Instituto Federal Sul-rio-grandense – Campus Camaquã
Disciplina: Banco de Dados II – Turma 2024
Professor: Vinícius Alves Hax
Assunto: Lista de exercícios 5 - Revisão da etapa 2

Para os exercícios 6 à 10 utilize as tabelas e registros abaixo:

```
CREATE TABLE teacher (
        id INT,
        name VARCHAR(200),
        side VARCHAR(200),
        level REAL
);

CREATE TABLE stats (
        id SERIAL,
        average_level REAL,
        teachers_count INT
);
```

1) Qual é a diferença entre um trigger BEFORE e AFTER, e como a escolha entre eles pode afetar a lógica de uma aplicação que utiliza o banco de dados?

Um trigger do tipo BEFORE vai ser executado ANTES que o evento tenha efeitos na base de dados. Pode ser usado para conferir uma determinada condição necessária para o comando ser executado, por exemplo, antes de deletar um registro podemos conferir se esse registro é chave estrangeira de algum outro registro.
Os triggers AFTER são executados após as mudanças do comando são efetivadas. Podemos usar isso, por exemplo, para atualizar estatísticas sobre sobre os registros ou fazer um log dos comandos.

2) Qual das alternativas descreve corretamente a função de um trigger no PostgreSQL?

a) Um trigger é uma estrutura usada apenas para otimizar consultas SQL.
b) Um trigger é uma função executada automaticamente em resposta a certos eventos em uma tabela, como uma inserção, atualização ou exclusão de dados.
c) Um trigger serve exclusivamente para testar a presença de chaves primárias em uma tabela.
d) Um trigger é um mecanismo de replicação entre bancos de dados PostgreSQL.

3) Quais dos seguintes eventos podem acionar um trigger no PostgreSQL?

a) Apenas eventos de atualização (UPDATE) e exclusão (DELETE).
b) Somente inserção (INSERT) e exclusão (DELETE).
c) Inserção (INSERT), atualização (UPDATE), exclusão (DELETE) e também eventos como TRUNCATE.
d) Apenas exclusão (DELETE) e seleção (SELECT).

4) Qual é a diferença entre um trigger FOR EACH ROW e FOR EACH STATEMENT?

Quando configuramos um trigger do tipo FOR EACH ROW a função relacionada é chamada para cada linha da tabela. Por exemplo, supondo que tenhamos em uma tabela

T três registros com o valor “idade” > 50.
Se houver um trigger atrelando uma função F ao evento UPDATE, quando o comando abaixo for executado:
UPDATE T SET experiencia = ‘alta’ WHERE idade > 50
Será chamada três vezes a função F, uma delas para cada linha.
Se usarmos um trigger usando FOR EACH STATEMENT a função F só será chamada uma vez.

5) O que representam as variáveis OLD e NEW dentro de um trigger?

A variável OLD representa o registro da tabela antes da execução do comando e é usado principalmente nos eventos DELETE e UPDATE. De maneira equivalente a variável NEW representa um registro que ainda não existe na base de dados no caso do INSERT ou então o novo valor do registro no caso do UPDATE.

6) Crie um trigger e uma função correspondente que sempre que o nível (coluna level) de um professor (tabela teacher) mudar, a coluna average_level (tabela stats) da será atualizada com a média dos níveis de todos os professores.

-- https://github.com/viniciusalveshax/aulas-2024/blob/master/banco-de-dados-2/lista-de-exercicios5/questao6.sql

```
CREATE OR REPLACE FUNCTION calc_average()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
AS $$
DECLARE
  average REAL;
BEGIN

  SELECT AVG(level) INTO average FROM teacher;

  raise notice 'Média %', average;

  UPDATE stats SET average_level = average;

END;
$$;

CREATE TRIGGER trigger_after_update2
  AFTER UPDATE
  ON teacher
  FOR EACH ROW
      EXECUTE PROCEDURE calc_average();
```

