Avaliação CP1 – Programação PLSQL

Prof. Ms. Marcel Thomé Filho

Considere as seguintes três tabelas em um banco de dados Oracle:

1. Tabela departamento:

```
CREATE TABLE departamento (
   id_departamento NUMBER PRIMARY KEY,
   nome_departamento VARCHAR2(50)
);
```

2. Tabela funcionario:

```
CREATE TABLE funcionario (
   id_funcionario NUMBER PRIMARY KEY,
   nome_funcionario VARCHAR2(50),
   id_departamento NUMBER,
   salario_atual NUMBER,
   FOREIGN KEY (id_departamento) REFERENCES departamento (id_departamento)
);
```

3. Tabela salario:

```
CREATE TABLE salario (
   id_salario NUMBER PRIMARY KEY,
   id_funcionario NUMBER,
   data_alteracao DATE,
   salario_anterior NUMBER,
   novo_salario NUMBER,
   FOREIGN KEY (id_funcionario) REFERENCES funcionario (id_funcionario)
```

);

Crie um bloco de programação PL/SQL que seja utilizado para inserir dados nas tabelas departamento, funcionario e salario. Em seguida, crie uma função chamada calcular_novo_salario que recebe o ID do funcionário e o percentual de aumento como parâmetros, calcula o novo salário com base no aumento percentual e retorna o resultado. Por fim, crie um procedimento chamado exibir_salarios que exibe o nome do funcionário, o salário anterior e o novo salário para todos os funcionários.

Material de consulta:

/* declaração de variáveis de memória –opcional
Begin
/* instruções de funcionamento –processamento, ifs
Exception
/* tratamento de exceções
opcional
End
/* finalização do bloco

If–then-elsif–then–else–endif
IF<condição> THEN
<instruções>;
ELSIF<condição> THEN
<instruções>;
ELSE
<instruções>;
END IF;

SELECT NOME_DA_COLUNA INTO NOME_DA_VARIAVEL FROM NOME_DA_TABELA
WHERE...;

FOR < contador> IN <valor inicial> .. <valor final> LOOP
< instrução (ões) >;
END LOOP;

CURSOR NOME_DO_CURSOR IS
SELECT COLUNA_1, COLUNA_2, ... , COLUNA_N FROM NOME_DA_TABELA;

DECLARE
CURSOR C_exibeIS SELECT nm_fun, salario FROM funcionario;
BEGIN

```
FOR V_exibeIN C_exibeLOOP
dbms_output.put_line('Nome: '||v_exibe.nm_fun||' -Salário: '||v_exibe.salario);
END LOOP;
END;

DECLARE

        ...

BEGIN

        ...

        EXCEPTION

                WHEN NOME_DA_EXCEÇÃO THEN

                RELAÇÃO_DE_COMANDOS;

                WHEN NOME_DA_EXCEÇÃO THEN

                RELAÇÃO_DE_COMANDOS;

                ...

END;

CREATE OR REPLACE FUNCTION nome_função (p1 in/out ou in/out, p2...)

RETURN tipo_dados;

IS

   variaveis locais

BEGIN

   programação

 RETURN nome_função;

END;


SELECT SOMA (4,6) FROM DUAL;
```

Ou via bloco de programação

Variavel := `SOMA (parametro1, parametro2);`

IN (padrão): Passa um valor do ambiente chamador para procedure e este valor não pode ser alterado dentro dela (passagem de parâmetro por valor).

OUT: Passa um valor da procedure para o ambiente chamador (passagem de

parâmetro por referência).

IN OUT: Passa um valor do ambiente chamador para a procedure. Esse valor pode ser alterado dentro da procedure e retornar com o valor atualizado para o ambiente chamador (passagem de parâmetro por referência).

```
CREATE OR REPLACE PROCEDURE nome_procedure
(argumento1 IN | OUT |IN OUT tipo_de_dados,
argumento2 IN | OUT |IN OUT tipo_de_dados,
...
argumentoN IN | OUT |IN OUT tipo_de_dados) IS | AS
variáveis locais, constantes, ...
BEGIN
...
END nome_procedure;

EXEC PROC_NOME_ALUNO(111222333);
```