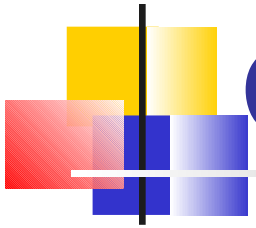
Laboratório de Bases de Dados

Prof. José Fernando Rodrigues Júnior

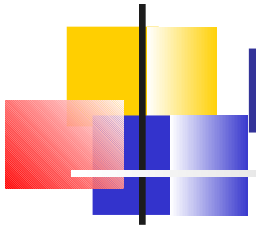
Aula 6 – PL/SQL (Procedural Language/Structured Query Language)

Material: Profa. Elaine Parros Machado de Sousa



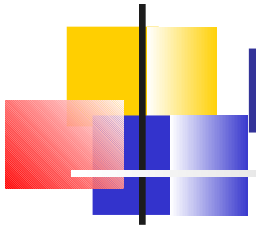
Contexto de programação

- **1GL** – linguagem de máquina, 0´s e 1´s
- **2 GL** – assembly, mnemônicos como LOAD e STORE
- **3 GL** – de alto nível, como C, Java, ...
- **4 GL** – declarações que abstraem os algoritmos e estruturas, como SQL
- **5 GL** – programação visual



PL/SQL

- PL/SQL combina flexibilidade da SQL (4 GL) com construções procedimentais do PL/SQL (3GL)
  - estende SQL:
    - variáveis e tipos
    - estruturas de controle
    - procedimentos e funções
    - tipos de objeto e métodos



PL/SQL

- **PL/SQL *engine*** ⇒ tecnologia
  - compila e executa blocos PL/SQL
  - pode ser instalado em:
    - **servidor Oracle**
      - *stored procedures* e *triggers*
      - blocos anônimos. Ex:
        - **Ferramentas de desenvolvimento PL/SQL**: SQLPlus, SQL Developer, Rapid SQL, DBPartner, SQL Navigator, TOAD, SQL-Programmer, PL/SQL Developer, ...
        - Pré-compiladores (ex: Pro*C/C++), ODBC, JDBC, OCI ...
    - **ferramentas Oracle**
      - Oracle Forms
      - Oracle Reports

PL/SQL

- **PL/S**
  - co
  - pc
    - **Outras combinações 3GL/4GL:**
      - PostgreSQL – PL/pgSQL
      - IBM DB2 – SQL PL
      - Microsoft SQL Server - **Transact-SQL**

      Developer, Rapid SQL, DBPartner, SQL Navigator, TOAD, SQL-Programmer, PL/SQL Developer, ...
      - Pré-compiladores (ex: Pro*C/C++), ODBC, JDBC, OCI ...
    - **ferramentas Oracle**
      - Oracle Forms
      - Oracle Reports

