In this project, I will use data wrangling methods to process OpenStreetMap data of xml format by following steps:

- Introduce the dataset.
- Audit the data.
- Provide an overview of the dataset
- Provide some statistical discovery.

1. Introduce the Dataset

Map Area : Beijing, China
link https://mapzen.com/data/metro-extracts/metro/beijing_china/

I choose Beijing because I'm familiar with it. Since the size of dataset is less than 200 MB, which I think is not large. Besides, in order to survy the whole dataset, I ues the whole dataset instead of sampling to observe it.

2. Audit the Data

Unique Element Type Numbers

bounds: 1
member: 60824
nd: 892099
node: 747571
osm: 1
relation: 5618
tag: 329157
way: 110210

Contribution Numbers

1633

Number of 4 Kinds of Tag Categories

tags that contain only lowercase letters and are valid : 309200
tags with a colon in the value : 19026
tags with problematic characters : 5
other tags : 926

Problems Encountered

Since there may be some unqualified data in the set. I must audit the dataset and make some correction strategies to process the dataset. I choose these 5 important attributes to audit: **addr:postcode**, **addr:city**, **addr:street**, **addr:street:en**, **addr:country**. After auditing those 5 attributes, I find 3 problems as following:

- Since the first 2 places of postcode in Beijing is '10' and the code has a length of 6, there are incorrect codes (010-62332281, 10043, 10043, 10080, 3208)
- Most city names are correct, except string 'yes'.
- Some English overabbreviated street names are incorrect, like 'Jie','Lu','St','road','Rd','Str','Rd.', and so on.

Here are the methods to solve those problems:

Postal Codes

There are only 6 incorrect codes: 010-62332281, 10043, 10043, 10080, 3208. Depend on the characteristic of incorrect codes, I make following strategies to rectify the wrong codes:

- For those codes shorter than 6 , if the first 2 place is not '10' and the length is 4 , then I will add '10' in front of the code. If the first 2 place is '10', I will add '0' at the end of the code till the length reach 6. Otherwise I will remove the code.
- For the code longer than 6 with '-' in the string, it's obviously a phone number, I will remove it.
- For code '110101', it represents another city in China, so I will remove it.

Here are the results:

```
'010-62332281' => ''
'10040' => '100400'
'110101' => ''
'10043' => '100430'
'10080' => '100800'
'3208' => '103208'
```

City Name

Since there is only one incorrect city name: 'yes'. I just change string 'yes' to 'Beijing'.

Overabbreviated Street Names

Srtreet names in Chinese are correct. But some translated street names are incorrect. Considering the complicated diversification of Chinese language, it's difficult to arrange all the street types. So I will recify incorrect names like this:

- First, collecting incorrect English names.
- Then based on it, creating a mapping of correct names.
- *Finally, updating the incorrect street names with correct ones.

Here are the mappings:

```
mapping = { 'Ave':'Avenue',
        'ave.':'Avenue',
        'Rd':'Road',
        'Rd.':'Road',
        'Lu':'Road',
        'road':'Road',
        'St': 'Street',
        'St.': 'Street',
        'Str':'Street',
        'street':'Street',
        'Jie':'Street',
        'jie':'Street' }
```

3. Overview of Dataset

File Sizes

beijing*china.osm .... 166.3 MB*
OpenStreetMap.db .... 85.7 MB
nodes.csv .... 61.3 MB
*nodes*tags.csv .... 2.9 MB
ways.csv .... 6.5 MB
ways*tags.csv .... 7.7 MB*
*ways*nodes.csv .... 21.5 MB

Number of Unique Users

```
select count(distinct user) as user_number
from(select user from nodes
     union all
     select user from ways) t;
```

```
1614
```

Number of Nodes

```
select count(id) as number
from ways;
```

```
110210
```

Top 10 Contributors

```
select user, count(id) as total_number
from ( select id, user from nodes
       union all
       select id, user from ways) t
group by user
order by total_number desc
limit 10 ;
```

```
Chen Jia : 193838
R438 : 150042
hanchao : 64094
ij_ : 52045
katpatuka : 23828
m17design : 21823
Esperanza36 : 18652
nuklearerWintersturm : 17155
RationalTangle : 14356
u_kubota : 9386
```

Number of Hospitals

```
select count(distinct id) as number
from nodes_tags
where value = 'hospital';
```

```
104
```

Number of Schools

```
select count(distinct id)
from nodes_tags
where value = 'school';
```

```
163
```

4. Additional Discovery of Data

Total Shop Number

```
select count(distinct id) as number
from nodes_tags
where key='shop'
group by key;
```

```
1383
```

