PostGIS

16 de maio de 2007

Sumário

Parte I

Sobre essa Apostila

Conteúdo

O conteúdo dessa apostila é fruto da compilação de diversos materiais livres publicados na internet, disponíveis em diversos sites ou originalmente produzido no CDTC em *http://www.cdtc.org.br*.

O formato original deste material bem como sua atualização está disponível dentro da licença *GNU Free Documentation License*, cujo teor integral encontra-se aqui reproduzido na seção de mesmo nome, tendo inclusive uma versão traduzida (não oficial).

A revisão e alteração vem sendo realizada pelo CDTC (*suporte@cdtc.org.br*) desde outubro de 2006. Críticas e sugestões construtivas são bem-vindas a qualquer tempo.

Autores

A autoria deste é de responsabilidade de Christian do Santos Ferreira (christian.linux@gmail.com).

O texto original faz parte do projeto Centro de Difusão de Tecnologia e Conhecimento, que vem sendo realizado pelo ITI (Instituto Nacional de Tecnologia da Informação) em conjunto com outros parceiros institucionais, atuando em conjunto com as universidades federais brasileiras que tem produzido e utilizado Software Livre, apoiando inclusive a comunidade Free Software junto a outras entidades no país.

Informações adicionais podem ser obtidas através do email *ouvidoria@cdtc.org.br*, ou da *home page* da entidade, através da URL **http://www.cdtc.org.br**.

Garantias

O material contido nesta apostila é isento de garantias e o seu uso é de inteira responsabilidade do usuário/leitor. Os autores, bem como o ITI e seus parceiros, não se responsabilizam direta ou indiretamente por qualquer prejuízo oriundo da utilização do material aqui contido.

Licença

Parte II

Informações Básicas

Sobre o CDTC

Objetivo Geral

O Projeto CDTC visa a promoção e o desenvolvimento de ações que incentivem a disseminação de soluções que utilizem padrões abertos e não proprietários de tecnologia, em proveito do desenvolvimento social, cultural, político, tecnológico e econômico da sociedade brasileira.

Objetivo Específico

Auxiliar o Governo Federal na implantação do plano nacional de software não-proprietário e de código fonte aberto, identificando e mobilizando grupos de formadores de opinião dentre os servidores públicos e agentes políticos da União Federal, estimulando e incentivando o mercado nacional a adotar novos modelos de negócio da tecnologia da informação e de novos negócios de comunicação com base em software não-proprietário e de código fonte aberto, oferecendo treinamento específico para técnicos, profissionais de suporte e funcionários públicos usuários, criando grupos de funcionários públicos que irão treinar outros funcionários públicos e atuar como incentivadores e defensores de produtos de software não proprietários e código fonte aberto, oferecendo conteúdo técnico on-line para serviços de suporte, ferramentas para desenvolvimento de produtos de software não proprietários e de seu código fonte livre, articulando redes de terceiros (dentro e fora do governo) fornecedoras de educação, pesquisa, desenvolvimento e teste de produtos de software livre.

Guia do aluno

Neste guia, você terá reunidas uma série de informações importantes para que você comece seu curso. São elas:

- Licenças para cópia de material disponível
- Os 10 mandamentos do aluno de Educação a Distância
- Como participar dos foruns e da wikipédia
- Primeiros passos

É muito importante que você entre em contato com TODAS estas informações, seguindo o roteiro acima.

Licença

Os 10 mandamentos do aluno de educação online

- 1. Acesso à Internet: ter endereço eletrônico, um provedor e um equipamento adequado é pré-requisito para a participação nos cursos a distância.
- 2. Habilidade e disposição para operar programas: ter conhecimentos básicos de Informática é necessário para poder executar as tarefas.
- 3. Vontade para aprender colaborativamente: interagir, ser participativo no ensino a distância conta muitos pontos, pois irá colaborar para o processo ensino-aprendizagem pessoal, dos colegas e dos professores.
- 4. Comportamentos compatíveis com a etiqueta: mostrar-se interessado em conhecer seus colegas de turma respeitando-os e fazendo ser respeitado pelo mesmo.
- 5. Organização pessoal: planejar e organizar tudo é fundamental para facilitar a sua revisão e a sua recuperação de materiais.
- 6. Vontade para realizar as atividades no tempo correto: anotar todas as suas obrigações e realizá-las em tempo real.
- 7. Curiosidade e abertura para inovações: aceitar novas idéias e inovar sempre.
- 8. Flexibilidade e adaptação: requisitos necessário à mudança tecnológica, aprendizagens e descobertas.
- 9. Objetividade em sua comunicação: comunicar-se de forma clara, breve e transparente é ponto - chave na comunicação pela Internet.
- 10. Responsabilidade: ser responsável por seu próprio aprendizado. O ambiente virtual não controla a sua dedicação, mas reflete os resultados do seu esforço e da sua colaboração.

Como participar dos fóruns e Wikipédia

Você tem um problema e precisa de ajuda?

Podemos te ajudar de 2 formas:

A primeira é o uso dos fóruns de notícias e de dúvidas gerais que se distinguem pelo uso:

. O fórum de notícias tem por objetivo disponibilizar um meio de acesso rápido a informações que sejam pertinentes ao curso (avisos, notícias). As mensagens postadas nele são enviadas a

todos participantes. Assim, se o monitor ou algum outro participante tiver uma informação que interesse ao grupo, favor postá-la aqui.
Porém, se o que você deseja é resolver alguma dúvida ou discutir algum tópico específico do curso. É recomendado que você faça uso do Forum de dúvidas gerais que lhe dá recursos mais efetivos para esta prática.

. O fórum de dúvidas gerais tem por objetivo disponibilizar um meio fácil, rápido e interativo para solucionar suas dúvidas e trocar experiências. As mensagens postadas nele são enviadas a todos participantes do curso. Assim, fica muito mais fácil obter respostas, já que todos podem ajudar.
Se você receber uma mensagem com algum tópico que saiba responder, não se preocupe com a formalização ou a gramática. Responda! E não se esqueça de que antes de abrir um novo tópico é recomendável ver se a sua pergunta já foi feita por outro participante.

A segunda forma se dá pelas Wikis:

. Uma wiki é uma página web que pode ser editada colaborativamente, ou seja, qualquer participante pode inserir, editar, apagar textos. As versões antigas vão sendo arquivadas e podem ser recuperadas a qualquer momento que um dos participantes o desejar. Assim, ela oferece um ótimo suporte a processos de aprendizagem colaborativa. A maior wiki na web é o site "Wikipédia", uma experiência grandiosa de construção de uma enciclopédia de forma colaborativa, por pessoas de todas as partes do mundo. Acesse-a em português pelos links:

- Página principal da Wiki - http://pt.wikipedia.org/wiki/

Agradecemos antecipadamente a sua colaboração com a aprendizagem do grupo!

Primeiros Passos

Para uma melhor aprendizagem é recomendável que você siga os seguintes passos:

- Ler o Plano de Ensino e entender a que seu curso se dispõe a ensinar;
- Ler a Ambientação do Moodle para aprender a navegar neste ambiente e se utilizar das ferramentas básicas do mesmo;
- Entrar nas lições seguindo a seqüência descrita no Plano de Ensino;
- Qualquer dúvida, reporte ao Fórum de Dúvidas Gerais.

Perfil do Tutor

Segue-se uma descrição do tutor ideal, baseada no feedback de alunos e de tutores.

O tutor ideal é um modelo de excelência: é consistente, justo e profissional nos respectivos valores e atitudes, incentiva mas é honesto, imparcial, amável, positivo, respeitador, aceita as idéias dos estudantes, é paciente, pessoal, tolerante, apreciativo, compreensivo e pronto a ajudar.

