PostgreSQL

Diogo Cezar Teixeira Batista

diogocezar@utfpr.edu.br

Universidade Tecnológica Federal do Paraná
Campus Cornélio Procópio
UTFPR-CP

Cornélio Procópio - 2008

Agenda I

1. Postgres
   - ACID
   - Alguns tipos de dados
   - Funcionalidades Avançadas
   - Linguagem de Definição de Dados(DDL)
     - Create Table
     - Alter Table
     - Drop Table
     - Comment
     - Create Index
   - Linguagem de Modificação de Dados (DML)
     - Insert
     - Update
     - Delete
     - Truncate

Uma breve história do PostGreSQL

- Banco de dados objeto-relacional;
- Derivado do pacote POSTGRES (Universidade da Califórnia em Berkeley);
  - 1986 início do projeto;
  - 1987 primeira versão do Postgres;
  - 1989 liberação para usuários restritos da versão 1;
  - 1991 versão 3 com as principais funcionalidades atuais;
  - 1993 versão 4.2, última lançada pela Berkeley;
  - 1994 Andrew Yu e Jolly Chen criaram a versão conhecida como Postgre95 com interpretador para a linguagem SQL;
  - 1997 Nome do projeto muda para PostgreSQL, a versão 6 é lançada;
  - 2000 versão 7 lançada com suporte a Foreign Key;
  - 2005 versão 8 lançada com versão nativa (sem uso do CYGWIN) para Windows, TABLESPACES, SAVEPOINTS, POINTINTIMERECOVERY. etc.

ACID

- *Atomicidade*: transações não podem ficar pela metade (tudo ou nada);
- *Consistência*: transações devem transformar um estado consistente do banco em outro estado consistente;
- *Isolamento*: transações são isoladas umas das outras, elas não "enxergam" dados gravados por transações concorrentes;
- *Durabilidade*: uma vez efetuada a transação os dados devem permanecer no banco.

Alguns tipos de dados

Tipo	Nome	Aliases	Descrição
Binário	bytea		dados binários ("matriz de bytes")
Boleano	boolean	bool	booleano lógico (verdade/falso)
Caracter	text		cadeia de caracteres de comprimento variável
Caracter	character [(n)]	char [(n)]	cadeia de caracteres de comprimento fixo
Caracter	character varying [(n)]	varchar [(n)]	cadeia de caracteres de comprimento variável [c]
Data e hora	timestamp [(p)] with time zone	timestamptz	data e hora, incluindo a zona horária
Data e hora	timestamp [(p)] [without time zone]		data e hora
Data e hora	time [(p)] with time zone	timetz	hora do dia, incluindo a zona horária
Data e hora	time [(p)] [without time zone]		hora do dia
Data e hora	date		data de calendário (ano, mês,dia)

Figura: Alguns tipos de dados

Alguns tipos de dados II

Monetário	money		quantia monetária (não recomendado)
Numérico	serial	serial4	inteiro de quatro bytes com auto-incremento
Numérico	smallint	int2	inteiro de dois bytes com sinal
Numérico	real	float4	número de ponto flutuante de precisão simples
Numérico	numeric [(p, s)]	decimal [(p, s)]	numérico exato com precisão selecionável
Numérico	integer	int, int4	inteiro de quatro bytes com sinal
Numérico	double precision	float8	número de ponto flutuante de precisão dupla [d]
Numérico	bigserial	serial8	inteiro de oito bytes com auto-incremento
Numérico	bigint	int8	inteiro de oito bytes com sinal [a]

Figura: Alguns tipos de dados

No PostgreSQL você pode:

- Criar validações de dados;
- Criar tipos de dados personalizados;
- Disparar *stored procedures* a partir de eventos das tabelas;
- Dividir o banco de dados em esquemas;
- Alocar bases inteiras, esquemas ou tabelas em locais diferentes (disco ou rede);

Componentes do PostgreSQL:

- Esquemas;
- Herança;
- Views;
- Funções;
- Restrições;
- Gatilhos;

Create Table I

Utilizado para criar tabelas.

Código 1: Sintaxe da linguagem DDL para criação de tabelas

```
1  CREATE TABLE nome_da_tabela (
2          { <nome_da_coluna> <tipo_de_dado> [ DEFAULT
               expressão_padrão ] [[ restrição_de_coluna ] | [
               restrição_de_tabela]]}
3  )[WITH OIDS | WITHOUT OIDS ]
4  onde restrição_de_coluna é:
5  [ CONSTRAINT nome_da_restrição ]
6  { NOT NULL | NULL | UNIQUE [ USING INDEX TABLESPACE
       espaço_de_tabelas ] | PRIMARY KEY [ USING INDEX
       TABLESPACE espaço_de_tabelas ] |
7  CHECK (expressão) | REFERENCES tabela_referenciada [ (
       coluna_referenciada ) ] }
8  e restrição_de_tabela é:
9  [ CONSTRAINT nome_da_restrição ]
```