Top 10 Shop Types

Here is top 10 different kinds of shops and it's amount.

```
select value, count(distinct id) as number
from nodes_tags
where key='shop' and value != 'yes'
group by value
order by number desc
limit 10;
```

```
supermarket : 239
convenience : 202
clothes : 90
bakery : 70
hairdresser : 66
kiosk : 58
bicycle : 40
electronics : 36
beauty : 32
copyshop : 27
```

Total Cafe Number

```
select count(distinct id) as number
from nodes_tags
where key='amenity' and value = 'cafe';
```

```
228
```

Top 10 Cafes

Here is top 10 cafes based on it's amount.

```
select t2.value, count(distinct t2.id) as number
from( select id from nodes_tags where key='amenity' and value = 'cafe' ) t1
 join (select id, value from nodes_tags where key='name') t2 on t1.id = t2.id
group by t2.value
order by number desc
limit 10;
```

```
星巴克 : 21
Starbucks : 19
Costa Coffee : 7
上岛咖啡 : 5
UBC Coffee : 3
Jazz Cafe : 2
Paradiso Coffee : 2
Paris Baguette : 2
Zoo Cafe : 2
雕刻时光 : 2
```

We have to notice that '星巴克' is the Chinese name of Starbucks, So there are 40 Starbucks stores. It's interesting that Starbucks holds a safe lead over other cafe shops.

Node Amount of Different Areas

```
select position, count(distinct id) as node_number
from(select
        case when t1.lat < t2.middle_lat and t1.lon < t2.middle_lon then 'SouthWest'
            when t1.lat < t2.middle_lat and t1.lon > t2.middle_lon then 'SouthEast'
            when t1.lat > t2.middle_lat and t1.lon < t2.middle_lon then 'NorthWest'
            when t1.lat > t2.middle_lat and t1.lon > t2.middle_lon then 'NorthEast'
        end as position, t1.id
     from nodes t1
     join(select (max(lat) + min(lat))/2 as middle_lat,
            (max(lon)+ min(lon))/2 as middle_lon from nodes ) t2 on 1=1 ) t
group by position
order by node_number desc ;
```

```
NorthWest : 272793
NorthEast : 166357
SouthEast : 165665
SouthWest : 142753
None : 3
```

There is an interesting discovery that node number in northwest of Beijing is far ahead in 4 areas. Next, let's compute amounts of other important atrributs of those 4 areas, see if there exists the same conclusion.

School Amount of Different Areas

```
select position, count(distinct id) as node_number
from(select
        case when t3.lat < t4.middle_lat and t3.lon < t4.middle_lon then 'SouthWest'
            when t3.lat < t4.middle_lat and t3.lon > t4.middle_lon then 'SouthEast'
            when t3.lat > t4.middle_lat and t3.lon < t4.middle_lon then 'NorthWest'
            when t3.lat > t4.middle_lat and t3.lon > t4.middle_lon then 'NorthEast'
        end as position, t3.id
     from(select distinct t2.lat, t2.lon, t1.id
           from nodes_tags t1
            join nodes t2 on t1.id = t2.id
          where t1.value = 'school') t3
     join(select (max(lat) + min(lat))/2 as middle_lat,
             (max(lon)+ min(lon))/2 as middle_lon from nodes ) t4 on 1=1) t
group by position
order by node_number desc ;
```

```
NorthWest : 72
NorthEast : 39
SouthWest : 29
SouthEast : 23
```

Hospital Amount of Different Areas

```
select position, count(distinct id) as node_number
from(select
        case when t3.lat < t4.middle_lat and t3.lon < t4.middle_lon then 'SouthWest'
            when t3.lat < t4.middle_lat and t3.lon > t4.middle_lon then 'SouthEast'
            when t3.lat > t4.middle_lat and t3.lon < t4.middle_lon then 'NorthWest'
            when t3.lat > t4.middle_lat and t3.lon > t4.middle_lon then 'NorthEast'
        end as position, t3.id
     from(select distinct t2.lat, t2.lon, t1.id
          from nodes_tags t1
            join nodes t2 on t1.id = t2.id
          where t1.value = 'hospital') t3
     join(select (max(lat) + min(lat))/2 as middle_lat,
              (max(lon)+ min(lon))/2 as middle_lon from nodes ) t4 on 1=1 ) t
group by position
order by node_number desc ;
```

```
NorthWest : 46
NorthEast : 25
SouthWest : 17
SouthEast : 16
```

