CURSO
Administração de PostgreSQL

MÓDULO 01

GINEAD

www.ginead.com.br

AVISO DE RESPONSABILIDADE

As informações contidas neste material de treinamento são distribuídas “NO ESTADO EM QUE SE ENCONTRAM”, sem qualquer garantia, expressa ou implícita. Embora todas as precauções tenham sido tomadas na preparação deste material, a Processor Alfamídia LTDA. não têm qualquer responsabilidade sobre qualquer pessoa ou entidade com respeito à responsabilidade, perda ou danos causados, ou alegadamente causados, direta ou indiretamente, pelas instruções contidas neste material ou pelo software de computador e produtos de hardware aqui descritos.

Conteúdo retirado do site do grupo de desenvolvimento do PostgreSQL, traduzido e adaptado por Processor Alfamídia LTDA.

PostgreSQL - Administração

Unidade 1: Instalação e configuração em Linux

Sobre o Curso

O Curso PostgreSQL Administração oferece aos alunos uma visão ampla da estrutra do física e lógica do banco de dados , bem como dos aplicativos operacionais do mesmo.

Ao final do curso os alunos serão capazes de instalar e configurar um banco de dados para uso em aplicações comerciais.

Formato do Curso

Esse curso é dividido em 11 unidades e um manual de referência, a maioria delas apresentando novas informações e contendo demonstrações de uso do banco de dados.

Objetivos do Curso

Após completar esse curso, você será capaz de

- Instalar e configurar um servidor de Banco de Dados
- Administrar e gerenciar o PostgreSQL

Pré-requisitos do Curso

Para obter o máximo desse treinamento, você deve já estar familiarizado com:

- Os sistemas operacionais de mercado.
- Linguagem SQL .

Unidade 1:
Instalação e configuração em Linux

Objetivos da Unidade

Após completar essa unidade, você será capaz de:

- Obter o PostgreSQL.
- Instalar o PostgreSQL
- Inicializar o Banco de Dados

Para instalação do Banco de Dados PostgreSQL é interessante uma básica noção do sistema operacional LINUX. Nesta apostila veremos passo a passo como instalar e inicializar o Banco de Dados.

Tópicos da Unidade

- Obtendo o PostgreSQL
- Instalando o PostgreSQL

Obtendo o software de instalação

No site do grupo de desenvolvimento do PostgreSQL (**www.postgresql.org**) podemos fazer o download do software sem custos. O software pode ser obtido em duas formas:

Fonte: baixando o fonte do PostgreSQL talvez seja possível instalar o Banco de Dados em um sistema LINUX que não se encontra na lista dos suportados.

Pacote: em uma segunda fase o grupo disponibiliza os pacotes (RPM's) de instalação do PostgreSQL.

Requerimentos

Em geral, uma plataforma moderna tipo Unix-compatível deveria ser capaz de rodar o PostgreSQL. O resumo de plataformas que receberam testes específicos e as outras não suportadas estão listadas logo abaixo nesta apostila. No diretório doc da distribuição que você estiver usando você encontra diversos FAQ específicos de várias plataformas caso você necessite de ajuda.

Os pré-requisitos que existem para a compilação do PostgreSQL:

• O GNU make é requerido; outros pragramas make podem *não funcionar*. O GNU make é frequentemente instalado com o nome de gmake; essa apostila irá sempre se referir à ele por gmake. (Em alguns sistemas GNU make é a ferramente default com o nome make.) Para testar se você possui o GNU make instalado, digite

gmake --version

É recomendado o uso da versão 3.76.1 ou posterior.

• Você precisa um compilador C ISO/ANSI. As versões recentes do GCC são recomendáveis, mas o PostgreSQL é conhecido por poder ser construído por uma enorme variedade de compiladores de diferentes empresas.

• gzip é também necessário para se deszipar a própria distribuição (caso você baixe os fontes).

• A Readline library do GNU (para edição confortável de linhas e busca de comandos no arquivo de history) será automaticamente usada se for encontrada. Você pode desejar instalá-la antes de proceder, mas isso não é essencial. (No NetBSD, a library libedit é Readline compatível e será usada se libreadline não for encontrada.)