PL/SQL *Engine*

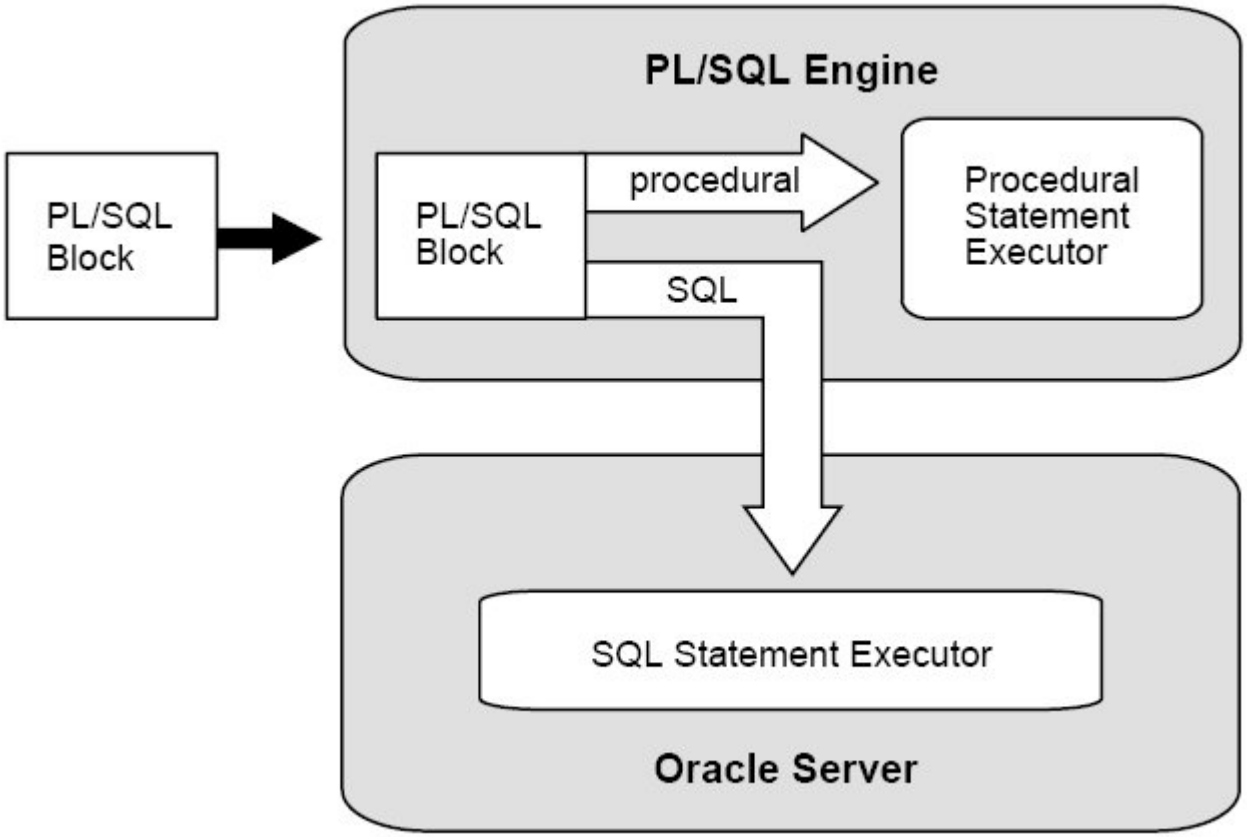
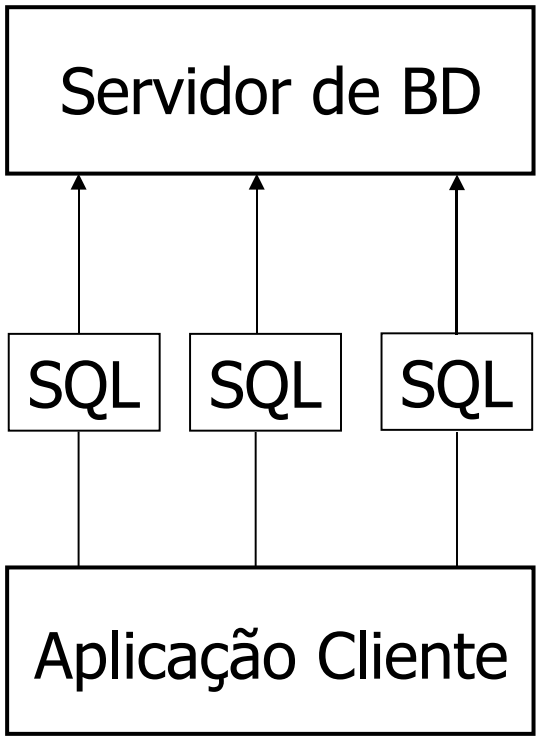
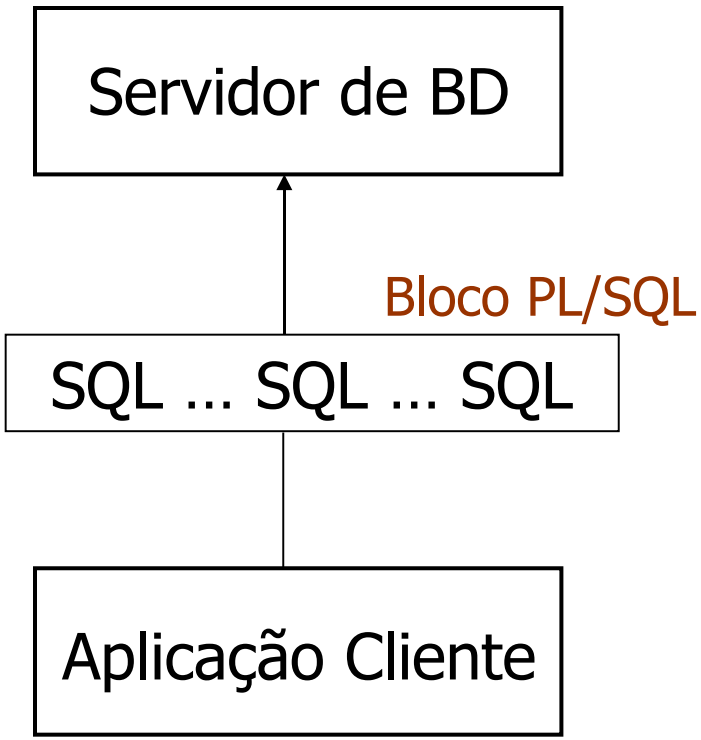
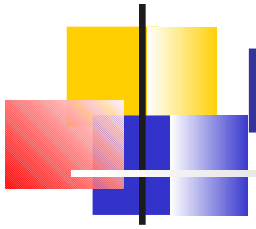