7) Crie dois triggers que funcionem da seguinte maneira: Sempre que um professor (teacher) for adicionado um trigger deve ser disparado e o valor da coluna teachers_count deve ser incrementado. De maneira inversa, crie um trigger que diminua o valor da mesma coluna sempre que um professor for deletado.

```
-- https://github.com/viniciusalveshax/aulas-2024/blob/master/banco-de-dados-2/lista-de-exercicios5/questao7.sql

CREATE OR REPLACE FUNCTION increment_teacher_count()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
AS $$
DECLARE
  counter INTEGER;
BEGIN

  SELECT teachers_count INTO counter FROM stats;

  counter := counter + 1;

  UPDATE stats SET teachers_count = counter;

END;
$$;

CREATE OR REPLACE FUNCTION decrement_teacher_count()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
AS $$
DECLARE
  counter INTEGER;
BEGIN

  SELECT teachers_count INTO counter FROM stats;

  counter := counter - 1;

  UPDATE stats SET teachers_count = counter;

END;
$$;

CREATE TRIGGER trigger_after_insert2
  AFTER INSERT
  ON teacher
  FOR EACH ROW EXECUTE PROCEDURE increment_teacher_count();


CREATE TRIGGER trigger_after_delete2
  AFTER DELETE
  ON teacher
  FOR EACH ROW EXECUTE PROCEDURE decrement_teacher_count();
```

8) Faça um programa em PL/pgSQL que crie dados sintéticos (falsos) para preencher a tabela teachers com 10 registros usando o comando WHILE. Nesse caso o ID e o nome não são importantes e podem ter qualquer valor. O nível (level) deve começar em 15 e

subir de cinco em cinco (ou seja, 15, 20, 25, 30, etc). Se o nível terminar em cinco o side (lado) deverá ser “Light”, caso contrário deverá ser “Dark”.

-- https://github.com/viniciusalveshax/aulas-2024/blob/master/banco-de-dados-2/lista-de-exercicios5/questao8.sql

```
do $$
declare
numero integer;
contador integer;
begin
   numero := 15;
   contador := 0;
   while contador < 10 loop
       contador := contador + 1;
       raise notice 'Número vale %', numero;
          if (contador % 2) != 0 then
               raise notice '% termina em cinco', numero;
                    insert into teacher (name, level, side) values ('yyy', numero, 'Light side');
             else
               raise notice '% não termina em cinco', numero;
                    insert into teacher (name, level, side) values ('yyy', numero, 'Dark side');
             end if;
             numero := numero + 5;

   end loop;
end; $$
```

9) Faça um programa em PL/pgSQL que consulte a tabela teachers e utilize a estrutura CASE para exibir uma mensagem de acordo com o level. Se o level for menor que cinquenta deverá aparecer a mensagem “Fraco”. Se o nível for entre 50 e 70 deverá aparecer a mensagem “Médio”. Se for maior que 70 e menor que 90 “Forte” e se for maior ou igual do que 90 deverá aparecer “Muito forte”.

-- https://github.com/viniciusalveshax/aulas-2024/blob/master/banco-de-dados-2/lista-de-exercicios5/questao9.sql

```
DO $$
DECLARE

NAME TEXT;
LEVEL REAL;

BEGIN

-- Usamos um loop FOR para iterar sobre cada linha da tabela 'teacher'
FOR NAME, LEVEL IN SELECT teacher.name, teacher.level FROM teacher
LOOP

        -- Exibimos os valores das colunas da linha atual
        RAISE NOTICE 'Professor(a) % tem nível %', NAME, LEVEL;
```

```
        CASE
                WHEN LEVEL < 50 THEN
                    RAISE NOTICE 'Professor %: Fraco', NAME;
                WHEN LEVEL >= 50 AND LEVEL < 70 THEN
                    RAISE NOTICE 'Professor %: Médio', NAME;
                WHEN LEVEL >= 70 AND LEVEL < 90 THEN
                    RAISE NOTICE 'Professor %: Forte', NAME;
                ELSE
                    RAISE NOTICE 'Professor %: Muito forte', NAME;
        END CASE;

END LOOP;

END $$;
```

10) Verifique se após executar os exercícios 6, 7 e 8 se os valores da tabela stats são os esperados.