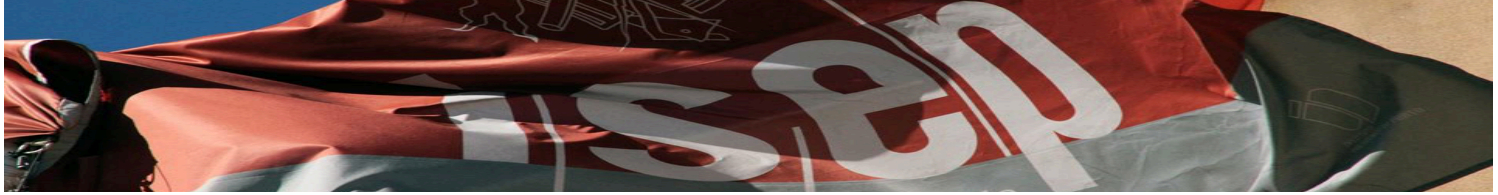
BASE DE DADOS

Oracle PL/SQL – Triggers

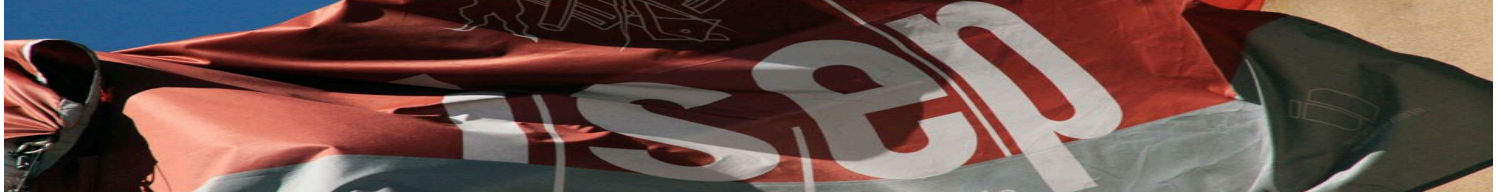
Teórico-Práticas
Ano Lectivo 2016/2017
Rosa Reis

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA



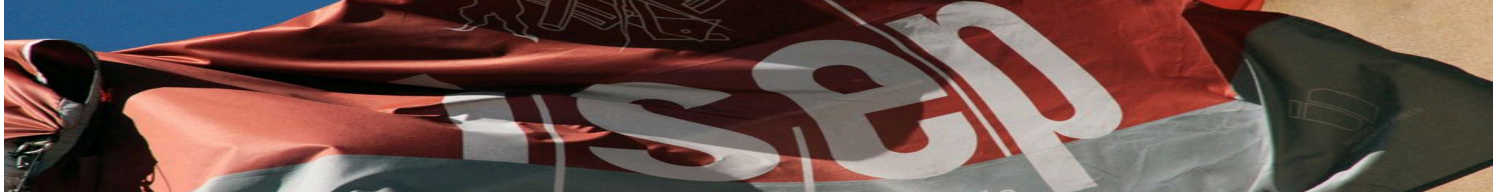
Objetivos

1. Conceitos
2. Triggers de Linha
3. Triggers de Instrução
4. Acedendo a valores de atributos em triggers de linha
5. Exercício