• Os programas GNU Flex e Bison são necessários para a compilação quando se for começar do zero, mas eles *não* são necessários quando ela for feita apartir de um pacote de fontes pois os arquivos pré-gerados de saída já estão incluidos no pacote.

Verifique se você tem espaço livre suficiente em disco. É necessário ter pelo menos 30 MB de espaço para a árvore dos fontes durante a compilação e cerca de 10 MB para o diretório de instalação. Um cluster vazio de bancos de dados usam cerca de 20 MB, os bancos de dados usam mais ou menos cinco (5) vezes mais espaço em disco do que arquivos texto com o mesmo conteúdo em dados. Se você for fazer alguns testes de regressão, você irá necessitar temporáriamente de uns extra 20 MB. Use o comando **df -h** para checar o espaço em disco.

Antes de instalar

Você precisa estar logado com super-usuário (root) para instalar o PostgreSQL. Verifique se já existe uma versão anterior do Banco de Dados rodando em seu sistema. Caso exista , pare o serviço e remova o aplicativo.

Para verificar a existência de uma versão anterior digite os seguintes comandos :

rpm –qa | grep –i postgre

ou

ps –aux | grep –i postmaster

Nota : caso exista uma versão anterior esta deve ser removida.

Instalação pelo pacote RPM

Seguindo regras a instalação através de pacote é muito simples, o cuidado que devemos ter é a confirmação de que o pacote adquirido é o suportado para nossa versão do LINUX. Feito esta verificação para instalar utilize o comando seguinte :

root# rpm –iv <nome do pacote.rpm>

onde nome do pacote equivale normalmente a versão do banco de dados.

Exemplo :

root# rpm –iv postgresql-7.2.1-i386.rpm

Instalação rápida pelo fonte

Para se instalar a partir do código fonte é necessário que se tenha instalado o compilador "GNU gcc", e utilitários de desenvolvimento, como o "GNU make". Esses programas fazem parte da distribuição original do LINUX. Para a instalação através do fonte siga os passos a seguir:

1. Faça login como root e mude para o diretório onde baixou o fonte.

2. Instale os fonte com o seguinte comando:

   root# tar –xzvf postgresql-7.2.1.tar

3. Mude para o diretório do PostgreSQL que foi criado:

   root# cd postgresql-7.2.1/

4. Execute o programa de configuração :

   root# ./configure

5. Execute o programa make e make install :

   root# gmake
   root# gmake install

Pronto, o software do banco de dados PostgreSQL está instalado.

Se você está fazendo um Upgrade

O formato interno de gravação dos dados do PostgreSQL mudam com os novos releases. Portanto, se você está fazendo um upgrade de uma instalação existente que não tem o número da versão "7.2.x", você deve fazer um backup e restaurar seus dados como mostrado aqui. Essas instruções assumem que sua instalação existente está no diretório `/usr/local/pgsql`, e que a área de dados está no `/usr/local/pgsql/data`. Substitua seus paths apropriadamente.

1. Tenha certeza de que o seu banco de dados não está sofrendo updates durante ou após o backup. Isso não afeta a integridade do backup, mas com certeza os dados alterados durante os updates não estarão inclusos. Se necessário, edite as permissões no arquivo `/usr/local/pgsql/data/pg_hba.conf` (ou equivalente) para disabilitar o accesso ao banco de todos, exceto você.

2. Para dumpear a sua instalação, digite:

pg_dumpall > *arquivo_de_saída*

Se você necessita preservar os OIDs (por exemplo quando você os estiver usando como foreign keys), use então a opção `-o` quando for rodar o **pg_dumpall**. O **pg_dumpall** não salva objetos largos. Cheque a seção XXXXXXX se você precisar fazer isso. Tenha certeza também de estar usando o comando **pg_dumpall** da versão que você está atualmente usando. O **pg_dumpall** não da versão 7.2 não deveria ser usando em bancos de dados de versõs anteriores a 7.2.

