FIAP GRADUAÇÃO

Application Development For Databases

Prof. Alan Reis

Criando Instruções Executáveis

Objetivos

Ao concluir esta lição, você será capaz de:

- Identificar as unidades lexicais de um bloco PL/SQL
- Usar funções SQL predefinidas no código PL/SQL
- Descrever quando ocorrem conversões implícitas e quando conversões explícitas devem ser usadas
- Criar blocos aninhados e qualificar variáveis com labels
- Criar código legível com recuos adequados
- Usar sequências em expressões PL/SQL

Agenda

- Criando instruções executáveis em um bloco PL/SQL
- Criando blocos aninhados
- Usando operadores e desenvolvendo código legível

Unidades Lexicais de um Bloco PL/SQL

Unidades lexicais:

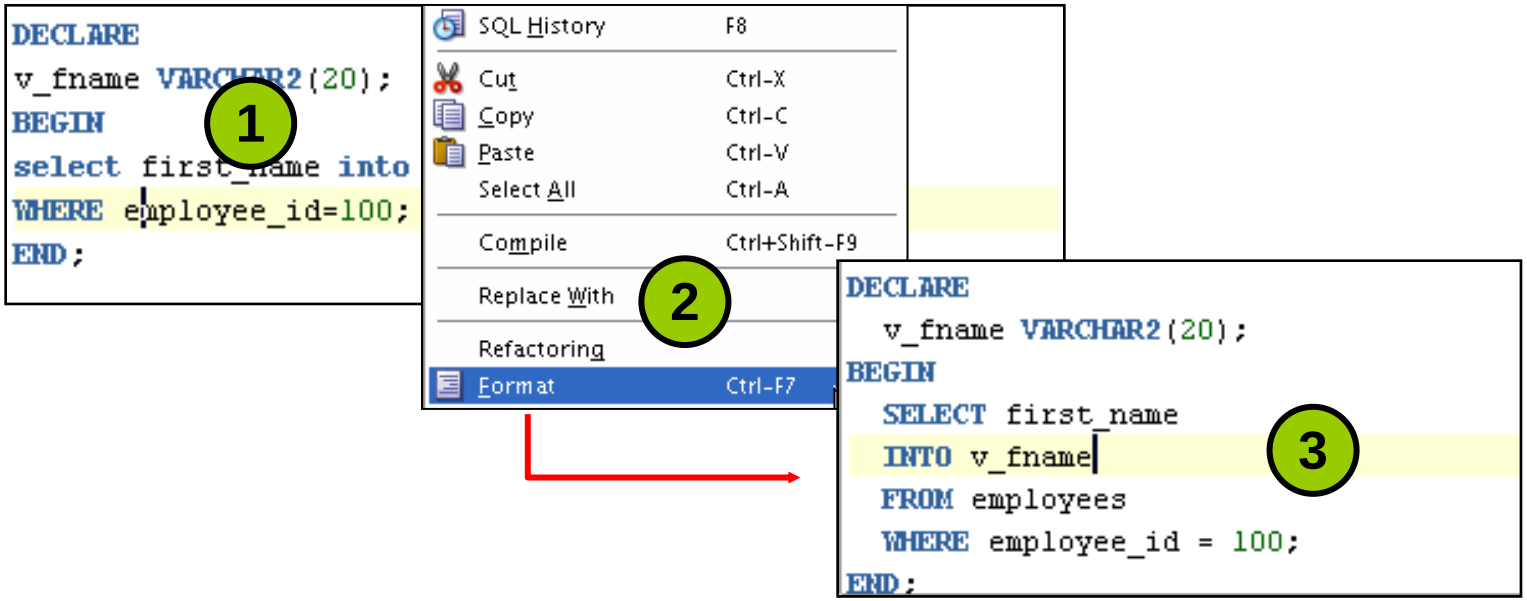
- São blocos de construção para qualquer bloco PL/SQL
- São sequências de caracteres, incluindo letras, dígitos, tabulações, espaços, quebras de linha e símbolos
- Podem ser classificadas como:
  - Identificadores: `v_fname, c_percent`
  - Delimitadores: `; , +, -`
  - Literais: `John, 428, True`
  - Comentários: `--, /* */`

Diretrizes e Sintaxe de Blocos PL/SQL

- Usando Literais
  - Os literais de caractere e de data devem ser delimitados por aspas simples.
  - Os números podem estar em valores simples ou em notação científica.

```
v_name := 'Henderson';
```

- Formatando código: as instruções podem abranger várias linhas.



