**Instalação Chado**

Se você tiver problemas, envie-os por e-mail para a [lista de discussão gmod-schema](https://github.com/GMOD/Chado/blob/master/chado/gmod-schema@lists.sourceforge.net) .

Esta versão funcionará com a versão mais recente do Generic Genome Browser (GBrowse) versão 1.68 ou superior. Se você tiver dificuldades com GBrowse e Chado, você pode querer dar uma olhada em obter um checkout svn do branch gbrowse-stable. As instruções de instalação do GBrowse estão incluídas nesse pacote. Além disso, para trabalhar com GBrowse, você precisará dos módulos Bio :: DB :: Das :: Chado que podem ser obtidos no CPAN.

**Pré-requisitos**

* PostgreSQL
* BioPerl
* go-perl
* formiga
* Módulos Perl

Há alguma configuração necessária para vários pré-requisitos documentados abaixo:

**Configuração do PostgreSQL**

Atualmente, os desenvolvedores GMOD estão usando 8.1 ou superior (PostgreSQL 9 não foi testado).

Itens a fazer com o Postgres para torná-lo pronto para uso:

* Faça-o aceitar conexões TCP / IP adicionando esta linha ao postgresql.conf (deve ser feito como usuário root ou postgres; o banco de dados deve ser reiniciado para que esta mudança tenha efeito):
* tcpip\_socket = true

(Esta opção não está disponível e não é necessária no PostgreSQL 8.1 ou superior.)

* Crie um usuário de banco de dados com permissão para eliminar e adicionar bancos de dados; o nome de usuário do banco de dados deve ser igual ao seu nome de usuário Unix para permitir que a construção do software avance sem problemas (deve ser feito como usuário postgres; createuser é um programa de linha de comando que vem com o pacote PostgreSQL):
* $ sudo su - postgres
* $ createuser --createdb <your username>
* $ exit # to exit out of the postgres user's shell
* Diga ao postgres que ele pode usar a linguagem plpgsql (como usuário postgres; createlang é um programa de linha de comando que vem com o pacote PostgreSQL):
* $ sudo su - postgres
* $ createlang plpgsql template1
* $ exit # to exit out of the postgres user's shell
* Edite o pg\_hba.conf (como o usuário 'root' ou 'postgres') para dar permissão ao usuário criado acima para acessar o banco de dados. Leia os comentários em pg\_hba.conf sobre as permissões. Um exemplo de pg\_hba.conf se parece com isto (que tem permissões muito vagas):
* # TYPE DATABASE USER CIDR-ADDRESS METHOD
* local all all trust
* # IPv4 local connections:
* host all all 127.0.0.1/32 trust
* # IPv6 local connections:
* host all all ::1/128 trust

NOTA: Se você estiver configurando uma instância de produção do Chado, agora é um bom momento para decidir como deseja definir os usuários e fazer a autenticação do cliente para seu banco de dados. Postgresql suporta vários métodos para definir usuários, incluindo o uso de usuários do sistema operacional, LDAP, Kerberos e muitos outros. Consulte o manual Postgesql para mais informações: <http://www.postgresql.org/docs/8.4/interactive/client-authentication.html><http://www.postgresql.org/docs/8.4/interactive/user-manag.html>

* Para a Pg 8.1+, se você quiser permitir conexões remotas, a opção listen\_addresses pode precisar ser modificada; ele permite um caractere curinga '\*', que corresponde a todas as interfaces de IP disponíveis ( *não* especifica os endereços de IP que podem se conectar). Defina isso no arquivo postresql.conf, que está no mesmo diretório que pg\_hba.conf.

Para obter informações sobre como ajustar o postgres para desempenho, consulte

http://gmod.org/wiki/PostgreSQL\_Performance\_Tips

e

http://www.varlena.com/varlena/GeneralBits/Tidbits/perf.html

Os dois parâmetros mais críticos para ajustar são shared\_buffers e effective\_cache size. O ajuste desses parâmetros pode exigir a modificação das configurações de memória em /etc/sysctl.conf, consulte a página de manual do sysctl para obter detalhes. Também crítica para o desempenho contínuo do postgres é a execução regular do comando VACUUM FULL ANALYZE. Este comando limpa dados antigos excluídos e analisa a estrutura do banco de dados para que o planejador de execução possa prever a maneira mais rápida de executar uma determinada consulta.

