

# PostgreSQL 8.4

Diogo Biazus  
diogob@gmail.com

# PostgreSQL 7.X

- Totalmente ACID
- MVCC
- Tipos e operadores customizáveis
- Diversas linguagens procedurais
- Hot backup
- Código aberto (BSD)
- Facilidade de extensão
- PL/pgSQL, PL/Python, PL/Perl, PL/Java, PL/PHP, PL/Ruby, etc.

# PostgreSQL 8.0

- Win32
- Pontos de salvamento
- Recuperação a partir de logs
- Tablespaces
- Tratamento de erro em PL/pgSQL
- Otimizações

# PostgreSQL 8.3

- Warm-standby
- Commit em duas etapas
- Consultas RETURNING
- FTS integrado ao banco
- XML integrado
- Tipos UUID e ENUM
- Integração LDAP

# PostgreSQL 8.4 e Avante

- 8.4
  - Window functions
  - CTE e recursividade
  - Permissão de colunas
- Futuro
  - Hot-standby (rep. assíncrona nativa)
  - Replicação síncrona nativa
  - Bitmap index no disco
  - SEPostgres

# Portes e pacotes

- Disponibilidade do fonte e de pacotes para:
  - \*BSD
  - Linux
    - pacotes em .rpm e .deb.
  - Windows
    - Pacotes msi (a partir da 8.0)

# Instalando o servidor no Linux

- Debian
  - apt-get install postgresql-8.3
- Compilação simples
  - Fazer download do fontes em:  
<http://www.postgresql.org/ftp/source/>
  - Descompactar
  - ./configure
  - make install

# Instalando o servidor no Linux

- Se não for através do sistema de pacotes deve-se iniciar o agrupamento com o initdb
- É necessário criar a pl/pgsql nos bancos em que ela for utilizada
- Para instalar módulos do contrib basta instalar o pacote postgresql-contrib (se for usado sistema de pacotes) ou entrar no diretório contrib correspondente e digitar make install