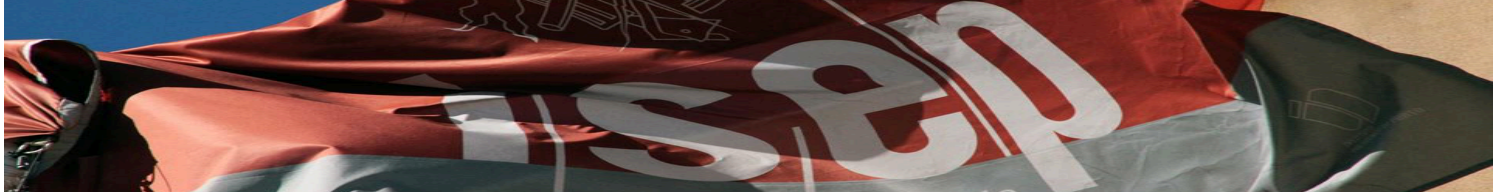
TRIGGERS

- Semelhante aos Procedimentos e às Funções.
- Associados a Tabelas e a Views
- Executados automaticamente quando:
  - Há modificação de dados (DML Trigger)
    - INSERT, UPDATE, coluna UPDATE ou DELETE
  - modificação de esquema (DDL Trigger)
  - eventos do sistema, login / logoff do utilizador (Sistema de Trigger)
- A utilização de triggers deve ser muito cuidadosa (apenas quando necessário) o uso excessivo de triggers pode resultar em interdependências complexas (Cascadind Triggers) que dificultam a manutenção de grandes aplicações.



TRIGGERS

- USADOS:
  - geração automática de dados
    - Auditoria (log), estatísticas
    - dados derivados
    - replicação de dados
  - Restrições especiais
    - Restrições baseadas no tempo
    - Restrições distribuídas
    - Restrições que não podem ser asseguradas pelo SGBDDB (regras de negócio complexas que não é possível impor através de CONSTRAINTS
  - Atualizações de Views complexas



TRIGGERS

CREATE [OR REPLACE] TRIGGER *trigger_name*

AFTER | BEFORE | INSTEAD OF *a_trigger_event* → ***instrução de triggering***

ON *table_name (or view_name)*

[**FOR EACH ROW[WHEN trigger_condition]**] → ***restrição***

DECLARE (opcional)

BEGIN (obrigatório)

Executado unicamente quando a condição de Trigger é TRUE

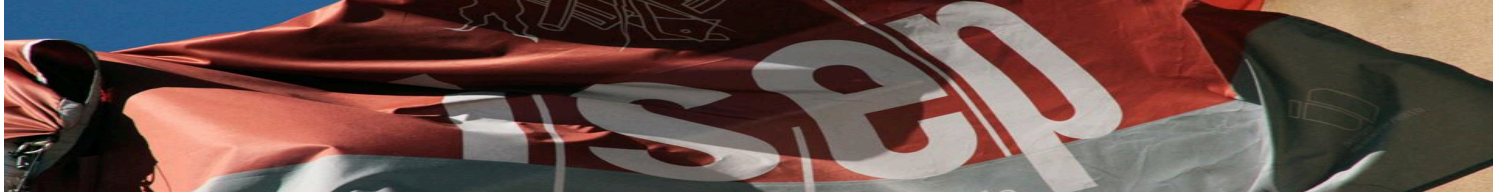
EXCEPTION (opcional)

Exception Section

END; (obrigatório)

ação ou corpo do trigger

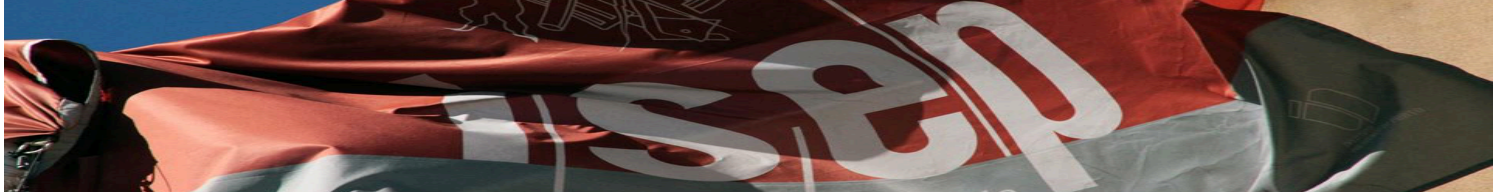
NOTA: Um *trigger_event* pode ser qualquer combinação de um INSERT, DELETE e/ou UPDATE numa tabela ou view



