EXEMPLO: Caso haja alguma modificação nos dados da tabela de salários, uma Trigger será disparada.

PostgreSQL

Criando a função que vai atualizar a tabela

```
CREATE OR REPLACE FUNCTION
atualiza_salario() RETURNS trigger AS $$
BEGIN
 INSERT INTO atualizacao_salarial
    (matricula, data_alteracao, salario)
 VALUES
    (NEW.matricula, NOW(), NEW.salario);
RETURN NULL;
END;
$$ LANGUAGE plpgsql;
```

sal_antigo,

OLD.salario,

EXEMPLO: Caso haja alguma modificação nos dados da tabela de salários, uma Trigger será disparada.

PostgreSQL

Criando a trigger em funcionarios

```
CREATE TRIGGER tg_atualiza_salario
   AFTER INSERT OR UPDATE
   ON funcionarios
FOR EACH ROW
   EXECUTE PROCEDURE atualiza_salario();
```

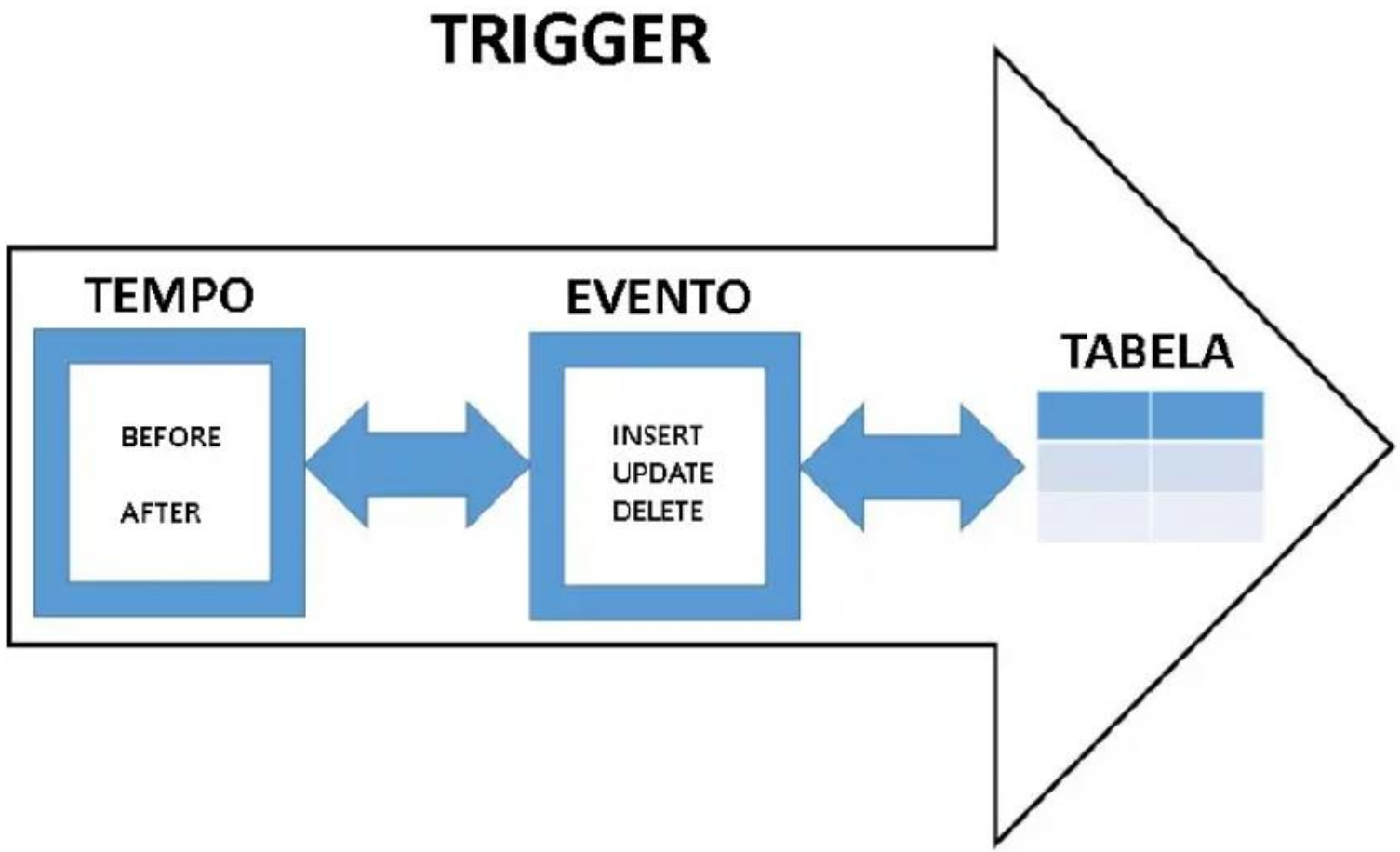
EXEMPLO: Caso haja alguma modificação nos dados da tabela de salários, uma Trigger será disparada.

