Simon Zhang

Data Wrangling Project

I chose Houston, Tx because I was born and raised in Houston! Below is a link to the OSM
file I downloaded:
https://mapzen.com/data/metro-extracts/metro/houston_texas/

DATA WRANGLING AND CLEANING:
The two main issues with this dataset that I cleaned up where Street Type abbreviations
and postcodes. I programmatically standardized both those pieces of data.

After auditing the street types,
here is a mapping of the main street types I wanted to change in my dataset:

```
mapping = { 'Ave': 'Avenue',
                    'Ave.': 'Avenue',
                    'Blvd': 'Boulevard',
                    'Blvd.': 'Bouelvard',
                    'Dr': 'Drive',
                    'Frwy': 'Freeway',
                    'Fwy': 'Freeway',
                    'HIGHWAY': 'Highway',
                    'Ln': 'Lane',
                    'Pkwy': 'Parkway',
                    'Rd': 'Road',
                    'St': 'Street',
                    'Stree': 'Street',
                    'street': 'Street'
        }
```

I used this function to clean the street types:

```
def update_name(name, mapping):

    list_name = name.split(' ')
    last_type = list_name.pop()
    if last_type in mapping:
            new_type = mapping[last_type]
            list_name.append(new_type)
            name = ' '.join(list_name)

            return name
    else:
            return name
```

After auditing the postcodes, I found that there were codes such as:
TX77024 with TX in front of the zip code
and,

77384-xxxx with '-xxxx' after the zipcode

I wrote a cleaning function that took just the 5 digit zip code and ignored the TX and
the '-xxxx' parts:

```
def fix_postcode(code):
        if len(code) >= 5:
                start_postcode = code.find('7')
                end_postcode = start_postcode + 5
                new_code = code[start_postcode:end_postcode]
                return new_code
        else:
                return code
```

I noticed there were also many tags with 'tiger' as attributes (The Topologically Integrated Geographic Encoding and Referencing system). After looking on the Udacity forums about the details of these tags, I decided to leave these 'tiger' attributes how they were since it was a reliable source of data.

STATISTICAL OVERVIEW OF DATASET:

1. SIZE OF FILE
I used the following queries to find the file size of houston.db:
PRAGMA page_size;
4096
PRAGMA page_count;
121466

Then I multipied 4096 * 121466 and divided by 1,000,000 to convert to Mb from bytes=
497.52 Mb

2. NUMBER OF UNIQUE USERS
I ran the query:

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

Returned:
1600 unique users

3. NUMBER OF NODES AND WAYS
I ran the query:

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

3031834

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

366755

4. NUMBER OF CHOSEN TYPE OF NODES
I ran this query to find the number of poles in Houston:

```
SELECT COUNT(*)
FROM nodes_tags
WHERE value='pole';
```

11190

I ran this query to find the number of places of worship in Houston:

```
SELECT COUNT(*)
FROM nodes_tags
WHERE value='place_of_worship';
```

2219

5. ADDITIONAL STATISTICS
I ran this query to find the Top 3 Users who made a 'pole' node:

```
SELECT user, COUNT(*) as num
FROM nodes, nodes_tags
WHERE nodes.id=nodes_tags.id and nodes_tags.value = 'pole'
GROUP BY user
ORDER BY num DESC
LIMIT 3;
```

42429, 10564
Rallysta74, 379
beweta, 64

I ran this query to find the Top 10 contributing 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;
```

```
woodpeck_fixbot,569278
TexasNHD,544483
afdreher,473340
scottyc,205303
cammace,193159
claysmalley,136594
brianboru,118555
skquinn,86265
RoadGeek_MD99,82255
Memoire,56679
```

6. SUGGESTIONS ON IMPROVING DATA OR ANALYSIS

This dataset can be improved by partnering up with a game such as Pokemon Go to get all users of the game involved in fixing the dataset. If there were certains spaces in the map that were still unknown, Pokemon Go could just put some kind of rare Pokemon or some kind of rare item incentive for the player of the game to go to that location and input the data about that place in order to get the certain in-game incentive. That way it gets all users of the game to help fix the map.

I feel like the input should also have restricted inputs as well.
If there were more drop down menus with 'street type' or had a drop down menu with a 5 number restricted zip code input then it would also automatically make the data cleaner.

LIST OF REFERENCES:

I used Udacity course videos and Udacity Forums and www.sqlite.org to complete this
project.