TRIGGERS - Tipos

- Quando se define um Trigger é possível especificar se este deve ser executado:
  - para cada linha afectada pela instrução de triggering, tal como um Update statement que actualiza 'n' linhas. (triggers de linha)
  - para cada instrução de triggering, independentemente do numero de linhas que afecte (triggers de instrução)
  - antes da instrução de triggering
  - depois da instrução de triggering

- É possível ter vários triggers do mesmo tipo para a mesma tabela



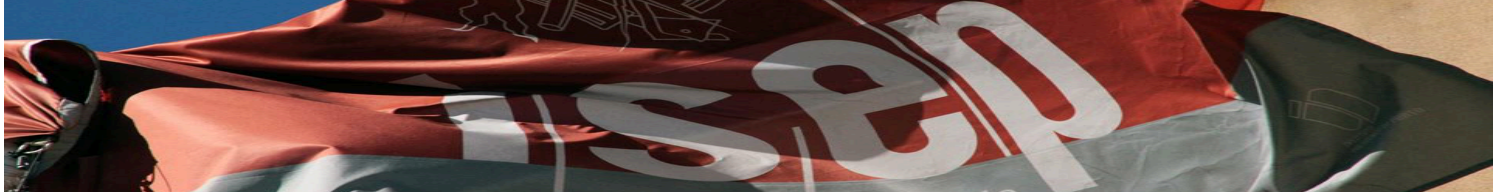
TRIGGERS – Exemplo de Trigger de linha

- **Criando um trigger de Linha**

```
/* Validando o domínio de um salário */
create or replace trigger testa_salario
   before insert or update of salario on funcionario
   for each row
   begin
     if :new.salario > 8000 then
       raise_application_error(-20000,'VALOR INCORRETO');
     end if;
   end;
/
```

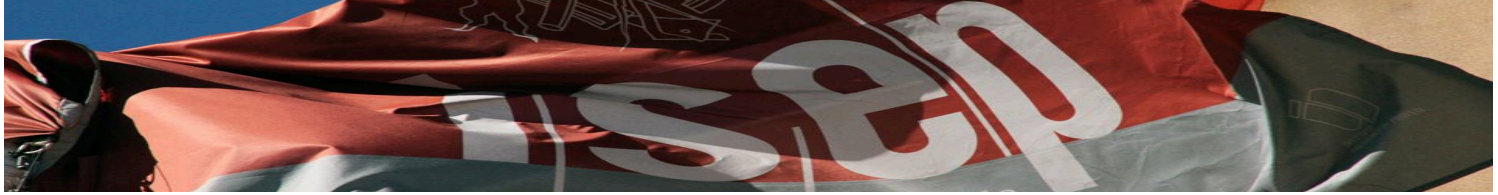
Obs: RAISE_APPLICATION_ERROR (número do erro, mensagem do erro);

-> *número do erro compreendido entre* -20000 e –20999



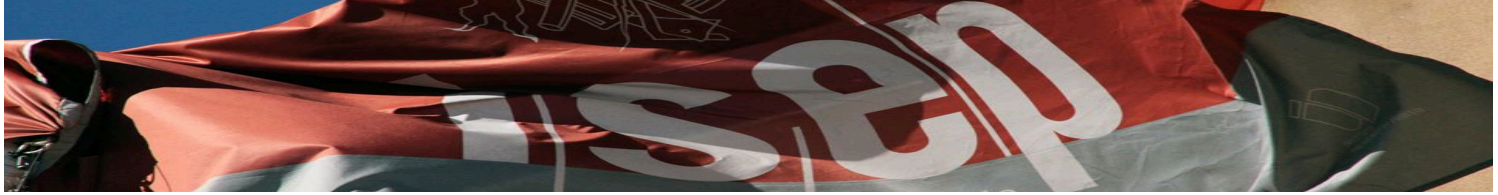
TRIGGERS – for each row

- Esta opção, quando especificada, "dispara" o trigger em cada registo afectado pela instrução de triggering.
- A ausência desta opção indica que o trigger só é executado uma única vez para cada instrução e não separadamente para cada registo afectado
- Quando se especifica com uma condição (cláusula WHEN)
  - a condição será avaliada para todos os registos afectados pelo trigger. Se a avaliação resultar em TRUE para o registo, então a acção do trigger é executada em relação a esse registo, caso contrário não será executada.
  - A expressão na cláusula WHEN deve ser uma expressão SQL e não pode incluir subqueries;