A classificação por um tutor desta natureza proporciona o melhor feedback possível, é crucial, e, para a maior parte dos alunos, constitui o ponto central do processo de aprendizagem.' Este tutor ou instrutor:

- fornece explicações claras acerca do que ele espera, e do estilo de classificação que irá utilizar;
- gosta que lhe façam perguntas adicionais;
- identifica as nossas falhas, mas corrige-as amavelmente', diz um estudante, 'e explica porque motivo a classificação foi ou não foi atribuída';
- tece comentários completos e construtivos, mas de forma agradável (em contraste com um reparo de um estudante: 'os comentários deixam-nos com uma sensação de crítica, de ameaça e de nervossismo')
- dá uma ajuda complementar para encorajar um estudante em dificuldade;
- esclarece pontos que não foram entendidos, ou corretamente aprendidos anteriormente;
- ajuda o estudante a alcançar os seus objetivos;
- é flexível quando necessário;
- mostra um interesse genuíno em motivar os alunos (mesmo os principiantes e, por isso, talvez numa fase menos interessante para o tutor);
- escreve todas as correções de forma legível e com um nível de pormenorização adequado;
- acima de tudo, devolve os trabalhos rapidamente;

Parte III

PostGIS

Capítulo 1

O que é o PostGIS

O PostGIS é uma extensão espacial gratuita e de código fonte livre. Sua construção é feita sobre o sistema de gerenciamento de banco de dados objeto relacional (SGBDOR) PostgreSQL, que permite o uso de objetos GIS (Sistemas de Informação Geográfica) ser armazenado em banco de dados. PostGIS inclui suporte para índices espaciais GiST e R-Tree, além de funções para análise básica e processamento de objetos GIS.

Capítulo 2

Plano de ensino

2.1 Objetivo

Capacitar os profissionais a especificarem e implantarem projetos utilizando o PostGIS (banco de dados espaciais).

2.2 Público Alvo

Técnicos que desejam trabalhar com PostGIS.

2.3 Pré-requisitos

Noções básicas de Linux e Geoprocessamento.

2.4 Descrição

O curso de PostGIS será realizado na modalidade EAD e utilizará a plataforma Moodle como ferramenta de aprendizagem. Ele é composto de dois módulos de aprendizado e uma avaliação final. O material didático estará disponível on-line de acordo com as datas pré-estabelecidas no calendário. A versão utilizada para o PostGIS será a 1.1.3.

Todo o material está no formato de lições, e estará disponível ao longo do curso. As lições poderão ser acessadas quantas vezes forem necessárias. Aconselhamos a leitura de "Ambientação do Moodle", para que você conheça o produto de Ensino a Distância, evitando dificuldades advindas do "desconhecimento"sobre o mesmo.

Ao final de cada semana do curso será disponibilizada a prova referente ao módulo estudado anteriormente que também conterá perguntas sobre os textos indicados. Utilize o material de cada semana e os exemplos disponibilizados para se preparar para prova.

Os instrutores estarão a sua disposição ao longo de todo curso. Qualquer dúvida deve ser disponibilizada no fórum ou enviada por e-mail. Diariamente os monitores darão respostas e esclarecimentos.

2.5 Metodologia

O curso está dividido da seguinte maneira:

2.6 Cronograma

- Introdução, instalação e configuração
- Importando e exportando dados, consultas, criação de índices e acesso a partir de outros softwares

As lições contém o contéudo principal. Elas poderão ser acessadas quantas vezes forem necessárias, desde que esteja dentro da semana programada. Ao final de uma lição, você receberá uma nota de acordo com o seu desempenho. Responda com atenção às perguntas de cada lição, pois elas serão consideradas na sua nota final. Caso sua nota numa determinada lição for menor do que 6.0, sugerimos que você faça novamente esta lição.
Ao final do curso será disponibilizada a avaliação referente ao curso. Tanto as notas das lições quanto a da avaliação serão consideradas para a nota final. Todos os módulos ficarão visíveis para que possam ser consultados durante a avaliação final.
Aconselhamos a leitura da "Ambientação do Moodle"para que você conheça a plataforma de Ensino a Distância, evitando dificuldades advindas do "desconhecimento"sobre a mesma.
Os instrutores estarão a sua disposição ao longo de todo curso. Qualquer dúvida deverá ser enviada no fórum. Diariamente os monitores darão respostas e esclarecimentos.

2.7 Programa

O curso de PostGIS oferecerá o seguinte conteúdo:

- Introdução ao PostGIS
- Instalação
- Configurando o PostgreSQL e o PostGIS
- Importando e exportando dados
- Fazendo consultas espaciais
- Construindo índices
- Acessando os dados do PostGIS em outros softwares

2.8 Avaliação

Toda a avaliação será feita on-line.
Aspectos a serem considerados na avaliação:

- Iniciativa e autonomia no processo de aprendizagem e de produção de conhecimento;

- Capacidade de pesquisa e abordagem criativa na solução dos problemas apresentados.

Instrumentos de avaliação:

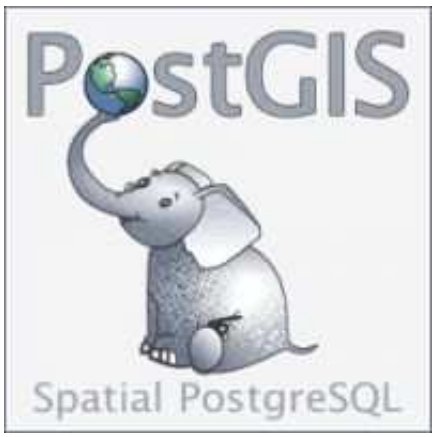
- Participação ativa nas atividades programadas.
- Avaliação ao final do curso.
- O participante fará várias avaliações referente ao conteúdo do curso. Para a aprovação e obtenção do certificado o participante deverá obter nota final maior ou igual a 6.0 de acordo com a fórmula abaixo:
- Nota Final = ((ML x 7) + (AF x 3)) / 10 = Média aritmética das lições
- AF = Avaliações

2.9 Bibliografia

- PostgreSQL: http://www.postgresql.org
- PostGIS: http://postgis.refractions.net
  - p/ Fedora 4 e 5: http://ftp.gwdg.de/pub/misc/freegis/intevation/freegis/fedora/
  - p/ Debian Sarge: http://pkg-grass.alioth.debian.org/cgi-bin/wiki.pl
  - p/ Debian Etch (Testing) e Sid (Unstable): PostGIS já é incluso nos softwares empacotados para esse distribuição.
  - p/ MacOS X: http://wwwamb.bologna.enea.it/forgrass/download.htm
  - p/ Windows: http://postgis.refractions.net/download/windows/
- GEOS: http://geos.refractions.net
- Proj4: http://proj.maptools.org
- OGR: http://ogr.maptools.org/
- QGIS: http://qgis.org/
- Mapserver: http://mapserver.gis.umn.edu/
- OGS e padrões OpenGIS: http://www.opengeospatial.org
- Especificações Web Map OpenGIS: http://www.opengis.org/techno/specs/01-047r2.pdf

Capítulo 3

Introdução



Uma pequena introdução sobre o PostGIS e os padrões OpenGIS.

3.1 Lição 1 - Introdução ao PostGIS

3.1.1 PostGIS

PostGIS é um módulo de estensão do Banco de Dados (DB) gratuito PostgreSQL. O PostGIS adiciona "capacidades espaciais"ao PostgreSQL e permitindo que esse se torne um repositório de dados para os Sistemas de Informações Geográficas (SIG). Isso ocorre nos mesmos moldes de soluções pagas como o SDE - Spatial Database Engine da ESRI® a extensão Spatial da Oracle® e o Spatial Extender do DB2 da IBM®.

No PostGIS está incluso suporte para todas as funcionalidades e objetos definidos na especificação "Simple Features for SQL"(SFSQL) do padrão OpenGIS® (Open Geospatial Consortium - OGC). Nele são definidos funções que permitem consultas e manipulações de dados espaciais através de comandos SQL no PostgreSQL.

