DATA WRANGLING

I've chosen Pune City because, it's one most of the most commercial cities in India .Most of the street Names, Places and areas are named just like any other cities in USA .so, It will be reliable to identify the names and cleaning them

Reference link for Pune.osm: https://mapzen.com/data/metro-extracts/metro/pune_india/

Size: 293MB (uncompressed)

Firstly, I've identified some street types from the imported data by matching the last name of street name (i.e. usually a street name ends with a street type) with regular expression function I've found that many of the last lame of the street name were entered wrongly or by using shortcuts (i.e. Road as rd or Rd. etc..) and some street names ended with numerical numbers representing the serial number of respective street name(eg: Lane 1, Avenue 4 etc..) and some are ended with ', pune '.

Secondly, since Pune is a district I want to differentiate Pune city from Pune district ,I found that Postal code can be used essentially to differentiate pune city from pune District ie..areas with postal codes ranging from 411001 and 411053 lie inside pune city region and rest lie outside pune city .I found that 37 postal codes lie outside pune city ,647 line inside pune city circumference and 13 postal codes were wrongly specified

AUDITING:

1.I've created a list for Wrongly specified street types and updated the wrongly entered street names by mapping, removed the numerical numbers at the end of street names and replaced "," or ",pune" with ""

2.In the first iteration of postal code Auditing ,I found and updated the postal codes which has white space charecters present in postal codes.

3.In the second iteration,I found that some Postal codes were entered as string and In third iteration I found some pincodes wrongly entered as a string and colon " : " present in the string with help of regular expression("^([a-z]|_)+:") I've updated the both postal codes as NONE

4.Below there are some data mentioned before and after Auditing

STREET NAMES:

Before Auditing		After Auditing
Kalyani nagar	-	Kalyani Nagar
Gulawani Maharaj Rd	-	Gulawani Maharaj Road
Hinjewadi Phase 1	-	Hinjewadi Phase
Echinus Court road	-	Echinus Court Road
Vega Center Swargate,	-	Vega Center Swargate
finolex chaulk	-	finolex Chowk

POSTAL CODES:

Before Auditing		After Auditing
410506	-	None
Paschimanagari	-	None
411 021	-	411021

Preparing for Data Base:

After auditing the data , the cleaned data is exported into respective dictionaries nodes,nodes_tags,ways,ways_nodes and ways _tags in CSV format the sizes of the files are mentioned below

FILE		SIZE
nodes.csv	-	113 MB
nodes_tags.csv	-	490 KB
ways.csv	-	15.7 MB
ways_nodes.csv	-	39 MB
ways_tags.csv	-	8.98 MB

QUERIES USING DATA BASE:

I've created a data base named pune.db with schema as specified explored the following questions using queries

1. Number of Nodes:

```
#number of "node" tags
import sqlite3
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT count(*)as num from nodes;
'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
print df
db.close()
```

```
         0
0  1416239
```

> There are **1416239** nodes

2. Number of Ways:

```
# number of 'way'tags
import sqlite3
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT count(*)as num from ways;
'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
print df
db.close()
```

```
        0
0  270156
```

> There are **270156** ways

3. Number of Unique users:

```
# number of unique users
import sqlite3
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT DISTINCT(user) from (select user from nodes UNION ALL select user from ways);

'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
#print df
print len(df)

db.close()
```

```
670
```

> There are 670 unique users

4.Number of amenities available in pune:

```
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT value,count(*)as num from (select value,key from nodes_tags  UNION ALL select value,key from ways_tags)
where key='amenity'
group by value
order by num desc
;
'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
print "Number of amenities  available in PUNE CITY :", len(df)
db.close()
```

```
Number of amenities  available in PUNE CITY : 52
```

5.Top 5 Amenities:

```
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT value,count(*)as num from (select value,key from nodes_tags  UNION ALL select value,key from ways_tags)
where key='amenity'
group by value
order by num desc
limit 5
;
'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
print df
db.close()
```

```
           0    1
0  restaurant  160
1      school  112
2        fuel   69
3    hospital   56
4        cafe   55
```

> so **'restaurant'** tops the amenities list with **160** followed by **"School"** with **112**

ADDITIONAL IDEAS:

User and Contribution facts:

```
#user with higest contribution
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT user,count(user)  from (select user from nodes UNION ALL select user from ways)
group by user
order by  count(user) desc
;
'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
print df[1].describe()
print ("\n")
print "Total Contributions :",df[1].sum()

db.close()
```

```
count      670.000000
mean      2517.007463
std       9387.334990
min          1.000000
25%          1.000000
50%          6.000000
75%         53.750000
max      96812.000000
Name: 1, dtype: float64


Total Contributions : 1686395
```

- Total 1686395 contributions were received.
- User named Singleton has contributed more than any other contributors.
- 179 users holds the least contribution rate with only one post of contribution.
- The Average Contribution of posts per user is 2517 .

Additional Data Exploration:

Top 5 Contributers:

```
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT user,count(user)  from (select user from nodes UNION ALL select user from ways)
group by user
order by  count(user) desc
limit 5;
'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
print df
db.close()
```

```
              0      1
0      singleton  96812
1    harishvarma  60144
2  jasvinderkaur  57697
3        sramesh  57627
4       praveeng  56788
```

Cuisines available and their count :

```
#amenities and their count
db = sqlite3.connect("pune.db")
c = db.cursor()
QUERY = '''SELECT value,count(*)as num from (select value,key from nodes_tags  UNION ALL select value,key from ways_tags)
where key='cuisine'
group by value
order by num desc
;
'''
c.execute(QUERY)
rows = c.fetchall()
import pandas as pd
df = pd.DataFrame(rows)
print "Number of amenities  available in PUNE CITY :", len(df)
print (df)

db.close()
```

```
Number of amenities  available in PUNE CITY : 14
                                    0   1
0                              indian  58
1                              burger  18
2                           coffee_shop  14
3                                 pizza   9
4                              sandwich   9
5                               italian   6
6                               Mastani   3
7                           North_Indian   3
8   Regional,_India,_Tandoor,_Chinese   3
9                                 cafe   3
10                             chicken   3
11                             chinese   3
12                               rolls   3
13                                thai   3
```

Suggestion:

When it comes to user Contribution, I'm reminded of "gamification" as an interesting force for contribution within the context of the OpenStreetMap, if user knowledge were a lot of conspicuously displayed, maybe others would take associate degree initiative in submitting a lot of edits to the map. And, if everybody sees that solely a few of power users area unit making over 80-90% a of given map, that may spur the creation of a lot of economical bots, particularly if bound gamification components were gift, like rewards, badges, or a leader board.

Advantages and Disadvantages :

The main advantage is that Open Street Map data is open-source and therefore free to use. This means anyone can use the data to create their own maps (and then use services like Map Box to generate and host customised map tiles). This means the developer doesn't have to work within Google's constraints.

The only imaginable downside to me is quality. Get me right, 99% is the good stuff, but as all crowd sourced data it's hard to maintain consistent quality control. Of course is free to alter and complete the data, but there's no guarantees as there would with a company behind it.

Conclusion:

After series of iterations in Auditing process, I believe that the data has been cleaned precisely and analysed well in exploration phase .