Figura retirada de PL/SQL User's Guide and Reference (Release 2 (9.2))

PL/SQL – Tráfego em Rede

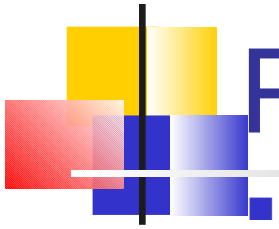






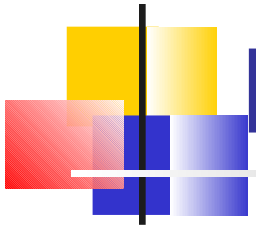
PL/SQL

- Vantagens
  - suporte a SQL
  - suporta a programação OO
  - performance
  - produtividade
  - integração com Oracle
  - resolve “encruzilhadas” SQL



PL/SQL

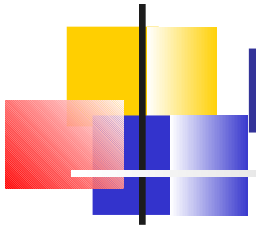
- Recursos
  - estrutura em blocos
  - variáveis e tipos
  - tratamento de erros
  - estruturas de controle
    - condicionais
    - repetição
  - cursores
  - procedimentos e funções
  - pacotes
  - coleções
  - conceitos OO



Princípios básicos PL/SQL

- PL/SQL **não** tem comandos de entrada e saída próprios
- A entrada e saída depende de recursos externos à linguagem
  - Pacote UTL_FILE ⇒ ler/escrever arquivo
  - Pacote DBMS_OUTPUT ⇒ saída em tela

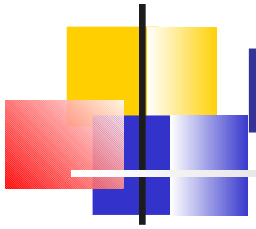
```
set serveroutput on;   /*habilita saída*/
begin
  dbms_output.put_line('Hello World!');
end;
```



Princípios básicos PL/SQL

- Estrutura em 3 blocos

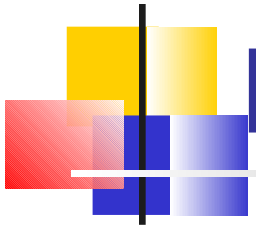
```
DECLARE
      /*variáveis, tipos, cursores, subprogramas, ... */

BEGIN
      /* instruções... */
EXCEPTION
      /*tratamento de exceções*/
END;
```



Princípios básicos PL/SQL

- Declaração/Inicialização de Variáveis

```
nome [CONSTANT] tipo [NOT NULL]
     [DEFAULT] [:= valor]
```



Princípios básicos PL/SQL

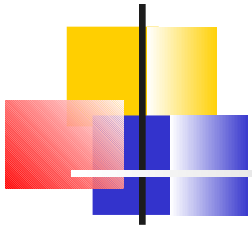
- Exemplo

```
SET SERVEROUTPUT ON;
DECLARE
      v_count NUMBER;
BEGIN
   SELECT count(*) INTO v_count FROM  aluno;
   dbms_output.put_line('NAlunos = ' || v_count);
END;
```

Exemplo

```
DECLARE
        v_nome LBD01_VINCULO_USP.nome%TYPE;
        v_idade LBD01_VINCULO_USP.NROUSP%TYPE;

BEGIN
  SELECT NOME, NROUSP INTO v_nome, v_idade
         FROM LBD01_VINCULO_USP A
         WHERE A.nrousp = 41;

  dbms_output.put_line('Aluno '|| v_nome ||
                       ', NroUSP ' || v_idade);

EXCEPTION /* exceções associadas ao SELECT INTO */
  WHEN NO_DATA_FOUND THEN
           dbms_output.put_line('Aluno não encontrado');
  /*se nusp não fosse único...*/
  WHEN TOO_MANY_ROWS THEN
           dbms_output.put_line('Há mais de um aluno
                                 com este NUSP');
END;
```

- Exemplo



```
DECLARE
    v_nome LBD01_VINCULO_USP.nome%TYPE;
    v_idade LBD01_VINCULO_USP.NROUSP%TYPE;
```

Equivale a:

```
DECLARE
    v_nome VARCHAR2(100);
    v_nusp NUMBER(7,0);
```

➔ O %TYPE faz com que o SGBD descubra qual é o tipo daquele dado no bd.

