Project 4: Street Map Case Study

Location:

Houston, TX

- https://www.openstreetmap.org/node/27526178

Reason: The Reason I choose Houston, TX because it the city I was born and raise and it have a special value to me. It been home for me for the last 25 years so I have a good knowledge of the area and its surroundings.

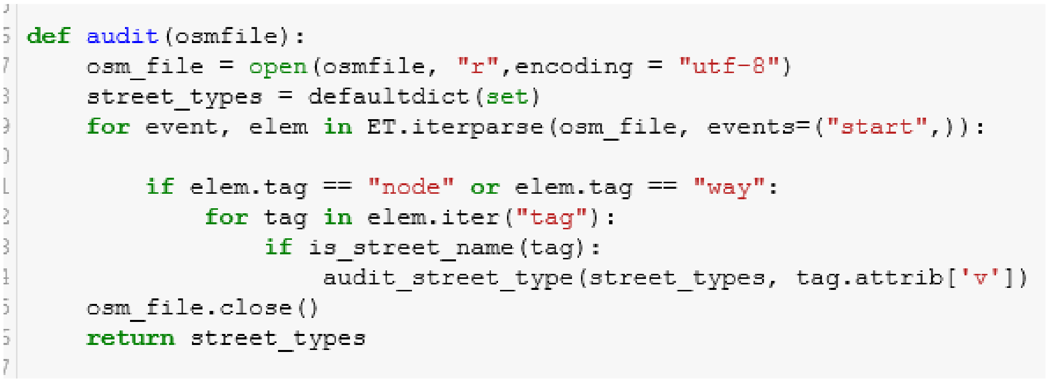
Problem: The First problem that I see is that most of the street name end with abbreviation for example:

- St should be Street
- Ave should be Avenue
- Blvd should be Boulevard
- E should be East
- Dr should be Drive

Another problem that I see that some key that are equal to Highway have a value with a lower case character instead of an upper case. For Example:

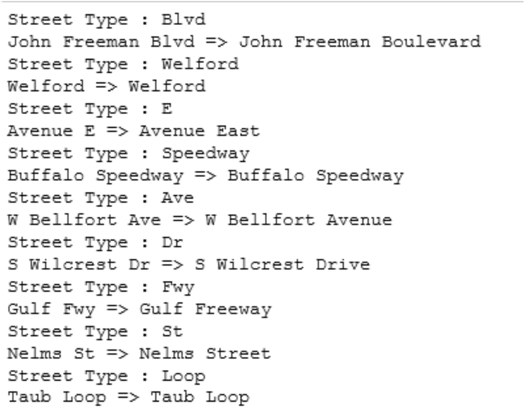
- residential should be Residential
- service should be Service
- motorway_link should be Motorway_Link

Cleaning: Before converting everything to XML format(csv) I have to clean the data so that my result are more accurate. This also help to be more organize with the data. The first code cleans the data for street abbreviation so instead of having Dr. in will convert to Drive.



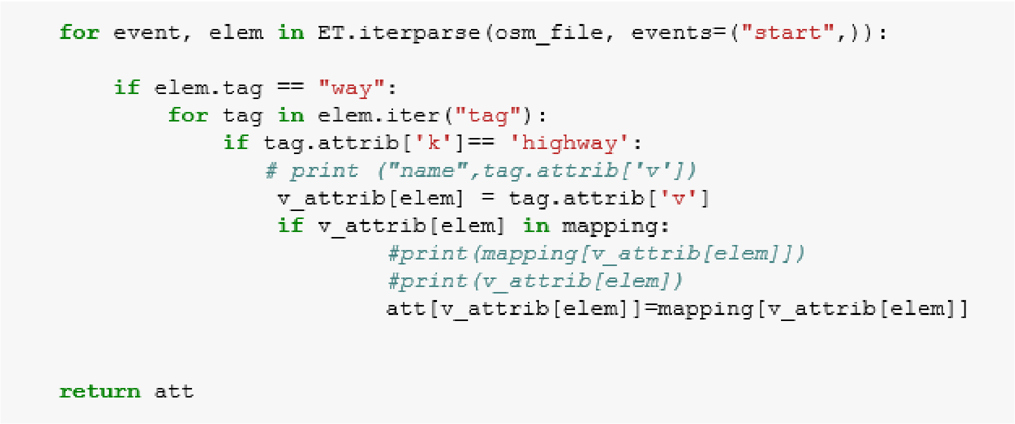
```
def audit(osmfile):
    osm_file = open(osmfile, "r",encoding = "utf-8")
    street_types = defaultdict(set)
    for event, elem in ET.iterparse(osm_file, events=("start",)):

        if elem.tag == "node" or elem.tag == "way":
            for tag in elem.iter("tag"):
                if is_street_name(tag):
                    audit_street_type(street_types, tag.attrib['v'])
    osm_file.close()
    return street_types
```



```
Street Type : Blvd
John Freeman Blvd => John Freeman Boulevard
Street Type : Welford
Welford => Welford
Street Type : E
Avenue E => Avenue East
Street Type : Speedway
Buffalo Speedway => Buffalo Speedway
Street Type : Ave
W Bellfort Ave => W Bellfort Avenue
Street Type : Dr
S Wilcrest Dr => S Wilcrest Drive
Street Type : Fwy
Gulf Fwy => Gulf Freeway
Street Type : St
Nelms St => Nelms Street
Street Type : Loop
Taub Loop => Taub Loop
```