TRIGGERS – de Instrução

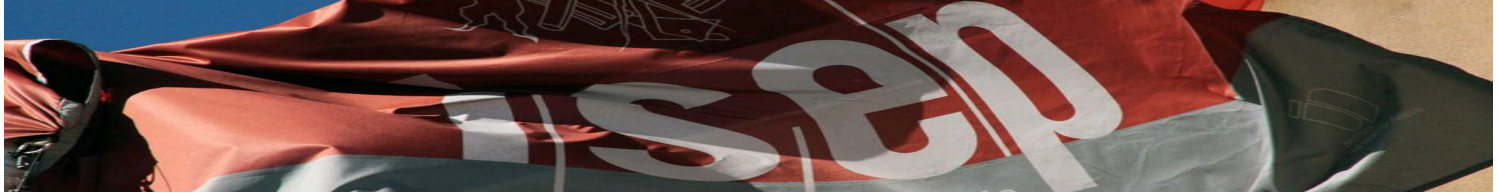
- Tem a finalidade de tratar a execução de ações sobre tabelas independentemente de quantas linhas forem afetadas.
- Através deste tipo de Trigger podemos registar a execução de comandos INSERT, UPDATE e DELETE contra tabelas que tenham Triggers contemplando essas ações.
- Caso um comando UPDATE atualize 1000 linhas, um Trigger deste tipo apenas dispararia 1 única vez. Este tipo de Trigger não pode referenciar qualquer valor contido em uma coluna da tabela. Isso ocorre porque o mesmo dispara uma única vez.



TRIGGERS – Exemplo de Trigger de Instrução

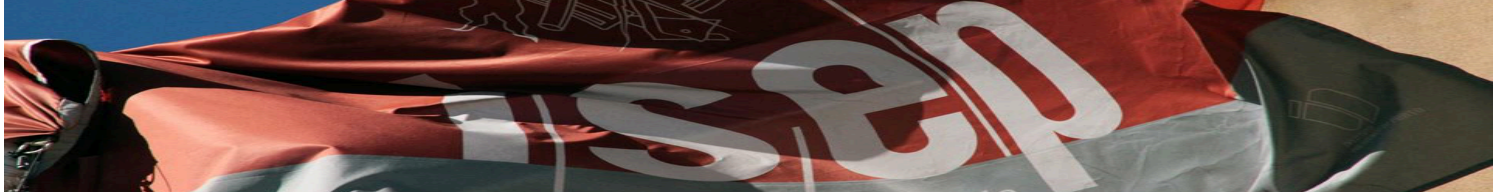
```
CREATE OR REPLACE TRIGGER trg_aud_trn
BEFORE INSERT OR DELETE OR UPDATE  ON transportador
BEGIN
    IF TO_NUMBER (TO_CHAR (SYSDATE, "hh24")) NOT BETWEEN 9 AND 18
    THEN
    raise_application_error(-20001,"Operação não pode ser executada fora do horário
    de expediente.");
    END IF;
END;
/
```