- Exemplo

```
DECLARE
    v_vinculo LBD01_VINCULO_USP%ROWTYPE;
```

Equivale a:

```
DECLARE
     v_vinculo VARCHAR2(100),v_nusp NUMBER(7,0),...
```

➔ O %ROWTYPE faz com que o SGBD descubra qual é o tipo de tuplas inteiras

Princípios básicos PL/SQL

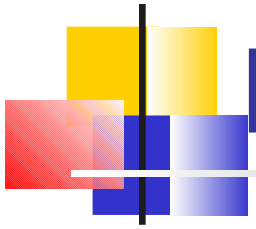
- Estruturas de controle de fluxo
  - `IF ... THEN .... END IF;`
  - `IF ... THEN .... ELSE ... END IF;`
  - `IF ... THEN ....`
    `ELSIF ... THEN...`
    `ELSE ... END IF;`
  - `CASE <variável>`
    `WHEN <valor> THEN <instruções>`
    `WHEN ... THEN...`
    `....`
    `ELSE ... /*opcional*/`
    `END CASE;`

Exemplo - insert

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

```
DECLARE
  v_count_turma NUMBER;
  v_count_aluno NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_count_turma FROM  lbd07_TURMA L
         WHERE L.CODDISC = 'SSC0722' and
           L.ano = EXTRACT (YEAR FROM SYSDATE) and L.NROTURMA = 1;

  IF v_count_turma = 0 THEN
    INSERT INTO LBD07_TURMA VALUES(1,EXTRACT (YEAR FROM SYSDATE),'SSC0722',31);
    dbms_output.put_line('Nova turma criada');
  END IF;

  SELECT COUNT(*) INTO v_count_aluno FROM  lbd08_matricula M
         WHERE M.CODDISC = 'SSC0722' and
             M.ano = EXTRACT (YEAR FROM SYSDATE) and M.NROTURMA = 1;

  IF v_count_aluno < 5 THEN
     INSERT INTO lbd08_matricula(NROUSP,CODDISC,ANO,NROTURMA,NOTA)
     VALUES (1,'SSC0722',EXTRACT (YEAR FROM SYSDATE),1, 0);
     dbms_output.put_line('Aluno matriculado');

  ELSE dbms_output.put_line('Turma lotada');
  END IF;
END;
```

```
DECLARE
        v_count_aluno NUMBER;
        exc_lotada EXCEPTION;
BEGIN
  SELECT COUNT(*) INTO v_count_aluno FROM  lbd08_matricula M
         WHERE M.CODDISC = 'SSC0722' and
               M.ano = EXTRACT (YEAR FROM SYSDATE) and M.NROTURMA = 1;

  IF v_count_aluno < 5 THEN
     INSERT INTO lbd08_matricula(NROUSP,CODDISC,ANO,NROTURMA,NOTA)
     VALUES (6,'SSC0722',EXTRACT (YEAR FROM SYSDATE),1, 0);

  ELSE RAISE exc_lotada;
  END IF;

  EXCEPTION
    WHEN exc_lotada
       THEN dbms_output.put_line('Turma lotada');
    WHEN OTHERS
       THEN dbms_output.put_line('Erro nro:   ' || SQLCODE
                              || '. Mensagem: ' || SQLERRM );
END;
```



Princípios básicos PL/SQL

- Estruturas de Repetição
  - **LOOP *<instruções>***

    EXIT WHEN *<condição de parada>* END LOOP;
  - **WHILE *<condição de parada>* LOOP**

    <instruções>

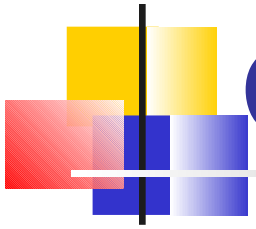
    END LOOP;
  - **FOR *<contador>* IN [REVERSE] *<min>*..*<max>***

    LOOP *<instruções>*

    END LOOP;