Devido à sua aderência pelo formato SFSQL, o PostGIS torna-se uma ferramenta confiável por usar funções padronizadas e abertas ao manipular os dados espaciais. Com isso muitas

aplicações (Mapserver, JUMP, QGIS, etc) podem lidar com o PostGIS e acessar/manipular seus dados. Entretanto, no PostGIS também são inclusas um número enorme de funções SQL que permitem interagir com dados espaciais sem a necessidade de um SIG.

Além disso, utilizar o PostGIS em um servidor poderoso (leia-se, máquina com grande porte de processamento) livra o usuário e seu software cliente da tarefa de lidar com o processamento das consultas ao DB. Isso evita que o usuário tenha que armazenar e processar um grande número de informações, e ainda permite que a base de dados seja acessada e consultada por um grande número usuários (na intranet ou internet).

3.1.2 Padrões OpenGIS (do Open Geospatial Consortium) e sua relação com o PostGIS

Em 1994 foi criado o consórcio internacional Open Geospatial (OGC - Open Geospatial Consortium), denomidado anteriormente de OpenGIS. Ele possui a missão de desenvolver especificações para interfaces espaciais que serão disponibilizadas livremente para uso geral.

Essa iniciativa nasceu para garantir a interoperabilidade de softwares através da padronização das funções que tratam dados espaciais e permitir que todos os softwares de SIG troquem dados entre si (abrir, ler e salvar). Apesar disso ser o ideal, o que pode ser visto é que alguns fabricantes procuram criar seus próprios formatos de dados, atormentando assim a troca de informações entre os diferentes SIG e aprisionando o usuário dentro de formatos proprietários.

Além de garantir interoperabilidade, outro fator interessante dos padrões OpenGIS é que ao trabalhar-se dentro desses formatos (e sem copyright ou patentes), o usuário torna-se independente do software, garantindo assim uma vida útil e longa para os seus projetos com dados espaciais.

Por uma coincidência histórica, os padrões abertos criados pelo OGC começaram a ganhar maturidade no mesmo período em que projetos de softwares livres começaram a surgir para o setor de Geoprocessamento. Assim os softwares já "nascem"com o OpenGIS "embutido"(como o PostGIS).

Atualmente os sistemas proprietários também implementam alguns padrões OGC; porém são os softwares livres que aderem mais rapidamente ao OpenGIS, tornando a escolha pelos softwares livres a escolha ideal para criar uma arquitetura seguindo os padrões abertos OGC.

Capítulo 4

Instalação

Instalação do PostGIS usando binários e através do código fonte.

4.1 Lição 2 - Instalação

4.1.1 Instalação a partir dos pacotes binários

Para a instalação do PostgreSQL e PostGIS o usuário geralmente tem duas opções: compilar a partir do código-fonte ou usar binários. Nesta seção abordaremos a instalação utilizando pacotes pré-compilados.

Como o GNU/Linux é um sistema operacional também focado para servidores, usar os binários do PostgreSQL é simples, já que a maioria das distribuições GNU/Linux possuem esse software pré-compilado e de fácil instalação.

Para o PostGIS a distribuição Debian (Etch/Testing e Sid/Unstable) já dispõe desse pacote que pode ser facilmente instalado com os comandos (como root):

> apt-get update

> apt-get install postgis

Caso o PostgreSQL ainda não esteja presente/instalado ele será automaticamente selecionado e instalado antes de instalar o PostGIS. Nesse procedimento todas as dependências do PostGIS e PostgreSQL também serão automaticamente instaladas.

No Debian Sarge (Stable) o pacote do PostGIS também está disponível, mas nesse caso é necessário utilizar um outro repositório (DebianGIS), já que esse software ainda não fazia parte

da "árvore"oficial do Debian na época do lançamento do Sarge. Para adicioná-lo basta inserir no arquivo source.list (no /etc/apt) do apt-get a seguinte linha:

deb http://pkg-grass.alioth.debian.org/debian-gis stable main non-free contrib

Feito isso basta repetir o procedimento de instalação descrito anteriormente para as outras versões do Debian.

Para outra distribuição GNU/Linux, o Fedora Core 4 e 5, também existem pacotes do PostGIS. Entretanto, esses não são atualizados com a mesma frequência do Debian. Nesse caso o procedimento de instalação consiste em fazer download manual do PostGIS e instalá-lo com o seguinte comando (como root):

> rpm -ivh postgis_pacote.rpm

(onde "postgis_pacote"é o nome do arquivo do download, e nesse caso se assume que o PostgreSQL e todas as dependências do PostGIS (Proj4 e GEOS) já estejam previamente instaladas).

4.1.2 Compilando o PostGIS a partir do código fonte

Para a grande maioria das distribuições GNU/Linux (OpenSuSE, Ubuntu, Slackware, Mandriva, etc) não há binários do PostGIS, então nesse caso a única maneira é compilá-lo a partir do código-fonte. Para o PostgreSQL não será abordado esse procedimento já que ele está disponível (e empacotado) para todas as grandes distribuições.

Compilar (e instalar) o PostGIS é geralmente uma tarefa relativamente simples e rápida. Porém é necessário cumprir alguns requisitos:

- PostgreSQL 7.2 ou superior instalado (inclusive o pacote "dev"ou arquivos de desenvolvimento);
- Compilador GNU C (gcc). Outros compiladores ANSI C podem ser usados mas existem maiores chances de haver problemas;

- GNU Make (gmake ou make). Todas as distribuições o incluem, mas nem todas o instalam por padrão. Outras versões do make podem não conseguir processar o arquivo Makefile;
- (recomendado) Proj4 - biblioteca de projeções cartográficas. Essa biblioteca é necessária caso o usuário queira ter suporte a reprojeção de coordenadas. Caso a sua distribuição não possua o Proj4 ele esta disponível para download nesse endereço: http://proj.maptools.org;
- (recomendado) GEOS - biblioteca de geometrias. É necessária para executar consultas no PostGIS que testam e fazem operações com geometrias. Caso a sua distribuição não possua o GEOS, ele está disponível para download nesse endereço: http://geos.refractions.net.

Após tudo estar checado e em ordem, chegou a hora de compilar o PostGIS. Os passos são os seguintes:

- Fazer download da última versão do PostGIS (em nosso caso a 1.1.3 - versão utilizada durante a confecção desse curso) no endereço: http://postgis.refractions.net;
- Após download o próximo passo é descomprimir o nosso arquivo, no exemplo chama-se "postgis-1.1.3.tar.gz", com o comando abaixo (sim, realmente têm um sinal de - (menos) no final da linha):

  > gzip -d -c postgis-1.1.3.tar.gz | tar xvf -

- Entre no diretório do postgis-1.1.3 e rode:

  > ./configure

NOTA: caso o Proj4 ou GEOS não sejam encontrados isso será informado ao final desse comando. Daí será necessário especificar no configure o caminho (PATH) para as bibliotecas que não forem encontrados. O comando ficaria:

> ./configure –with-proj=PATH

NOTA: Erros no configure (ou nos comandos que virão a seguir) em geral são devidos a pacotes adicionais que estão faltando. Logo, é necessário instalá-los para prosseguir até o final da instalação.

- Agora chegou a hora de compilar e instalar o PostGIS. Os comandos são:

  > make

  > make install

Capítulo 5

Configuração

Configuração do PostgreSQL e do PostGIS.

5.1 Lição 3 - Configurando o PostgreSQL e o PostGIS

5.1.1 Configurando e testando o PostgreSQL

Nessa seção iremos inicializar o PostgreSQL e o PostGIS, e preparar nosso primeiro banco de dados para receber os dados espaciais.