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA



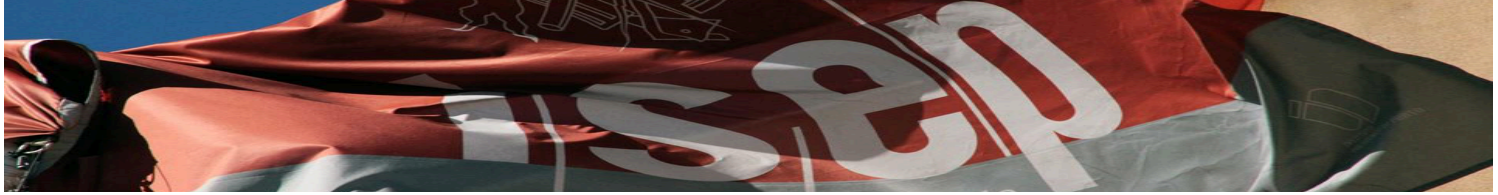
TRIGGERS - Síntese

	Linha	**Instrução**	**Before**	**After**
Execução	Executado sempre que uma tabela é afectada pela instrução de triggering	Executado tendo em consideração a instrução de triggering, independentemente do número de registos afectados	Executa a ação do trigger antes da instrução de triggering;	Executa a ação do trigger depois de executada a instrução de triggering
Utilidade	Se o código contido na ação do trigger depender dos dados resultantes da instrução de triggering ou dos registos afectados	Se o código na ação do trigger não depender dos dados resultantes da instrução de triggering ou dos registos afectados	Permite eliminar processamento desnecessário da instrução de triggering e o seu eventual rollback (casos em que se geram excepções na ação do trigger)	Controlar o timming dum trigger;
Aplicação		Questões de segurança relacionadas com o utilizador; Registos de Auditoria;	Cálculos de valores de colunas específicas antes da instrução de triggering (INSERT ou DELETE) estar completa	



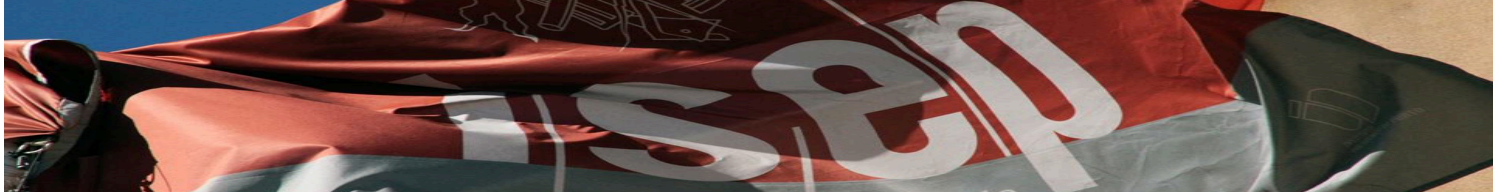
TRIGGERS – Síntese

Tipo Trigger	Caracteristicas
BEFORE instrução	A ação do trigger é executada antes da instrução de triggering;
AFTER instrução	A ação do trigger é executada depois de executada a instrução de triggering
BEFORE linha	A ação do trigger é executada: 1. de acordo com a restrição do trigger; 2. antes de cada linha ser afectada pela instrução de triggering; 3. antes da verificação das restrições de integridade.
AFTER linha	A ação do trigger é executada para cada registo de acordo com a restrição do trigger e depois de modificados os registos pela instrução de triggering. É feito o lock dos registos afectados.



TRIGGERS – Aceder aos valores de atributos

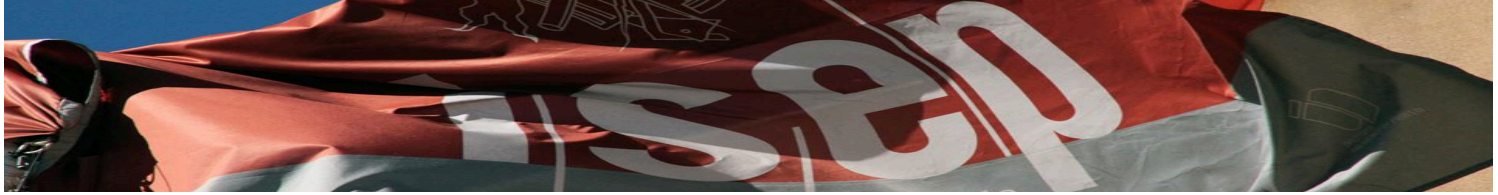
- No corpo dum trigger é possível aceder aos valores antigos e novos dos atributos do registo afectado pela instrução de triggering.
- Existem dois nomes de correlação para cada coluna da tabela a ser modificada:
  - um para o valor antigo (:OLD)
    - **:OLD.nome_atributo -** indica o valor anterior de um campo que está a ser alterado por um comando DELETE ou UPDATE
  - outro para o valor novo (:NEW):
    - **:NEW.nome_atributo .** Indica um novo valor para um campo que está a ser alterado por um comando INSERT ou UPDATE