Exemplo

```
DECLARE
  v_disciplina LBD07_TURMA.CODDISC%TYPE;
  v_anoTurma   LBD07_TURMA.ANO%TYPE;
BEGIN
  v_disciplina := 'SSC0722';
  v_anoTurma := EXTRACT (YEAR FROM SYSDATE);

  /* insere 6 turmas na disciplina SCC103 */
  FOR nroTurma IN 1..8 LOOP
    INSERT INTO LBD07_TURMA
         VALUES (nroTurma, v_anoTurma, v_disciplina, 31);
         dbms_output.put_line('Turma ' || nroTurma || ' criada.');
  END LOOP;

  EXCEPTION
   WHEN OTHERS
    THEN dbms_output.put_line('Erro nro:  ' || SQLCODE
                              || '. Mensagem: ' || SQLERRM );
END;
```



Cursores

- **Área de contexto**
  - área de memória com informações de processamento de uma instrução
  - inclui **conjunto ativo** ⇒ linhas retornadas por uma consulta
- **Cursor**
  - *handle* para uma área de contexto (cursor **NÃO** é uma variável de memória)
  - tipos:
    - implícito
    - explícito

Cursor Explícito

```
DECLARE
CURSOR c1 IS SELECT empno, ename, job
                FROM emp
               WHERE deptno = 20;
```

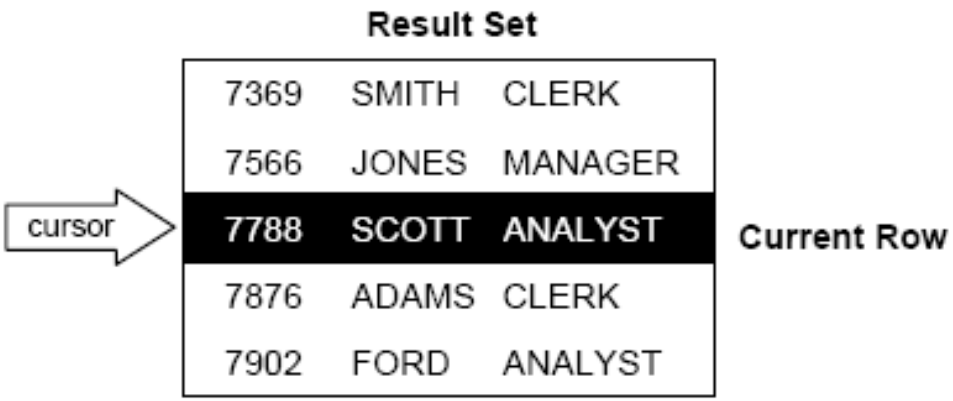
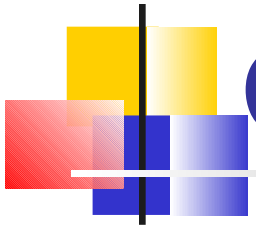
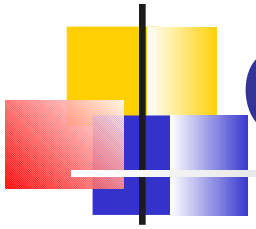


Figura retirada de PL/SQL User's Guide and Reference (Release 2 (9.2))



Cursor Explícito

- Passos:
  - declarar o cursor
  - abrir o cursor
    - **`OPEN`**
  - buscar resultados
    - **`FETCH`** – retorna uma tupla por vez e avança para a próxima no conjunto ativo
  - fechar cursor
    - **`CLOSE`**



Cursor Explícito

- Atributos do tipo *CURSOR*
  - **`FOUND`**
    - **`NULL`** se ainda não houve nenhum **`FETCH`**
    - **`true`** se o **`FETCH`** anterior retornou uma tupla
    - **`false`** caso contrário
  - **`NOTFOUND: !FOUND`**
  - **`ISOPEN`**
  - **`ROWCOUNT`**
    - nro de tuplas **já lidas** por **`FETCH`**

Exemplo – Cursor Explícito

```
DECLARE
  CURSOR c_alunos IS SELECT * FROM lbd03_aluno;
  v_alunos c_alunos%ROWTYPE;

BEGIN

 OPEN c_alunos;  /*abre cursor - executa consulta */

 LOOP
   FETCH c_alunos INTO v_alunos; /*recupera tupla*/
   /*sai do loop se não há mais tuplas*/
   EXIT WHEN c_alunos%NOTFOUND;

   dbms_output.put_line('NUSP: ' || v_alunos.nrousp ||
                        ' - Idade: ' || v_alunos.idade);
 END LOOP;

 CLOSE c_alunos; /*fecha cursor*/

END;
```