Os passos a seguir devem ser executados como usuário "postgres". Para fazer isso use o comando "su postgres"(é necessário acesso a senha desse usuário).

Para inicializar o PostgreSQL o comando é:

> /etc/init.d/postgresql-8.1 start

NOTA: o local e nome do PostgreSQL podem variar dependendo do sistema GNU/Linux utilizado.

Depois deverá aparecer a seguinte mensagem:

* Starting PostgreSQL 8.1 database server [ok]

Ok, já estamos rodando o PostgreSQL!

Os seguintes comandos deverão ser executados como o usuário postgres, que é o usuário primário do PostgreSQL. Para se tornar o usuário postgres, execute:

 # su postgres

Pronto. Agora, para testar e ver quais os bancos de dados que estão disponíveis use o comando:

> psql -l

O resultado (no exemplo) foi:

Lista dos bancos de dados

Nome | Dono | Codificação
———————+———————+———————-
postgres | postgres | UTF8
project1 | postgres | UTF8
template0 | postgres | UTF8
template1 | postgres | UTF8
(4 registros)

Agora vamos criar o banco de dados exemplo com o comando:

> create database meu_DB

O resultado será: CREATE DATABASE

5.1.2 Configurando e testando o PostGIS

O PostGIS necessita que a linguagem procedural PL/pgSQL seja ativada no banco de dados que o iremos utilizar. Para isso o comando é:

> create language plpgsql meu_DB

Agora carregue o arquivo lwpostgis.sql, ele serve para carregar no PostgreSQL definições de objetos e funções do PostGIS. Na nossa instalação, por código-fonte, o arquivo está no diretório

/usr/share/postgresql/8.1/contrib:

> psql -d meu_DB -f /usr/share/postgresql/8.1/contrib/8.1/lwpostgis.sql

Por último é carregado o arquivo (spatial_ref_sys.sql) de definições do sistema de coordenadas EPSG

> psql -d meu_DB -f /usr/share/postgresql/8.1/contrib/8.1/spatial_ref_sys.sql

IMPORTANTE: Caso a instalação tenha sido feita através do apt-get, esses arquivos podem estar em locais diferentes no sistema de arquivos. Por exempo, após instalar o postgis no Debian testing, esses arquivos se encontam em "/usr/share/postgresql-8.1-postgis/". Para descobrir onde estão esses arquivos, você pode executar, como root:

- updatedb
- locate lwpostgis.sql
- locate spatial_ref_sys.sql

NOTA: uma mensagem de erro como:

psql:/usr/share/postgresql/8.1/contrib/lwpostgis.sql:3509: ERROR: current transaction is aborted, commands ignored until end of transaction block

ou

psql:/usr/share/postgresql/8.1/contrib/spatial_ref_sys.sql:10781:
ERROR:
current transaction is aborted, commands ignored until end of transaction block
ROLLBACK
VACUUM

Apenas significam que o objetos e funções do PostGIS já foram carregados em nosso banco de dados.

Testando o PostGIS

Para verificar se o PostGIS está funcionando corretamente no nosso banco de dados récem criado (meu_DB) vamos rodar alguns comandos simples de SQL.

O primeiro comando testa se as tabelas relacionadas ao PostGIS foram devidamente carregadas em nosso banco de dados.

meu_DB=# \dt

Lista de relações

Esquema | Nome | Tipo | Dono
————+———————————-+———+———-
public | geometry_columns | tabela | postgres
public | spatial_ref_sys | tabela | postgres
(2 registros)

No segundo comando iremos ver o número de registros que existem na tabela "spatial_ref_sys".

meu_DB=# SELECT count(*) FROM spatial_ref_sys;

count
——-
2671
(1 registro)

Esse resultado mostra que a esse tabela possui 2671 entradas. Algo que é perfeitamente normal. NOTA: esse valor pode variar em diferentes versões do PostGIS.

O terceiro comando seria verificar se as funções do PostGIS (coluna 'Nome') foram devidamente carregadas em nosso banco (meu_DB). NOTA: dependendo da versão do PostGIS o resultado pode ser diferente.

meu_DB-# \df **postgis***

Lista de funções

```
 Esquema | Nome | Tipo de dado do resultado | Tipos de dado do argumento
————————+————————————+———————————————————————+———————————————————————
public | postgis_full_version | text |
public | postgis_geos_version | text |
public | postgis_gist_joinsel | double precision | internal, oid, internal, smallint
public | postgis_gist_sel | double precision | internal, oid, internal, integer
public | postgis_jts_version | text |
public | postgis_lib_build_date | text |
public | postgis_lib_version | text |
public | postgis_proj_version | text |
public | postgis_scripts_build_date | text |
public | postgis_scripts_installed | text |
public | postgis_scripts_released | text |
public | postgis_uses_stats | boolean |
public | postgis_version | text |
(13 registros)
```

No último teste vamos verificar a versão do PostGIS que está instalada através do seguinte comando:

```
meu_DB=# SELECT postgis_version();

postgis_version
———————————————————
1.1 USE_GEOS=1 USE_PROJ=1 USE_STATS=1
(1 registro)
```

Aqui podemos ver que a versão instalada é a 1.1(.3). Também temos a confirmação que as bibliotecas GEOS e PROJ estão sendo utilizadas pelo PostGIS (já que elas tem o status igual á 1).

Capítulo 6

Importando e exportando dados no PostGIS

Lição sobre como inserir e extrair os dados do banco.

6.1 Lição 4 - Importando e exportando dados

6.1.1 Importando dados no PostGIS

Uma vez criado o banco de dados com funções espaciais (meu_DB) estamos prontos para inserir os dados de GIS nele. Para isso existem vários caminhos para entrar com os dados espaciais no banco de dados PostGIS/PostgreSQL. Aqui iremos abordar dois: usando comandos de SQL ou usando os programas de importação e exportação de arquivo Shapefile para formato SQL (shp2pgsql) e o carregador de dados (psql).

6.1.2 Usando SQL

Se você pode converter dados para modo texto, então usando SQL formatado poderia ser o modo mais fácil de entrar com seus dados em PostGIS. Como no Oracle e outros bancos de dados de SQL, os dados podem ser carregados em grandes quantidades concatenando um grande arquivo de texto cheio de declarações SQL ("INSERT") dentro do terminal SQL.

A importação de um arquivo texto (teste.sql por exemplo) poderia se parecer como o exemplo a seguir:

```
insert into points values ( 'POINT(0 0)', 'Origin' );
insert into points values ( 'POINT(5 0)', 'X Axis' );
insert into points values ( 'POINT(0 5)', 'Y Axis' );
```

Para realizar essa operação crie um arquivo .TXT com as três linhas de comando SQL acima e nomeie o arquivo de "teste.sql".

Antes de importar esse arquivo é necessário, no entanto, criar a tabela onde os dados serão inseridos. Para realizar isso dê o seguinte comando:

meu_DB=# CREATE table points (pt geometry, name varchar);
CREATE TABLE

Com essa tabela podemos a seguir importar o arquivo de dados para dentro do PostgreSQL usando o comando "psql":

> psql -d meu_DB -f teste.sql

INSERT 0 1
INSERT 0 1
INSERT 0 1

Agora para testar os dados iremos realizar uma consulta que mede a distância entre todos os pontos (0,0), (5,0) e (0,5) com o ponto (5,5). O comando para realizamos essa tarefa é o seguinte:

meu_DB=# SELECT name, AsText(pt), Distance(pt, 'POINT(5 5)') from points;

E o resultado deve ser:

name | astext | distance
———-+————+————————
Origin | POINT(0 0) | 7.07106781186548
X Axis | POINT(5 0) | 5
Y Axis | POINT(0 5) | 5

6.1.3 Convertendo dados com o shp2pgsql e importando com o psql

Para a importação de dados no PostGIS também podemos utilizar o comando shp2pgsql. Ele converte arquivos ESRI Shapefile para o formato SQL, e que depois podem ser inseridos no PostGIS usando o comando psql.