3. Se você está instalando uma nova versão na mesma localização da anterior então dê um shut down no antigo servidor antes de instar os novos arquivos :

kill -INT `cat /usr/local/pgsql/data/postmaster.pid`

Versões anteriores à 7.0 não têm o arquivo `postmaster.pid`. Se você está usando uma versão que não têm esse arquivo tente achar o id do processo usando o comando **ps ax | grep postmaster**, e usar o comando **kill** após.

Em sistemas que tem o PostgreSQL starteado na hora do boot, existe provávelmente um arquivo de start-up que irá fazer a mesma coisa. Por exemplo, no Red Hat Linux o comando **/etc/rc.d/init.d/postgresql stop** deve funcionar. Outra possibilidade é **pg_ctl stop**.

4. Se você está instalando uma nova versão na mesma localização da anterior é também uma boa idéia manter a anterior, em caso de você ter problemas e necessite reverter o processo. Use este comando para renomear a antiga instalação :

mv /usr/local/pgsql /usr/local/pgsql.old

Após você ter instalado o PostgreSQL 7.4, crie um novo diretório para a base de dados e starte o novo servidor. Lembre que você de executar esses comandos estando logado como postgres (o qual você já deve ter já que isso é um upgrade).

/usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data

/usr/local/pgsql/bin/postmaster -D /usr/local/pgsql/data

Finalmente, restore seus dados com

/usr/local/pgsql/bin/psql -d template1 -f *outputfile*

usando o *novo* psql.

Você pode também instalar a nova versão em paralelo com a velha para diminuir seu tempo com o banco parado.

Instalação normal pelo fonte

1. Configuração

O primeiro passo do procedimento de instalação é o de configurar árvore dos fontes para o seu sistema e escolher as opções que você gostaria. Isso é feito rodando o script chamado `configure`.

Para uma instalação default simplesmente digite

```
./configure
```

Esse script rodará uma série de testes para descobrir os valores das variáveis dependentes do sistema e detectar algumas peculiaridades do seu sistema operacional, finalmente ele irá criar diversos arquivos na árvore para gravar o que ele achou.

A configuração default irá construir o servidor e os utilitários, assim como todas as aplicações clientes e as interfaces que requerem apenas um compilador C. Todos os arquivos serão instalados por default no diretório `/usr/local/pgsql`.

Você pode customizar o processo de instalação e construção informando um ou mais opções de linha de comando para o `configure`:

`--prefix=`*`PREFIXO`*

Instala todos os arquivos dentro do diretório *`PREFIXO`* ao invés de `/usr/local/pgsql`. Na verdade, os arquivos serão instalados dentro de vários subdiretórios; nunca nenhum arquivo será instalado diretamente dentro do diretório *`PREFIXO`*.

Se você tem necessidades especiais, você pode também customizar subdiretórios individuais com as seguintes opções.

`--exec-prefix=`*`EXEC-PREFIXO`*

Você pode instalar arquivos que são independentes da estrutura dentro de um diretório diferente, *`EXEC-PREFIXO`*, do que *`PREFIXO`* foi setado. Isso pode ser útil para compartilhar arquivos independentes da estrutura entre vários hosts.

Se você omitir isso, então *EXEC-PREFIXO* é setado igual à *PREFIXO* e ambos os arquivos que são dependentes e independentes da estrutura serão instalados dentro da mesma árvore, o que provávelmente será isso que você quer.

`--bindir=`*DIRETORIO*

Especifica o diretório para os programas executáveis. O diretório default é *EXEC-PREFIXO*`/bin`, o que normalmente significa `/usr/local/pgsql/bin`.

`--datadir=`*DIRETORIO*

Seta o diretório para os arquivos de dados read-only usados pelos programas instalados. O default é *PREFIXO*`/share`. Note que isso não tem nada a ver com o local em que os arquivos do seu banco de dados serão colocados.

`--sysconfdir=`*DIRETORIO*

É o diretório para os vários arquivos de configuração, *PREFIXO*`/etc` por default.

