

# OpenStreetMap Data Case Study



7point5 (/u/27d144331d89)  
2018.02.26 20:55\* 字数 931 阅读 0 评论 0 喜欢 0  
(/u/27d144331d89)

编辑文章 (/writer#/notebooks/22666263/notes/24470518)

## OpenStreetMap Data Case Study

### Map Area

Isle of Wight, England

- Relation: <https://www.openstreetmap.org/relation/154350> (<https://link.jianshu.com?t=https%3A%2F%2Fwww.openstreetmap.org%2Frelation%2F154350>)
- Download Link: <https://download.geofabrik.de/europe/great-britain/england/isle-of-wight.html> (<https://link.jianshu.com?t=https%3A%2F%2Fdownload.geofabrik.de%2Feurope%2Fgreat-britain%2Fengland%2Fisle-of-wight.html>)

The Isle of Wight is an island in south England I always want to visit. I am looking forward to digging out some interesting information as well as helping out improving OpenStreetMap.

### 1. Problems Encountered in the Map

- irregular street names (*Leeson Road*<http://wightpaths.co.uk/rowmaptiles/> ([{zoom}/{x}/{y}.png](https://link.jianshu.com?t=Roadhttp%3A%2F%2Fwightpaths.co.uk%2Frowmaptiles%2F))
- inconsistent street name spelling ('King Edwards Close', 'Littletown lane')
- inconsistent attribute spelling
- incomplete postcode

#### 1.1 irregular street names:

When I ran an audit of the stree names, I used a defaultDict(int) to check the street types. If the name has a white space in it, i added the last part to the defaultDict. For those has only one word for their street names, I add the whole name into the defaultDict. I found an unusually long one, i.e. normally it is a short word like street or road. To correct this type of errors, I inserted a re function and split function into the shape\_element function to separate the street name and kept the correct name before export the data.

```
'''re and split functions i used to correct the data;
item is a node of the document'''
pattern = re.compile(r'.*http')

if item.attrib['k'] and item.attrib['k'] == 'addr:street' and pattern.search(item.attrib['v']):
    item.attrib['v'] = item.attrib['v'].split('http')[0]
```

#### 1.2 inconsistent street name spelling



Some street names only has the first letter capitalized ('Littletown lane'). I unified the format by using `split()` and `upper()`.

```
'''key is the value for attribute 'k' in <tag>;
value is the value for attribute 'v' in <tag> '''
if 'addr:street' in key and ' ' not in value:
    value = value[0].upper() + value[1:]
elif 'addr:street' in key and ' ' in value:
    value = ' '.join([i[0].upper() + i[1:] for i in value.split(' ')])
```

### 1.3 inconsistent attribute spelling

During auditing tag attributes, it turned out that some tags' **k** attributes are spelt as '**addr:Street**' whereas some others are spelt like '**addr:street**'. This could potentially cause trouble when analysing data with different types of spellings. To fix it, I used `.upper()` before sepearate the **colons(:)** between certain **k** tags.

### 1.4 incomplete postcode

UK postcodes are alphanumeric. Each post code is divided into two parts separated by a single space: the outward code and the inward code respectively. 4 postcodes were found with only outward half, e.g. 'PO39'.

After chekcing the addresses with incomplete postcodes, it turned out that those addresses are neighbourhood in which they reside. Thus they should be updated with the same postcodes (PO39 0EX).

```
'''check the geo info of the locations with incomplete postcodes'''
sqlite> SELECT id, key, type, value FROM node_tags
WHERE id IN (SELECT id FROM node_tags WHERE key = 'postcode' AND value = 'PO39') ;
```

```
'''update postcodes'''
sqlite> UPDATE node_tags SET value = 'PO39 0EX'
WHERE id IN (select id from node_tags where key = 'postcode' and value = 'PO39')
AND key = 'postcode';
```

## 2. Data Overview

This section contains basic statistics about the dataset and the MongoDB queries used to gather them.

