INSTITUTO FEDERAL
Pernambuco

Triggers

...

Trigger

INSTITUTO FEDERAL
Pernambuco

- Tradução: **GATILHO**
- Procedimentos armazenados no SGBD que são automaticamente ativados em resposta a determinadas mudanças que ocorrem no BD
  - Em geral não são modificados e executam várias vezes
  - São executados implicitamente sempre que ocorre o evento que os dispara
  - Não aceitam parâmetros

Trigger

INSTITUTO FEDERAL
Pernambuco

- É muito utilizada para ajudar a manter a consistência dos dados ou para propagar alterações em um determinado dado de uma tabela para outras.
- Composta de 3 partes:
  - **Momento:** corresponde ao tempo em que a trigger irá executar. BEFORE, AFTER ou INSTEAD OF
  - **Evento:** corresponde à operação relacionada: INSERT, DELETE ou UPDATe
  - **Ação:** corresponde a determinar qual trigger-função será disparada.
- Acesso aos objetos:
  - **.OLD:** versão antiga (Acessível apenas no BEFORE)
  - **.NEW:** versão nova (dados atualizados que dispararam a trigger)

Triggers
- Sintaxe de definição -

INSTITUTO FEDERAL
Pernambuco

```
CREATE FUNCTION nome_funcao_trigger ()
RETURNS TRIGGER
LANGUAGE PLPGSQL AS $$
DECLARE
   -- variáveis
BEGIN
   -- lógica_da_trigger
   return NEW;
END; $$
```

```
CREATE TRIGGER trigger_name
momento evento ON table_name
FOR EACH ROW
EXECUTE PROCEDURE nome_funcao_trigger ( );
```

Trigger
- Exemplos -

INSTITUTO FEDERAL
Pernambuco

```
create table times (
  id serial primary key,
  nome varchar(30) not null,
  num_pontos integer,
  vitorias integer,
  derrotas integer,
  empates integer,
  gols_afavor integer,
  gols_sofridos integer
);

insert into times values
(1,'Sport',0,0,0,0,0,0),
(2,'Náutico',0,0,0,0,0,0),
(3,'Santa Cruz',0,0,0,0,0,0);
```

```
create table jogos (
  id_mandante integer,
  id_visitante integer,
  gols_mandante integer,
  gols_visitante integer,
  PRIMARY key
     (id_mandante, id_visitante),
  FOREIGN key (id_mandante)
REFERENCES times (id),
  FOREIGN key (id_visitante)
REFERENCES times (id)
);
```

Trigger
- Exemplos -

INSTITUTO FEDERAL
Pernambuco

- Quando houver uma inserção de um novo jogo e for um empate, atualizar o número de empates dos times que participaram do jogo.

Trigger
- Exemplos -

INSTITUTO FEDERAL
Pernambuco

- Quando houver uma inserção de um novo jogo e for um empate, atualizar o número de empates dos times que participaram do jogo.

```
CREATE or replace FUNCTION atualiza_empate()
RETURNS TRIGGER
LANGUAGE PLPGSQL AS $$
BEGIN
    if new.gols_mandante = new.gols_visitante then
            update times set empates = empates+1
                    where id = new.id_visitante or id=new.id_mandante;
    end if;
    return new;
END; $$

CREATE TRIGGER tr_empates
before INSERT ON jogos
FOR EACH ROW
EXECUTE PROCEDURE atualiza_empate();
```

Trigger
- Exercícios -

INSTITUTO FEDERAL
Pernambuco

1. Atualizar a função ***atualizar_time()*** para todas as atualizações necessárias listadas abaixo:
   - Se a partida houver um vencedor, atualizar a quantidade de derrotas e de vitórias;

```
CREATE or replace FUNCTION atualizar_empate()
RETURNS TRIGGER
LANGUAGE PLPGSQL AS $$
BEGIN
   if new.gols_mandante = new.gols_visitante THEN
                update times set empates = empates+1 where id=new.id_mandante or id=new.id_visitante;
   elseif new.gols_mandante > new.gols_visitante THEN
                update times set vitorias=vitorias+1 where id=new.id_mandante;
      update times set derrotas=derrotas+1 where id=new.id_visitante;
   ELSE
                update times set vitorias=vitorias+1 where id=new.id_visitante;
      update times set derrotas=derrotas+1 where id=new.id_mandante;
   end if;
   return new;
END; $$;

CREATE or replace TRIGGER tr_empate
after insert ON jogos
FOR EACH ROW
EXECUTE PROCEDURE atualizar_empate();
```

Trigger
- Exercícios -

INSTITUTO FEDERAL
Pernambuco

2. É necessário contabilizar as informações de gols sofridos e feitos, além de atualizar a pontuação dos times após um jogo ser realizado, seguindo as seguintes regras:
   - Atualizar as informações de gols pro e gols contra;
   - Atualizar a pontuação dos times de acordo com a situação abaixo:
     - Vencedor: +3 pontos
     - Empate: +1 ponto para cada time
     - Perdedor: não altera a pontuação

Continuação

INSTITUTO FEDERAL
Pernambuco

3. Crie uma visão para identificar a classificação do campeonato, que deve seguir as seguintes regras de ordenação: pontuação, número de vitórias e saldo de gols.
4. Crie um subprograma que insere um time recebendo apenas o nome do time como argumento e defina o número de ID que deve ser utilizado e inicializando com 0 os demais atributos.
5. Crie um subprograma que retorne a diferença de pontos entre o primeiro e último colocado.
6. Crie um subprograma que recebe o resultado de um jogo entre dois times e realiza a atualização desses dados.
7. Crie um subprograma que retorne os nomes dos times e a quantidade de pontos ordenados de acordo com o critério escolhido: 1 (qtde de pontos), 2 (qtde de vitórias), ou 3 (saldo de gols). **RETURN QUERY**

Continuação

INSTITUTO FEDERAL
Pernambuco

O sistema agora deve contabilizar em um atributo quantos jogos cada time fez, então solicito:

8. Criar um subprograma que cria a coluna num_jogos do tipo inteiro na tabela time e atualiza o valor dessa coluna com a soma dos valores nos seguintes 3 campos: empates, vitórias e derrotas.
9. Executar o subprograma acima, e logo depois deletá-lo.
10. O número de jogos de um time deve ser incrementado antes que um jogo que ele participou seja cadastrado no banco. Criar essa regra de negócio via *trigger*, lembrando de atualizar a quantidade de jogos dos dois participantes da partida!!

INSTITUTO FEDERAL
Pernambuco

Triggers

...