OPENSTREET MAP DATA CASE STUDY -DAND PROJECT

MAP AREA FOR THE PROJECT:

- Vidyaranyapura, Bangalore,Karnataka,India

REFERENCE BELOW - CUSTOM AREA

https://mapzen.com/data/metro-extracts/your-extracts/6cad1da9443e

CHECKING TAGS

The OSM file size for the downloaded file is 89MB and below would be the high level tags in the osm file

```
# High level TAGS in the Vidyaranyapura.osm file
def count_tags(filename):
    dicttag = {}
    for event,elem in  ET.iterparse(filename):
        if elem.tag in dicttag:
            dicttag[elem.tag]=dicttag[elem.tag]+1
        else:
            dicttag[elem.tag]=1
    return dicttag
```

{'bounds': 1, 'member': 734, 'nd': 514904, 'node': 412443, 'osm': 1, 'relation': 111, 'tag': 111437, 'way': 97646}

PROBLEMS ENCOUNTERED IN THE OSM FILE

1.IDENTIFYING PROBLEMATIC KEY VALUES IN THE TAGS

{'lower': 108454, 'lower_colon': 2877, 'other': 105, 'problemchars': 1}

```
# Checking if the keys inside the tags have any problems

lower = re.compile(r'^([a-z]|_)*$')
lower_colon = re.compile(r'^([a-z]|_)*:([a-z]|_)*$')
problemchars = re.compile(r'[=\+/&<>;\'"\?%#$@\,\. \t\r\n]')

def key_type(element, keys):
    if lower.findall(element.attrib['k']):
        keys['lower']= keys['lower']+1
    elif lower_colon.findall(element.attrib['k']):
        keys['lower_colon'] = keys['lower_colon']+1
```

```
    elif problemchars.findall(element.attrib['k']):
        print(element.attrib['k'],element.attrib['v'])
        keys['problemchars'] = keys['problemchars']+1

    else:
        keys['other']= keys['other']+1
    return keys

def process_map(filename):
    keys = {"lower": 0, "lower_colon": 0, "problemchars": 0, "other": 0}
    with open('Vpura1.osm','w')as output:
        output.write('<?xml version="1.0" encoding="UTF-8"?>\n')
        output.write('<osm>\n  ')
        osm_file = open(filename, "r",encoding ="utf-8")
        for event,element in ET.iterparse(osm_file,events = ('start','end')):
            if event == 'end' and (element.tag in
["node","way","relation","bounds"]):
                for child in element.iter('tag'):
                    keys = key_type(child, keys)
                    if child.attrib['k']== "□□□□□□□□□ □□□ □□□□□":
                        child.attrib['k']="addr:street"
                        child.attrib['v'] = "Rajagopala Nagar"# as per google
translate
                output.write(ET.tostring(element).decode("utf-8"))
        output.write('</osm>')
        osm_file.close()
    return keys
```

Upon investigating further on the problemchars we got the below as the key value where the address is written in the local language (Kannada)

□□□□□□□□□ □□□ □□□□□

Further translation of the same helped in identifying the english version as Rajagopala Nagar. The py translate function is not usable(for free) and hence since it is only a single value hich required modification handled the same using if statement The entry could have been ignored for further exploration but wanted to try practising writing in a new file and hence the above. The correction is written into a newfile Vpura1.osm which will henceforth be used in the analaysis.

CURRENT STATUS OF Keys {'lower': 108454, 'lower_colon': 2878, 'other': 105, 'problemchars': 0}

2. AUDITING STREET NAMES

Abbreviation was not common in this osm file However there were a lot of repetitive keys used with a different case or with a appended comma Examples Below

1.Smaller case road used instead of Road

'road': {'3rd main road', '80 feet road', 'Abbigere Main road', 'Shivanahalli main road', 'c i l main road'}}

2.Comma appended at the end of the word as below 'Road,': {'Jalahalli Cross, Tumkur Road,'}, 'Post,':

{'60ft road Sanjaynagar,No 401/1C,Dream Paradise, Gubbalala Main ' 'Road,Subramanyapura Post,'},
The mapping function has been used to correct the above issues and the correction is stored in a new file Vpura2.osm

```
# CHECKING THE STREET TYPES
def audit(osmfile):
    with open('Vpura2.osm','w')as output:
        output.write('<?xml version="1.0" encoding="UTF-8"?>\n')
        output.write('<osm>\n  ')
        osm_file = open(osmfile, "r",encoding ="utf-8")
        street_types = defaultdict(set)
        for event, elem in ET.iterparse(osm_file,events = ('start','end')):
            if event == 'end' and (elem.tag in ["node","way","relation","bounds"]):
                if elem.tag == "node" or elem.tag == "way":
                    for tag in elem.iter("tag"):
                        if is_street_name(tag):
                            new_value =audit_street_type(street_types,
tag.attrib['v'])
                            tag.attrib['v']= new_value
                output.write(ET.tostring(elem).decode("utf-8"))
        output.write('</osm>')
        osm_file.close()
    return street_types

def update_name(name, mapping = Mapping):
    if name.find(",")!= -1: # replacing the unwanted commas
        name = name.replace(',','')
    else:# replacing the last word
        split = name.rsplit(' ', 1)
        if len(split)>1:
            newname =split[0]
            keyword = split[1]
            if keyword in mapping:
                keyword = mapping[keyword]
            name = newname+' '+keyword
    return name
```