# Instalando o servidor no Linux

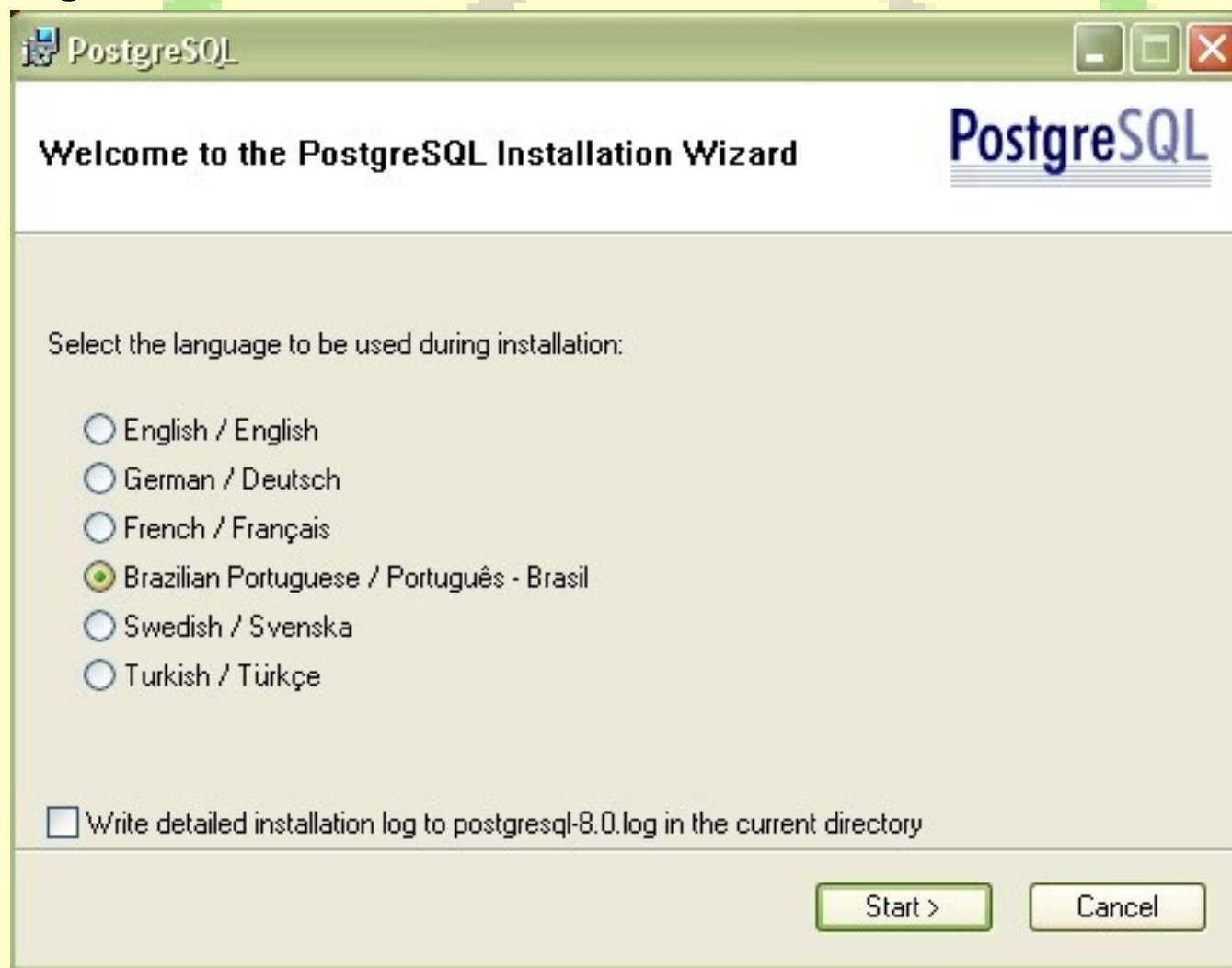
- initdb
  - Usar o flag --locale
  - O diretório deve pertencer ao usuário que vai rodar o banco. Máscara de permissão devem ser 700
  - Deve ser executado com o usuário que vai rodar o banco
  - O primeiro usuário do banco terá o mesmo nome do usuário do sistema que executou o initdb

