清洗openstreetmap数据

1. 项目综述

Adelaide是澳大利亚的一座港口城市，南澳大利亚洲的首府。阿德莱德文化古迹景点众多，治安良好，自然环境优美，被《经济学家》评为2016年世界最适宜人类居住的城市之一。因此本项目将整理清洗Adelaide的openstreetmap数据，原始文件adelaide_sample.osm大小为237M。

2. 数据中的问题

将原始osm数据整理成json格式，导入到mongodb后，主要发现了两个问题：

- postcode编码格式不一致
- phone格式不一致或有错误。

osm2json.py文件在将osm数据整理时，针对数据库查询时的问题，调用auditosm.py文件中的清洗方法，完成对数据的清洗过程。下面主要陈述这三方面的问题和解决方法。

2.1 postcode编码不一致

数据库名称是adelaide，集合名称是osm，查询postcode数据，运行下面的命令：

```
db.osm.find({"addr.postcode": {"$exists":1}},{"_id":0,"addr.postcode":1})
```

结果如下：

```
{ "addr" : { "postcode" : "5035" } }
{ "addr" : { "postcode" : "5072" } }
{ "addr" : { "postcode" : "5096" } }
...
{ "addr" : { "postcode" : "SA 5118" } }
...
{ "addr" : { "postcode" : "SA 5052" } }
...
```

可以看出邮政编码格式不统一，正常的postcode应该是四位数字，出现问题的格式只有类似"SA 5052"一种，因此在审查程序auditosm.py中专门处理这种有问题的编码：

```
def audit_postcode(postcode):
'''
根据查询结果，不一致的情况是“SA 5608”。 如果后四位是数字，返回后四位。
'''
return postcode[-4:] if re.match(r'[\d]{4}$', postcode) else None
```

2.2 电话号码存在错误

在查询电话号码时，发现有几个问题：1.格式不一致，如空格数不一致，有的有括号，有点带斜杠；2.号码位数缺失，有的缺少“+61”，有的缺失区号，有的号码完全不对。

查询命令如下：

```
{ "phone" : "+61882234550" }
{ "phone" : "+61 8370 9521" } #格式不一致，中间有空格
{ "phone" : "0882101000" }
{ "phone" : "08 8260 2055" }
...
{ "phone" : "(08) 8370 9844" }
{ "phone" : "+61 8 83391899" }
{ "phone" : "+61-8-82952491" }
...
{ "phone" : "82404068" } #缺少区号
{"phone" : "13 22 11" } #位数错误
```

auditosm.py中的audit_phone(phone)函数用来清洗phone字段。

```
def audit_phone(phone):
num_patt = re.compile('[\d]{1}') #匹配数字
num_lst = num_patt.findall(phone) #找出phone中的所有数字
if len(num_lst) == 11: #属于正常的手机号，只需要清理格式
country_num = ''.join(num_lst[0:2]) #格式：“+61 123456789”
return "+" + country_num + ' '+ ''.join(num_lst[2:])\
        if country_num == '61' else None

elif len(num_lst) == 10: #属于固定电话
district_num = ''.join(num_lst[0:2])
    return district_num + ' ' + ''.join(num_lst[2:]) if district_num in ['02',
'03', '04', '07', '08'] else None #要求满足正确的区号，格式“08 #12345678”

elif len(num_lst) == 9: #缺少“+61”的手机号
return "+61" + ''.join(num_lst[:])
else: #缺少区号的固定电话号码没法补全
return None
```

3. 数据概览

3.1 数据大小

adelaide_sample.osm : 237M

adelaide_sample.json: 353M

mongodb中adelaide 数据库大小：101M

3.2 查询唯一用户数

在mongodb shell中运行以下命令：

```
> db.osm.aggregate([
... {"$match": {"created.user": {"$exists": 1}}},
... {"$group": {"_id": "$created.user",
... "count": {"$sum": 1}}},
... {"$group": {"_id": "Total user number",
... "total user number": {"$sum": 1}}}
... ])

{ "_id" : "Total user number", "total user number" : 798 }
```