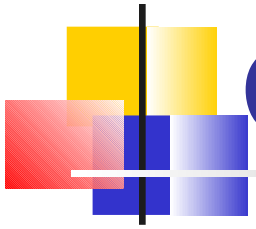
Exemplo – Cursor Explícito

```
DECLARE
  CURSOR c_alunos IS SELECT * FROM lbd03_aluno;
  v_alunos c_alunos%ROWTYPE;

BEGIN

  OP

  LO

    EXIT WHEN c_alunos%NOTFOUND;

    dbms_output.put_line('NUSP: ' || v_alunos.nrousp ||
                         ' - Idade: ' || v_alunos.idade);
  END LOOP;

  CLOSE c_alunos; /*fecha cursor*/

END;
```

O % funciona como operador de propriedade:

%: objeto x propriedade_nome → propriedade_valor

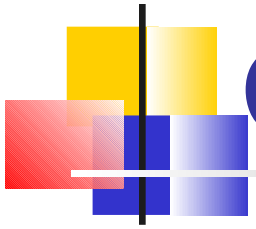
```
DECLARE
   CURSOR c_alunos IS SELECT M.nrousp, A.nome, M.nota
      FROM lbd08_matricula M JOIN lbd01_vinculo_usp A
                             ON M.nrousp = A.nrousp
      WHERE M.coddisc='SSC0722' AND M.ano=2009 FOR UPDATE OF M.nota;
   /*FOR UPDATE OF - registros ficam bloqueados para a seção corrente*/

   v_resultado c_alunos%ROWTYPE; /*ROWTYPE associado a cursor*/

BEGIN
 OPEN c_alunos;
 LOOP
   FETCH c_alunos INTO v_resultado;
   EXIT WHEN c_alunos%NOTFOUND;
   dbms_output.put_line('Aluno: ' || v_resultado.nrousp || ' - ' ||
              v_resultado.nome || ' Nota: ' || v_resultado.nota);
   IF v_resultado.nota = 4.99 THEN
     UPDATE lbd08_matricula
        SET nota = 5.0
      WHERE CURRENT OF c_alunos; /*para update ou delete*/
      /*CURRENT OF se refere necessariamente a um único registro*/
      /*o uso é vinculado a cursores FOR UPDATE OF para update e delete*/
   END IF;
 END LOOP;
 COMMIT; /*Release FOR UPDATE records*/
 CLOSE c_alunos;
END;
```



Cursor Implícito - SQL

- Todas as instruções SQL são executadas dentro de uma área de contexto, então...
  - existe um **cursor implícito** que aponta para essa área de contexto → `cursor SQL`

- PL/SQL implicitamente abre o cursor SQL, processa a instrução SQL e fecha o cursor



Cursor Implícito - SQL

- Utilizado para processar as instruções:
  - **`INSERT`**
  - **`UPDATE`**
  - **`DELETE`**
  - **`SELECT ... INTO`**

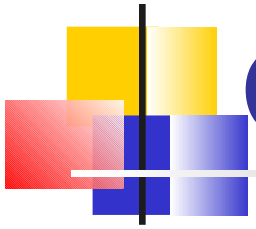
Cursor Implícito - SQL

- **SELECT**
  - **FOUND**
    - **TRUE:** se o comando anterior retornou alguma tupla
    - **FALSE**: caso contrário – no entanto a exceção NO_DATA_FOUND é lançada imediatamente
  - **NOTFOUND**
    - **TRUE:** se o comando anterior não retornou nenhuma tupla - acessível apenas no bloco de exceção
    - **FALSE:** caso contrário
  - **ROWCOUNT:** nro de tuplas retornadas pelo comando anterior
    - **se #tuplas = 0 ➔ ROWCOUNT** == 0 exceção NO_DATA_FOUND - acessível apenas no bloco de exceção
    - se **#tuplas** > 1 exceção TOO_MANY_ROWS - acessível apenas no bloco de exceção com **ROWCOUNT** = 1
    - se **#tuplas = 1 ➔ ok, ROWCOUNT** = 1
  - **ISOPEN**
    - sempre FALSE – propriedade útil apenas para cursores explícitos

Cursor Implícito - SQL

- **SELEC**
  - **FO**
    -
    - çada
  - **NO**
    - enas
      no bloco de exceção
    - **FALSE:** caso contrário
  - **ROWCOUNT:** nro de tuplas retornadas pelo comando anterior
    - **se #tuplas = 0 ➔ ROWCOUNT** == 0 exceção NO_DATA_FOUND - acessível apenas no bloco de exceção
    - se **#tuplas** > 1 exceção TOO_MANY_ROWS - acessível apenas no bloco de exceção com **ROWCOUNT** = 1
    - se **#tuplas = 1 ➔ ok, ROWCOUNT** = 1
  - **ISOPEN**
    - sempre FALSE – propriedade útil apenas para cursores explícitos

Conclusão: o Oracle só permite a utilização de um cursor de seleção implícito caso ele selecione exatamente uma única tupla.