PINCODE AUDIT

Below were the cases where the Pincode was wrongly keyed in with additional special characters

'560032,' '- 560094,'

```
#Code for correction of PinCodes
def Correct_Pincode(postcode,Corrected_Pin):
    new_postcode = postcode
    if len(postcode) >6:
        print("Problem PostCodes:")
        print (postcode)
        new_postcode = postcode.replace(",","").replace(" ","").replace("-","")
        if new_postcode not in Corrected_Pin:
            Corrected_Pin.append(new_postcode)
```

```
    return new_postcode,Corrected_Pin
```

Problem PostCodes: 560032, Problem PostCodes: - 560094, Problem PostCodes: 560 064 Corrected PostCodes: ['560032', '560094', '560064']

DATABASE OVERVIEW

Basic Overview about the data is covered below

File Sizes

- Vpura2.osm 93MB
- Vidyaranyapura.db....58MB
- nodes.csv.....34MB
- nodes_tags.csv.....295KB
- ways.csv.....5MB
- ways_nodes.csv.....12MB
- ways_tags.csv.....3MB

```
<h5> Number of Nodes </h5>
```

SELECT COUNT(*) FROM NODES; 412444

Number of Ways

SELECT COUNT(*) FROM WAYS; 97647

Number of Unique Users

SELECT COUNT(DISTINCT(United.UID)) FROM (SELECT UID FROM NODES UNION ALL SELECT UID FROM WAYS)United; 386

Top 10 Users and Contribution

```
SELECT United.User,count(*)as num
FROM (SELECT UID,User FROM NODES UNION ALL SELECT UID,User FROM WAYS)United
GROUP BY United.UID
ORDER BY NUM DESC
LIMIT 10;

himabindhu|67454
shekarn|61009
jasvinderkaur|39063
kranthikumar|33655
saikumar|32520
sampath reddy|25837
himalay|21103
```

```
vamshiN|19593
harishk|19465
harisha|17811
```

Average User Contribution

```
SELECT avg(num) FROM
(SELECT United.User as user ,count(*)as num FROM
(SELECT UID,User FROM NODES UNION ALL SELECT UID,User FROM WAYS)United
GROUP BY United.UID)grouped;


1321.4792746114
```

Additional Data Exploration

1. Top 10 Amenities in the area

```
SELECT value,count(*) as num
FROM Node_tags
WHERE key= "amenity"
GROUP BY value
ORDER by num desc
LIMIT 10;

restaurant|103
atm|86
bank|78
place_of_worship|65
pharmacy|50
fuel|45
fast_food|42
school|35
hospital|31
cafe|27
```

2. Popular banks (based on presence of branch and ATM)

```
SELECT value,count(*)as num from Node_tags
WHERE id in (select id FROM Node_tags WHERE key="amenity" and (value ="atm" or
value ="branch"))
and key ="operator"
GROUP BY value
ORDER BY num DESC
LIMIT 10;

Axis Bank|8
Canara Bank|8
State Bank of India|7
ICICI Bank|7
```

```
HDFC|6
Citibank|2
Dhanalakshmi Bank|2
Vijaya Bank|2
IDBI bank|1
Indian Bank|1
```

3.Popular Cuisines

```
SELECT value,count(*)as num from Node_tags
WHERE id in (select id FROM Node_tags WHERE key="amenity" and value ="restaurant")
and key ="cuisine"
GROUP BY value
ORDER BY num DESC
LIMIT 5;

indian|17
regional|13
vegetarian|9
chinese|8
chicken|4
```

CONCLUSION

The User entered OSM data for Vidyaranyapura has much more opportunities for exploration. The major issue faced in this exploratory study is that of making the entered data conform to a uniform set of naming standards which would help in retrieval of correct data on the groupby operations. There were junk values found in certain tags and the same has been cleaned to a certain extent to practice and apply the techniques learned in the course.

However the accuracy of the data entered by users is beyond the scope of this project.

OTHER IDEAS

1. There should be a form of input language check at the time of submission of data which prevents user from entering data in native language
2. Validation of Pincodes with respect to number of digits can be done at the time of input

3.A general acceptable format for various input fields to cross validate the data entered before commit

4.Famous Landmarks in the area to be made available to the viewers which in turn will drive more trafiic and hence more popularity for the project

PROBLEMS WITH INPUT PARSER

1.Too much of checking and validation might dissuade the users from making entries making the addition of data a very long drawn process

REFERENCES

-https://docs.python.org/2/library/sqlite3.html

-http://stackoverflow.com/questions/2887878/importing-a-csv-file-into-a-sqlite3-database-table-using-python

-https://docs.python.org/2/library/sqlite3.html

-http://www.dreamincode.net/forums/topic/190453-importing-csv-into-sqlite3-db/