`--libdir=`*DIRETORIO*

É a localização onde serão instaladas as libraries e os módulos dinamicamente lidos. O default é *EXEC-PREFIXO*`/lib`.

`--includedir=`*DIRETORIO*

É o diretório para a instalação dos arquivos header do C e do C++. O default é *PREFIXO*`/include`.

`--docdir=`*DIRETORIO*

Os arquivos de documentação, exceto as “man pages”, serão instalados nesse diretório. O default é *PREFIXO*`/doc`.

`--mandir=`*`DIRETORIO`*

As man pages que vêm com o PostgreSQL serão instaladas nesse diretório, nos seus respectivos `man`*`x`* subdiretórios. O default é *`PREFIXO`*`/man`.

Note: Todo o cuidado foi tomado para tornar possível instalar o PostgreSQL dentro de shared locations (tal como `/usr/local/include`) sem interferência com o resto do sistema. Primeiro, a string "`/postgresql`" é automaticamente adicionada à `datadir`, `sysconfdir`, e `docdir`, à menos que o nome do diretório já contenha a string "`postgres`" ou "`pgsql`". Por exemplo, se você escolher `/usr/local` como prefixo, a documentação será instalada em `/usr/local/doc/postgresql`, mas se o prefixo for `/opt/postgres`, então ela será colocada em `/opt/postgres/doc`. Segundo, o layout de instalação dos arquivos headers do C e do C++ foram reorganizados no release 7.2. Os arquivos headers públicos das interfaces dos clientes serão instalados dentro do `includedir`. Os arquivos headers internos e do servidor serão instalados dentro de diretórios privados dentro do `includedir`. Finalmente, um subdiretório privado será também criado, se apropriado, abaixo do `libdir` para os módulos dinamicamente lidos.

`--with-includes=`*`DIRETORIOS`*

`DIRETORIOS` é uma lista de diretórios separados por dois pontos (:) que serão adicionados à lista que o compilador usa para procurar por arquivos de header. Se você tem pacotes opcionais (tipo o GNU Readline) instalado em uma localização não standard, você deve usar essa opção e provávelmente a opção correspondente `–withlibraries`.

Exemplo: `--with-includes=/opt/gnu/include:/usr/sup/include`.

`--with-libraries=`*`DIRETORIOS`*

`DIRETORIOS` é uma lista de diretórios separados por dois pontos (:) para a procura de libraries. Você precisará usar essa opção (e a opção correspondente `--with-includes`) se você tiver packages instaladas em localizações não standard.

Exemplo: `--with-libraries=/opt/gnu/lib:/usr/sup/lib`.

`--enable-locale`

Habilita o suporte à locais. Há uma penalização de performance associada ao puporte à locias, mas se você estiver em um ambiente fora do Inglês, talvez essa seja uma boa opção.

`--enable-recode`

Habilita o suporte à caracteres single-byte.

`--enable-multibyte`

Habilita o suporte à codificação de caracteres multibyte (includindo o Unicode) e conversão de caracter sets. Note que algumas interfaces (tipo Tcl ou Java) esperam que todas as strings de caracter estejam em Unicode, essa opção é então required para o correto suporte à essas interfaces.

`--enable-nls[=`*`LIGUAGENS`*`]`

Habilita o Native Language Support (NLS), isso é, a habilidade de display de mensagens em uma outra língua que o Inglês. *`LINGUAGENS`* é uma lista separada por espaços de linguas que você quer obter suporte, por exemplo `--enable-nls='de fr'`. (A interseção entre a sua lista e o set das já atualmente suportadas será computada automáticamente.) Se você não especificar uma lista, então todas as traduções disponíveis serão instaladas.

Para usar essa opção, você precisará da implementação gettext API. Alguns sistemas operacionais já tem isso built-in (exemplo, Linux, NetBSD, Solaris), para outros sistemas você pode fazer um download um pacote add-on daqui:

http://www.postgresql.org/~petere/gettext.html.