# Instalando no Windows

- Fazer o download do msi
- Descompactar
- Iniciar o instalador

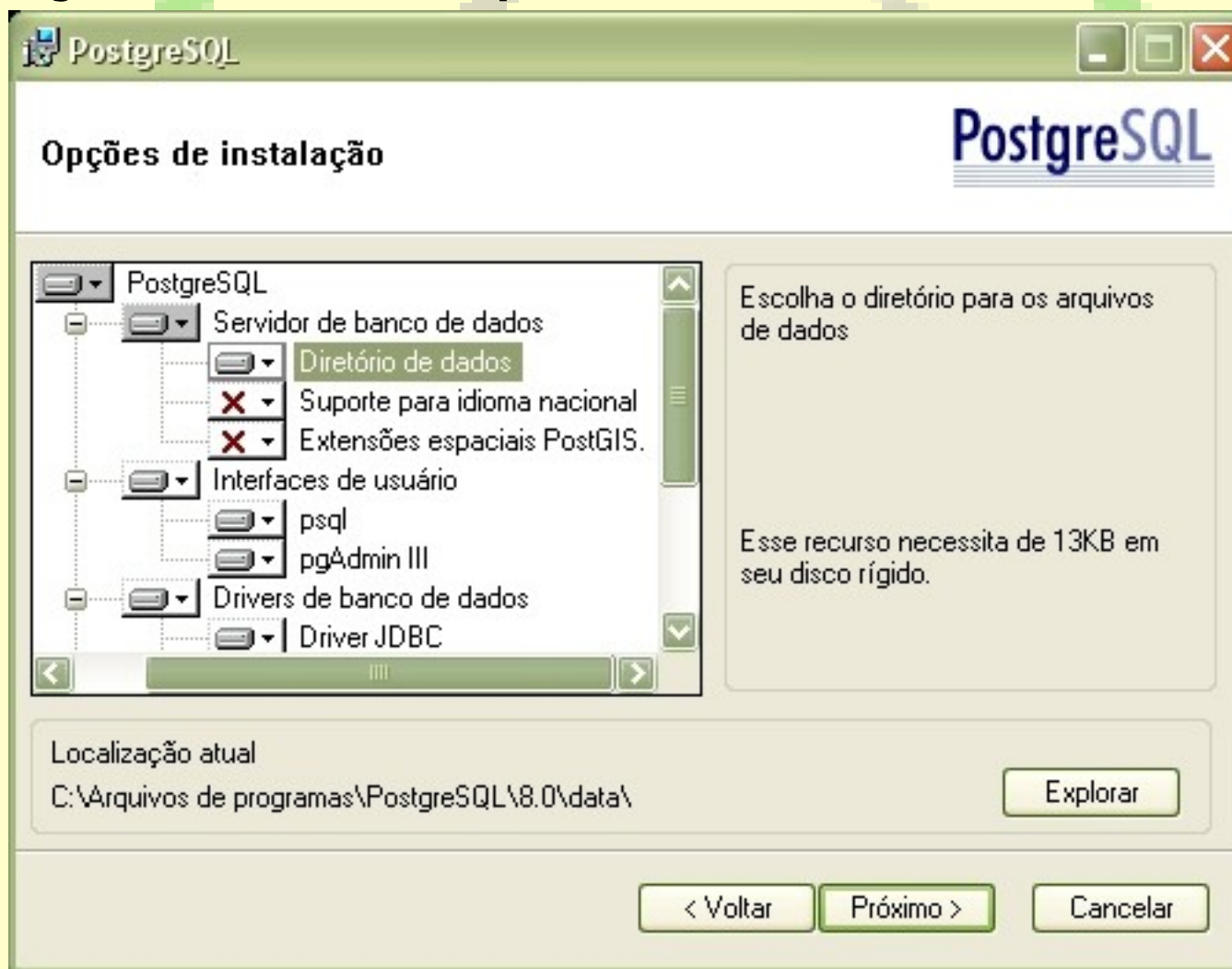
# Instalando no Windows

- Seleção de idioma do instalador



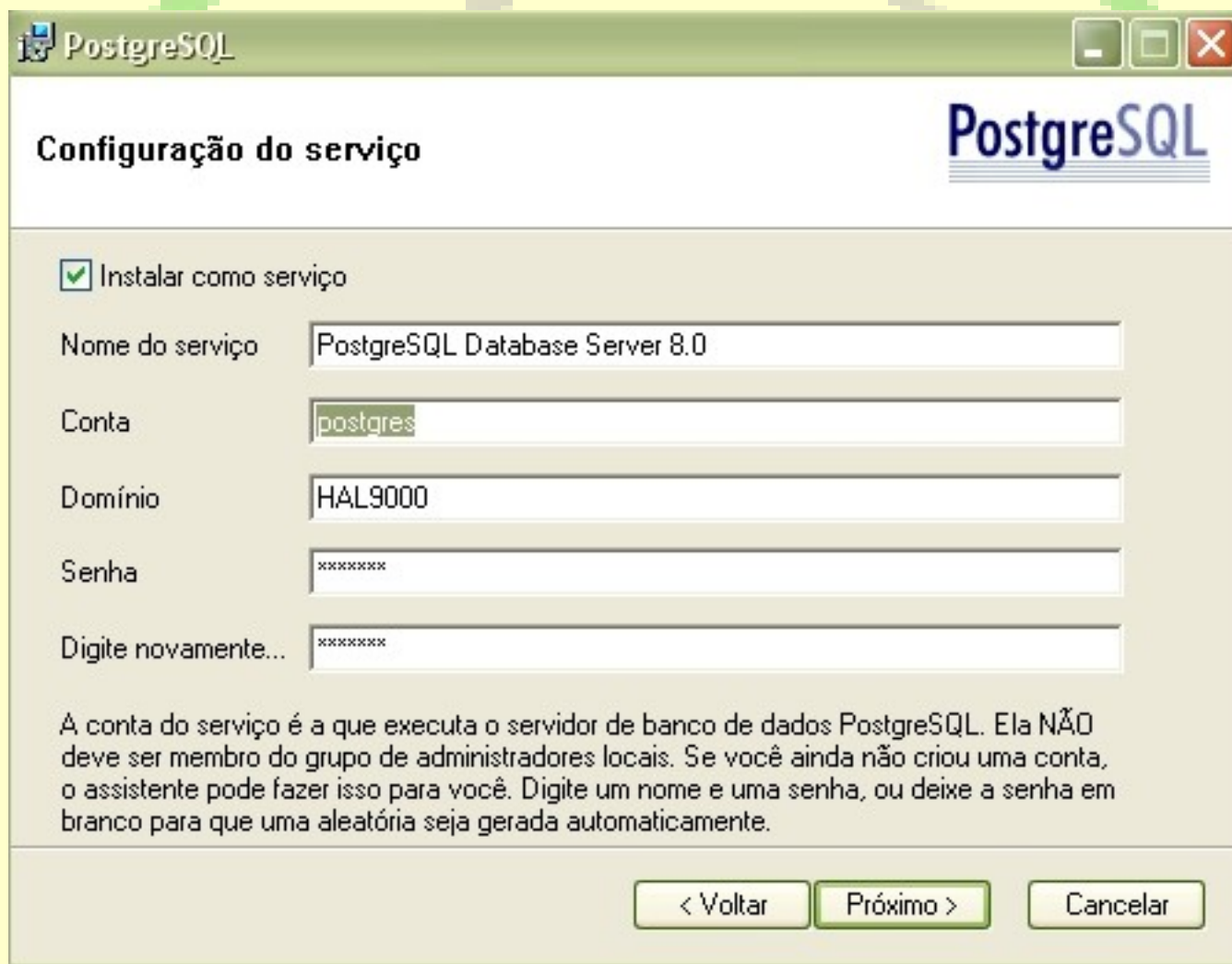
# Instalando no Windows

- Seleção de componentes



# Instalando no Windows

- Configuração de serviço



The screenshot shows the 'PostgreSQL' window titled 'Configuração do serviço'. It features the PostgreSQL logo in the top right corner. The main content area has a checkbox labeled 'Instalar como serviço' which is checked. Below this are five input fields: 'Nome do serviço' (PostgreSQL Database Server 8.0), 'Conta' (postgres), 'Domínio' (HAL9000), 'Senha' (masked with asterisks), and 'Digite novamente...' (also masked). At the bottom, there is a paragraph of text explaining the service account requirements and three buttons: '< Voltar', 'Próximo >', and 'Cancelar'.

