Project OpenStreetMap Data Wrangling with sql

Name : ELDHOSE PETER

Map Area

New Delhi, India

Dataset Link:"https://mapzen.com/data/metro-extracts/metro/new-delhi_india/"
New Delhi is the capital city of my country, India.

Unique Tags

There are different types of tags in the XML dataset.
mapparser.py is used to get the count of different tags used in the xml file.

In [ ]:

```
tags={
'bounds': 1,
'node': 3406322,
'nd': 4215461,
'relation': 6206,
'way': 694044,
'tag': 820659,
'member': 27991,
'osm': 1
}
```

Problems Encountered In The Map

There were problems with street names.Regional languages were used to describe speciffic terms like "marg" for "Road". Street names are cleaned and updated using is_street_name(elem) and update_name(name,mapping) functions in audit.py

Abbreviations

Rd : Road

Mispelling

Socity: Society

Hindi Names

Marg : Road

Lower Case

janakapuri:Janakapuri vasundhara:Vasundhara park:Park

In [ ]:

```
mapping = {"St": "Street",
"St.": "Street",
"delhi": "Delhi",
"Ave": "Avenue",
"avenue": "Avenue",
"vihar": "Vihar",
"Rd.": "Road",
"Rd": "Road",
"Marg": "Road",
"marg": "Road",
"road": "Road",
"Roads": "Road",
"nagar": "Nagar",
"lane": "Lane",
"colony": "Colony",
"society": "Society",
"soc.": "Society",
"Socity": "Society",
"janakapuri": "Janakapuri",
"vasundhara": "Vasundhara",
"park": "Park"
}
```

Street names are updated using the functions given below.

In [ ]:

```
def is_street_name(elem):
return (elem.attrib['k'] == "addr:street")

###Function to update street names####

def update_name(name, mapping=mapping):
newnm = name.split(' ')
l1 = []
for nam in newnm:
token = 0
for mapp in mapping.items():
if mapp[0] == nam:
var = mapp[1]
token = 1
l1.append(var)
break
if token!= 1:
l1.append(nam)
string = ' '.join(l1)
return string
```

Indian postal code is called PIN code. PIN code consist only 6 digits.

In [4]:

```
def tag_attributes(element, default_tag_type, child):
    tg = {}
    tg['id'] = element.attrib['id']
    if ':' not in child.attrib['k']:
        tg['key'] = child.attrib['k']
        tg['type'] = default_tag_type
    else:
        colpos = child.attrib['k'].index(':')
        pos_col = colpos + 1
        tg['key'] = child.attrib['k'][pos_col:]
        tg['type'] = child.attrib['k'][:colpos]
    if is_street_name(child):
        street_name = update_name(child.attrib['v'])
        tg['value'] = street_name

###====Code to remove invalid postal code====###
    elif tg['key']=='postcode':
        pin_code=child.attribute['v']
        m=POST_CODE.match(pin_code)
        if m is not None:
            if len(pin_code)==6:
                tg['value']=pin_code

            else:
                return None
####==============xxxxxxx====================####
    else:
        tg['value'] = child.attrib['v']
    return tg
```

File Sizes

new-delhi_india.osm : 751.0 MB
nodes.csv : 283.7 MB
nodes_tags.csv: 1.5 MB
ways.csv: 42.1 MB
ways_nodes.csv: 10.5 MB
ways_tags.csv: 25.4 MB
NewDelhi.db : 518.7 MB

Number of nodes

In [ ]:

```
sqlite> SELECT COUNT(*) FROM NODES;
```

In [ ]:

```
Output: 3406322
```

In [ ]:

```
sqlite> SELECT COUNT(*) FROM WAYS;
```

In [ ]:

```
output:694044
```

Top users

In [ ]:

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

In [ ]:

```
Output:
```

In [ ]:

```
Oberaffe|266490
premkumar|164820
saikumar|159947
Naresh08|136512
anushap|133776
sdivya|130065
anthony1|125879
himabindhu|122876
sathishshetty|122414
Apreethi|114311
```

Unique Users

In [ ]:

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

In [ ]:

```
Output:
```

In [ ]:

```
1445
```

Popular Ammenities

In [ ]:

```
sqlite>SELECT value,COUNT(*) as num
...>FROM nodes_tags
...>WHERE key='amenity'
...>GROUP BY value
...>ORDER BY num DESC
...>LIMIT 10;
```

In [ ]:

```
Output:
```

In [ ]:

```
restaurant|223
fuel|213
atm|200
place_of_worship|175
bank|168
school|159
fast_food|131
parking|90
hospital|87
cafe|76
```

Biggest Religion

In [ ]:

```
sqlite> SELECT nodes_tags.value, COUNT(*) as num
FROM nodes_tags
JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE value='place_of_worship') i
ON nodes_tags.id=i.id
WHERE nodes_tags.key='religion'
GROUP BY nodes_tags.value
ORDER BY num DESC
LIMIT 1;
```

In [ ]:

```
Output:
```

In [ ]:

```
hindu|70
```

Popular Cuisines

In [ ]:

```
sqlite>SELECT nodes_tags.VALUE,COUNT(*) AS num
...>FROM nodes_tags
...>JOIN (SELECT DISTINCT(id) FROM nodes_tags WHERE VALUE='restaurant') i
...>ON nodes_tags.id=i.id
...>WHERE nodes_tags.key='cuisine'
...>GROUP BY nodes_tags.Value
...>ORDER BY num DESC
...>LIMIT 10;
```

In [ ]:

```
Output:
```

In [ ]:

```
indian|22
regional|11
pizza|8
North_Indian|5
chinese|5
vegetarian|5
burger|4
asian|2
korean|2
sandwich|2
```

Conclusion

The data set was large. For the purpose of the project, I have cleaned the data set using iterparsing. There were errors in street names.Most of errors are found to be seen in the 'tag' element of OSM file.
I have updated street names. Still there are lot more areas to be imrpoved.There is also a lot of outdated data. This must be removed.

Additional Thoughts

*** We can build a mechanism by which data is be cleaned periodically.**
*** Users must be restricted to use native languages.**
*** We can limit data types of particular fields. Ex: Indian postal codes are of format 999999.**
So we can limit the use data types other than digits from 0 to 9. So the postal code data types must be limited based on countries.
***Outdated data must be removed.**
***The phone numbers must be verified.**

Benefits:

***Data will be updated and will be less prone to errors.The users will be able to access most recent information.**
***Limiting the data types of particular fields will make the process of data wrangling much easier.The restriction of native languages will provide uniformity of information as only a single language is used.This also helps to reduce confusions among people regarding names of streets or roads.**

Anticipated issues:

***Users will find it annoying to fill the forms and update the information periodically.So it would be difficult to find and remove outdated data.**
***Some users would find difficult to use languages other than native languages.Restricting the use of native languages will affect the contribution of considerable proportion of the users.**