TRIGGERS – Exemplo

```
Set serveroutput on; // Necessário para visualizar a saída
/* Imprimindo o valor antigo e o novo do salário */
create or replace trigger saldif
before delete or insert or update on funcionario
for each row
  declare
  sal_diff funcionario.salario%type;
  begin
            if (:new.cod_pessoa > 0)  then
              sal_diff := :new.salario - :old.salario;
              dbms_output.put(' antigo: '||:old.salario);
              dbms_output.put(' novo: '||:new.salario);
              dbms_output.put_line(' Diferença:'||sal_diff);
        end if;
    end;
/
```

Operação	:OLD	:NEW
INSERT	x	√
UPDATE	√	√
DELETE	√	x

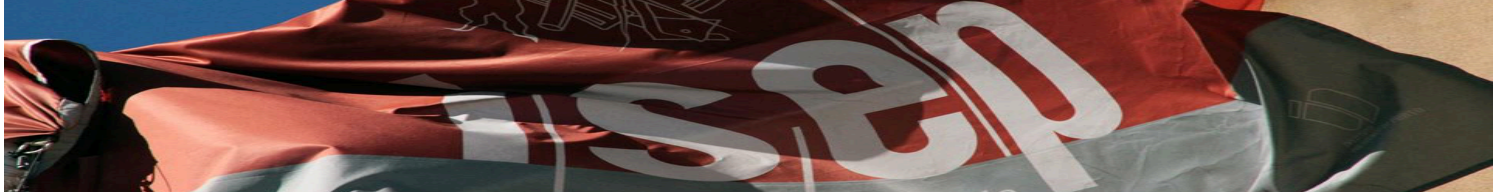


EXERCÍCIO

Considere as seguintes tabelas:

Investigador= {código_I, nome, instituição, data_nascimento, qtd_artigos}
Evento = { código_E, nome, sigla, ano}
Artigo= { código_A, título, idioma, nota, idioma, código_E}
Escrita= { *código_I, código_A*}
País = {código_País, nome_País}

- **Escreva um trigger** que atualize o atributo qtd_artigos da tabela Investigador quando é efectuada uma inserção de um registo na tabela Escrita. Se o valor de qtd_artigos ainda for nulo deve fazer uma atualização contando quantos artigos esse investigador já escreveu; mas se o valor não for nulo basta incrementar o valor atual de qtd_artigos de uma unidade.



EXERCÍCIO – Resolução

```
CREATE OR REPLACE TRIGGER CONTROLARTIGOS
BEFORE INSERT ON Escrita
FOR EACH ROW
DECLARE
        Qtd_artigos_actual  NUMBER;
BEGIN
    SELECT I.qtd_artigos INTO qtd_artigos_atual
    FROM Investigador I
    WHERE I.código_I = :NEW.código_I;
     IF (qtd_artigos_atual IS NULL) THEN
        UPDATE Investigador SET qtd_artigos = ( SELECT COUNT(*)
                                                FROM Escrita E
                                               WHERE E.código_I = :NEW.código_I);
    ELSE
        UPDATE Investigador SET qtd_artigos = qtd_artigos + 1
        WHERE código_I = :NEW.código_I
    END IF;
END;
```