PostgreSQL

## Configuração do serviço

☒ Instalar como serviço

Nome do serviço: PostgreSQL Database Server 8.0

Conta: postgres

Domínio: HAL9000

Senha: xxxxxxxx

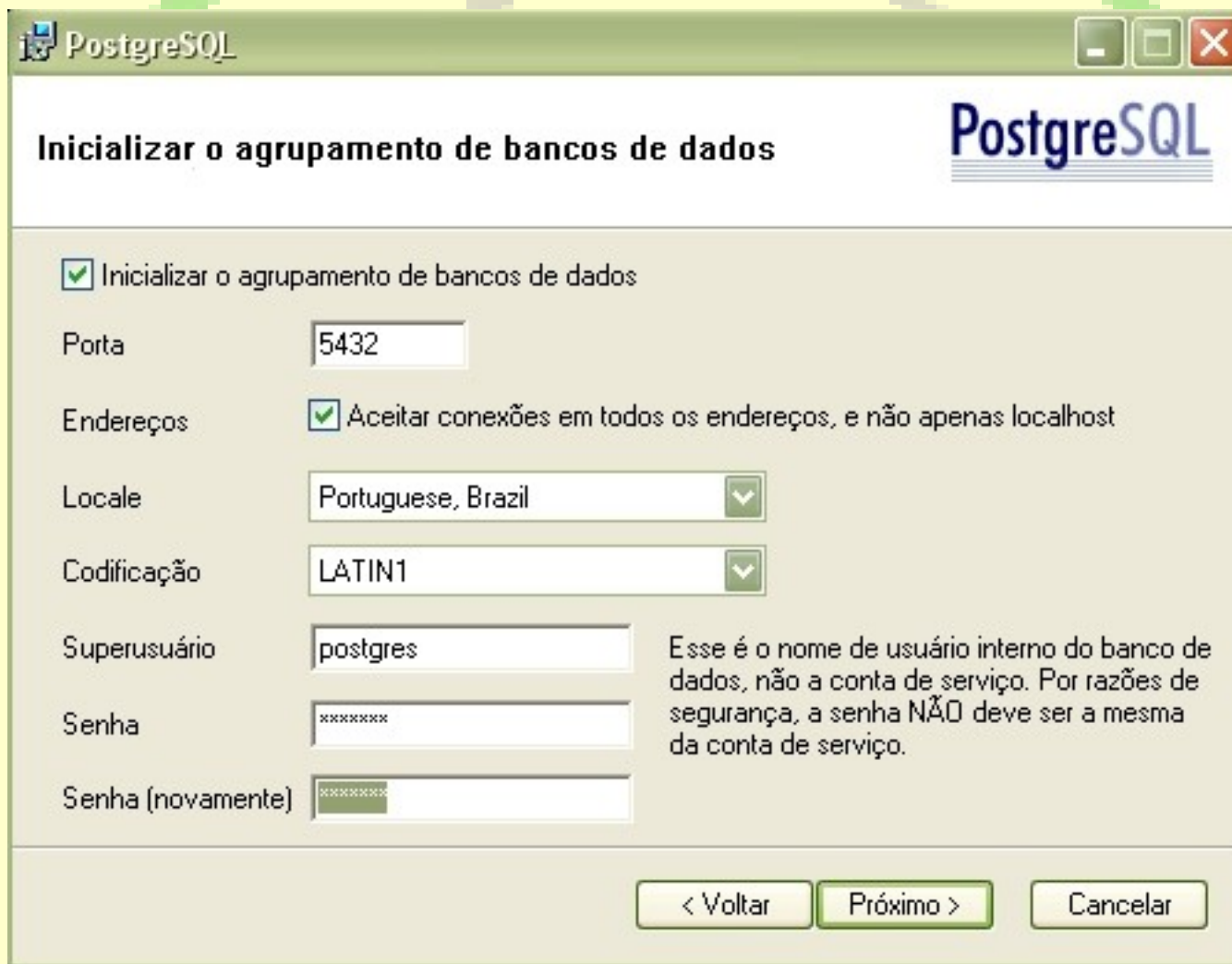
Digite novamente...: xxxxxxxx

A conta do serviço é a que executa o servidor de banco de dados PostgreSQL. Ela **NÃO** deve ser membro do grupo de administradores locais. Se você ainda não criou uma conta, o assistente pode fazer isso para você. Digite um nome e uma senha, ou deixe a senha em branco para que uma aleatória seja gerada automaticamente.

< Voltar   Próximo >   Cancelar

# Instalando no Windows

- `initdb -locale=pt_BR -D diretorio`



PostgreSQL

## Inicializar o agrupamento de bancos de dados

☒ Inicializar o agrupamento de bancos de dados

Porta: 5432

Endereços: ☒ Aceitar conexões em todos os endereços, e não apenas localhost

Locale: Portuguese, Brazil

Codificação: LATIN1

Superusuário: postgres

Senha: xxxxxx

Senha (novamente): xxxxxx

Esse é o nome de usuário interno do banco de dados, não a conta de serviço. Por razões de segurança, a senha NÃO deve ser a mesma da conta de serviço.

