

Trabalhando com rotas nos dados do OpenStreetMap

Para esta série de artigos, vou pressupor que você já possui um [ambiente OSM instalado](#) e ainda guarda com você o arquivo **south-america-latest.osm.pbf**. Se você não possui nada disso, acompanhe primeiro [esta série](#) antes de prosseguir.

Vou mostrar como criar uma tabela de vértices (topologia) dos dados de ruas do OSM para então calcular rotas com eficiência.

Vamos precisar baixar o programa *Osm2Po*, que faz todo o trabalho de criação da topologia sem que você precise de muito esforço. De quebra ele ainda oferece um serviço WEB onde você já poderá calcular suas rotas, mas não é minha intenção usar serviços de terceiros! vamos criar nosso próprio sistema de cálculo de rotas com o [Geoserver](#) como servidor de mapas e o [OpenLayers](#) como interface com o usuário.

```
$ wget http://osm2po.de/releases/osm2po-5.1.0.zip
```

```
$ unzip osm2po-5.1.0.zip
```

Após baixar e descompactar o arquivo, edite o arquivo *osm2po.config* e retire os comentários das seguintes linhas:

```
postp.0.class = de.cm.osm2po.plugins.postp.PgRoutingWriter
postp.0.writeMultiLineStrings = true
postp.1.class = de.cm.osm2po.plugins.postp.PgVertexWriter
postp.2.class = de.cm.osm2po.plugins.postp.PgPolyWayWriter
postp.3.class = de.cm.osm2po.plugins.postp.PgPolyRelWriter
```

```
postp.4.class = de.cm.osm2po.postp.GeoExtensionBuilder
postp.5.class = de.cm.osm2po.postp.MlgExtensionBuilder
postp.5.id = 0
postp.5.maxLevel = 3, 1.0
```

```
postp.6.class = de.cm.osm2po.sd.postp.SdGraphBuilder
```

```
# Pg*Writer usually create sql files. Enable the following
# parameter to redirect them to stdout (console)
```

```
postp.pipeOut = false
```

Minha máquina possui 8GB de RAM, então eu serei generoso separando 5GB para a execução do programa. Se você quiser modificar isso, altere o parâmetro -*Xmx5g* no comando de execução abaixo.

```
java -Xmx5g -jar osm2po-core-5.1.0-signed.jar  
tileSize=30x60,10 south-america-latest.osm.pbf
```

Digite “yes” para aceitar a licença. Só precisará fazer isso uma vez. Dependendo da sua quantidade de memória, isso poderá demorar um pouco. Ao final da conversão, se tudo correr bem, o programa continuará rodando e você deverá ter um servidor web ouvindo na porta 8888:

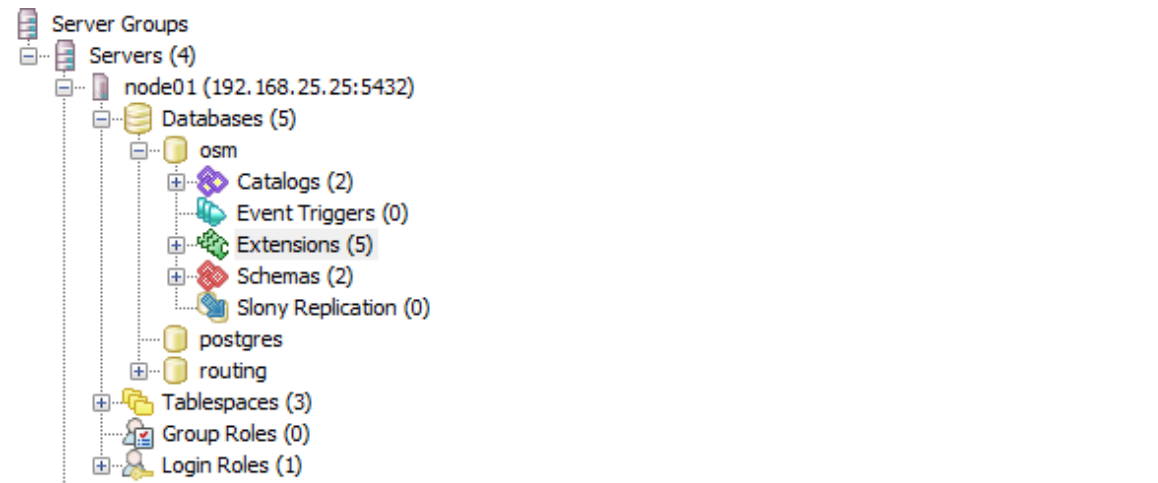
```
http://localhost:8888/Osm2poService
```

