OpenStreetMap 项目

地区 - 上海

> 中国，上海市

- https://mapzen.com/data/metro-extracts/metro/shanghai_china/
- 位置简介：

  上海市，简称**沪**，别称**申**，中华人民共和国直辖市，中国大陆经济、金融、贸易中心城市，国家财政收入支柱城市。其港口为世界最大的集装箱港，主要产业包括商贸流通、金融、信息、制造。上海位于中国东部弧形海岸线的正中间，地处长江三角洲最东部，东向东海，隔海与日本九州岛相望，南濒杭州湾，西部与江苏、浙江两省相接，最北部为处于长江入海口中的崇明岛。上海是中国人口最多的城市之一，常住人口2419.70万，其中本地户籍人口为1439.50万人，约占总人口数的59%[参 1]。江南的吴越文化传统与各地移民带入的多样文化融合，逐渐形成了特有的海派文化。 — 来自 维基百科
- 选择原因：我现在在上海工作，对上海比较熟悉。

数据清洗

验证邮政编码

我使用以下代码对 Node 中的邮政编码进行验证：

```
rex_postcode = re.compile(r'[1-9]\d{5}(?!\d)')
    with open("shanghai.osm", 'rb') as f:
        soup = BeautifulSoup(f.read(), "lxml")
        postcodes = soup.find_all('tag', k='addr:postcode')
        for post in postcodes:
            if rex_postcode.fullmatch(post['v']) == None:
                print(post['v'])

        print("Done!")
```

发现 6 条异常值：

```
2000080
201315 上海
201315 上海
2000080
20032
20032
```

数据修正

对于发现的异常邮政编码数据，可以尝试修复，如果实在不能修复则需要删除。

修复办法：

- 对于 2000080 这条数据，我查询了它在源文件中的位置：

```
<tag k="name" v="Shanghai Bund South China Harbour View Hotel"/>
                <tag k="phone" v="+86-21-56660808"/>
                <tag k="rooms" v="0"/>
                <tag k="stars" v="4"/>
                <tag k="name:zh" v="Shanghai Bund South China Harbour View
Hotel"/>
                <tag k="smoking" v="no"/>
                <tag k="tourism" v="hotel"/>
                <tag k="website"
v="http://www.bundsouthchinaharbourviewhotel.com/"/>
                <tag k="addr:city" v="Shanghai"/>
                <tag k="addr:street" v="Huangpu Road"/>
                <tag k="addr:postcode" v="2000080"/>
                <tag k="opening_hours" v="24/7"/>
                <tag k="internet_access" v="wlan"/>
                <tag k="addr:housenumber" v="53"/>
                <tag k="internet_access:fee" v="no"/>
```

其位置为：上海中南海滨酒店，实际邮编应该为：200080

- 对于 `201315 上海` 这条数据，其位置为 `秀沿路1028弄2支弄`，实际邮编应该为：201315
- 对于 20032 这条数据，其位置为 `中山南二路` 正大乐城，实际邮编为：200032

验证经纬度

经纬度数据大部分为小数点后面保留 7 位，有少部分小数点后面只有 1~6 位，可能会影响数据的准确性。我使用以下代码来判断异常经纬度数据。

```
with open("shanghai.osm", 'rb') as f:
        soup = BeautifulSoup(f.read(), "lxml")
        nodes = soup.find_all('node')

        print("lat")
        for node in nodes:
            if len(node['lat']) < 6:
                print(node['lat'])

        print("lon")
        for node in nodes:
            if len(node['lon']) < 7:
                print(node['lon'])

        print("Done!")
```

小数点后少于 3 位的数据如下：

```
lat:
31.22
31.14
31.14
31.14

lon:
121.37
121.34
121.66
121.65
```

数据修正

对于经纬度数据出现的不精确问题，我们无法从已有的数据去推断这些数据的真实值，考虑到异常数据比较少，修复起来比较困难，并且删除的话并不会对整体的数据造成明显影响，所以建议进行删除处理。

数据概览

Shanghai.osm 137.9MB
sample.osm 1.4MB

文件	大小	行数
Node.db	46.5MB	625646
Way.db	5.5MB	90315
Relation.db	96KB	1046

行数统计

由于我使用了 peewee 这个 python 库，所以没有直接使用原生 SQL 语句，在将数据都插入到 MySQL 以后，使用以下代码可以统计 node，way，relation 的行数：

```
nodes = SelectQuery(Node).select()
ways = SelectQuery(Way).select()
relations = SelectQuery(Relation).select()

print("nodes count = ", nodes.count())
print("ways count = ", ways.count())
print("relations count = ", relations.count())
```