< Voltar Próximo > Cancelar

# Instalando no Windows

- createlang 'plpgsql' template1





# Instalando no Windows

- Instalando módulos do contrib



**Habilitar Módulos Contrib**

Módulos Contrib proporcionam funcionalidades adicionais, muitas vezes especializadas. Selecione aqueles que você deseja que sejam instalados. Todos os arquivos serão instalados, de modo que os módulos podem ser adicionados executando-se a instrução SQL apropriada.

<input type="checkbox"/> B-Tree GiST	<input type="checkbox"/> ISBN e ISSN	<input type="checkbox"/> R-Tree GiST	<input type="checkbox"/> TSearch2
<input type="checkbox"/> Chkpass	<input type="checkbox"/> Large Objects (lo)	<input type="checkbox"/> SEG	<input type="checkbox"/> User Lock
<input type="checkbox"/> Cube	<input type="checkbox"/> L-Tree	<input type="checkbox"/> AutoInc	
<input type="checkbox"/> DBlink	<input type="checkbox"/> Misc. Utilities	<input type="checkbox"/> Insert Username	
<input checked="" type="checkbox"/> DBsize	<input type="checkbox"/> No Update	<input type="checkbox"/> ModDateTime	
<input type="checkbox"/> Earth Distance	<input type="checkbox"/> Trigram Matching	<input type="checkbox"/> RefInt	
<input type="checkbox"/> Fuzzy String Match	<input checked="" type="checkbox"/> Suporte pgAdmin	<input type="checkbox"/> Time Travel	Módulos obsoletos:
<input type="checkbox"/> Integer Aggregator	<input type="checkbox"/> Funções Cripto	<input type="checkbox"/> String IO	<input type="checkbox"/> Full Text Index
<input type="checkbox"/> Integer Array	<input type="checkbox"/> PGStatTuple	<input type="checkbox"/> Table Functions	<input type="checkbox"/> TSearch

< Voltar   Próximo >   Cancelar



# Configurações básicas

- Arquivos dentro do diretório de dados
- pg\_hba
  - configuração de como os hosts devem se autenticar
- postgresql.conf
  - demais configurações do banco

# Configurações básicas

- pg\_hba
  - Regras de como autenticar, lidas sequencialmente do início ao fim do arquivo.
  - Tipos de autenticação:
    - trust, reject, password, md5, crypt, krb4, krb5, ident, pam
  - Ex.:

Tipo	BD	Usuario	IP/Rede	Método
local	all	all		ident sameuser
host	all	all	127.0.0.1/32	md5

# Configurações básicas

- postgresql.conf
  - listen\_addresses = '\*'
    - O banco escuta em todas interfaces de rede
  - max\_connections = 100
    - Máximo de conexões concorrentes

# Acessando o banco

- psql template1 postgres
  - comandos do psql:
    - \l – lista bancos de dados
    - \c bd – conecta no banco bd
    - \dt – lista relações
    - \i – executa arquivo contendo comandos sql
    - \? – mostra help do psql
    - \h – mostra referência do SQL
    - \h comando – mostra referência do comando

# SQL Básico

- Nomes
  - Devem ser auto-explicativos
  - " " Para nomes de objetos
  - ' ' Para literais

# SQL Básico

- DDL

CREATE DATABASE nome

[ [ WITH ] [ OWNER [=] dono ]

[ TEMPLATE [=] modelo ]

[ ENCODING [=] codificação ]

[ TABLESPACE [=] tablespace ] ]

DROP DATABASE nome

# SQL Básico

- DDL

```
CREATE [ TEMPORARY ] TABLE nome_tabela (  
    { nomecoluna tipo [ DEFAULT expressão ] }  
    [, ... ]  
)  
[ WITH OIDS | WITHOUT OIDS ]  
[ TABLESPACE tablespace ]
```

```
DROP TABLE nome [ CASCADE | RESTRICT ]
```

# SQL Básico

- DDL

GRANT permissao ON objeto TO usuario  
[ WITH GRANT OPTION ]

REVOKE permissao ON objeto FROM usuario



# SQL Básico

- CREATE DATABASE minicursos;
- CREATE TABLE instrutores  
(  
    id\_instrutor serial PRIMARY KEY,  
    nome varchar(100) NOT NULL  
);