Cursor Implícito - SQL

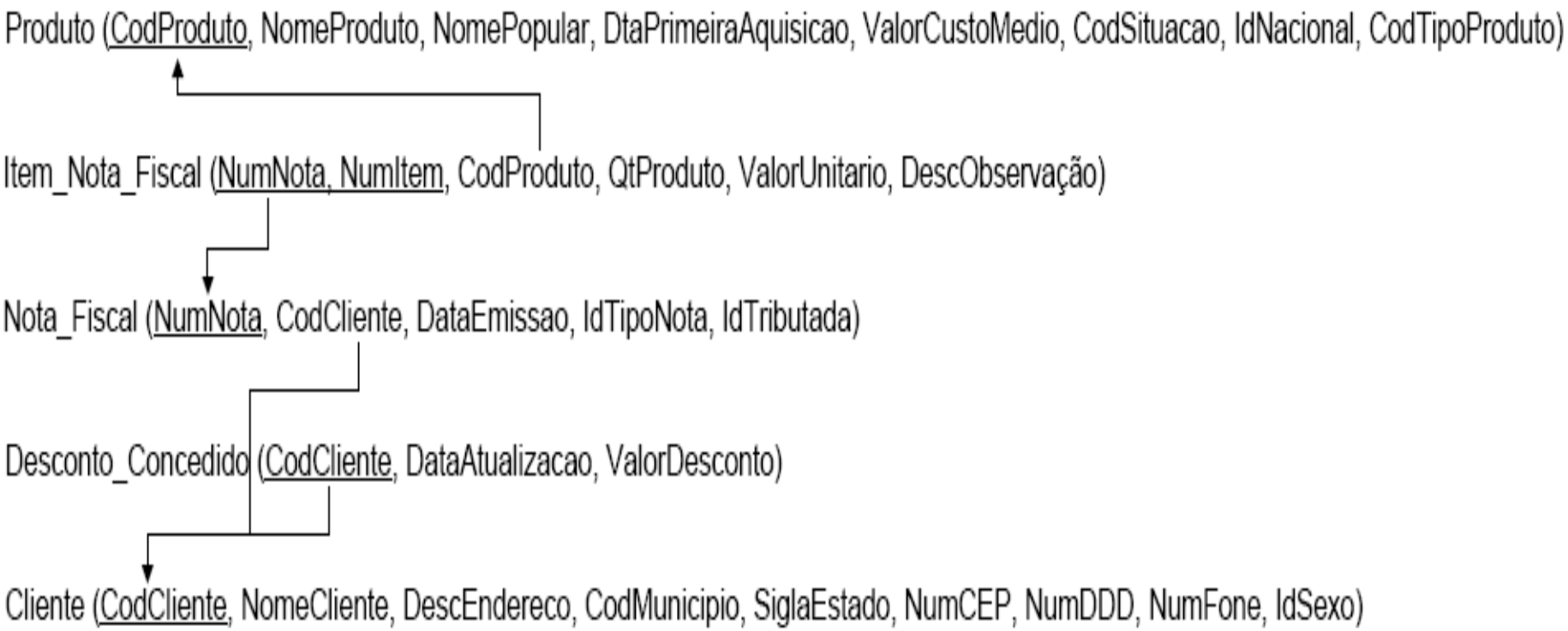
- **INSERT/UPDATE/DELETE**
  - **`FOUND`**
    - **`TRUE:`** se o comando anterior alterou alguma tupla
    - **`FALSE`**: caso contrário
  - **`NOTFOUND`**
    - **`TRUE:`** se o comando anterior não alterou nenhuma tupla
    - **`FALSE:`** caso contrário
  - **`ROWCOUNT:`** nro de linhas alteradas pelo comando anterior
  - **`ISOPEN`**
    - sempre `FALSE` – propriedade útil apenas para cursores explícitos

