Triggers (gatilhos)

Uma trigger fica "ouvindo" eventos (inserção, update,...) nas tabelas desejadas, caso ocorra algum desses eventos uma série de instruções podem ser executadas.

Tem-se dois tipos de triggers

- **Trigger de nível de linha (FOR EACH ROW)**
  - Executa várias vezes, de acordo com número de registros afetados
- **Tigger de nível de instrução (FOR EACH STATEMENT)**
  - Executa uma única vez

```
CREATE TRIGGER nome_trigger { BEFORE | AFTER } { eventos [ OR ... ] }
ON nome_tabela
  FOR  EACH ROW
EXECUTE PROCEDURE nome_funcao ( arguments )
```

PostgreSQL

SINTAXE Função TIPO TRIGGER

```
CREATE OR REPLACE FUNCTION <nome> RETURNS
trigger AS $$
BEGIN
    //comandos
RETURN NULL;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TABLE Produto
(
    cod_prod    integer PRIMARY KEY,
    descricao   VARCHAR(50) UNIQUE,
    qtde_disponivel INT NOT NULL DEFAULT 0
);

INSERT INTO Produto VALUES (1, 'Feijão', 15);
INSERT INTO Produto VALUES (2, 'Arroz', 10);

CREATE TABLE ItensVenda
(
    cod_venda       integer,
    id_produto      integer,
    qtde_vendida    integer,
    FOREIGN KEY (id_produto) REFERENCES Produto(cod_prod) ON DELETE CASCADE
);


CREATE TRIGGER t_atualiza_estoque
BEFORE INSERT ON ItensVenda
FOR EACH ROW
EXECUTE PROCEDURE atualiza_estoque();

CREATE OF REPLACE FUNCTION atualiza_estoque() RETURNS TRIGGER
AS
$$
BEGIN

END
$$ LANGUAGE plpgsql;
```

TRIGGERS (GATILHOS) - EXEMPLO

```
/* Vamos criar uma função que insere na tabela
de auditoria o código do funcionário e a data
de alteração da tabela Funcionarios */
CREATE OR REPLACE FUNCTION funcionario_log_func()
RETURNS trigger AS $$
BEGIN
INSERT INTO funcionarios_auditoria
(codigo_func, data_alteracao)
VALUES
(NEW.codigo, current_date);
RETURN NEW; -- indica que nova linha será inserida
END;
$$ LANGUAGE plpgsql;
```

TRIGGERS (GATILHOS) - EXEMPLO

```
/* Finalmente, criamos nossa trigger que
informa que devemos executar a
funcionario_log_func() toda vez que houverem
inserções ou atualizações na tabela
Funcionários */
CREATE TRIGGER log_trigger
AFTER INSERT OR UPDATE ON funcionarios
FOR EACH ROW
EXECUTE PROCEDURE funcionario_log_func();
```

TRIGGERS (GATILHOS)

- Em pgplsql, para criarmos triggers, devemos criar uma função que retorna a trigger e em seguida criar a trigger em si

```
CREATE TRIGGER name
       { BEFORE | AFTER | INSTEAD OF }
       { event [ OR ... ] } ON table
       [ FROM referenced_table_name ]
      [ FOR [ EACH ] { ROW | STATEMENT } ]
      EXECUTE PROCEDURE function_name ( arguments )
```

evento pode ser um dentre: `INSERT UPDATE [ OF column_name [, ... ] ] DELETE TRUNCATE`

TRIGGERS (GATILHOS)

- Os eventos podem ser executados por linha alterada (for each row) ou por comando executado (for each statement)
- A diferença é transacional
  - FOR EACH ROW executa cada linha como uma transação
  - FOR EACH STATEMENT executa cada comando como uma transação

trigger