Laboratório 1 – Programação Avançada

**Exercícios básicos I**

**Criar um bloco anônimo para contar de 1 a 10 e no final imprimir a mensagem: Acabei**

set serveroutput on;

declare

i number := 1;

begin

while i <= 10 loop

dbms\_output.put\_line(i);

i := i + 1;

end loop;

dbms\_output.put\_line('Acabei');

end;

**Criar um bloco anônimo para contar quantos pares existem entre 1 e 10 e imprimir o resultado. A função do oracle que testa o resto é mod()**

﻿SET SERVEROUTPUT ON;

DECLARE

i NUMBER := 1;

contador\_pares NUMBER := 0;

BEGIN

WHILE i <= 10 LOOP

IF MOD(i, 2) = 0 THEN

contador\_pares := contador\_pares + 1;

END IF;

i := i + 1;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Quantidade de números pares entre 1 e 10: ' || contador\_pares);

END;

**Criar um bloco anônimo para imprimir o nome do funcionário com employee\_id=110**

﻿SET SERVEROUTPUT ON;

declare

var\_nome varchar2(20);

begin

select FIRST\_NAME

into var\_nome

from EMPLOYEES

Where employee\_id = 110;

DBMS\_output.put\_line('Nome do usuario: ' || var\_nome);

end;

**Criar um bloco anônimo que monte a string abaixo para o funcionário 110 imprimindo na tela.**

**﻿**SET SERVEROUTPUT ON;

declare

var\_nome varchar2(20);

var\_telefone varchar2(20);

var\_salario number;

begin

select FIRST\_NAME, PHONE\_NUMBER, SALARY

into var\_nome, var\_telefone, var\_salario

from EMPLOYEES

Where employee\_id = 110;

DBMS\_output.put\_line('Nome do usuario: ' || var\_nome);

DBMS\_output.put\_line('Telefone: ' || var\_telefone);

DBMS\_output.put\_line('Salario: ' || var\_salario);

end;

**Criar Procedure**

**Crie uma procedure capaz de alterar o salário de um empregado de um departamento.**

**﻿**create or replace procedure alterar\_salario\_empregado(

p\_employee\_id IN employees.employee\_id%TYPE,

p\_department\_id IN employees.department\_id%TYPE,

p\_novo\_salario IN employees.salary%TYPE

)

IS

BEGIN

UPDATE employees

SET salary = p\_novo\_salario

WHERE employee\_id = p\_employee\_id

AND department\_id = p\_department\_id;

IF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Nenhum funcionario encontrado com esse id e departamento');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Salario atualizado com sucesso.');

END IF;

END alterar\_salario\_empregado;

/

BEGIN

alterar\_salario\_empregado(110, 100, 7000);

END;

/

**Os parâmetros devem ser código do empregado, o código do departamento e o percentual de reajuste.**

**﻿**create or replace procedure alterar\_salario\_empregado(

p\_employee\_id IN employees.employee\_id%TYPE,

p\_department\_id IN employees.department\_id%TYPE,

p\_percentual\_salario IN number

)

IS

v\_salario\_atualizado employees.salary%TYPE;

BEGIN

UPDATE employees

SET salary = salary \* (1 + p\_percentual\_salario)

WHERE employee\_id = p\_employee\_id

AND department\_id = p\_department\_id;

IF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Nenhum funcionario encontrado com esse id e departamento');

ELSE

SELECT salary INTO v\_salario\_atualizado

FROM employees

WHERE employee\_id = p\_employee\_id;

DBMS\_OUTPUT.PUT\_LINE('Salario atualizado com sucesso.');

DBMS\_OUTPUT.PUT\_LINE('Novo salario: ' || TO\_CHAR(v\_salario\_atualizado));

END IF;

END alterar\_salario\_empregado;

/

BEGIN

alterar\_salario\_empregado(110, 100, 0.10);

END;

/

COMMIT

**Criar Função**

**Criar uma função capaz de retornar o total de funcionários de um departamento.**

**﻿**CREATE OR REPLACE FUNCTION total\_funcionarios\_por\_departamento(

p\_department\_name IN VARCHAR2

) RETURN NUMBER

IS

v\_total NUMBER;

BEGIN

SELECT COUNT(\*)

INTO v\_total

FROM employees e

JOIN departments d ON e.department\_id = d.department\_id

WHERE d.department\_name = p\_department\_name;

RETURN v\_total;

END total\_funcionarios\_por\_departamento;

/

set serveroutput on;

declare

v\_total NUMBER;

begin

v\_total:= total\_funcionarios\_por\_departamento('Finance');

DBMS\_OUTPUT.PUT\_LINE('Total de funcionarios no departamento: ' || v\_total);

end;

/

commit