PL/SQL : Cursores, funções, procedimentos e *triggers*

MSc. Eugénio Alberto Macumbe

Nel Chamo

O que iremos discutir?....

PL/SQL

- Cursores;
- Funções;
- Procedimentos;
- Triggers.

PL/SQL - Cursores

- Conjunto privado de resultados;

Dois tipos:

- Implícitos;
- Explícitos;

PL/SQL - Cursores

- Implícitos
  - Criados automaticamente quando um comando SQL é executado;
  - Atributos:
    - %FOUND;
    - %NOTFOUND;
    - %ISOPEN;
    - %ROWCOUNT;
  - Acessado através de :
    - sql%nome_attributo

PL/SQL - Cursores

- EX:

```
Declare
    contagem number;
Begin
    update estudante set saldo = saldo + 200;

    if sql%notfound then
            dbms_output.put_line('estudantes não encontrados');
    else if sql%found then
            contagem := sql%rowcount;
            dbms_output.put_line('encontrados ' || contagem || '
estudantes');
    end if;
    end if;
End;
```

PL/SQL - Cursores

- Explícitos
  - Declarados pelo programador dentro do PL/SQL;
  - É criado através do comando SELECT que retorna mais de uma linha;
  - Declaração:
    - **CURSOR** nome_cursor **IS** comando_select**;**
  - Passos:
    - Declare;
    - Open;
    - Fetch;
    - Close;

PL/SQL - Cursores

- EX:

```
Declare
    v_id estudante.id%type;
    v_nome estudante.nome%type;
    CURSOR c_estudante IS select id, nome from estudante;
Begin
    open c_estudante;
    loop
            fetch c_estudante into v_id, v_nome;
            exit when c_estudante%notfound;
            dbms_output.put_line(v_id || ' ' ||v_nome);
    end loop;
    close c_estudante;
End;
```

PL/SQL - Cursores

- EX2(For loop):

```
Declare
    CURSOR c_estudante IS select id, nome from estudante where
email is not null;
v_estudante c_estudante%rowtype;
Begin
    for v_estudante in c_estudante loop
    dbms_output.put_line(v_estudante.id || ' ' ||v_estudante.nome);
    end loop;
 End;
```

PL/SQL - Functions

- Funções criadas pelo usuário e armazenadas na DB, executadas quando chamadas;
- EX1:

```
Create or replace function total_estudantes()
Return number
IS
   total number := 0;
 BEGIN
   select count(*) into total from estudantes;
   return total;
END;
```

PL/SQL - Functions

- EX2:

```
Create or replace function cria_email (id_estudante number)
Return varchar2
IS
    v_email varchar2(25);
    v_nome estudante.nome%type;
BEGIN
    select nome into v_nome from estudante where id =
id_estudante;
    if sql%found then
            v_email := v_nome || '@up.ac.mz';
    endif;
    return v_email;
END;
```

PL/SQL - Procedures

- Chamdas através do comando EXEC;
- EX:

```
Create or replace procedure define_email
AS
    CURSOR c_estudante IS select id from estudante where email
is null;
    v_id c_estudante%type;
Begin
    open c_estudante;
    fetch c_estudante into v_id;
    while c_estudante%found loop
            update estudante set email = cria_email(v_id) where
id = v_id;
    end loop;
Close c_estudante;
End;
```

PL/SQL – Procedures vs Functions

- Procedures executam uma ou diversas funções enquanto que funções executam uma tarefa especifica;
- Procedures podem ou não retornar valores enquanto que funções retornam valores;
- Funções podem ser chamadas em comandos SQL;
- Funções podem ser chamadas dentro de um Procedure, o contrário já não;

PL/SQL – Triggers

- Código PL/SQL executado automaticamente em resposta à um evento, normalmente DML;
- Propósitos:
  - Gerar dados automaticamente;
  - Obrigar a integridade dos dados;
  - Definir politicas de segurança;
  - Replicação de dados;
  - Auditoria de dados;

PL/SQL – Triggers

```
Create or replace trigger <trigger name>
[before/after] [insert/update/delete] on <table_name>
[for each statement/for each row]
[when <condition>];
```

- Tipos:
  - Before (Antes da acção);
  - After (Depois da acção);
  - For each row (Para cada linha da acção);
  - For each statement (Para cada comando);

PL/SQL – Triggers

- Existem duas variáveis (:old e :new) que guardam os valores das colunas a serem actualizadas na base de dados;
- Existem três variáveis (inserting, updating, deleting) que retornam os valores true ou false de acordo com a operação que está a ser executada;
- alter table <tabela> enable all triggers;
- alter table <tabela> disable all triggers;

PL/SQL – Triggers

- EX:

```
Create or replace trigger maiusculas
before insert or update on estudante
For each row
Declare
Begin
      :new.apelido := upper(:new.apelido);
      :new.nome := upper(:new.nome);
End;
```

PL/SQL – Triggers

- EX2:

```
CREATE OR REPLACE TRIGGER log_estudante
  BEFORE INSERT OR DELETE OR UPDATE of curso ON estudante
  FOR EACH ROW
DECLARE
  v_tipo CHAR(1);
  v_sid varchar2(10);
BEGIN
  IF INSERTING THEN
          v_tipo := 'I';
          v_sid := :new.sid;
  ELSIF UPDATING THEN
          v_tipo := 'U';
          v_sid := :new.sid;
  ELSE
          v_tipo := 'D';
          v_sid := :old.sid;
END IF;
INSERT INTO CUR_AUDIT (tipo, usuario, data,
   SID, antigo, novo)
   VALUES (v_tipo , USER, SYSDATE, v_sid, :old.curso, :new.curso);
END;
```