OpenStreetMap Project 3
Data Wrangling with SQL

San Jose CA, United States
https://www.openstreetmap.org/relation/112143

(1) I manually looked over a 1/1000 sample of the file. The only thing that jumped out at me is that ZIP codes were mostly five digit but some were nine digit. While nine digit ZIP codes, might be useful for some purposes, for most purposes, five digit will suffice. If I knew that nine digits would be useful, I might dummy out the five digits as 94123-0000. I created a **fiz_zip function** to truncate all ZIPs at five digits.

```
def fix_zip(zip):
    new_zip = zip[:5]
    print new_zip
    return new_zip
```

(2) I then ran Unexpected Street Types on the entire file. The results are located in **unexpected.pdf**. There were only 57 results so I was able to check all them manually. Some changes were pretty obvious to make, such as:
Ln > Lane
ave > Avenue
Boulvevard > Boulvevard
I placed these corrections in the **Mapping dictionary**.

```
mapping = { "St": "Street",
            "St.": "Street",
            "Ave":"Avenue",
            "Rd.":"Road",
            "6":"",
            "Winchester":"Winchester Boulevard",
            "Ln":"Lane",
            "Rd.":"Road",
            "114":"",
            "A":"",
            "ave":"Avenue",
            "20":"",
            "Boulvevard":"Boulevard",
            "1":"",
            "Hamilton":"Hamilton Road",
            "Hwy":"Highway",
            "Dr":"Drive",
            "CA":"",
            "0.1":"",
            "Bellomy":"Bellomy Street",
            "Cir": "Circle",
            "Franklin":"Franklin Street",
            "Bascom":"Bascom Rd",
            "Julian":"Julian Street",
            "street":"Street",
```

```
            "Blvd":"Boulevard",
            "Ct":"Court",
            }
```

Others errors were more one-off data entry type of mistakes.
For example, “7.1” came from “Hwy 17 PM 7.1”, so I decided to change this to Highway 17, as I am familiar with that road. If this was a “real life” project I would have done a little research to verify that these changes were correct. I placed these changes in the **function fix_misc**.

```
def fix_misc(snippet):
    if snippet == "Great American Pkwy Ste 201":
        new_snippet == "Great American Parkway"
        print "**"
        print new_snippet
        return new_snippet
    elif snippet == "rio robles":
        new_snippet = "Rio Robles Drive"
        print "**"
        return new_snippet
    elif snippet == "Rio Robles":
        new_snippet = "Rio Robles Drive"
        print "**"
        return new_snippet
    elif snippet == "Zanker Road, San Jose,":
        print "**"
        new_snippet = "Zanker Road"
        return new_snippet
    elif snippet == "Zanker Road, San Jose,":
        print "**"
        new_snippet = "Zanker Road"
        return new_snippet
    elif snippet == "wilcox Avenue":
        print "**"
        new_snippet = "Wilcox Avenue"
        return new_snippet
    elif snippet == "Ala 680 PM":
        print "**"
        new_snippet = "Unknown"
        return new_snippet
    elif snippet == "Hwy 17 PM 7.1":
        print "**"
        new_snippet = "Highway 17"
        return new_snippet
    elif snippet.endswith("#"):
        print "**"
        new_snippet = snippet[:-2]
```

```
        return new_snippet
    else:
        new_snippet = snippet
        return snippet
```

(3) Here are the file sizes

```
sanjose.osm ......... 287 MB
OSM.db .......... 808 MB
nodes.csv ............. 107 MB
nodes_tags.csv ........ 2.5 MB
ways.csv .............. 10   MB
ways_tags.csv ......... 21 MB
ways_nodes.cv ......... 35 MB
```

(4) Some SQL queries

# of nodes
SELECT COUNT(*) FROM nodes;
7691549

# of ways
SELECT COUNT(*) FROM ways;
1021757

# of UIDs
SELECT COUNT(DISTINCT(e.uid))
FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) e;
1241

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;
nmixter,1730268
mk408,915108
"Bike Mapper",495300
samely,466572
dannykath,432264
RichRico,428760
karitotp,340860
n76,229458
matthieun,196626
"Minh Nguyen",196590

Top 10 ZIP codes

```
SELECT tags.value, COUNT(*) as count
FROM (SELECT * FROM nodes_tags
      UNION ALL
      SELECT * FROM ways_tags) tags
WHERE tags.key='postcode'
GROUP BY tags.value
ORDER BY count DESC
LIMIT 10;
```

```
95014,21023
95070,1365
94087,1142
94086,1050
95051,944
95037,933
95054,545
95127,538
95125,519
95050,476
```

Top 10 amenities

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

```
restaurant,3825
fast_food,1845
cafe,1120
place_of_worship,940
bicycle_parking,875
bench,870
school,760
toilets,735
fuel,615
bank,580
```

Top 10 restaurant cuisines

```
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;
chinese,305
vietnamese,295
mexican,255
pizza,245
japanese,195
indian,145
italian,130
thai,125
american,115
sushi,95

(4) The node and way tags in San Jose generally seemed pretty sparse. If we needed more data we would need to think about how to incentivize mappers.
Since this exercise was for data cleaning, not data analysis per se, most of my thoughts revolved around cleaning data. Since there are 7,691,549 nodes and 1,021,757, there is no way I caught all the errors a first time around. And in fact my **fix_misc function** does contain a few corrections that I noticed from my first pass ata data cleaning. For instance, Unexpected Street Types turned up several streets ending in a digit. There turned out mostly to be apartment numbers, e.g., "#6". So in the mapping dictionary I replaced each of those instanced with "" This left a trailing "#" in the address so in **fix_misc** I deleted trailing "#". Also, while reviewing the cleaned data from the first time, I just happened to notice additional miscellaneous errors and added them **fix_misc** and the mapping dictionary.
Thus it must be important in a real project to understand the requirements and intents, in order to clean up the data from the right perspective and to an adequate standard. In a file this large, it would be very time-consuming to clean up all the data (diminished returns for amount of effort). So likely a certain standard such as 99.9% might be defined. It is also likely an iterative process, where you get asymptotically closer to perfect each iteration.