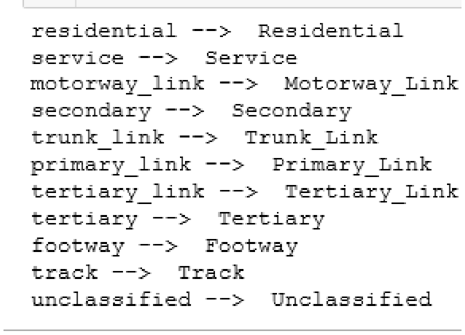
The code above going in and find the key and value of the address and get the last value of the street and replace with the full name of the street example Fwy into Freeway. This help tell the difference between the streets.



```
for event, elem in ET.iterparse(osm_file, events=("start",)):

    if elem.tag == "way":
        for tag in elem.iter("tag"):
            if tag.attrib['k']== 'highway':
               # print ("name",tag.attrib['v'])
                v_attrib[elem] = tag.attrib['v']
                if v_attrib[elem] in mapping:
                        #print(mapping[v_attrib[elem]])
                        #print(v_attrib[elem])
                        att[v_attrib[elem]]=mapping[v_attrib[elem]]


return att
```

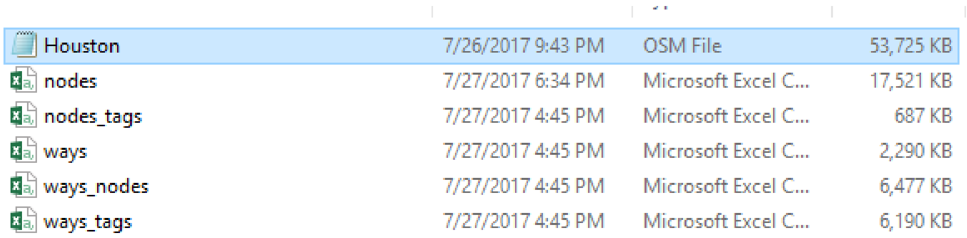


```
residential -->  Residential
service -->  Service
motorway_link -->  Motorway_Link
secondary -->  Secondary
trunk_link -->  Trunk_Link
primary_link -->  Primary_Link
tertiary_link -->  Tertiary_Link
tertiary -->  Tertiary
footway -->  Footway
track -->  Track
unclassified -->  Unclassified
```

In this code above it goes into file and find the key highway and locate the value and replace it with an uppercase letters. This something important that should be uppercase than lowercase.

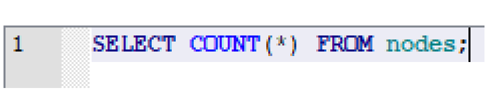
SQL: After Cleaning the code I use SQL find information about Houston like find the most zip code being use , The type of food in the area, and more information like that. This help give a picture to the reader a better idea about the data that is given.

Data Overview:

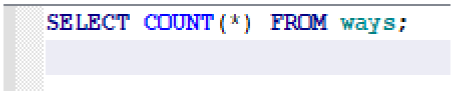


Houston	7/26/2017 9:43 PM	OSM File	53,725 KB
nodes	7/27/2017 6:34 PM	Microsoft Excel C...	17,521 KB
nodes_tags	7/27/2017 4:45 PM	Microsoft Excel C...	687 KB
ways	7/27/2017 4:45 PM	Microsoft Excel C...	2,290 KB
ways_nodes	7/27/2017 4:45 PM	Microsoft Excel C...	6,477 KB
ways_tags	7/27/2017 4:45 PM	Microsoft Excel C...	6,190 KB

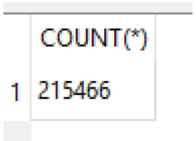
Nodes and Ways:



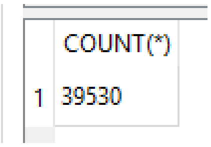
```
SELECT COUNT(*) FROM nodes;
```



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

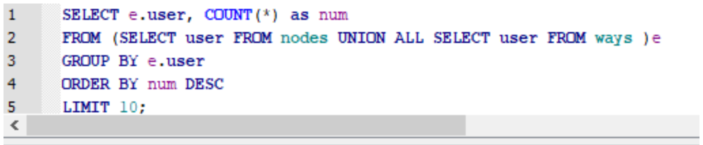


	COUNT(*)
1	215466

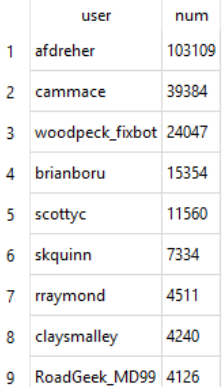


	COUNT(*)
1	39530

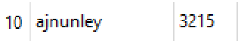
Top 10 Users:



```
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;
```