Se você está usando a implementação gettext do GNU C library então adicionalmente você irá precisar do pacote GNU gettext para alguns programas utilitários. Para qualquer outra implementação você não precisará disso.

`--with-pgport=`*`NUMERO`*

Selecione *`NUMERO`* como a porta default para o servidor e os clientes. A porta default é 5432. A porta pode sempre ser modificada depois, mas se você especifica isso aqui, então ambos os clientes e o servidor terão o mesmo número default compilado, o que pode ser muito conveniente. Geralmente a única boa razão para não selecionar um valor default é se você pretende rodar múltiplos servidores PostgreSQL em uma mesma máquina.

`--with-CXX`

Monta a C++ interface library.

`--with-perl`

Monta o módulo de interface Perl. Ele será instalado no local usual para os Perlmodules (tipicamente dentro do `/usr/lib/perl`), você deve ter acesso do root para executar o processo de instalação (veja o passo 4). Você necessita ter o Perl 5 instalado para usar essa opção.

`--with-python`

Monta o módulo de interface Python. Você deve ter acesso do root para executar o processo de instalação do Python no seu diretório default (`/usr/lib/python`*x*`.`*y*). Você necessita ter o Python instalado para usar essa opção e o seu sistema necessita suportar shared libraries. Se ao invés de usar o Python você deseja montar um completamente novo interpretador binário, você deve fazer isso manualmente.

`--with-tcl`

Monta os componentes que requerem Tcl/Tk, que são os libpgtcl, pgtclsh, pgtksh, PgAccess, e PL/Tcl. Mas veja abaixo sobre `--without-tk`.

`--without-tk`

Se você especificar `--with-tcl` e essa opção, então os programas que requerem Tk (pgtksh e PgAccess) serão excluídos.

`--with-tclconfig=`*DIRETORIO*

`--with-tkconfig=`*DIRETORIO*

Tcl/Tk instala os arquivos `tclConfig.sh` e `tkConfig.sh`, os quais contém informações de configuração necessárias para montar os módulos de interface para o Tcl ou Tk. Esses arquivos são normalmente encontrados automáticamente nas suas conhecidas localizações, mas se você quiser usar uma versão diferente de Tcl ou Tk você pode especificar o diretório onde encontrá-los.

`--enable-odbc`

Monta o driver ODBC. Por default, ele será independente de um gerenciador de drivers. Para se trabalhar melhor com um gerenciador de drivers já instalado no seu sistema, use uma das próximas opções em conjunto com essa.

`--with-iodbc`

Monta o driver ODBC para uso com iODBC.

`--with-unixodbc`

Monta o driver ODBC para uso com unixODBC.

`--with-odbcinst=`*`DIRETORIO`*

Especifica o diretório onde o driver ODBC irá encontrar o seu arquivo de configuração : `odbcinst.ini`. O default é `/usr/local/pgsql/etc` ou outro qualquer que você especificou com `--sysconfdir`. Isso deveria estar arranjado de forma que o driver lê o mesmo arquivo que o gerenciador de drivers lê.

Se `--with-iodbc` ou `--with-unixodbc` são usados, essa oção será ignorada porque nesse caso o gerenciador de drivers cuida da localização do arquivos de configuração.

`--with-java`

Monta o driver JDBC e as packages associadas. Essa opção requere que o programa Ant também seja instalado (bem como a JDK, claro).

`--with-krb4[=`*`DIRETORIO`*`]`

`--with-krb5[=`*`DIRETORIO`*`]`

Monta o suporte para a autenticação Kerberos. Você pode usar tanto o Kerberos versão 4 ou 5, mas não ambos. O argumento *`DIRETORIO`* especifica o diretório root da instalação Kerberos; `/usr/athena` é assumido como default. Se os arquivos header importantes e as libraries não forem ser encontradas abaixo de um diretório pai em comum, então você deve usar as opções `--with-includes` e `--with-libraries` em conjunto com essa opção. Se, por outro lado, os arquivos requeridos se encontrarem em uma localização

que é varrida por default (exemplo, `/usr/lib`), então você pode esquecer o argumento. `configure` irá checar pelos arquivos header files e libraries para ter certeza de que a sua instalação Kerberos é suficiente antes de proceder.