Exemplo – Cursor Implícito

```
DECLARE
 v_nota CONSTANT lbd08_matricula.nota%TYPE := 5.0;

 BEGIN

  UPDATE lbd08_matricula SET nota = v_nota
       WHERE nota > 3.0 AND nota < 6.0
             AND coddisc = 'SSC0722';

  IF SQL%FOUND /*cursor implícito associado ao UPADATE*/
  THEN dbms_output.put_line(SQL%ROWCOUNT || ' alunos
                            tiveram a nota alterada');
  ELSE dbms_output.put_line('Nenhum aluno teve a nota
                             alterada');
 END IF;

END;
```

```
DECLARE
     v_aluno lbd01_vinculo_usp.nrousp%TYPE;
     v_nome lbd01_vinculo_usp.nome%TYPE;
BEGIN
 v_nome := 'andre';

 --SELECT nrousp INTO v_aluno FROM lbd01_vinculo_usp WHERE nome LIKE 'ANDRE';
 --UPDATE lbd01_vinculo_usp SET nome = 'ANDRE' WHERE nome LIKE 'b%';
 INSERT INTO lbd01_vinculo_usp VALUES(10,3,'adao','04/05/1978','05/08/1985','y');
 IF SQL%FOUND THEN
  dbms_output.put_line('Alteracao TRUE');
 ELSE
  dbms_output.put_line('Alteracao FALSE');
 END IF;

 IF SQL%NOTFOUND THEN
  dbms_output.put_line('Sem alteracao TRUE ' || SQL%ROWCOUNT);
 ELSE
  dbms_output.put_line('Sem alteracao FALSE ' || SQL%ROWCOUNT);
 END IF;

 EXCEPTION
  WHEN OTHERS THEN
       IF SQL%FOUND THEN
        dbms_output.put_line('Except: alteracao TRUE ' || SQL%ROWCOUNT);
       ELSE
        dbms_output.put_line('Except: alteracao FALSE ' || SQL%ROWCOUNT);
       END IF;
       IF SQL%NOTFOUND THEN
        dbms_output.put_line('Except: sem alteracao TRUE ' || SQL%ROWCOUNT);
       ELSE
        dbms_output.put_line('Except: sem alteracao FALSE ' || SQL%ROWCOUNT);
       END IF;
END;
```

Esquema para os Exemplos de Cursor Implícito



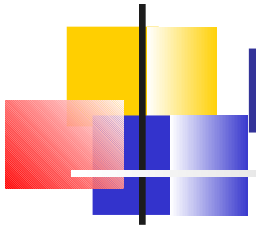
Exemplo – Cursor Implícito

```
DECLARE
       v_cliente Cliente.Cd_Cliente%TYPE := 1;
       v_valor   Nota_Fiscal.Vl_Total%Type;
BEGIN
  SELECT Vl_Total INTO v_valor FROM Nota_Fiscal
         WHERE Cd_Cliente = v_cliente;

  IF v_valor > 0 THEN
     UPDATE Desconto_Concedido
        SET Vl_Desconto = Valor * 0.1,
            Dt_Atualizacao = sysdate
      WHERE Cd_Cliente = Aux_Cliente;
     IF SQL%NOTFOUND THEN
        INSERT INTO Desconto_Concedido
          (Cd_Cliente, Vl_Desconto, Dt_Atualizacao)
          VALUES
          (Aux_Cliente, Valor * 0.1, Sysdate)
     END IF;
  END IF;
END;
```

TEM ERRO!!!!

Exemplo – Cursor Implícito

```
DECLARE
        Vendas Number(5);
        Cursor Cur_Produtos IS SELECT Cd_Produto, Vl_Custo_Medio
                                FROM Produto
                               FOR UPDATE OF Vl_Custo_Medio;
BEGIN
  FOR Reg_Produtos IN Cur_Produtos LOOP
    SELECT Count (*)
      INTO Vendas
      FROM Item_Nota_Fiscal
     WHERE Cd_Produto = Reg_Produtos.Cd_Produto;
     IF Vendas < 4 THEN
       Dbms_Output.Put_line ('Desconto para o produto '||
                              Reg_Produtos.Cd_Produto);
       UPDATE Produto
          SET Vl_Custo_Medio = Vl_Custo_Medio * 0.95
        WHERE CURRENT OF Cur_Produtos;
       IF SQL%NOTFOUND THEN
         Dbms_Output.Put_Line('Erro na atualização.');
       END IF;
     ELSE
       Dbms_Output.Put_Line('Sem alteração no produto '||
                             Reg_Produtos.Cd_Produto);
     END IF;
END;
```

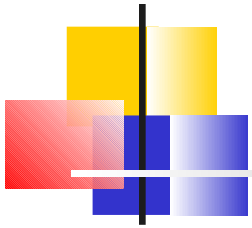


PL/SQL

- Manual de consulta:

PL/SQL

User's Guide and Reference



Prática 5