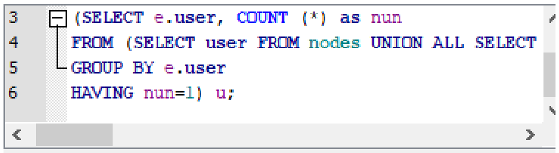


	user	num
1	afdreher	103109
2	cammace	39384
3	woodpeck_fixbot	24047
4	brianboru	15354
5	scottyc	11560
6	skquinn	7334
7	rraymond	4511
8	claysmalley	4240
9	RoadGeek_MD99	4126

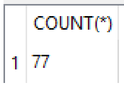


10	ajnunley	3215

Users appearing more than Once :



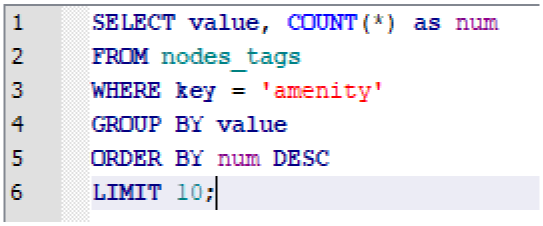
```
(SELECT e.user, COUNT (*) as nun
FROM (SELECT user FROM nodes UNION ALL SELECT
GROUP BY e.user
HAVING nun=1) u;
```



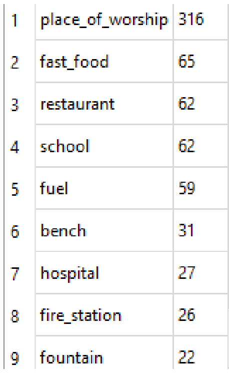
	COUNT(*)
1	77

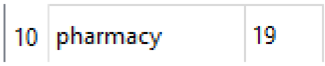
Additional Information

Amenity:



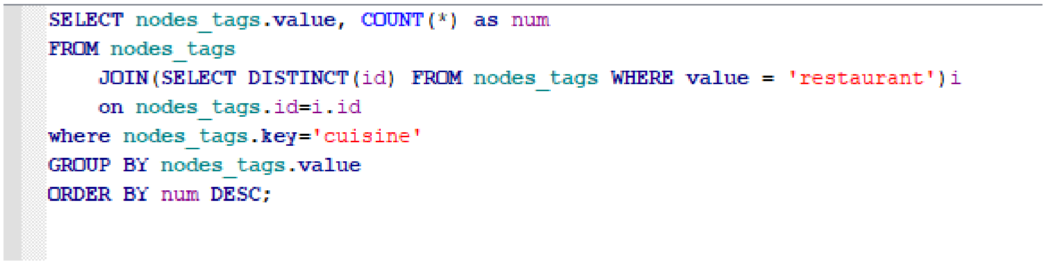
```
SELECT value, COUNT(*) as num
FROM nodes_tags
WHERE key = 'amenity'
GROUP BY value
ORDER BY num DESC
LIMIT 10;
```



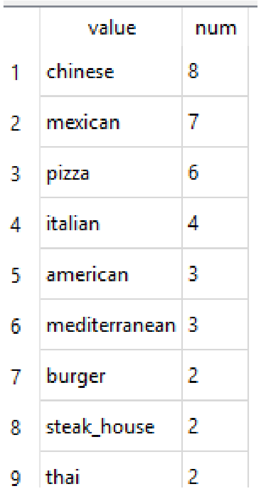


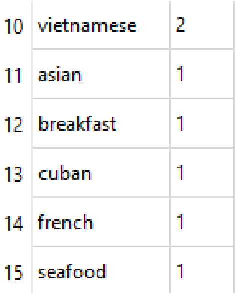
1	place_of_worship	316
2	fast_food	65
3	restaurant	62
4	school	62
5	fuel	59
6	bench	31
7	hospital	27
8	fire_station	26
9	fountain	22
10	pharmacy	19

Cuisine :



```
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;
```





	value	num
1	chinese	8
2	mexican	7
3	pizza	6
4	italian	4
5	american	3
6	mediterranean	3
7	burger	2
8	steak_house	2
9	thai	2
10	vietnamese	2
11	asian	1
12	breakfast	1
13	cuban	1
14	french	1
15	seafood	1

Additional Ideas:

You can go more in dept. by categorizing them for example find the food and how many of them are burger shop, and how many of them are McDonalds and How many McDonalds are in a specific range. The bad thing from this idea is that some food shop is bundle together so you might have a McDonalds that in a gas station that might not count.

Limitation :

There was a couple of limitation with the xml data. The data was mostly clean their wasn't a lot of problem with values. I have to pay attention more closely to the data to spot a mistake that could be clean.

Conclusion:

In Conclusion this project help understand better how to take data from a xml and search and clean the data and convert it to a csv format. Also have a better understanding how SQL work and get information from tables that we already have. Overall this give me a better understanding how the data wrangling works and how it important it is to Data Science .