### 2.1 File sizes

```
isle-of-wight-latest ----- 127 MB
data.db ----- 138 MB
nodes.csv ----- 47.6 MB
nodes_tags.csv ----- 19 MB
ways.csv ----- 5.9 MB
ways_tags.csv ----- 20.7 MB
ways_nodes.csv ----- 144.6 MB
```

### 2.2 Number of documents in the osm file

```
def file_check(osm_file):
    data = defaultdict(int)
    tree = ET.parse(osm_file)
    for item in tree.iter():
        data[item.tag] += 1
    return data
```



```
'osm': 1,  
'bounds': 1,  
'node': 588853,  
'tag': 259899,  
'way': 101359,  
'nd': 759141,  
'relation': 1691,  
'member': 20227
```

2.3 Number of nodes

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

588853

2.4 Number of ways

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

101359

2.5 Number of unique users

```
select count(distinct(uid)) from (select uid from nodes union all select uid from ways);
```

507

2.6 Top 10 contributing users

```
select user, count(1) as num  
from (select uid, user from nodes union all select uid, user from ways)  
group by user  
order by num desc  
limit 10;
```

"Mappernerd"	374511
"iwhs"	122522
"dmgroom_ct"	81467
"dmgroom"	18411
"Andy Street"	9933
"jpennycook"	7831
"Rondon237"	6705
"DrMark"	6445
"UniEagle"	6395
"Gostman"	5028

2.7 Number of users only contributed 1 post

```
select count(*)  
from (select user, count(1) as num  
from (select uid, user from nodes union all select uid, user from ways)  
group by user  
having num = 1)  
;
```

66

3. Additional Data Exploration

3.1 Number of tag types



```
select count(*)
from (select *
from (select key, value from node_tags union all select key, value from way_tags)
group by key)
;
```

571

3.2 Top 10 appearing tag types

```
select key, count(1) as num
from (select key, value from node_tags union all select key, value from way_tags)
group by key
order by num desc
limit 10
;
```

```
"building"  53454
"source"    28232
"highway"   21093
"natural"   14638
"barrier"   14490
"name"      12105
"city"      9622
"street"    9144
"access"    5825
"house"     5129
```

3.3 Number of building types

```
select count(*)
from (select *
from (select key, value from node_tags union all select key, value from way_tags)
where key = 'building'
group by value)
;
```

46

3.4 Top 10 building types

```
select value, count(1) as num
from (select key, value from node_tags union all select key, value from way_tags)
where key = 'amenity'
group by value
order by num desc
limit 10
;
```

```
"yes"  22528
"residential"  18364
"house"  8861
"chalet"  2178
"apartments"  323
"garage"  291
"hut"  207
"commercial"  153
"terrace"  124
"barn"  78
```

3.5 Number of data provided by The National Public Transport Access Nodes (NaPTAN)



```
select count(*)
from (select *
from (select * from node_tags union all select * from way_tags)
where type = 'naptan'
group by id)
;
```

1335

### 3.6 General information of naptan locations

```
select key, count(1) as num
from (select *
from (select * from node_tags union all select * from way_tags)
where id in
(select id
from (select * from node_tags union all select * from way_tags)
where type = 'naptan'
group by id)
and type = 'regular')
group by key
order by num desc
;
```

```
"name" 1334
"source" 1328
"highway" 1321
"shelter" 82
"operator" 12
"note" 11
"bench" 9
"local_ref" 9
"alt_name" 8
"railway" 4
"network" 2
"physically_present" 2
"wheelchair" 2
"covered" 1
"fixme" 1
"seats" 1
```

### 3.7 The naptan location without a name attribute

```
select *
from (select * from node_tags union all select * from way_tags)
where id in
(select id
from (select *
from (select * from node_tags union all select * from way_tags)
where id in
(select id
from (select * from node_tags union all select * from way_tags)
where type = 'naptan'
group by id)
)
group by id
)
)
and id not in
(select id
from (select *
from (select * from node_tags union all select * from way_tags)
where id in
(select id
from (select * from node_tags union all select * from way_tags)
where type = 'naptan'
group by id)
)
)
)
where key = 'name'
)
;
```



```

"533813619" "source"      "regular"      "naptan_import"
"533813619" "Street"      "naptan"       "East Street"
"533813619" "Bearing"     "naptan"       "SW"
"533813619" "AtcoCode"   "naptan"       "230000007750"
"533813619" "verified"    "naptan"       "no"
"533813619" "Indicator"  "naptan"       "Adj"
"533813619" "CommonName" "naptan"       "Cineworld"
"533813619" "NaptanCode" "naptan"       "iowgjgt"