# SQL Básico

- CREATE TABLE cursos

(

id\_curso serial PRIMARY KEY,  
id\_instrutor int REFERENCES instrutores,  
nome varchar(100) NOT NULL  
);

# SQL Básico

- DML

```
SELECT [ ALL | DISTINCT ]  
      * | expressão [ AS nome ] [, ...]  
[ FROM relacao [, ...] ]  
[ WHERE condição ]  
[ GROUP BY expressão [, ...] ]  
[ HAVING condição [, ...] ]  
[ ORDER BY expressão [ ASC | DESC |  
[ LIMIT { quantidade | ALL } ]  
[ OFFSET inicio ]
```

# SQL Básico

- DML

INSERT INTO tabela [ ( coluna [, ...] ) ]

{

DEFAULT VALUES

| VALUES ( { expressão | DEFAULT } [, ...] )

| consulta

}

# SQL Básico

- INSERT INTO instrutores VALUES (DEFAULT, 'Diogo Biazus');
- INSERT INTO cursos VALUES (DEFAULT, 1, 'Mini-curso de PostgreSQL');

# SQL Básico

- `SELECT * FROM cursos;`
- `SELECT * FROM instrutores;`
- `SELECT * FROM cursos JOIN instrutores  
USING (id_instrutor);`

# SQL Básico

- DML

UPDATE tabela SET coluna = { expressão |  
DEFAULT } [, ...]  
[ WHERE condição ]

# SQL Básico

- DML

DELETE FROM tabela [ WHERE condição ]



# SQL Básico

- DCL

BEGIN

COMMIT

SAVEPOINT nome

ROLLBACK TO [ SAVEPOINT ] nome

ROLLBACK

# SQL Básico

- BEGIN;
- UPDATE cursos SET nome = 'teste';
- SELECT \* FROM cursos;
- SAVEPOINT atualiza;
- DELETE FROM cursos;
- SELECT \* FROM cursos;
- ROLLBACK TO atualiza;
- SELECT \* FROM cursos;
- ROLLBACK;
- SELECT \* FROM cursos;

# Manutenção básica

- VACUUM
  - Remove versões antigas de registros
  - Deve ser agendado no mínimo diariamente se o autovacuum for desabilitado
  - ANALYZE para atualizar estatísticas

# Manutenção básica

- Backups
  - Podem ser feitos com o banco no ar
  - pg\_dump
    - Formatos:
      - Plain, Custom e Tar
  - pg\_restore
    - Pode-se usar listas para restauração

# Tablespaces

- Área de dados separada fisicamente
- `CREATE TABLESPACE nome [ OWNER dono ] LOCATION 'diretorio'`

# Funções em PL/pgSQL

- Porque PL/pgSQL?
- Ela é bonita? Não.
- Ela é OO? Não.
- Ela é popular? Não.
- Mesmo assim ela é ótima no que faz!

# Funções em PL/pgSQL

```
CREATE FUNCTION primeira_funcao() RETURNS  
    VOID AS
```

```
'
```

```
BEGIN
```

```
RETURN;
```

```
END;
```

```
' LANGUAGE 'plpgsql';
```

```
DROP FUNCTION primeira_funcao();
```

# Funções em PL/pgSQL

- Estrutura básica

```
[DECLARE
```

```
nome_da_variável tipo;
```

```
...
```

```
]
```

```
BEGIN
```

```
comandos;
```

```
END;
```



# Funções em PL/pgSQL

```
CREATE OR REPLACE FUNCTION primeira_funcao()  
  RETURNS VOID AS  
  
$body$  
BEGIN  
  RAISE NOTICE 'Minha primeira rotina em PL/pgSQL';  
  RETURN;  
END;  
  
$body$  
LANGUAGE 'plpgsql';
```

# Funções em PL/pgSQL

- Declarando variáveis:

DECLARE

numero int4 NOT NULL DEFAULT 10;

- Atribuindo valores:

numero := 15;