Shop Amount of Different Areas

```
select position, count(distinct id) as node_number
from(select
        case when t3.lat < t4.middle_lat and t3.lon < t4.middle_lon then 'SouthWest'
            when t3.lat < t4.middle_lat and t3.lon > t4.middle_lon then 'SouthEast'
            when t3.lat > t4.middle_lat and t3.lon < t4.middle_lon then 'NorthWest'
            when t3.lat > t4.middle_lat and t3.lon > t4.middle_lon then 'NorthEast'
        end as position, t3.id
     from(select distinct t2.lat, t2.lon, t1.id
          from nodes_tags t1
            join nodes t2 on t1.id = t2.id
          where t1.key = 'shop') t3
     join(select (max(lat) + min(lat))/2 as middle_lat,
             (max(lon)+ min(lon))/2 as middle_lon from nodes ) t4 on 1=1 ) t
group by position
order by node_number desc ;
```

```
NorthEast : 677
NorthWest : 444
SouthEast : 161
SouthWest : 101
```

Apartment Amount of Different Areas

```
select position, count(distinct id) as node_number
from(select
        case when t3.lat < t4.middle_lat and t3.lon < t4.middle_lon then 'SouthWest'
            when t3.lat < t4.middle_lat and t3.lon > t4.middle_lon then 'SouthEast'
            when t3.lat > t4.middle_lat and t3.lon < t4.middle_lon then 'NorthWest'
            when t3.lat > t4.middle_lat and t3.lon > t4.middle_lon then 'NorthEast'
        end as position, t3.id
     from(select distinct t2.lat, t2.lon, t1.id
          from nodes_tags t1
            join nodes t2 on t1.id = t2.id
          where t1.value like '%apartment%') t3
     join(select (max(lat) + min(lat))/2 as middle_lat,
               (max(lon)+ min(lon))/2 as middle_lon from nodes ) t4 on 1=1 ) t
group by position
order by node_number desc ;
```

```
NorthWest : 23
NorthEast : 9
SouthEast : 5
SouthWest : 2
```

Office Amount of Different Areas

```
select position,count(distinct id) as node_number
from(select
        case when t3.lat < t4.middle_lat and t3.lon < t4.middle_lon then 'SouthWest'
            when t3.lat < t4.middle_lat and t3.lon > t4.middle_lon then 'SouthEast'
            when t3.lat > t4.middle_lat and t3.lon < t4.middle_lon then 'NorthWest'
            when t3.lat > t4.middle_lat and t3.lon > t4.middle_lon then 'NorthEast'
        end as position,  t3.id
    from(select distinct t2.lat, t2.lon, t1.id
         from nodes_tags t1
            join nodes t2 on t1.id = t2.id
         where t1.key = 'office') t3
    join(select (max(lat) + min(lat))/2 as middle_lat,
            (max(lon)+ min(lon))/2 as middle_lon from nodes ) t4 on 1=1 ) t
group by position
order by node_number desc ;
```

```
NorthEast : 72
NorthWest : 44
SouthEast : 26
SouthWest : 10
```

We can that the amounts of apartment, shop, hospital, school in northwest are all far ahead over other three areas. Which means this area is the gathering place of students and residents, and lots of people live here. While the amount of office in northeast is the largest. Which means it's the gathering place of companies, and lots of people work here.

Conclusion

Through out the process of audditting data, analyzing data by sql. We can find some meaningful things, especially the probably population distribution such as aggregate region of residents and students, and the gathering place of companies. During the process of wrangling dataset, I thought about two suggestions for improvement and analyzing data.

- As for the map data of Beijing, some attributes only have values in Chinese, some only have English version, and some have both of them. Which will cause reading difficulties. To solving this problem, I suggest a rule that for some specified attributes, contributors should provide information both in English and Chinese, and other specified attributes should be only in English.
    - **Benefit**: The dataset will be more consistency, clear presented and readable. For analyzers like us can query data more convenient, do not have to distinguish between different languages.
    - **Anticipated Problem**: Some attributes are hard to translate between two different languages, and contributors are on various levels of language proficiency.

- The data of some areas is not adequate. As in this example, there are some main streets I know but not appear in the map. I think it's owing to lack of popularizing and contributors. We should strengthen publicity and attact more contributors. Besides, I think seeking subsidize from those who use OSM to award contributors is a good way to attract more and more contributors.

- **Benefit**: With more contributors, OSM will contain larger amount of information, and leading to more uesers, which in return will lead to more contributors. That's a good circulation.
- **Anticipated Problem**: Since there are lots competitiors do the similar thing, how to make OSM more attractive will not be easy.