```

### 3.8 The naptan stop that needs to be fixed

tag attributes:

```

select key, value
from (select * from node_tags union all select * from way_tags)
where id in
      (select id
       from (select *
            from (select * from node_tags union all select * from way_tags)
              where id in
                    (select id
                     from (select * from node_tags union all select * from way_tags)
                       where type = 'naptan'
                     group by id
                    )
              and key = 'fixme'
       )
      )
;

```

```

"name"      "Bannock Road South"
"fixme"     ",ay not be in use"
"source"    "naptan_import"
"highway"   "bus_stop"
"Street"    "Bannock Road"
"Bearing"   "S"
"AtcoCode"  "230000005890"
"verified"  "yes"
"Indicator" "towards Blackgang"
"CommonName" "Bannock Road South"
"NaptanCode" "iowdtmg"

```

lat and lon info:

```

select lat, lon
from nodes
where id in
      (select id
       from (select *
            from (select * from node_tags union all select * from way_tags)
              where id in
                    (select id
                     from (select * from node_tags union all select * from way_tags)
                       where type = 'naptan'
                     group by id
                    )
              and key = 'fixme'
       )
      )
;

```

lat: 50.597754

lon: -1.2654259

### 3.9 Number of locations required to be fixed with name or street information



```
select count(*)
from (select *
      from (select * from node_tags union all select * from way_tags)
      where id in
            (select id
             from (select * from node_tags union all select * from way_tags)
             where key = 'fixme' or (key = 'note' and value like '%fixme%'))
      and key in ('name', 'street')
      group by id
      )
;
```

139

3.10 Number of nodes with name or street information

nodes with a name:

```
select count(*)
from (select id, key, value from (select * from node_tags)
      where key = 'name')
;
```

3899

nodes with address (street information):

```
select count(*)
from (select id, key, value from (select * from node_tags)
      where key = 'street')
;
```

3613

nodes with either a name or address:

```
select count(*)
from (select id
      from (select id, key, value from (select * from node_tags)
            where key in ('name', 'street'))
      group by id
      )
;
```

6060

4. Facts and suggestions

- There are 507 unique users contributed to the area.
- 66 users (13%) have only submitted 1 posts.
- Top 3 users contributed 98.2% of overall entries.
- Top 3 user contributed 63.6%, 20.8%, 13.8% of overall entries respectively.

After viewing the data and posting some entries myself, it came to me that editing an entries could cost at least 5 minutes (including checking the right input). A fair amount of users might find the procedure to contribute time-consuming.

Suggestion

I would suggest OpenStreetMap to breakdown the fields of data, highlight the priorities (e.g. name and address are prior to whether or not smoking is allowed inside the building), preload location information (e.g. city and province can be preloaded if locations nearby both have same values of city/province).



Expected Enhancements

- Preloading verified information can reduce the time cost of inputting single POI data.
- Preloading verified information can keep certain fields in a consistent format.

Risks

- Preloaded data can be inaccurate when the POI is at the edge of two areas (e.g. city boarder).
- Verified information can be incorrect due to human error and system errors.
- Previously verified information can change due to causes like changes the municipal administration plans.

5. Conclusion

After wrangling and reviewing the dataset, I found out that the data was contributed by a small numbers of users mostly. As a result, the data might not be all accurate as not all inputs can be reviewed by others. In addition, most locations have different levels of missing data. Although enlarging the size of data and enhancing the accuracy are both important, OSM should consider encourage more entires to be submitted before enhancing the accuracy as it is easy to get a large amount of data if they cooperate with other GPS processors or introduce more users to participate.

小礼物走一走，来简书关注我

赞赏支持

📖 日记本 (/nb/22666263) © 著作权归作者所有



7point5 (/u/27d144331d89)

写了 931 字，被 0 人关注，获得了 0 个喜欢

(/u/27d144331d89)

喜欢



更多分享


(http://cwb.assets.jianshu.io/notes/images/2447051

被以下专题收入，发现更多相似内容 投稿管理

+ 收入我的专题

why stoc, (/p/27770f164b96?utm\_campaign=maleskine&utm\_content=...

Why Stock Markets CrashThis page intentionally left blankWhy Stock Markets CrashCritical Events in ComplexFinancial SystemsD i d i e r S ...