As ações são:
- **Insert;**
- **Update;**
- **Delete;**

Para cada ação os objetos que ficam disponíveis são:
- **Insert**: NEW;
- **Update**: NEW e OLD;
- **Delete**: OLD;

Os objetos trabalham da seguinte forma:
- **NEW**: Todos os dados que foram inseridos ou atualizados (insert e update, respectivamente);
- **OLD**: Todos os dados que foram deletados ou sobrescritos (delete e update, respectivamente);

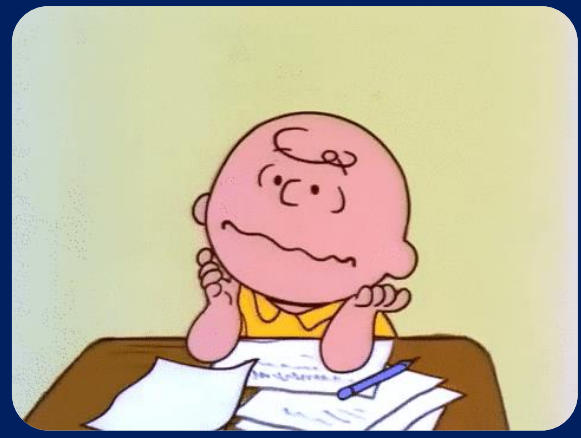
CICLO DE VIDA DE UM TRIGGER



EXERCÍCIO

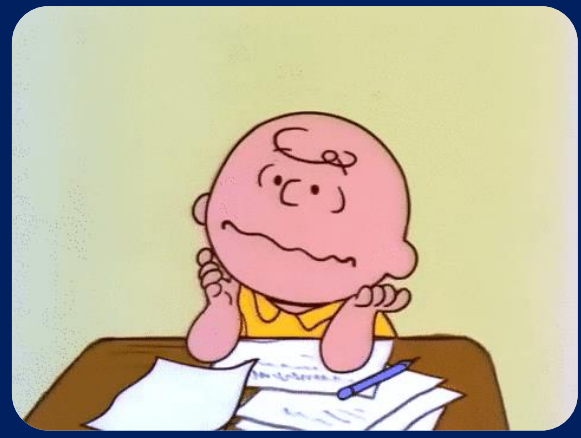
1) Criar uma trigger (trg_cad_clientes) para auditar todos os inserts realizados em uma tabela de cadastro de clientes.
Organizar a tabela de logs para armazenar todos os campos da tabela com a data do respectivo insert; e devem ser inseridos no mínimo 5 cadastros para testar o log de auditoria.

CREATE TABLE cadastro_cliente (cod_cliente serial primary key, nome_cliente varchar(30), sobrenome varchar(30), dt_cadastro date);



EXERCÍCIO - 2

2) Com a mesma tabela do exercício anterior (cadastro_cliente), criar um trigger para armazenar os valores antes de serem excluídos da tabela.
Devem ser inseridos 5 pessoas com diferentes nomes e o delete deve ser feito caso a caso com o respectivo nome.



EXERCÍCIO - 3

O departamento de Recursos Humanos deseja que os aumentos de salários maiores de R$ 1500,00 sejam armazenados em uma tabela de auditoria para posterior averiguação. Criar uma tabela de funcionário e uma trigger (trg_func) que audite somente esses aumentos com o novo valor do salário e o anterior.

```
create table salario (cd_func serial, nm_func varchar(30), salario float);
create table salario_aud(dt_audit date, cd_func int, nm_func
varchar(30), salario float);
```

```
INSERT INTO SALARIO VALUES(1,'MARCELO DINIZ', 1000);
INSERT INTO SALARIO VALUES(2,'MARIO QUINELLO',1000);
INSERT INTO SALARIO VALUES(3,'CARLOS ROBERTO', 3000);
INSERT INTO SALARIO VALUES(4,'JOSE SILVA', 5000);
INSERT INTO SALARIO VALUES(5,'MARINA BEZERRA', 10000);
```

EXERCÍCIO - 4

Criar um trigger para auditar a troca de departamento dos funcionários de uma determinada empresa, constando o novo e antigo departamento. Para criação dessa trigger, somente será dada a tabela de departamento e o novo e antigo departamento devem ser auditados. As estruturas como tabela de log e outras que julgarem necessárias, deverão ser criadas de acordo com a necessidade.

```
CREATE TABLE DEPT_EMP (CD_FUNCIONARIO SERIAL, CD_FILIAL INT,
NM_FUNCIONÁRIO VARCHAR(30), NM_DEPARTAMENTO VARCHAR(30));
```

```
INSERT INTO DEPT_EMP VALUES (1, 500, 'MARCELO DINIZ', 'TI');
INSERT INTO DEPT_EMP VALUES (2, 500, 'CARLOS ROBERTO', 'FINANÇAS');
INSERT INTO DEPT_EMP VALUES (3, 500, 'ROBERTO ALMEIDA', 'RH');
INSERT INTO DEPT_EMP VALUES (4, 500, 'MARISA INACIO', 'VENDAS');
```