Comentando o Código

- Inicie os comentários de uma única linha com dois hífens (--).
- Coloque um comentário em bloco entre os símbolos /* e */.

```
DECLARE
...
v_annual_sal NUMBER (9,2);
BEGIN
/* Compute the annual salary based on the
   monthly salary input from the user */
v_annual_sal := monthly_sal * 12;
--The following line displays the annual salary
DBMS_OUTPUT.PUT_LINE(v_annual_sal);
END;
/
```

Funções SQL no Código PL/SQL

- Disponíveis em instruções procedurais:
  - Funções de uma única linha
- Não disponíveis em instruções procedurais:
  - `DECODE`
  - Funções de grupo

Funções SQL no Código PL/SQL: Exemplos

- Obter o tamanho de uma string:

```
v_desc_size INTEGER(5);
v_prod_description VARCHAR2(70):='You can use this
product with your radios for higher frequency';

-- get the length of the string in prod_description
v_desc_size:= LENGTH(v_prod_description);
```

- Obtenha o número de meses que um funcionário trabalhou:

```
v_tenure:= MONTHS_BETWEEN (CURRENT_DATE, v_hiredate);
```

Usando Sequências em Expressões PL/SQL

Iniciando no 11*g*:

```
DECLARE
  v_new_id NUMBER;
BEGIN
  v_new_id := my_seq.NEXTVAL;
END;
/
```

Antes do 11*g*:

```
DECLARE
  v_new_id NUMBER;
BEGIN
  SELECT my_seq.NEXTVAL INTO v_new_id FROM Dual;
END;
/
```

Conversão de Tipo de Dados

- Converte dados em tipos de dados comparáveis
- É de dois tipos:
  - Conversão implícita
  - Conversão explícita
- Funções:
  - `TO_CHAR`
  - `TO_DATE`
  - `TO_NUMBER`
  - `TO_TIMESTAMP`

Conversão de Tipo de Dados

1

```
-- implicit data type conversion
v_date_of_joining DATE:= '02-Feb-2000';
```

2

```
-- error in data type conversion
v_date_of_joining DATE:= 'February 02,2000';
```

3

```
-- explicit data type conversion
v_date_of_joining DATE:= TO_DATE('February
02,2000','Month DD, YYYY');
```

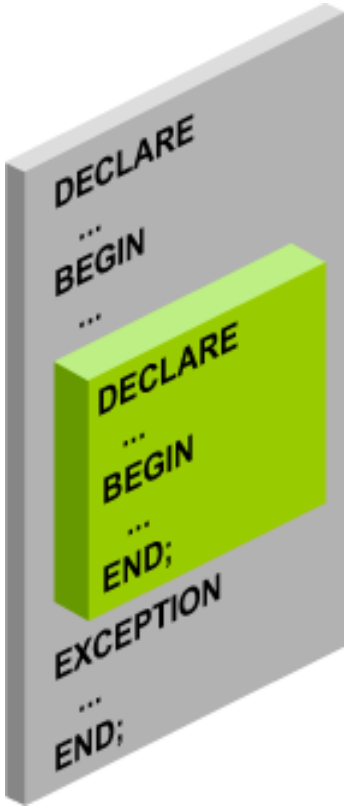
Agenda

- Criando instruções executáveis em um bloco PL/SQL
- Criando blocos aninhados
- Usando operadores e desenvolvendo código legível

Blocos Aninhados

Os blocos PL/SQL podem ser aninhados.

- Uma seção executável (`BEGIN … END`) pode conter blocos aninhados.
- Uma seção de exceções pode conter blocos aninhados.



Blocos Aninhados: Exemplo

```
DECLARE
 v_outer_variable VARCHAR2(20):='GLOBAL VARIABLE';
BEGIN
  DECLARE
   v_inner_variable VARCHAR2(20):='LOCAL VARIABLE';
  BEGIN
   DBMS_OUTPUT.PUT_LINE(v_inner_variable);
   DBMS_OUTPUT.PUT_LINE(v_outer_variable);
  END;
 DBMS_OUTPUT.PUT_LINE(v_outer_variable);
END;
```

```
anonymous block completed
LOCAL VARIABLE
GLOBAL VARIABLE
GLOBAL VARIABLE
```