`--with-krb-srvnam=`*`NOME`*

O nome do serviço principal do Kerberos. `postgres` é o default. Não há provávelmete razão para mudar isso.

`--with-openssl[=`*`DIRETORIO`*`]`

Monta o suporte para conexões SSL encriptadas. Isso requer o pacote OpenSSL instalado. O argumento *`DIRECTORIO`* especifica o diretório root da instalação do OpenSSL; o local default é `/usr/local/ssl`. `configure` irá checar pelos arquivos header files e libraries para ter certeza de que a sua instalação OpenSSL é suficiente antes de proceder.

`--with-pam`

Monta o suporte para PAM (Pluggable Authentication Modules).

`--enable-syslog`

Habilita o servidor PostgreSQL para usar o syslog. (O uso essa opção não significa que você tenha que se logar usando o syslog ou que isso será feito por default, isso simplesmente torna possível a opção de torná-lo on a qualquer hora.)

`--enable-debug`

Compila totdos os programas e libraries com os síbolos de debug. Isso que dizer que você pode rodar programas através de um debugger para analizar problemas. Isso faz crescer o tamanho dos executáveis considerávelmente, e para os compiladores não-GCC isso geralmente disabilita a otimização do compilador, causando travamentos e paradas no sistema. Entretanto, ter os símbolos disponível é extremamente bom quando surgir algum problema. Atualmente, essa opção é recomendada para instalaçoes que irão entrar em produção apenas se você usa GCC. Você deveria habilitar essa opção sempre que for fazer trabalho de desenvolvimento ou rodando uma versão beta do banco.

`--enable-depend`

Habilita a checagem automática de dependências. Com essa opção, os makefiles serão setados de modo que todos os objetos afetados irão ser recompilados quando qualquer arquivo header for modificado. Isso é muito útil quando você estiver fazendo algum desenvolvimento, mas isso causará um grande aumento de código se você pretende apenas compilar uma vez e instalar. Atualmente essa opção só funciona se você usar GCC. Se você prefere um compilador C ou C++ diferente dos que o `configure` usa, sete as variáveis de ambiente `CC` ou `CXX`, respectivamente, para o programa de sua escolha. Similarmente, você pode sobrescrever as variáveis flag default do compilador com as variáveis `CFLAGS` e `CXXFLAGS`. Por exemplo:

```
env CC=/opt/bin/gcc CFLAGS='-O2 -pipe' ./configure
```

2. Compilação

Para começar a compilação, digite

```
gmake
```

(Lembre de usar o GNU make.) A compilação pode levar de 5 minutos até meia-hora dependendo do seu hardware. A última linha mostrada deveria ser

```
All of PostgreSQL is successfully made. Ready to install.
```

3. Testes de Regressão

Se você quiser testar esse novo servidor compilado antes de instalá-lo, você pode rodar testes de regressão apartir de agora. Eles servem para verificar se o PostgreSQL rodará na sua máquina do jeito que os desenvolvedores planejaram. Digite

```
gmake check
```

(Isso não rodará se você estiver logado como root; faça isso como um usuário sem privilégios.) É possível que algum teste falhe, devido à diferenças de interpretação de mensagens de erro ou resultados de pontos flutuantes. Você pode repetir esse teste a qualquer hora mais tarde apenas repetindo esse comando.

4. Instalando os arquivos

Nota: Se você está fazendo upgrade de um sistema existente e está instalando os novos arquivos em cima dos anteriores, à esta altura, você já deveria ter feito um backup dos seus dados e dado um shutdown no servilor antigo, como já explicado na Seção anterior.

Para instalar o PostgreSQL digite

```
gmake install
```

Isso irá instalar os arquivos dentro dos diretórios onde foram especificados no passo 1. Tenha certeza de ter as permissões adequadas para gravar naquela área. Normalmente você precisa executar esse passo logado como root. Alternativamente, você poderia criar os diretórios destino antecipadamente e garantir que as permissões já estivessem sido dadas.