独立用户统计

我统计了 Node 表的独立用户数，代码如下，和 SQL 语句 `GROUP_BY` 的作用是一样的：

```
nodes = Node.select(Node.user,
fn.Count(Node.user).alias("user_count")).group_by(Node.user)
```

独立用户数为：1163

排名前 10 的用户以及他们贡献数据的数量

以下数据从 Node 表中统计得出，代码如下：

```
query = Node.select(Node.user,
fn.Count(Node.user).alias("user_count")).group_by(Node.user)
query = query.order_by(SQL('user_count DESC'))

for node in query[0:10]:
        print(node.user_count)
    print(node.user)
```

对应的数据为：

序号	user	count
1	zzcolin	66286
2	Xylem	62368
3	Stenive	49075
4	aighes	43568
5	yangfl	38028
6	HWST	36256
7	alberth2	19116
8	u_kubota	18795
9	z_i_g_o	18263
10	duxxa	13172

排名前 100 的用户数据总数

查询代码如下：

```
query = Node.select(Node.user,
fn.Count(Node.user).alias("user_count")).group_by(Node.user)
    query = query.order_by(SQL('user_count DESC'))

    num = 0
    for node in query[0:100]:
        print(node.user_count)
        print(node.user)
        num += node.user_count

    print('num = ', num)
    print('per = ', num/625646)
```

数量总数为：582789 占比：93.15%

仅有一条数据的用户数

查询代码如下：

```
query = Node.select(Node.user,
fn.Count(Node.user).alias("user_count")).group_by(Node.user).having(fn.Count
(Node.user) == 1)
```

结果为：288

数据量小于 100 的用户数

查询代码如下：

```
query = Node.select(Node.user,
fn.Count(Node.user).alias("user_count")).group_by(Node.user).having(fn.Count(
Node.user) < 100)
```

结果为：934

Node 中 Tag 为 fast_food 的数量

fast_food 数量的查询代码如下：

```
with open("shanghai.osm", 'rb') as f:
        soup = BeautifulSoup(f.read(), "lxml")
        fastfood = soup.find_all('tag', k='amenity', v='fast_food')

        sum = 0
        for food in fastfood:
            tags = food.find_next_siblings()
            for tag in tags:
                if tag['k'] == 'name:en':
                    sum +=1
                    print(tag['v'])

        print("Done!")
        print("sum = ",sum)
```

结果为：75，大多为 KFC 和 McDonald's。

改进建议

原始数据中，有 93% 的数据由数据量总数排名前 100 的用户统计而来，有多达 288 个用户提供的数据仅有一条。同时这些数据中有部分数据的经纬度数据并不精确，对于这些数据由于修复起来比较困难，所以我进行了删除处理。另外，邮政编码中，也有部分数据不规范，有的只有 5 位，有的超过了 6 位，甚至包含其他无效字符串，这部分数据我根据其所在源文件中的位置，结合上下文进行了修复。Node 中 fast_food 的数量比较少，仅有 75 个，但是就实际情况来看，75 个未免也太少了，猜测应该是数据统计不全面或者缺失造成的。

对于数据统计不全面或者缺失这个问题，我认为需要去寻找一些其他的数据源，作为补充，比如自己编程从一些高德地图、百度地图、Google Map 的 API 中获取。

由于 OSM 中的数据来自于不同的人，不同的设备（设备的定位精度可能不同），这些因素本身就有可能造成数据异常。在数据的清洗中，我们也发现了一些数据异常，所以我们能否相信这份数据是准确的呢？因此，我觉得需要引入一些其他的数据源作为补充，同时也可以对已有的数据进行交叉验证，从而提高数据质量。

值得注意的是，由于第三方的数据源格式也不尽相同。在处理中，可能会造成成本比较高，因此可以有针对性的去选择权重比较高，需要优先修复或补充的数据进行处理。

参考资料

- https://github.com/coleifer/peewee
- http://docs.peewee-orm.com/
- https://docs.python.org/2/library/decimal.html
- http://stackoverflow.com/questions/14010551/how-to-convert-between-bytes-and-strings-in-python-3
- http://www.liaoxuefeng.com/wiki/001431608955726 4a6b348958f449949df42a6d3a2e542c000/00
- http://www.ruanyifeng.com/blog/2007/10/ascii_unicode_and_utf-8.html