Pode parar o serviço usando CTRL+C. O que nos interessa é o conteúdo da pasta *osm* que foi criada. Dentro dela, entre outros arquivos, deverá conter o arquivo **osm_2po_4pgr.sql** (de tamanho um pouco exagerado para um SQL, mas é isso mesmo).

Vamos importar este SQL para o [banco de dados osm](#) do nosso OpenStreetMap:

```
$ su postgres (opcional, dependendo de seu ambiente. Você  
precisará estar em sudo -i para fazer isso)  
$ psql -U postgres -d osm -q -f "osm_2po_4pgr.sql" ( a  
conexão local precisa estar como "trust" )
```

Onde *postgres* é o usuário do servidor e *osm* é o nome do banco de dados. [Mais detalhes em como configurar a conexão local como trust](#). Certifique-se de que todas as extensões necessárias estão instaladas antes de executar a importação:



Extension	Owner	Comment
hstore	postgres	data type for storing sets of (key, value) pairs
pgrouting	postgres	pgRouting Extension
plpgsql	postgres	PL/pgSQL procedural language
postgis	postgres	PostGIS geometry, geography, and raster spatial types and functions
postgis_topology	postgres	PostGIS topology spatial types and functions

Banco de dados OSM

Ao final da importação teremos uma tabela chamada *osm_2po_4pgr* em nosso banco de dados *osm*. Pode apagar a pasta “*osm*” agora, caso tenha problemas de espaço em disco. Esta tabela contém basicamente os vértices de todas as ruas e estradas do banco de dados do OSM. Mas como funciona isso?

Inicialmente você deverá escolher as ruas de partida e destino da sua rota na tabela *osm_2po_4pgr* usando o nome da rua. O importador copia a geometria e o nome das ruas da tabela *planet_osm_line*. Se você precisar realizar uma busca usando outros critérios diretamente na tabela *planet_osm_line*, você poderá usar o atributo *osm_id* das duas para realizar o *join*.

Conforme foi dito, a tabela de topologia *osm_2po_4pgr* armazena os vértices que une as ruas. O que importa para o calculador de rotas é: “*dado um vértice, quais outros vértices eu consigo alcançar?*”. Para responder esta pergunta, a tabela contém dois atributos (campos) chamados *source* e *target*, que representam os vértices que conectam as ruas umas com as outras ou as segmentações da própria rua. É fácil concluir então que, dado o registro de uma rua qualquer, o seu *target* aponta para uma rua a qual ela está conectada (*source* com o mesmo valor) e seu *source* aponta para uma rua que se conecta nela (*target* com o mesmo valor). Basta então seguir a trilha de *sources* e *targets* da origem até o destino,

encontrando quais ruas possuem o *source* igual ao *target* da rua anterior. Como o OSM fragmenta as ruas, isso também vale para os segmentos da mesma rua. Na imagem abaixo podemos ver um trecho da Rua do Catete (no Rio de Janeiro) fragmentada e com seus vértices marcados em vermelho. Cada círculo vermelho é o *source* de um segmento e o *target* do segmento seguinte.



Vou terminar este post por aqui. Já criamos a tabela de topologia e vimos alguns conceitos sobre ela. [No próximo post](#) vou continuar explicando os conceitos de rotas para depois partir para a prática.