O comando psql tem vários parâmetros para serem utilizados na linha de comando:

-d Exclui a tabela do banco de dados antes de criarmos uma nova tabela com os dados do arquivo Shapefile.

-a Acrescenta dados de um arquivo Shapefile na tabela de banco de dados. Note que, para usar esta opção para carregar arquivos múltiplos, os arquivos têm que ter os mesmos atributos e os mesmos tipos de dados.

-c Cria uma nova tabela e insere dados do arquivo Shapefile. Este parâmetro é o modo padrão.

-p Produz somente o código do SQL da criação da tabela, sem adicionar nenhum dado ao banco de dados. Isto pode ser usado se você necessitar separar completamente as etapas da criação da tabela do carregamento dos dados.

-D Comando usado para importar séries de dados muito grandes. Usa o formato do "dump"do PostgreSQL para a saída de dados. Isto pode ser combinado com -a, -o e -d. É muito mais rápido carregar do que o default de inserção (comando "INSERT") do formato SQL.

-s <SRID> Cria e insere tabelas de geometria com o SRID especificado.

-k Conserva formato dos identificadores (coluna, esquema e atributos). Note que os atributos em Shapefile são todos MAIÚSCULOS.

-i Força todos os inteiros a usarem o formato de 32-bits (não cria inteiros de 64-bits, mesmo se o cabeçalho DBF aparenta permitir isso).

-I Cria um índice GIST na coluna de geometria.

-w Formato de saída WKT, para o uso com versões de PostGIS anteriores (0.x). Note que isto introduzirá deriva nas coordenadas e que deixará cair os valores M dos Shapefiles.

NOTA: -a, -c, -d e -p é mutuamente exclusivo.

Agora iremos realizar uma sessão de exemplo para usar o programa de tradução de Shapefile para formato SQL (shp2pgsql) para criar um arquivo em formato SQL, e em seguida importá-lo no banco de dados usando o comando psql:

> shp2pgsql bc_roads.shp bc_roads > bc_roads.sql

Nota: se o shp2pgsql não estiver disponível com um comando no seu terminal, é necessário para rodá-lo dar o caminho ("path") do executável.

> psql -d meu_DB -f roads.sql

A conversão e importação de dados podem ser feitos em um único comando se concatenarmos (como no UNIX) os dois comandos:

> shp2pgsql bc_roads.shp bc_roads | psql -d meu_DB

6.1.4 Exportando dados do PostGIS usando SQL

Os dados podem ser extraídos do banco usando comandos SQL ou o importardor/exportador de arquivos Shapefile.

O meio mais direto de extrair dados do banco de dados é usar um comando SQL (SELECT) para selecionar dados através de uma consulta, e exportar as colunas resultantes em um arquivo de texto:

meu_DB=# SELECT gid, AsText(the_geom) AS the_geom, name FROM bc_roads;

```
gid | the_geom | name
——+————————————————————————————————+————————-
1 | MULTILINESTRING((1205902.3 460882.2,1205906.3 460789.7)) | No 1 Rd
2 | MULTILINESTRING((1205906.3 460789.7,1205907.3 460776.4)) | No 1 Rd
3 | MULTILINESTRING((1205907.3 460776.4,1205913.6 460638.8)) | No 1 Rd
4 | MULTILINESTRING((1205913.6 460638.8,1205919.6 460513.4)) | No 1 Rd
5 | MULTILINESTRING((1205919.6 460513.4,1205920.8 460482)) | No 1 Rd
6 | MULTILINESTRING((1211706.8 462048.2,1211710.3 461877)) | Seaham Cres
7 | MULTILINESTRING((1212105 462099.5,1212160.1 462102.5)) | Seahaven Pl
(7 registros)
```

Porém, haverá momentos em que restrições serão necessárias para reduzir o número de campos retornados. No caso de restrições baseadas em atributos, apenas use a mesma sintaxe normal de SQL com uma tabela não espacial. No caso de restrições espaciais, os operadores seguintes são disponíveis/úteis:

&& Este operador diz se o retângulo envolvente (bounding box) de uma geometria intersecta o retângulo envolvente de outra.

∼**=** Estes operadores testam se duas geometrias são geometricamente idênticas. Por exemplo, se "POLYGON((0 0,1 1,1 0,0 0))"é igual à "POLYGON((0 0,1 1,1 0,0 0))"(nesse caso é!).

= Este operador é um pouco mais "ingênuo", e só testa se o retângulo envolvente (bounding box) das geometrias é o mesmo.

Logo, você pode usar estes operadores em consultas. Note que ao especificar geometrias e retângulos na linha de comando SQL, você tem que transformar as representações de caracteres (string) explicitamente em geometrias usando a função "GeometryFromText () ". Por exemplo:

SELECT name FROM bc_roads
WHERE the_geom = GeomFromText
('MULTILINESTRING((1212105 462099.5,1212160.1 462102.5))',-1);

A consulta anterior devolveria o único registro da tabela "bc_roads"no qual a geometria era igual aquele valor.

Ao usar o operador "&& ", você pode especificar tanto um retângulo (BOX3D) como característica de comparação como uma geometria. Quando você especifica uma geometria é o seu retângulo envolvente (bounding box) que será usado para a comparação.

SELECT name FROM bc_roads
WHERE the_geom && GeomFromText('POLYGON((1212105 462099.5,1212160.1 462102.5, 1212105 462099.5))',-1);

Então, a consulta anterior usará o retângulo envolvente do polígono para propósitos de comparação.

A consulta espacial mais comum provavelmente será uma consulta "frame-based", usada por um software cliente, como um visualizador de dados ou servidor de mapas na web, é obter o "map-frame"(representação gráfica - figura) para exibição. Usando um objeto "BOX3D"para o "frame", essa consulta iria ser como demonstrado abaixo:

SELECT AsText(the_geom) AS the_geom
FROM bc_roads
WHERE the_geom && SetSRID('BOX3D(1212105 462099.5,1212160.1 462102.5)'::box3d,-1);

Note que o uso do SRID, especifica a projeção do BOX3D. O valor -1 é usado para indicar que nenhum SRID é especificado.

6.1.5 Convertendo dados com o pgsql2shp e exportando com o psql

O programa de conversão pgsql2shp conecta diretamente ao banco de dados e converte uma tabela em um arquivo Shapefile. A sintaxe básica é:

pgsql2shp [<opções>] <banco de dados> [<esquema>.] <tabela>

As opções de linha de comando são:

-f <nome do arquivo> Escreve a saída para o nome de arquivo especificado.

-h <servidor> O servidor de banco de dados para conectar.

-p <porta> A porta para se conectar ao servidor de banco de dados.

-P <senha> A senha para usar a conexão do banco de dados.

-u <usuário> O nome do usuário para usar na conexão com o banco de dados.

-g <coluna de geometria> No caso de tabelas com colunas de múltiplas geometrias, esse opcão espesifica qual coluna de geometria deve ser usada quando escrever o arquivo Shapefile.

-b Usar um cursor binário. Isto fará a operação mais rápida, mas não irá funcionar se qualquer atributo "não-geométrico"na tabela estiver com texto incompleto.

-r Modo bruto. Não descarta o campo "gid"ou exclui os nomes das colunas.

-d Para a compatibilidade com versões anteriores: escreva um arquivo Shapefile 3-dimensional quando for extraí-lo do banco de dados de PostGIS (em versões pre-1.0.0 - neles o padrão é escrever um arquivo Shapefile 2-dimensional neste caso). Inicie de postgis-1.0.0 (ou versão posterior) e as dimensões são codificadas automaticamente.

Capítulo 7

Fazendo consultas espaciais

Como fazer consultas espaciais simples e complexas no PostGIS.

