1

spanish|1

sushi|1

Of course, fish & chips had to be amongst the top fast food ! Its number 2 ! Other wise, Chinese, Indian & Italian seem to to dominate the top places across both. (assuming one can call british pizza Italian!)

IV. Additional Improvement Ideas

With CO2 emission reduction and climate change at the forefront of mobility discussions, it is heartening to see 300 bicycle parkings in the city of Southampton

```
SELECT count(*)

from

(SELECT * FROM nodes_tags

WHERE value='bicycle_parking'

UNION ALL

SELECT * FROM ways_tags

WHERE value='bicycle_parking')e;
```

300

At the moment, there is no further detail provided about this bicycle parking spaces. As a regular cycle user, one of my main concerns is finding a secure parking space as possibility of theft is quite high otherwise. Hence, it would be very beneficial if there was data included on

the size of the parking space e.g. small (<10 spots), medium (10-50 spots) or large (>100 spots)

Benefits

Would further encourage cycle usage as people would be better informed if they have the possibility of finding a secure parking space for their cycles

Challenges

This data may need updating if the parking space capacity changes. Also, some large parking spaces may still have a high risk of being full if they are located in rush prone areas (e.g. near a train/underground metro rail station)

V. Conclusion

The Open Map data for Southampton is not yet complete and lags other privately owned map databases. However, it still gives some interesting insights into the city. Hopefully, its quality will improve with greater contribution from the people, benefiting both the travellers and the residents