Embora o link acima descreva o ajuste em geral, os exemplos dados para ajustar os parâmetros do kernel são específicos do Linux. Para definir shmmax nas caixas do Mac OS X, edite

| **Versão** | **Caminho** |
| --- | --- |
| OS X 10.2 | / System / Library / StartupItems / SystemTuning / SystemTuning |
| OS X 10.3 | / etc / rc |
| OS X 10.5 | /etc/sysctl.conf |

para aumentar os valores de shmmax e shmall, assim:

sysctl -w kern.sysv.shmmax = 52428800 # bytes: 50

megs sysctl -w kern.sysv.shmmin = 1

sysctl -w kern.sysv.shmmni = 32

sysctl -w kern.sysv.shmseg = 8

sysctl -w kern. sysv.shmall = 25600 # 4k páginas: 100 megs

(esses são os valores usados ​​em um Mac com 1,2 G de RAM) e reinicie.

Para uma caixa Linux com 512 MB de RAM, use estes valores em /etc/sysctl.conf:

kernel.shmall = 134217728

kernel.shmmax = 134217728

e faça essas alterações no arquivo postgresql.conf:

tcpip\_socket = true # Substituído por listen\_addresses no Postgres 8.0+

work\_mem = 2048 # Este é "sort\_mem" se estiver usando Postgres 7.x

max\_connections = 32

**Configuração do Bioperl Live**

bioperl-live ou uma versão 1.6.1 ou superior. Veja [http://bioperl.org](http://bioperl.org/) .

**go-perl**

Pode ser obtido no CPAN usando o shell cpan com o comando

cpan install GO::Parser

**Formiga**

Ao instalar a partir do svn, o ant é necessário para mover os arquivos GMODTools do schema / GMODTools para o schema / chado. Ao instalar a partir de uma distribuição, isso não é necessário, pois os arquivos já terão sido movidos como parte do processo de construção.

**Módulos Perl**

Os módulos perl podem ser instalados através do shell CPAN e emitindo o comando 'install Bundle :: GMOD' que instalará todos os módulos abaixo, exceto SQL :: Translator, que é opcional.

| **Módulo** | **Usado em** |
| --- | --- |
| CGI | GBrowse |
| GD | GBrowse |
| DBI | GBrowse, Chado |
| DBD :: Pg | GBrowse, Chado |
| SQL :: Translator | Chado (apenas para um esquema Chado personalizado) |
| Digest :: MD5 | GBrowse |
| Text :: Shellwords | GBrowse |
| Gráfico | Bio-Chaos |
| Data :: Stag | Chado |
| XML :: Parser :: PerlSAX | Chado |
| Módulo :: Construir | Chado |
| Class :: DBI | GMODWeb ou com um esquema Chado personalizado |
| Class :: DBI :: Pg | GMODWeb ou com um esquema Chado personalizado |
| Class :: DBI :: Pager | GMODWeb ou com um esquema Chado personalizado |
| DBIx :: DBStag | Chado |
| XML :: Simples | Chado |
| LWP | Chado |
| Modelo | Chado |
| Bio :: Chado :: Schema | Chado |

**Instalando o Esquema Chado**

**Preparação do Ambiente**

Primeiro, você deve definir algumas variáveis ​​em seu ambiente. Se você estiver usando o bash ou um shell do tipo bash, isso é feito por meio de um comando como este:

$ export VARNAME = valor

Se você estiver usando tcsh ou outro shell semelhante ao csh, é feito assim:

Valor de $ setenv VARNAME

Para tornar a vida mais fácil para você, provavelmente você também desejará colocar esses comandos em seu arquivo .tcshrc ou .bashrc para que as variáveis ​​de ambiente estejam sempre disponíveis quando você efetuar login.