7.1 Lição 5 - Fazendo consultas espaciais

7.1.1 Consultas

O argumento para a funcionalidade do banco de dados espacial está em executar consultas dentro do banco de dados que deveria requerer funcionalidade contidas em um software GIS. Usando PostGIS efetivamente requer conhecimento de quais funções espaciais estão disponíveis, e assegura-se que os índices espaciais apropriados estão criados permitindo bom desempenho nas consultas.

7.1.2 Exemplos de SQL espacial

Os exemplos nesta seção utilizarão duas tabelas, uma tabela de estradas lineares, e uma tabela de limites poligonal da municipalidade. As definições de tabela para a tabela dos bc_roads são:

Coluna | Tipo | Descrição
————-+————————+————————-
gid | integer | Unique ID
name | character varying | Road Name
the_geom | geometry | Location Geometry (Linestring)

A definição de tabela para a tabela do bc_municipality é:

Coluna | Tipo | Descrição
————+————————+————————-
gid | integer | Unique ID
code | integer | Unique ID
name | character varying | City / Town Name
the_geom | geometry | Location Geometry (Polygon)

7.1.3 Exemplos 1

Qual o comprimento total de todas as estradas, expressado nos quilômetros?

Você pode responder esta questão com um comando muito simples de SQL:

meu_DB=# SELECT sum(length(the_geom))/1000 AS km_roads FROM bc_roads;

km_roads
———————
57245.3213072718
(1 registro)

Qual é a área da cidade de Prince George, em hectares?

Esta consulta combina uma condição do atributo (no nome da municipalidade) com um cálculo espacial (de área):

meu_DB=# SELECT area(the_geom)/10000 AS hectares FROM bc_municipality WHERE name = 'PRINCE GEORGE';

hectares
———————
32657.9103824925
(1 registro)

Qual é a maior municipalidade na província, por área?

Esta consulta traz uma medição espacial na condição da busca. Há diversas maneiras de avaliar este problema, mas a mais eficiente está abaixo:

meu_DB=# SELECT name, area(the_geom)/10000 AS hectares FROM bc_municipality ORDER BY hectares DESC LIMIT 1;

name | hectares
———————————+———————
TUMBLER RIDGE | 155020.025561315
(1registro)

Note que para responder a esta consulta nós temos que calcular a área de cada polígono. Se nós fizermos isto muito, faria mais sentido acrescentar uma coluna com as áreas de cada polígono à tabela, para a usarmos como um índice que iria aumentar o desempenho das consultas. Requisitando que os resultados sejam ordenados em sentido decrescente, e usando o comando "LIMIT"de PostgreSQL, nós podemos facilmente pegar apenas o maior valor, sem usar uma função agregada como o max().

7.1.4 Exemplos 2

Qual é o tamanho completo das estradas contidas inteiramente dentro de cada municipalidade?

Este é um exemplo de um "cruzamento espacial"("spatial join"), porque nós estamos trazendo junto dados de duas tabelas (fazendo um cruzamento), mas estamos usando uma condição de interação espacial (contido ou "contained") como a condição de cruzamento, ao invés de usar uma abordagem relacional usando um atributo em comum:

```
meu_DB=# SELECT m.name, sum(length(r.the_geom))/1000 as roads_km
FROM bc_roads AS r,bc_municipality AS m
WHERE r.the_geom && m.the_geom
AND contains(m.the_geom,r.the_geom)
GROUP BY m.name
ORDER BY roads_km DESC;

name | roads_km
——————————+——————
SURREY | 1687.24857093241
VANCOUVER | 1550.37923490885
LANGLEY DISTRICT | 849.559744805505
PRINCE GEORGE | 737.646890951342
BURNABY | 732.348440798669
RICHMOND | 698.334802617971
KELOWNA | 689.282898281948
KAMLOOPS | 653.015753055733
DELTA | 645.412004316109
MATSQUI | 626.442542300976
... (segue)
```

Esta consulta leva um certo tempo para ser concluída, porque cada estrada é sumarizada no resultado final (cerca de 250 mil estradas para essa tabela do nosso exemplo). Para menores sobreposições ("overlays") (centenas de registros em milhares) a resposta pode ser muito mais rápida.

Criar uma nova tabela com todas as estradas contidas na cidade de Prince George.

Este é um exemplo de uma sobreposição ("overlay"), que lê em duas tabelas e cria uma nova tabela como saída, e que consiste dos resultados espacialmente sobrepostos ("clipped"). Ao contrário do cruzamento espacial ("spatial join") demonstrado acima, esta consulta realmente cria novas geometrias (com seu resultado). Uma sobreposição é como um cruzamento espacial turbinado, pois é útil para trabalhos mais minuciosos de análise:

```
meu_DB=# CREATE TABLE pg_roads as
SELECT intersection(r.the_geom, m.the_geom) AS intersection_geom, length(r.the_geom) AS
rd_orig_length, r.*
FROM bc_roads AS r, bc_municipality AS m
WHERE r.the_geom && m.the_geom
AND intersects(r.the_geom, m.the_geom)
AND m.name = 'PRINCE GEORGE';
```

Use o seguinte comando para verificar o resultado:

meu_DB=# SELECT name from pg_roads;

Nota: Usa a tecla espaço para avançar ou a tecla "q"para sair.

7.1.5 Exemplos 3

Qual é o tamanho em quilômetros, da rua "Douglas St.", em Victoria?

```
meu_DB=# SELECT sum(length(r.the_geom))/1000 AS kilometers
FROM bc_roads r, bc_municipality m
WHERE r.the_geom && m.the_geom
AND r.name = 'Douglas St'
AND m.name = 'VICTORIA';

kilometers
------------------
6.85339156709377
(1 registro)
```

Qual é o maior polígono de municipalidade que possui um buraco?

```
    meu_DB=# SELECT gid, name, area(the_geom) AS area
FROM bc_municipality
WHERE nrings(the_geom) > 1
ORDER BY area DESC LIMIT 1;

gid | name | area
——+——————+————————
13 | SPALLUMCHEEN | 257374619.430219
(1 registro)
```

Capítulo 8

Construindo índices

Como construir índices de modo a utilizar melhor o banco de dados espacial.

8.1 Lição 6 - Construindo índices

8.1.1 Construindo índices

Índices são o que faz possível usar grandes conjuntos de dados em um banco de dados espacial. Sem indexação, qualquer busca por uma feição iria requerer uma procura seqüencial passando por todos os registros no banco de dados. A indexação acelera as buscas por organizar os dados em uma árvore de procura (hierarquia) que pode ser rapidamente acessada para encontrar um dado em particular. O PostgreSQL suporta três tipos de índices por padrão:

B-Trees são usadas para dados que podem ser ordenados ao longo de um eixo; por exemplo, números, letra inicial, datas. Dados de GIS não podem ser racionalmente ordenados ao longo de um eixo. Qual é maior, (0,0) ou (0,1) ou (1,0)?. Assim, a indexação B-Tree é inútil para nós;

R-Trees dividem os dados em retângulos, sub-retângulos, e sub-sub retângulos, etc. R-Trees são usados por alguns bancos de dados espaciais para indexar dados de GIS, mas a implementação do R-Tree no PostgreSQL não é tão robusta como o GiST.

Índices GiST (Generalized Search Trees) dividem dados em "things to one side"(coisas para um lado), "things which overlap"(coisas que se sobrepõe), "things which are inside"(coisas que estão dentro) e podem ser usados em uma extensa gama de tipos de dados, inclusive dados de GIS. PostGIS usa um índice de R-Tree implementado em cima do GiST para indexar dados de GIS.

8.1.2 Índices GiST

GiST representa "Generalized Search Tree"e é uma forma generalizada de indexação. Além da indexação de GIS, GiST é usada para acelerar buscas em todos os tipos de estruturas de dados irregulares (arranjos de vetores, dados espectrais, etc) que não são normalmente suscetíveis a indexação B-Tree.