Escopo e Visibilidade da Variável

```
DECLARE
 v_father_name VARCHAR2(20):='Patrick';
 v_date_of_birth DATE:='20-Apr-1972';
BEGIN
  DECLARE
   v_child_name VARCHAR2(20):='Mike';
   v_date_of_birth DATE:='12-Dec-2002';
  BEGIN
   DBMS_OUTPUT.PUT_LINE('Father''s Name: '||v_father_name);
   DBMS_OUTPUT.PUT_LINE('Date of Birth: '||v_date_of_birth);
   DBMS_OUTPUT.PUT_LINE('Child''s Name: '||v_child_name);
  END;
DBMS_OUTPUT.PUT_LINE('Date of Birth: '||v_date_of_birth);
END;
/
```

Usando um Qualificador com Blocos Aninhados

```
BEGIN <<outer>>
DECLARE
 v_father_name VARCHAR2(20):='Patrick';
 v_date_of_birth DATE:='20-Apr-1972';
BEGIN
  DECLARE
   v_child_name VARCHAR2(20):='Mike';
   v_date_of_birth DATE:='12-Dec-2002';
  BEGIN
   DBMS_OUTPUT.PUT_LINE('Father''s Name: '||v_father_name);
   DBMS_OUTPUT.PUT_LINE('Date of Birth: '
                        ||outer.v_date_of_birth);
   DBMS_OUTPUT.PUT_LINE('Child''s Name: '||v_child_name);
   DBMS_OUTPUT.PUT_LINE('Date of Birth: '||v_date_of_birth);
  END;
END;
END outer;
```

Desafio: Determinação do Escopo de Variáveis

```
BEGIN <<outer>>
DECLARE
  v_sal       NUMBER(7,2) := 60000;
  v_comm      NUMBER(7,2) := v_sal * 0.20;
  v_message   VARCHAR2(255) := ' eligible for commission';
BEGIN
  DECLARE
        v_sal       NUMBER(7,2) := 50000;
        v_comm      NUMBER(7,2) := 0;
        v_total_comp  NUMBER(7,2) := v_sal + v_comm;
  BEGIN
        v_message := 'CLERK not'||v_message;
        outer.v_comm := v_sal * 0.30;
  END;
v_message := 'SALESMAN'||v_message;
END;
END outer;
/
```

1

2

Agenda

- Criando instruções executáveis em um bloco PL/SQL
- Criando blocos aninhados
- Usando operadores e desenvolvendo código legível

Operadores no Código PL/SQL

- Lógicos
- Aritméticos
- Concatenação
- Parênteses para controlar a ordem das operações

Iguais aos do SQL

- Operador exponencial (**)

Operadores no Código PL/SQL: Exemplos

- Incrementar o contador para um loop.

```
loop_count := loop_count + 1;
```

- Definir o valor de um flag booleano.

```
good_sal := sal BETWEEN 50000 AND 150000;
```

- Validar se o número de um funcionário contém um valor.

```
valid  := (empno IS NOT NULL);
```

Diretrizes de Programação

Facilite a manutenção do código:

- Documentando o código com comentários
- Desenvolvendo uma convenção de letras maiúsculas e minúsculas para o código
- Desenvolvendo convenções de nomes para identificadores e outros objetos
- Melhorando a legibilidade com o uso de recuos

Usando Recuos no Código

Para obter clareza, use recuos para cada nível de código.

```
BEGIN
  IF x=0 THEN
     y:=1;
  END IF;
END;
/
```

```
DECLARE
  deptno        NUMBER(4);
  location_id   NUMBER(4);
BEGIN
  SELECT  department_id,
          location_id
  INTO    deptno,
          location_id
  FROM    departments
  WHERE   department_name
          = 'Sales';
...
END;
/
```

Questionário

Você pode usar a maioria das funções SQL de uma única linha, como as de número, caractere, conversão e data em expressões PL/SQL.

a. Verdadeiro
b. Falso

Sumário

Nesta lição, você aprendeu a:

- Identificar as unidades lexicais de um bloco PL/SQL
- Usar funções SQL predefinidas no código PL/SQL
- Criar blocos aninhados para fragmentar funcionalidades relacionadas logicamente
- Decidir quando executar conversões explícitas
- Qualificar variáveis em blocos aninhados
- Usar sequências em expressões PL/SQL

Exercício 3: Visão Geral

Este exercício aborda os seguintes tópicos:

- Verificando regras de escopo e aninhamento
- Criando e testando blocos PL/SQL