# Funções em PL/pgSQL

- Estruturas de controle:
  - IF ... THEN ... ELIF ... THEN ... ELSE ... END IF;
  - FOR ... LOOP END LOOP;
  - LOOP ... END LOOP;
  - WHILE ... LOOP ... END LOOP;

# Funções em PL/pgSQL

- Executando SQL:

```
CREATE OR REPLACE FUNCTION exclui_cliente(pid_cliente int4)  
    RETURNS int4 AS
```

```
$body$
```

```
DECLARE
```

```
    vLinhas int4 DEFAULT 0;
```

```
BEGIN
```

```
DELETE FROM clientes WHERE id_cliente = pid_cliente;
```

```
GET DIAGNOSTICS vLinhas = ROW_COUNT;
```

```
RETURN vLinhas;
```

```
END;
```

```
$body$ LANGUAGE 'plpgsql';
```

# Domínios

- Podemos fazer validações mais complexas
  - CPF, CNPJ, email, dependência de arquivos, etc...
- E criar domínios para validar de forma consistente

# Domínios

```
CREATE OR REPLACE FUNCTION valida(p1 varchar)
RETURNS boolean AS $$
BEGIN
    RETURN (lower(p1) IN ('a', 'b', 'ab', 'o'));
END;
$$ LANGUAGE plpgsql;
```

# Domínios

```
CREATE DOMAIN tipo_sanguineo AS varchar  
CHECK(valida(VALUE));
```

```
CREATE TABLE pessoa  
(  
    nome varchar PRIMARY KEY,  
    sangue tipo_sanguineo NOT NULL  
);
```

# Domínios

```
INSERT INTO pessoa (nome, sangue)  
VALUES ('Diogo', 'm');
```

```
INSERT INTO pessoa (nome, sangue)  
VALUES ('Diogo', 'm');
```



# Gatilhos em PL/pgSQL

- Tipo de retorno TRIGGER
- Nível de linha e nível de comando
- Variáveis para nível de linha:
  - NEW
  - OLD
  - TG\_OP
  - TG\_WHEN

# Gatilhos em PL/pgSQL

```
CREATE TRIGGER nome { BEFORE |  
    AFTER } { evento [ OR ... ] }  
    ON tabela [ FOR [ EACH ] { ROW |  
    STATEMENT } ]  
    EXECUTE PROCEDURE  
    funcao( argumentos )
```

# Gatilhos em PL/pgSQL

```
CREATE TABLE usuarios
```

```
(
```

```
nome varchar(30) PRIMARY KEY,
```

```
senha char(32)
```

```
);
```

# Gatilhos em PL/pgSQL

```
CREATE FUNCTION md5_senha() RETURNS  
    TRIGGER AS $body$  
  
BEGIN  
    NEW.senha := md5(NEW.senha);  
    RETURN NEW;  
END;  
$body$  
LANGUAGE 'plpgsql';
```

# Gatilhos em PL/pgSQL

```
CREATE TRIGGER md5_senha BEFORE INSERT ON  
    usuarios FOR EACH ROW EXECUTE PROCEDURE  
    md5_senha();
```

```
INSERT INTO usuarios VALUES ('usuario', 'senha');
```

```
SELECT * FROM usuarios;
```

# Dicas de segurança

- O objetivo é minimizar os danos
- Sempre “engaiole” o seu usuário
- Conheça bem os comandos GRANT e REVOKE
- Use funções “SECURITY DEFINER” como interface para operações críticas
- Crie usuários comuns para criação e administração de bancos
- Crie usuários comuns para operação de bancos
- Usuários diferentes para bancos diferentes

# Extensões

- Funções em C
- DBLink: Consultas remotas
- DBI-Link: Consultas em fontes DBI
- Tsearch: Indexação de textos no PostgreSQL
- PgXML: Manipulação com Xquery
- PostGis: Dados geográficos

# Replicação

- Slony I
- PGPool II
- Warm-standby (hot-standby na 8.5)
- Skytools



# Referências

- Sobre o PgSQL
  - <http://www.postgresql.org.br/>
  - <http://www.postgresql.org/>