Uma vez que uma tabela de dados de GIS excede algumas milhares de linhas, você irá querer construir um índice para acelerar as buscas espaciais de dados (a menos que todas suas buscas sejam baseadas em atributos, nesse caso você iria construir um índice normal usando os campos de atributo).

A sintaxe para construir um índice de GiST em uma coluna de geometria é demonstrada à seguir:

CREATE INDEX [nome_do_índice] ON [nome_da_tabela]

USING GIST ([campo_de_geometria] GIST_GEOMETRY_OPS);

Construindo um índice espacial é um exercício computacionalmente intensivo: em tabelas ao redor de 1 milhão de linhas, em uma máquina de 300MHz Solaris, achamos que com a construção de um índice GiST leva aproximadamente 1 hora. Depois de construir um índice, é importante forçar PostgreSQL a coletar as estatísticas de tabela, e que são usadas para otimizar as busca:

VACUUM ANALYZE [nome_da_tabela] [nome_da_coluna];

– Isso é apenas necessário para instalações do PostgreSQL 7.4 ou anteriores

SELECT UPDATE_GEOMETRY_STATS([nome_da_tabela], [nome_da_coluna]);

No PostgreSQL índices GiST têm duas vantagens em relação aos índices R-Tree. Em primeiro lugar, índices GiST são "null safe"(seguro com nulos), significando que eles podem indexar colunas que incluem valores nulos. Em segundo lugar, índices GiST apóiam o conceito de "lossiness"(sem perda) que é importante quando lidamos com objetos de GIS com tamanhos maiores do que o tamanho de página do PostgreSQL (8Kb). "Lossiness"permite PostgreSQL armazenar somente a parte "importante"de um objeto em um índice - no caso de objetos de GIS, só o retângulo envolvente. Objetos de GIS maiores do que 8K causarão que os índices R-Tree falhem no processo de construção.

No exemplo iremos criar o índice para a tabela "bc_roads":

meu_DB=# CREATE INDEX bc_roads_gidx
ON bc_roads USING gist (the_geom gist_geometry_ops);

CREATE INDEX

Feito isso, para verificar se o seu índice foi criado rode o comando (\d bc_roads):

meu_DB=# \d bc_roads

Tabela "public.bc_roads"
Coluna | Tipo | Modificadores
————+————————+————————————————————
gid | integer | not null default
nextval('bc_roads_gid_seq'::regclass)
name | character varying(40) |
the_geom | geometry |

Índices:
"bc_roads_pkey"PRIMARY KEY, btree (gid)
"bc_roads_gidx"gist (the_geom)
... (segue)

Para atualizar as estatísticas do nosso banco de dados, rode o comando VACUUM ANALYZE para fazer um "update"geral, ou rode ele apenas na tabela na qual você gerou o índice, seguindo o exemplo:

meu_DB=# VACUUM ANALYZE bc_roads the_geom

Agora realizaremos duas consultas, a primeira sem usar o índice (não-indexada) e a segunda usando (indexada). O resultado da segunda consulta será bem mais rápida (isso pode ser facilmente percebido em computadores mais lentos):

```
meu_DB=# SELECT gid, name
FROM bc_roads
WHERE CROSSES (the_geom, GeomFromText ('LINESTRING(1220446 477473,1220417 477559)',
-1));
```

e

```
meu_DB=# SELECT gid, name
FROM bc_roads
WHERE the_geom && GeomFromText('LINESTRING(1220446 477473,1220417 477559)', -1)
AND CROSSES(the_geom, GeomFromText ('LINESTRING(1220446 477473,1220417 477559)',
```

-1));

8.1.3 Usando Índices

Índices aceleram de maneira invisível o acesso de dados: uma vez que o índice é construído, o "planejador de consultas"é quem decide quando usar a informação do índice para acelerar a consulta. Infelizmente, o "planejador de consultas"do PostgreSQL não otimiza bem o uso de índices de GiST, então às vezes as consultas que deveriam usar um índice espacial fazem uma busca seqüencial pela tabela inteira.

Se você acha que seus índices espaciais não estão sendo usados (ou seus índices de atributo para aquele assunto) existem algumas soluções para verificar se eles estão funcionando corretamente:

Primeiramente, certifique-se que as estatísticas são coletadas sobre o número e distribuições dos valores em uma tabela, para proporcionar para o "planejador de consultas"com melhor informações que permitam a tomada de decisão se usa os índices ou não. Para instalações PostgreSQL 7.4 e anteriores isso é feito rodando o seguinte comando:

update_geometry_stats([nome_da_tabela, nome_da_coluna])

(calcula as estatísticas de distribuições de valores em uma tabela)

e

VACUUM ANALYZE [nome_da_tabela] [nome_da_coluna]

(calcula as estatísticas a respeito do número de valores em uma tabela).

Começando com PostgreSQL 8.0 rodar VACUUM ANALYZE fará ambas as operações. Você pode usar o "VACUUM"("limpar") regularmente em seus bancos de dados, para isso basta agendar no "cron"para executá-lo fora dos horários de pico do servidor PostgreSQL.

Se "limpar"o seu banco de dados não funcionar, você pode forçar o planejador a usar as informações de índice com o seguinte comando:

SET ENABLE_SEQSCAN=OFF

Você deveria apenas usar este comando esporadicamente, e em consultas espacialmente indexadas. Já que em geral, o planejador sabe melhor do que você o que fazer a respeito na hora de escolher entre os índices B-Tree normais e os GiST. Uma vez que você rodou sua consulta, você deveria considerar a opção de reverter o ENABLE_SEQSCAN ao seu estado original, de maneira que em outras consultas o planejador funcionará de forma normal.

Nota: A partir de versão 0.6 do PostGIS, não deveria ser necessário forçar o planejador a usar os índices com ENABLE_SEQSCAN.

Se você acha que o planejador decide errado sobre o custo de consultas sequencial versus índice, tente reduzir o valor de "random_page_cost"em postgresql.conf ou usando SET "random_page_cost"=#. O valor padrão para o parâmetro é 4, tente ajustá-lo para 1 ou 2. Reduzir esse o valor faz com que o planejador fique mais inclinado a usar consultas usando o índice.

8.1.4 Vantagens dos Índices

Na construção de uma consulta é importante lembrarmos que somente os operadores baseados em retângulos envolventes (bounding box) como &&, podem tirar proveito do índice espacial de GiST. Funções como DISTANCE() não podem usar o índice para otimizar suas operações. Por exemplo, a seguinte consulta seria bastante lenta em uma tabela grande:

SELECT name FROM bc_roads
WHERE DISTANCE(the_geom, GeomFromText('POINT(1220000 470000)', -1)) < 100;

Esta consulta está selecionando todas as geometrias em geom_table que está dentro de 100 unidades do ponto (1220000 470000). Esta operação será relativamente lenta porque está calculando a distância entre cada ponto da tabela e nosso ponto específico (isto é, um cálculo de DISTANCE() para cada linha da tabela). Podemos evitar isto usando o operador && para reduzir o número de cálculos de distância necessários:

SELECT the_geom FROM bc_roads
WHERE the_geom && 'BOX3D(1219900 469900, 1220100 470100)'::box3d
AND DISTANCE(the_geom, GeomFromText('POINT(1220000 470000)', -1)) < 100

Esta consulta seleciona as mesmas geometrias, mas faz isto de um modo mais eficiente. Supondo que existe um índice GiST em the_geom, o "planejador de consultas"reconhecerá que pode usar o índice para reduzir o número de linhas antes de calcular o resultado da função distance (). Note que a geometria de BOX3D que é usada dentro da operação && é uma caixa quadrada de 200 unidades e centrada no ponto original - esta é nossa "caixa de procura". O

operador && usa o índice para reduzir rapidamente o resultado atribuído a nossa busca à apenas essas geometrias que têm um retângulo envolvente que se sobrepõem com a nossa "caixa de procura". Supondo que nossa "caixa de procura"é muito menor que as extensões da tabela de geometria inteira, isto reduzirá drasticamente o número de cálculos de distância que precisam ser feitos.