| **Variável** | **Uso** |
| --- | --- |
| GMOD\_ROOT | A localização da instalação do Chado (por exemplo, "/ usr / local / gmod"). Contém os arquivos de origem que definem o esquema, bem como as definições de configuração e espaço temporário. |
| CHADO\_DB\_NAME | O nome do seu banco de dados Chado |
| CHADO\_DB\_USERNAME | O nome de usuário para conectar ao Chado |
| CHADO\_DB\_PASSWORD | A senha para o usuário do banco de dados [opt] |
| CHADO\_DB\_HOST | O host no qual o banco de dados é executado (por exemplo, "localhost") [opt] |
| CHADO\_DB\_PORT | A porta na qual o banco de dados está escutando [opt] |

Conforme indicado, o host, a porta e a senha são opcionais.

Nota: existe um mecanismo para passar essas variáveis ​​diretamente para o instalador durante a etapa "perl Makefile.PL". Ao fornecer pares chave = valor, é possível evitar a configuração de variáveis ​​ambientais. A sintaxe é a seguinte:

$ perl Makefile.PL GMOD\_ROOT = / usr / local / gmod CHADO\_DB\_NAME = dev\_chado\_01

A compatibilidade com versões anteriores pode não ser mantida, pois este método de configuração do processo de instalação funcionará.

**Makefile**

No diretório chado (o mesmo diretório em que se encontra o INSTALL.Chado) execute o seguinte comando:

$ perl Makefile.PL

Serão solicitados vários valores de configuração usados ​​pelo Chado e suas ferramentas associadas:

\* Use the simple install (uses default database schema) [Y]

Answering yes eliminates the need to have SQL::Translator installed.

This is recomended, and that is all that is necessary in order to use

the full schema and run GBrowse and GMODWeb on top of it.

\* Use values in '/home/scott/gmod/build.conf'? [Y]

If `perl Makefile.PL` has been run before, answering yes to this

will cause Makefile.PL to use the configuration options from the

previous build.

\* What database server will you be using? [PostgreSQL]

Specify what database vendor to use. Currently only PostgreSQL works.

\* What is the Chado database name? [dev\_chado\_allenday\_05]

This will be the name of the created chado database.

\* What is the database username? [allenday]

Default user that the installed libraries should try to

connect to the database as.

\* What is the password for 'allenday'?

Password for the default user.

\* What is the database host? [localhost]

Host of the database daemon.

\* What is your database port? [5432]

Port of the database daemon.

\* Where shall downloaded ontologies go? [./tmp]

The directory where ontology files and there lock files will be stored

\* What is the default organism (common name, or "none")?

The organism name should be one what will be in the organism table.

When the database is created, several organisms will be there

by default; these include: human, fruitfly, mouse, mosquito,

rat, Arabidopsis thaliana, worm, zebrafish, rice, and yeast. (The

insert statements that create these default organisms are

contained in load/etc/initialize.sql).

\* Do you want to make this the default chado instance? [y]

You can have more than one Chado instance on a server, each with a

different name. You can supply one of those names when loading

GFF, for example "--dbprofile fly\_staging". If you don't supply the

--dbprofile option, it will just use the default database parameters.

If you answered 'No' to the simple install question, AutoDBI.pm

will now be created by SQL::Translator, see the CUSTOM DATABASE

SCHEMAS section below for more information.

**Instalação**

$ make # Faça o esquema

$ make install # Instale o script e os módulos

$ sudo make install # Para uma instalação global

Provavelmente precisa ser executado como root. Instala scripts de carregamento de dados no caminho do perl (normalmente / usr / local / bin ou / usr / bin), módulos perl, bem como coloca vários arquivos em $ GMOD\_ROOT e cria a infra-estrutura para registro de erros criando $ GMOD\_ROOT / logs e criando o arquivo /etc/log4perl.conf se ele ainda não existir.

$ make load\_schema # Instale o esquema no banco de dados. Cria banco de dados, apaga qualquer banco de dados com esse nome !!