共有798个不同的用户对osm有贡献。

3.3 查询餐馆数目

```
> db.osm.aggregate([ {"$match": {"amenity": "restaurant"}},
{"$group": {"_id": "$name", "count": {"$sum": 1}}},
{"$group": {"_id": "Total restaurant number",
                        "total restaurant number": {"$sum": 1}}}])

{ "_id" : "Total restaurant number", "total restaurant number" : 310 }
```

3.4 建筑类型排序

```
db.osm.aggregate([
... {"$match": {"building": {"$exists": 1}}},
... {"$group": {"_id": "$building",
... "count": {"$sum": 1}}},
... {"$sort": {"count": -1}},
... {"$limit": 12}
... ])
{ "_id" : "house", "count" : 33798 }
{ "_id" : "yes", "count" : 18049 }
{ "_id" : "industrial", "count" : 1339 }
{ "_id" : "apartments", "count" : 654 }
{ "_id" : "garage", "count" : 482 }
{ "_id" : "shed", "count" : 461 }
{ "_id" : "commercial", "count" : 443 }
{ "_id" : "school", "count" : 317 }
{ "_id" : "residential", "count" : 302 }
{ "_id" : "retail", "count" : 124 }
{ "_id" : "church", "count" : 85 }
{ "_id" : "roof", "count" : 81 }
```

值得一提的是，adelaide的工业建筑industrial排名第三，说明adelaide的工业比较发达。教堂建筑也排在了第十位，有85座教堂，如果有对宗教感兴趣的同学，adelaide应该是一个不错的旅游景点。

3.5 查询教堂名称

下面查询一下，到底有哪些教堂,只显示15座。

```
>db.osm.aggregate([
... {"$match": {"building": {"$in":["church", "cathedral"]}}},
... {"$group": {"_id": "$name"}},
... {"$limit": 15}
... ])
{ "_id" : "Our Lady of the Visitation Catholic Church" }
{ "_id" : "The Church of the Holy Cross" }
{ "_id" : "St Columba's Anglican Church" }
{ "_id" : "Hallett Cove Lutheran Church" }
{ "_id" : "Hallett Cove Baptist Church" }
{ "_id" : "Semaphore Uniting Church" }
{ "_id" : "Russian Orthodox" }
{ "_id" : "Elizabeth Revival Fellowship" }
{ "_id" : "Blackwood Uniting Church" }
{ "_id" : "Chapel" }
{ "_id" : "St Oswald Church" }
{ "_id" : "Grace Assembly Church" }
{ "_id" : "Anglican Church of the Epiphany" }
{ "_id" : "Friends' Meeting House" }
{ "_id" : "Lutheran Church" }
```

4. 其他的建议和改进

4.1 使用bash语句查询数据问题

bash语句灵活方便，可以用它来帮助我们快速定位有问题的数据，而且不需要改变原有数据文件，从而可以安全地制定清洗计划，

如

```
输入：cat adelaide_sample.osm | grep "postcode"
输出：<tag k="addr:postcode" v="5046" />
<tag k="addr:postcode" v="5046" />
<tag k="addr:postcode" v="5046" />
...
<tag k="addr:postcode" v="SA 5118" />
```

cat表示查看文件，|表示管道，将cat 查看的内容经过后面的语句再处理，grep是搜索匹配条件的内容。
但是这样可能存在漏掉关键词的问题，比如忽略"post_code"或"postal_code"关键词。
可以采用正则表达式匹配，如：
输入： cat adelaide_sample.osm | grep -E "post.*code"
输出结果中果然出现了postal_code关键词：

```
<tag k="postal_code" v="5082" />
```

于是又返回去重新增加一项对postal_code的清洗程序。
对于其他的关键词，如"shop"，我们有可能找不全同类型的关键词，这样很有可能会遗漏这些数据的清洗步骤，这是其中的一个问题。

5. 参考资料

- mongodb documentation
- python正则表达式 re findall
- 百度百科：阿德莱德
- linux中的grep命令