Uma dica: para medir o consumo de tempo e processamento um recurso útil é colocar o comando "EXPLAIN"no início de cada consulta.

Capítulo 9

Acessando os dados do PostGIS em outros softwares

Como acessar o PostGIS a partir de outros softwares como o QGIS.

9.1 Lição 7 - Acessando os dados do PostGIS em outros softwares

9.1.1 Acessando os dados do PostGIS em outros softwares

As consultas espaciais (e resultados) no PostGIS não possuem saída gráfica. Quando queremos apenas estatísticas ou respostas diretas, nada além disso é necessário. Porém em outros casos, a visualização gráfica nos permite ver e expressar certos detalhes com mais clareza.

Pensando dessa forma a seguir iremos abordar alguns softwares de podem visualizar dados do PostGIS.

9.1.2 Visualizando dados exportados do PostGIS

A maneira mais "primitiva"de visualizar dados do PostGIS em outros softwares seria exportar os dados do banco de dados. Um dos formatos mais acessível é o Shapefile.

Para exportar seus dados nesse formato basta realizar o procedimento descrito na seção do curso "Exportando dados no PostGIS".

Outra método alternativo de exportar (e também importar) dados seria usar o pacote de ferramentas GIS vetorias OGR (incluso no pacote GDAL). Um exemplo da sua utilização esta à seguir:

```
> ogr2ogr -f "ESRI Shapefile"nome_do_arquivo_criado.shp
"PG:dbname=meu_DB"nome_da_tabela_exportada
```

9.1.3 Usando o QGIS

Outra maneira de visualizar dados do PostGIS seria através de softwares desktop GIS. Dentre eles o QGIS é um dos destaques.

O QGIS é um software de visualização de dados espaciais, com um bom suporte a dados raster e vetorias. Outras funcionalidades adicionais podem ser implementadas através da construção de plugins (como o SPIT - "*Shapefile to PostgreSQL Import Tool*"). No entanto, para lidar com os dados do PostGIS no QGIS já existe suporte nativo à:

- simples visualização dos dados;
- realizar consultas usando a interface gráfico do QGIS diretamente no PostGIS e visualizar os resultados gráficamente.

O procedimento de conectar o QGIS ao PostGIS (e PostgreSQL) é feito da seguinte forma:

- criar uma conexão entre o QGIS e o PostgreSQL através de assistente gráfico;
- conectar-se ao banco de dados (PostgreSQL);
- selecionar a camada (tabela a ser visualizada geograficamente) a ser importada;
- adicionalmente podemos usar o WHERE para filtrar os dados à serem importados no QGIS;
- importar a camada.

As principais vantagens de usar diretamente dados que estão em um servidor PostGIS no QGIS seriam:

- conectar-se diretamente ao PostGIS e importar os dados garante um acesso amplo e rápido caso um grande número de pessoas utilize a mesma base de dados;
- se a base de dados é atualizada constantemente, usar diretamente os dados do servidor evita que pessoas tenham que exportar os dados, e também **diminui as chances que elas utilizem apenas arquivos que possam conter dados desatualizados.**

9.1.4 Usando Mapserver

Com o crescimento da Internet, ampliaram-se muito as maneiras de disponibilizar informações geográficas através da Web.

Dentre elas destacamos o Minnesota Mapserver (ou apenas Mapserver). Ele é um servidor de mapas Web para Internet que adapta-se a especificação do Servidor de Mapas na Internet OpenGIS. Com ele podemos criar sites que permitam ao usuários visualizar dados espaiais através da Web.

Outra funcionalidade interessante para nós é capacidade do Mapserver em utilizar diretamente dados do PostgGIS em websites (mesmo que cada servidor esteja instalado em um máquina diferente).

Para usar o PostGIS com Mapserver, você precisará conhecer como configurar o Mapserver (algo que está além da estensão desse curso). Ou requisitos mínimos para usar o PostGIS em conjunto com o Mapserver são:

- Versão 0.6 ou mais nova de PostGIS;
- Versão 3.5 ou mais nova de Mapserver.

O Mapserver acessa dados de PostGIS/PostgreSQL como qualquer outro cliente de PostgreSQL (usando a biblioteca **libpq**). Isto significa que o Mapserver pode ser instalado em qualquer máquina com acesso de rede (internet ou intranet) e para o servidor de PostGIS, e desde que o sistema tenha as bibliotecas cliente de PostgreSQL.

Os passos para usar o Mapserver são os seguintes:

1. Compile e instale Mapserver, com qualquer opções que você desejar, sem esquecer a opção de configuração -with-postgis"(para habilitar o suporte ao PostGIS);
2. Em seu arquivo de mapas Mapserver, acrescente uma camada de PostGIS. Por exemplo:

```
LAYER
        CONNECTIONTYPE postgis
        NAME "widehighways"
        # Connect to a remote spatial database
        CONNECTION "user=dbuser dbname=gisdatabase host=bigserver"
        # Get the lines from the 'geom' column of the 'roads' table
        DATA "geom from roads"
        STATUS ON
        TYPE LINE
        # Of the lines in the extents, only render the wide highways
        FILTER "type = 'highway' and numlanes >= 4"
        CLASS
```

```
                # Make the superhighways brighter and 2 pixels wide
                EXPRESSION ([numlanes] >= 6)
                COLOR 255 22 22
                SYMBOL "solid"
                SIZE 2
        END
        CLASS
                # All the rest are darker and only 1 pixel wide
                EXPRESSION ([numlanes] < 6)
                COLOR 205 92 82
        END
END
```

No exemplo acima, as diretivas de PostGIS-específicos são como segue:

CONNECTIONTYPE

Para camadas de PostGIS, este é sempre "postgis."

CONNECTION

A conexão de banco de dados é especificada por um conjunto de variáveis (com os valores padrões em <>):

user=<usuário> password=<senha_do_usuário>
dbname=<nome_do_banco_de_dados> hostname=<servidor> port=<5432>

no exemplo: CONNECTION "user=dbuser dbname=gisdatabase host=bigserver"

Alguns parâmetros podem ser omitidos. Mas no mínimo você deveria colocar o nome do banco de dados e usuário para se conectar.

DATA

O formato deste parâmetro é «coluna> from <nome_da_tabela>"onde a coluna é a coluna espacial a ser traduzida/visualizada no mapa, e tabela é a fonte dos dados.

FILTER

O FILTER (filtro) deve ser uma string SQL válida, e que correspondendo a uma consulta SQL (com a palavra-chave "WHERE"). Por exemplo, visualizar apenas as estradas com 6 ou mais pistas, use um filtro de "num_lanes >= 6."

3 No seu banco de dados espacial, assegure-se que você tem índices espaciais (GiST) construídos para todas as camadas que serão utilizadas pelo Mapserver.
CREATE INDEX [nome_do_índice]
ON [nome_da_tabela]
USING GIST ([coluna_de_geometria] GIST_GEOMETRY_OPS);

1. Se você for fazer consulta em suas camadas usando Mapserver também precisará de um índice "oid".

O Mapserver requer identificadores únicos para cada registro espacial ao fazer as consultas, e o módulo do Mapserver para manipular os dados do PostGIS usa o valor oid PostgreSQL para prover estes identificadores únicos. Um efeito colateral disto é que para fazer acesso randômico rápido de registros durante as consultas, um índice "oid"é necessário.

Para construir um índice "oid", use o SQL seguinte:

CREATE INDEX [nome_do_índice] ON [nome_da_tabela] (oid);