Se você compilar as interfaces do Perl ou do Python e você não foi o usuário root quando executou o comando acima, então parte da sua instalação provávelmente falhou. Nesse caso, logue como root e digite

```
gmake -C src/interfaces/perl5 install
```

```
gmake -C src/interfaces/python install
```

Se você não tem acesso como superusuário, então você está sozinho nessa: você ainda pode colocar os arquivos requeridos em outros diretórios onde o Perl ou o Python possam achá-los, mas como fazer isso é deixado apenas como um exercício.

A instalação standard provê apenas os arquivos header necessários para o desenvolvimento de aplicações clientes. Se você planeja em fazer algum desenvolvimento do lado do servidor (tipo funções customizadas ou tipos de dados escritos em C), então você deverá instalar toda a árvore include do PostgreSQL no seu diretório. Para fazer isso, digite

```
gmake install-all-headers
```

Isso irá gerar um ou dois megabytes a mais na sua instalação.

Instalação apenas do Cliente : se você quiser instalar apenas as aplicações clientes e as libraries de interface, use então os seguintes comandos :

```
gmake -C src/bin install
gmake -C src/include install
gmake -C src/interfaces install
```

```
gmake -C doc install
```

Para desfazer a instalação use o comando **gmake uninstall**. Entretanto, isso não irá remover qualquer diretório novo criado.

Após a instalação você pode abrir algum espaço removendo os arquivos compilados da árvore de fontes com o comando **gmake clean**. Isso irá preservar os arquivos feitos pelo programa configure, desse jeito você pode recompilar tudo novamente com **gmake** quando quiser. Para fazer voltar toda a árvore de fontes ao estado original em que ela foi distribuída, use **gmake distclean**. Se você for compilar o PostgreSQL para diversas plataformas usando a mesma árvore de fontes, você deve fazer sempre isso e reconfigurar para cada compilação.

Se você executar uma compilação e descobrir que suas opções estavam erradas, ou se você mudar alguma coisa que o configure investigue (por exemplo, você instala o GNU Readline), então é uma boa idéia rodar o **gmake distclean** antes de reconfigurar e recompilar. Sem isso, as suas mudanças de configuração podem não chegar onde elas devem.

Após a Instalação

Shared Libraries

Em alguns sistemas que têm shared libraries (a maioria deles têm) você deve dizer ao seu sistema como achar as shared libraries recentemente instaladas. Os sistems no qual isso *não* é necessário incluem BSD/OS, FreeBSD, HP-UX, IRIX, Linux, NetBSD, OpenBSD, Tru64 UNIX (conhecido antigamente como Digital UNIX), and Solaris.

O método para setar o path de pesquisa para as shared library varia entre as plataformas, mas o método mais usado é o setamento da variável de ambiente `LD_LIBRARY_PATH` mais ou menos assim:

Em Bourne shells (**sh**, **ksh**, **bash**, **zsh**)

```
LD_LIBRARY_PATH=/usr/local/pgsql/lib

export LD_LIBRARY_PATH
```

ou em **csh** ou **tcsh**

```
setenv LD_LIBRARY_PATH /usr/local/pgsql/lib
```

Troque `/usr/local/pgsql/lib` com o valor que você setou em `--libdir` no passo 1. Você deveria colocar esses comandos dentro de um arquivo shell de startup tipo o `/etc/profile` ou o `~/.bash_profile`. Você pode obter mais alguma boa informação associada com esse método em http://www.visi.com/~barr/ldpath.html.

Em alguns sistemas é preferível que você sete a variável de ambiente `LD_RUN_PATH` *antes* da compilação. Em caso de dúvida, dê uma olhada no manual do seu sistema (talvez **ld.so** ou **rld**). Se mais tarde você receber uma mensagem tipo

```
psql: error in loading shared libraries
libpq.so.2.1: cannot open shared object file: No such file or
directory
```

então talvez esse passo tivesse sido necessário. Simplesmente faça-o antes de prosseguir.

