Arquitetura e Desempenho de Bancos de Dados

Funções em SQL e Funções em PL/pgSQL

Profa. Socorro Vânia Alves

Funções no PostgreSQL

O PostgreSQL possui quatro tipos de função:

- funções internas.
- funções escritas na linguagem de comando SQL.
- funções escritas nas linguagens procedurais (funções escritas em, por exemplo, em PL/pgSQL).
- funções na linguagem C

Funções no PostgreSQL

- Todos os tipos de funções aceitam tipos base e tipos compostos como argumentos ou parâmetros.
- Além disso, todos os tipos de função podem retornar um tipo base ou um tipo composto.

Funções Definidas pelo Usuário

- **Funções na linguagem de comando SQL**
  - As funções SQL são as mais fáceis de serem definidas e, portanto a maior parte dos conceitos apresentados para as funções SQL podem ser levados para os outros tipos de função.

Funções Definidas pelo Usuário

Funções na linguagem de comando SQL

- As funções SQL executam uma lista arbitrária de declarações SQL, retornando o resultado da última consulta da lista.
- No caso mais simples a primeira linha do resultado da última consulta é retornada. Caso a última consulta não retorne nenhuma linha, é retornado o valor nulo.

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

- O corpo de uma função SQL deve ser uma lista contendo uma ou mais declarações SQL separadas por ponto-e-vírgula (;).
- O ponto-e-vírgula após a última declaração é opcional.
- A menos que a função seja declarada como retornando o tipo void, a última declaração deve ser um comando SELECT.

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

- Além de comandos SELECT, podem existir comandos de manipulação de dados (INSERT, UPDATE e DELETE), assim como outros comandos SQL.
- A única exceção é que não se pode colocar os comandos BEGIN, COMMIT, ROLLBACK ou SAVEPOINT na função SQL.

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

- Por exemplo, a função a seguir remove as linhas contendo salários negativos da tabela funcionarios:

```
CREATE OR REPLACE FUNCTION limpar_funcionarios ()
RETURNS void AS $$
    DELETE FROM funcionarios
    WHERE salario <= 0;
$$ LANGUAGE SQL;
```

```
SELECT limpar_funcionarios ();
```

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

- Quando a função SQL recebe argumentos, estes são referenciados no corpo da função utilizando a sintaxe $n.
- Assim, $1 se refere ao primeiro argumento, $2 ao segundo, e assim por diante.
- No exemplo abaixo deve ser observado que, dentro da função, os argumentos são referenciados como $1 e $2.

```
CREATE OR REPLACE FUNCTION somar (integer, integer)
RETURNS integer AS $$
        SELECT $1 + $2;
$$ LANGUAGE SQL;
```

```
SELECT somar(1,2) AS resposta;
```

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

No exemplo abaixo deve ser observado que, dentro da função, os argumentos são referenciados conforme o nome dado.

```
CREATE OR REPLACE FUNCTION somar ( a integer, b integer)
RETURNS integer AS $$
    SELECT a + b;
$$ LANGUAGE SQL;
```

```
SELECT somar(2,2) AS resposta;
```

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

http://pgdocptbr.sourceforge.net/pg80/xfunc-sql.html

Abaixo está mostrada uma função mais útil, que pode ser utilizada para realizar débitos em uma conta corrente no banco:

```
CREATE FUNCTION debitar (integer, numeric) RETURNS integer AS $$
    UPDATE conta_corrente
        SET saldo = saldo - $2
        WHERE numero_da_conta = $1;
    SELECT 1;
$$ LANGUAGE SQL;
```

O usuário pode executar esta função para debitar R$100.00 da conta 17 da seguinte maneira:

```
SELECT debitar(17, 100.0);
```

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

Provavelmente, na prática seria desejado que a função retornasse um resultado mais útil do que a constante "1" e, portanto, uma definição mais realística seria:

```
CREATE FUNCTION debitar (integer, numeric) RETURNS numeric AS $$
    UPDATE conta_corrente
        SET saldo = saldo - $2
        WHERE numero_da_conta = $1;
    SELECT saldo FROM conta_corrente WHERE numero_da_conta = $1;
$$ LANGUAGE SQL;
```

que atualiza o saldo e retorna o novo saldo.

Funções Definidas pelo Usuário

Funções na linguagem de comando (SQL)

Outro exemplo:

```
CREATE OR REPLACE FUNCTION listaFuncionario (integer)
RETURNS text AS $$
    SELECT nome||' : '||salario
    FROM funcionarios WHERE codigo=$1;
$$ LANGUAGE SQL;
```

```
SELECT listaFuncionario(1);
```

Funções Definidas pelo Usuário

Funções em Linguagem Procedural

- No PostgreSQL é possível definir funções pelo usuário além de SQL. Essas linguagens são genericamente chamadas de **linguagens procedurais (PL: Procedural Language**).
- São exemplos de PLs: PL/pgSQL, PL/Perl, PL/Python e outras linguagens definidas pelo usuário.

Funções Definidas pelo Usuário

Funções em Linguagem Procedural

- PL/pgSQL é uma linguagem procedural carregável feita para o PostgreSQL, a qual é similar à PL/SQL do Oracle.
- Enquanto declarações SQL são executadas individualmente pelo servidor, a linguagem PL/pgSQL agrupa um bloco de processamento e comandos em série dentro do servidor, somando o poder da linguagem procedural com a facilidade do SQL.

Funções Definidas pelo Usuário

Funções em Linguagem Procedural

- Em PL/pgSQL todos os tipos de dados, operadores e funções SQL podem ser utilizados.
- A linguagem PL/pgSQL é estruturada em blocos. O texto completo da definição da função deve ser um bloco.

Funções Definidas pelo Usuário

Funções em Linguagem Procedural

- Um bloco é definido como:

```
[ <<rótulo>> ]
        [ DECLARE
                declarações ]
        BEGIN
                instruções;
        END;
```

- ✓ Declarações e instruções dentro do bloco devem ser terminadas com ponto-e-vírgula. Um bloco dentro de outro bloco deve ter um ponto-e-vírgula após o END. O END final que conclui o corpo da função não requer o ponto-e-vírgula.
- ✓ Palavras chaves e identificadores podem ser escritos mesclando letras maiúsculas e minúsculas. As letras dos identificadores são convertidas implicitamente em minúsculas, a menos que estejam entre aspas.

Funções Definidas pelo Usuário

Funções em Linguagem Procedural

Exemplo:

```
CREATE OR REPLACE FUNCTION calcula_imposto(preco real, porcent_imp real)
RETURNS real AS $$
BEGIN
   RETURN preco*(porcent_imp/100);
END;
$$ LANGUAGE plpgsql;
```

Funções Definidas pelo Usuário

Funções em Linguagem Procedural

Exemplo:

```
CREATE OR REPLACE FUNCTION somar_tres_valores(v1 integer, v2 integer, v3 integer)
RETURNS integer AS $$
DECLARE
   resultado integer;
BEGIN
   resultado := v1 + v2 + v3;
   RETURN resultado;
END;
$$ LANGUAGE plpgsql;
```