$ make prepdb # Inserir dados básicos

Insere alguns itens úteis nas tabelas fundamentais do Chado. Ele usa load / etc / initialize.sql. Ele contém informações para vários organismos comuns e bancos de dados de origem (por exemplo, Genbank). Este arquivo pode ser editado para adicionar qualquer organismo ou banco de dados de origem, usando as instruções INSERT para os exemplos como um modelo. Observe também que o destino prepdb precisa ser executado antes do destino das ontologias, mas pode ser executado novamente mais tarde, se mais instruções insert forem adicionadas (por exemplo, para adicionar um novo organismo ou banco de dados).

$ make ontologies # Carregar ontologias

Obtém e instala várias ontologias. Requer uma conexão de rede. Absolutamente necessários são a Ontologia de Relacionamento e a Ontologia de Seqüência (SO). Todos os outros são opcionais, embora o vocabulário controlado por propriedade de recurso seja normalmente útil para carregar arquivos GFF, e a Ontologia Gene geralmente é útil para uma ampla variedade de anotações de recurso de gene. Observe que os arquivos de ontologia recuperados são armazenados no diretório especificado quando 'perl Makefile.PL' foi executado (o padrão é ./tmp). Para fazer uma instalação repetida, o diretório que contém a ontologia baixada deve ser removido. Além de 'rm -rf ./tmp', você também pode emitir omake clean, que limpará todos os arquivos e diretórios criados até este ponto da instalação. Observe também que carregar uma grande ontologia como a Gene Ontology levará vários minutos (talvez até uma hora).

Observe que, como make ontologiesbaixa os arquivos de ontologia de seus repositórios online, esta etapa está sujeita a falhas devido a problemas de rede.

Se você já possui os arquivos de ontologia desejados localmente, pode executar um comando para cada arquivo para carregá-lo. Observe novamente que a Ontologia de Relacionamento é necessária antes de todas as outras, e a Ontologia de Seqüência (SO) é absolutamente necessária para o funcionamento adequado do banco de dados. Os comandos para carregar uma ontologia são:

$ go2fmt.pl -p obo\_text -w xml / path / to / obofile | \

go-apply-xslt oboxml\_to\_chadoxml -> obo\_text.xml

Isso cria um arquivo chadoxml do arquivo obo - em seguida, execute:

$ stag-storenode.pl \

-d 'dbi:Pg:dbname=$CHADO\_DB\_NAME;host=$CHADO\_DB\_HOST;port=$CHADO\_DB\_PORT' \

--user $CHADO\_DB\_USERNAME --password $CHADO\_DB\_PASSWORD obo\_text.xml

Se você tiver outros arquivos de formato de ontologia, os comandos serão semelhantes; consulte a documentação de go2fmt.pl e go-apply-xslt para seu formato de arquivo.

É uma boa ideia neste ponto fazer um backup do banco de dados, especialmente se você carregou uma grande ontologia como GO. Para fazer um despejo completo do banco de dados, emita este comando:

$ pg\_dump db\_name > db\_dump.sql

e para restaurar o banco de dados, emita este comando:

$ psql db\_name < db\_dump.sql

**Carregando dados**

Com isso, a instalação do esquema está concluída. Por favor, veja os HOWTOs em <http://gmod.org/> para informações sobre como carregar o esquema Chado com dados.

**Esquemas de banco de dados personalizados**

Se você respondeu 'Não' à pergunta sobre fazer uma instalação simples durante perl Makefile.PL, você deve fornecer os arquivos default\_schema.sql e default\_nofuncs.sql. A melhor maneira de criar esses arquivos é usando bin / chado-build-schema.pl, um script perl com uma interface gráfica de usuário para construir interativamente um esquema Chado. Se estiver fornecendo suas próprias definições de tabela, você também terá que editar o arquivo chado-module-metadata.xml para definir como suas tabelas se relacionam com outras tabelas no esquema Chado. Embora não haja documentação do DTD desse arquivo, ele é relativamente simples. Consulte INSTALL.Custom para obter mais informações sobre como chado-build-schema.pl está relacionado ao processo de construção. Assim que seus arquivos default\_schema.sql e default\_nofuncs.sql estiverem no lugar no diretório de módulos, você pode executar perl Makefile.PL.