Se você está usando BSD/OS, Linux, ou SunOS 4 e você tem acesso como root execute o seguinte comando

```
/sbin/ldconfig /usr/local/pgsql/lib
```

(ou o equivalente diretório) após a instalação para habilitar o run-time linker e fazê-lo encontrar as shared libraries mais rápido.

Dê uma olhada na página **ldconfig** do manual para mais informação. No FreeBSD, NetBSD, e Open BSD o comando é

```
/sbin/ldconfig -m /usr/local/pgsql/lib
```

Outros sistemas não são conhecidos de terem um comando equivalente.

Variáveis de Ambiente

Se você instalou o PostgreSQL dentro do diretório `/usr/local/pgsql` ou algum outro que não é procurado pelos programas por default, adicione então `/usr/local/pgsql/bin` (ou o que você setou em

`--bindir` no passo 1) dentro do seu `PATH`. Para fazer isso, adicione a seguinte linha no seu arquivo de startup shell, tipo o `~/.bash_profile` (ou o `/etc/profile`, se você quiser passar isso para todos os usuários):

```
PATH=/usr/local/pgsql/bin:$PATH
```

Se você estiver usando o **csh** ou o **tcsh**, use então esse comando:

```
set path = ( /usr/local/pgsql/bin $path )
```

Para liberar o seu sistema a encontrar a documentação man, adicione a seguinte linha no seu arquivo de startup shell:

```
MANPATH=/usr/local/pgsql/man:$MANPATH
```

As variávies de ambiente `PGHOST` e `PGPORT` especificam para as aplicações clientes o host e a porta do servidor de banco de dados, sobrescrevendo as defaults compiladas. Se você indo rodar aplicações clientes remotamente então é conveniente que cada usuário que planeja usar o banco de dados sete o seu `PGHOST`. Isso não é requerido, entretanto:

os setups podem ser comunicados via linha de comando para a maioria dos programas cliente.

Plataformas Suportadas

O PostgreSQL foi verificado pelos seus desenvolvedores trabalhando nas plataformas listadas abaixo. Uma plataforma suportada geralmente significa que o PostgreSQL compila e instala de acordo com essas instruções e passa nos testes de regressão.

Nota: Se você está tendo problemas com a instalação em uma plataforma suportada, por favor, escreva para <pgsql-bugs@postgresql.org> ou <pgsql-ports@postgresql.org>, e não para as pessoas listadas aqui.

Inicializando a área de dados

O banco de dados padrão não foi pré-inicializado, sendo necessário efetuar este procedimento manualmente com os seguintes passos :

Neste momento subentende-se que já exista um usuário postgres do grupo postgres.

1. Crie um diretório para armazenar seus dados :

   root# mkdir “path do diretório”

2. Altere as permissões do diretório:

   root# chown postgres.postgres “path do diretório”

3. Mude para o usuário “postgres”

   root# su postgres

4. Posicione-se no diretório onde o postgreSQL foi instalado:

   postgres# cd /usr/local/pgsql/bin

5. Inicialize o diretório para receber os dados

   ./initdb –d “path do diretório”

Inicializando o banco de dados

Utilizando o editor vi inclua no .bash_profile do usuário postgres a linha seguinte:

```
PATH=/usr/local/pgsql/bin:$PATH
```

Faça um novo login com o usuário postgres.

Após ter concluído com sucesso a instalação do PostgreSQL e da área de dados inicial, você deve inicializar o servidor digitando:

pg_ctl –D “path_do_diretório” start

ou

postmaster –D “path_do_diretório”

Resumo da Unidade

- O PostgreSQL é 100% Open Source.
- Existem empresas especializadas no suporte do PostgreSQL.

Revisão da Unidade

1. Onde conseguimos o código fonte do PostgreSQL ?

2. Quanto custa a versão mais atualizada do PostgreSQL ?

GINEAD

Semeando Conhecimento