UDACITY

Nanodegree Analista de Dados

Projeto: Limpando dados do OpenStreetMap

Baía de Guanabara, Rio de Janeiro - Brasil

por Fábio Corrêa Cordeiro

Link Open Street Map: https://osm.org/go/OVdg2Wg-

Nesse projeto será feita a limpeza dos dados do Open Street Map referente ao entorno da Baía de Guanabara, no estado do Rio de Janeiro – Brasil. Esse mapa possui trechos das cidades da região metropolitana do Rio de Janeiro sendo elas: Rio de Janeiro, Duque de Caxias, Magé, São Gonçalo e Niterói. Inclui também diversas ilhas da Baía de Guanabara, sendo as principais a Ilha do Governador, Ilha do Fundão e Paquetá.

1 – Problemas Encontrados no mapa

Inicialmente foi baixado uma pequena porção do mapa que representava apenas o bairro Jardim Guanabara, arquivo com 3.204 KB. Nessa versão de teste identificamos os seguintes problemas que também seriam encontrados na versão definitiva do mapa:

- Os "street_type" na lingua portuguesa estão em posição diferentes do que na lingua inglesa.
- Em português há caracteres que não existem em inglês.
- Foram encontrados novos "expected".
- Foram encontradas abreviações ou erros de digitação nos "street_type"

Os "street_type" na lingua portuguesa estão em posição diferentes do que na lingua inglesa.

No código desenvolvido para o "Estudo de Caso: Dados do Street Map" considerava que o "street_type" seria a última palavra do nome da rua. Por exemplo Arlington **Street**, 7th **Avenue** e Central **Park**. No entanto, em português, e em outras linguas latinas, o "stree_type" seria a primeira palavra. Como em **Rua** Marquês de Paraná, **Avenida** Brasil e **Largo** da Carioca.

Portanto a Regular Expression usada para selecionar a última palavra foi modificada para capturar a primeira palavra

```
street_type_re = re.compile(r'^\S+\b', re.IGNORECASE)
```

Em português há caracteres que não existem em inglês.

Quando usado o código desenvolvido no Estudo de Caso, algumas comparações não foram feitas corretamente devido a caracteres e acentuações específicas da língua portuguesa (como ç, ã, á por exemplo). Foi necessário acrescentar **".encode('utf8')"** na comparação de strings da função "audit_street_type".

```
def audit_street_type(street_types, street_name):
    m = street_type_re.search(street_name)
    if m:
        street_type = m.group()
        if street_type.encode('utf8') not in expected:
            street_types[street_type].add(street_name)
```

Foram encontrados novos "expected"

Foram encontrados os seguintes "expected street types" que foi incluido na lista "expected":

expected = ["Rua", "Avenida", "Quadra", "Via", "Estrada", "Caminho", "Estacionamento", "Parque", "Praia", "Alameda", "Beco", "Campo", "Ladeira", "Largo", "Mirante", "Rodovia", "Travessa", "Praça", 'Boulevard', "Calçadão" , "Condomínio"]

Foram encontradas abreviações ou erros de digitação nos "street_type"

Foram identificados os seguintes erros de digitação e abrviações no "street_type" que foram incluídos no dicionário mapping para serem corrigidos:

```
mapping = {
        'Av ': "Avenida ",
        'Av. ': "Avenida ",
        'PLAZA ': "Praça ",
        'Pca ': "Praça ",
        'Praca ': "Praça ",
        'R. ': "Rua ",
        'R ': "Rua ",
        'Rue ': "Rua ",
        'rua ': "Rua ",
        'Rod ': "Rodovia ",
        'Rod. ': "Rodovia ",
        'Trav ': "Travessa ",
        }
```

Outras auditorias

Como auditoria complementar foi verificado se as tags "addr:city" referenciavam as cidades do entorno da Baía de Guanabara. Essa auditoria foi feita para todos os registros, não apenas para os nós e vias. Para atualizar esses registros foi criado um novo dicionário "mapping2" que segue abaixo:

```
mapping2 = {'Camboinha': 'Niterói',
        'Camboinhas': 'Niterói',
        'Caramujo': 'Niterói',
        'Várzea das Moças, São Gonçalo': 'São Gonçalo',
        'rio de Janeiro': 'Rio de Janeiro',
        'SãoGonçalo': 'São Gonçalo',
        'Maracanã': 'Rio de Janeiro',
        'Rio de janeiro': 'Rio de Janeiro',
        'Leblon': 'Rio de Janeiro',
        'niterói': 'Niterói',
        'Colubandê': 'São Gonçalo',
        'Fonseca': 'Niterói',
        'Niteroi': 'Niterói',
        'rio de janeiro': 'Rio de Janeiro',
        'Alcântara': 'São Gonçalo',
        'niteroi': 'Niterói',
        'Cafubá': 'Niterói',
```

```
'RIo de Janeiro': 'Rio de Janeiro',
'Rua Monsenhor Magaldi': 'Rio de Janeiro',
'Rio de Janeiro,': 'Rio de Janeiro',
'Méier': 'Rio de Janeiro',
'São Lourenço': 'Niterói',
}
```

2 – Visão geral dos dados

Nesta seção será apresentado algumas informações sobre os arquivos, a base de dados geradas e algumas consultas.

Arquivos de teste:

Jardim_Guanabara_OSM ----------3.204 KB

Jardim_Guanabara_OSM.json----------3.399 KB

Arquivos do projeto:

Guanabara_Bay_OSM----------165.872 KB

Guanabara_Bay_OSM.json----------175.669 KB

Número de documentos e tamanho da base:

```
collstats = db.command("collstats",'OSM')
print 'count: ', collstats['count']
print 'size: ', collstats['size']
> count:  855.779
> size:  200.965,246 KB
```

Número de vias ("ways"):

```
query = {'type': 'way'}
db.OSM.find(query).count()
> 104.867
```

Número de nós ("nodes"):

```
query = {'type': 'node'}
db.OSM.find(query).count()
> 750.912
```

Número de usuários distintos que criaram vias ("way") ou nós ("node") :

```
len(db.OSM.distinct('created.uid'))
> 1.099
```

Número de nós em algumas ruas famosas:

```
def street_nodes(street_name):
        query = {'address.street': street_name, 'type': 'node'}
```

```
    return db.OSM.find(query, projection).count()


street_nodes("Avenida Presidente Vargas")
> 6
street_nodes("Avenida Brasil")
> 9
street_nodes("Calçadão de Copacabana")
> 1
street_nodes("Avenida Atlântica")
> 20
street_nodes("Avenida Rio Branco")
> 43
```

3 - Ideias adicionais

Ao analisar o nome das "street types" foi verificado que algumas palavras foram abreviadas ou escritas erradas. Como proposta para continuidade do trabalho, essa verificação poderia ser feita para todas as palavras que compõem o "street_name".

Cada palavra do street_name poderia ser comparada com um dicionário em português e com uma lista de nomes próprios (http://centraldedados.pt/nomes_proprios/) que fariam o papel da variável "expected" do código atual. A partir das palavras que não casarem com as duas listas seria montado um novo dicionário "mapping" para a correção das palavras.

Os benefícios dessa proposta é poder identificar programáticamente um grande número de erros nos dados. Além disso o dicionário "mapping" poderia ser usado para auditar outras áreas no Brasil ou em cidades de língua portuguesa.

Entretanto, os problemas esperados para essa solução é a dificuldade de se montar o dicionário "mapping" manualmente, pois ele deverá ser relativamente maior que o de "street types". Támbém teremos dificuldades de identificar palavras e nomes em outras línguas.