Adam\_潜 (/u/45b83aea7859?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

rljs (/p/84be537f1929?utm\_campaign=maleskine&utm\_content=note&...

rljs by sennchi Timeline of History Part One The Cognitive Revolution 1 An Animal of No Significance 2 The



Tree of Knowledge 3 A Day in t...



sennchi (/u/4dde9ad6c8ed?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 今日学术视野(2017.11.16) (/p/d86b136b3f41?utm\_campaign=maleskine...

cs.AI - 人工智能cs.CL - 计算与语言cs.CR - 加密与安全cs.CV - 机器视觉与模式识别cs.CY - 计算与社会  
cs.DC - 分布式、并行与集群计算cs.DS - 数据结构与算法cs.HC - 人机接口cs.IR - 信息检索cs.IT - 信息论...



爱可可\_爱生活 (/u/ZQtGe6?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 今日学术视野(2017.11.1) (/p/85bcb2620f37?utm\_campaign=maleskine&...

cs.AI - 人工智能cs.CL - 计算与语言cs.CV - 机器视觉与模式识别cs.CY - 计算与社会cs.DC - 分布式、并行与  
集群计算cs.DL - 数字图书馆cs.IR - 信息检索cs.IT - 信息论cs.LG - 自动学习cs.LO - 计算逻辑cs....



爱可可\_爱生活 (/u/ZQtGe6?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 今日学术视野(2018.2.14) (/p/de3a0b34bcf5?utm\_campaign=maleskine&...

cs.AI - 人工智能cs.CL - 计算与语言cs.CR - 加密与安全cs.CV - 机器视觉与模式识别cs.CY - 计算与社会  
cs.DB - 数据库cs.DC - 分布式、并行与集群计算cs.DS - 数据结构与算法cs.GT - 计算机科学与博弈论cs.H...



爱可可\_爱生活 (/u/ZQtGe6?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

(/p/7a70ffc052c2?)



utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendation)

### 没有耐心的宝宝很可怕！ (/p/7a70ffc052c2?utm\_campaign=maleskine&u...

孩子缺乏耐心的后果，做事半途而废 有些孩子做什么事情，都是断断续续完成的，或者做到一半，就干脆不做了，这种半途而废的现象，完全与孩子的缺乏耐心是有关系的。 1、容易分散注意力 孩子做事情精神无...



女生日志 (/u/b5f970dd5ef6?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### 源码推荐(04.29)：iOS动画Demo，HR\_W仿QQ页面 (/p/ff95f7049f13?utm...

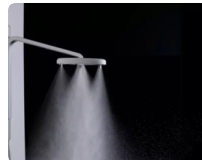
写了个demo，展示了几个iOS中的动画。用到了UIView动画，CoreAnimation，贝塞尔曲线，CALayer等 仿照QQ的样子做了一小部分.新手学习中.主要功能 实现了的segment的点击滑动视图.点击+ 出现一个...



moshanguhuakai88 (/u/701e82d7a202?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

(/p/3972d948b820?)



utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendation)

### PH每日快报 | Nebia Shower，沐浴vs.淋浴 (/p/3972d948b820?utm\_cam...

PH快报是 Product X 项目下的一个媒体专栏，由一群产品爱好者自发编译来自Producthunt榜单上的产品，每日更新，期待您的关注与支持！翻译志愿者招募中，有兴趣的童鞋请站内信我们，标题请注明 PH翻译...




Product\_X (/u/74b70605e944?)

utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

### #RIA联合训练营#拆页七--《幸福课》--七组--高宇 (/p/6a43081b1920?utm...



片段七：《幸福课》161-162 页 用 woop 思维来克服拖延症有一位心理学家 加布里埃尔·厄廷根 发明了一套能够增加执行力的思维方式。过程大概需要十几分钟，却能带来意想不到的收益。W (wish): 愿望或心事...


 宇非门 (/u/f1adfc58db7c?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

(/p/8d7e5e1b3408?



utm\_campaign=maleskine&utm\_content=note&utm\_medium=seo\_notes&utm\_source=recommendation)  
**6247-20170430-Day7 (/p/8d7e5e1b3408?utm\_campaign=maleskine&ut...**

基本要素模仿练习

 DrDancy (/u/7e484b5b3fb8?  
utm\_campaign=maleskine&utm\_content=user&utm\_medium=seo\_notes&utm\_source=recommendation)