Create Table II

```
10  { UNIQUE ( nome_da_coluna [, ... ] ) [ USING INDEX
        TABLESPACE espaço_de_tabelas ] | PRIMARY KEY (
        nome_da_coluna [, ... ] ) [ USING INDEX TABLESPACE
11  espaço_de_tabelas ] | CHECK ( expressão ) | FOREIGN KEY (
        nome_da_coluna [, ... ] )
12  REFERENCES tabela_referenciada [ ( coluna_referenciada [,
        ... ] ) ] }
```

Usando corretamente o Create Table

Código 2: Boa prática usando Create Table

```
1  CREATE TABLE fatura (
2          id_fatura integer,
3          total numeric(9,2) NOT NULL,
4          desconto numeric(6,2),
5          id_cliente integer NOT NULL,
6          id_vendedor integer NOT NULL,
7          data timestamp(0) DEFAULT current_timestamp(0),
8          CONSTRAINT fatura_pk PRIMARY KEY (id_fatura),
9          CONSTRAINT fatura_cliente_fk FOREIGN KEY id_cliente
               REFERENCES cliente,
10         CONSTRAINT fatura_vendedor_fk FOREIGN KEY
               id_vendedor REFERENCES vendedor
11 ) WHITHOUT OIDS;
```

Usando o Create Table sem convenção

Código 3: Má prática usando Create Table

```
1  create table Faturas
2  (IdFatura integer primary key,
3  Total money not null,
4  Desconto numeric(6,2),
5  IdCliente integer not null references Cliente,
6  IdVendedor integer not null references Vendedor,
7  Data timestamp default now());
```

Alter Table I

Utilizado para alterar tabelas.

Código 4: Sintaxe da linguagem DDL para alteração de tabelas

```
1  ALTER TABLE nome ação [, ... ]
2  ALTER TABLE nome RENAME [ COLUMN ] coluna TO
       novo_nome_da_coluna
3  ALTER TABLE nome RENAME TO novo_nome
4  onde ação é uma entre:
5           ADD [ COLUMN ] coluna tipo [ restrição_de_coluna [
                ... ] ]
6           DROP [ COLUMN ] coluna [ CASCADE ]
7           ALTER [ COLUMN ] coluna TYPE tipo [ USING expressão
                ]
8           ALTER [ COLUMN ] coluna SET DEFAULT expressão
9           ALTER [ COLUMN ] coluna DROP DEFAULT
10          ALTER [ COLUMN ] coluna { SET | DROP } NOT NULL
11          ADD restrição_de_tabela
```

Alter Table II

```
12          DROP CONSTRAINT nome_da_restrição [ RESTRICT |
                CASCADE ]
13          OWNER TO novo_dono
14          SET TABLESPACE nome_do_espaço_de_tabelas
```

Drop Table

Remove uma ou mais tabelas.

Código 5: Sintaxe da linguagem DDL para remover tabelas

```
1  DROP TABLE nome [, ...] [ CASCADE ]
```

Comment

Adiciona comentários em uma tabela, coluna, indice, restrição, função, indice, etc.

Código 6: Sintaxe da linguagem DDL para fazer comentários

```
1 COMMENT ON tipo_de_objeto nome_do_objeto IS 'text'
```

Create Index

Cria um índice numa tabela.

Código 7: Sintaxe da linguagem DDL para criar índices

```
1  CREATE [ UNIQUE ] INDEX nome_do_índice ON tabela [ USING
       método ]
2          ( { coluna | ( expressão ) } [ classe_de_operadores
               ] [, ...] )
3          [ TABLESPACE espaço_de_tabelas ]
4          [ WHERE predicado ]
```

Insert

Insere um registro em uma tabela.

Código 8: Sintaxe da linguagem DML para inserir registro

```
1  INSERT INTO tabela [ ( coluna [, ...] ) ]
2          { VALUES ( { expressão | DEFAULT } [, ...] ) |
               consulta }
```

Update

Atualiza um registro em uma tabela.

Código 9: Sintaxe da linguagem DML para atualizar registro

```
1  UPDATE tabela SET coluna = { expressão | DEFAULT } [, ...]
2          [ FROM lista_do_from ]
3          [ WHERE condição ]
```

Delete

Remove um registro em uma tabela.

Código 10: Sintaxe da linguagem DML para remover registro

```
1  DELETE FROM tabela [ WHERE condição ]
```

Truncate

Zera os registros de uma tabela.

Código 11: Sintaxe da linguagem DML para zerar os registros de uma tabela

```
1 TRUNCATE [ TABLE ] nome
```