1 –

Pelo fato de que a linguagem SQL não tem tanta expressividade quanto outras linguagens de programação como C, Python ou Java, ela tem a necessidade de ser acoplada à uma linguagem para melhor manipulação dos dados inseridos no banco, essa linguagem ao qual é acoplado a SQL é chamada de linguagem host.

Usando a sintaxe da SQL Embutida a linguagem hospedeira pode acessar e atualizar os dados no banco. Para identificar a requisição da linguagem hospedeira ao banco é usado a sintaxe ‘ EXEC SQL < > END-EXEC ’ (END-EXEC pode ser alterado para ‘ ; ’ no caso da linguagem hospedeira ser a C).

2 – //Criação da tabela categoria

CREATE TABLE CATEGORIA

(CODIGO\_CAT INTEGER,

PERCENTUAL NUMERIC);

INSERT INTO CATEGORIA VALUES (1, 9.2);

INSERT INTO CATEGORIA VALUES (2, 7.6);

INSERT INTO CATEGORIA VALUES (3, 2.3)

=============================================================================

//Atualização da função 'AtualizaSalario' para buscar o percentual dado o codigo especifico do empregado

CREATE OR REPLACE FUNCTION AtualizaSalario (codigo\_emp INTEGER )

RETURNS integer AS $$

DECLARE

codigoe INTEGER;

salarioe NUMERIC;

ctr INTEGER = 0;

c1 CURSOR FOR SELECT codigo\_emp, salario FROM empregado;

BEGIN

OPEN c1;

LOOP

FETCH c1 INTO codigoe, salarioe;

ctr: = ctr + 1;

EXIT WHEN ctr > 5;

percentual := select BuscaPercentual(categoria);

salarioe := salarioe \* (1 + (percertual / 100));

UPDATE empregado

SET salario = salarioe

WHERE codigo = codigoe;

END LOOP;

CLOSE c1;

RETURN 1;

END;

$$ LANGUAGE plpgsql

=============================================================================

//Criação da busca de de percentual pela chave Categoria inserida na tabela empregado

CREATE OR REPLACE FUNCTION BuscaPercentual(categoria in INTEGER)

RETURNS NUMERIC AS $$

BEGIN

RETURN select cat.percentual

from categoria as cat

where categoria = cat.codigo\_cat

END;

$$ LANGUAGE plpgsql;

=============================================================================

//calcular o aumento sem o update

CREATE OR REPLACE FUNCTION CalculaAumento(salario IN NUMERIC,

categoria IN NUMERIC)

RETURNS NUMERIC AS $$

BEGIN

RETURN salario \* (1 + (select BuscaPercentual(categoria) / 100));

END;

$$ LANGUAGE plpgsql;

=============================================================================

//Atualizações e inserções no Banco

drop table empregado;

drop table categoria;

CREATE TABLE EMPREGADO

(CODIGO\_EMP INTEGER,

NOME VARCHAR(50),

SALARIO NUMERIC,

CATEGORIA INTEGER);

DELETE FROM EMPREGADO;

INSERT INTO EMPREGADO (CODIGO\_EMP, NOME, SALARIO, CATEGORIA) VALUES (1, 'Empregado 1', 1500.00, 1);

INSERT INTO EMPREGADO (CODIGO\_EMP, NOME, SALARIO, CATEGORIA) VALUES (2, 'Empregado 2', 2500.00, 2);

INSERT INTO EMPREGADO (CODIGO\_EMP, NOME, SALARIO, CATEGORIA) VALUES (3, 'Empregado 3', 3200.00, 1);

INSERT INTO EMPREGADO (CODIGO\_EMP, NOME, SALARIO, CATEGORIA) VALUES (4, 'Empregado 4', 900.00, 2);

INSERT INTO EMPREGADO (CODIGO\_EMP, NOME, SALARIO, CATEGORIA) VALUES (5, 'Empregado 5', 1130.00, 3);

ALTER TABLE EMPREGADO ADD COLUMN CODIGO\_CAT INTEGER;

alter table empregado add constraint pk\_codigo\_emp primary key (codigo\_emp);

alter table empregado add constraint fk\_codigo\_cat foreign key (codigo\_cat) references categoria (codigo\_cat);

alter table categoria add constraint pk\_codigo\_cat primary key (codigo\_cat);

3-

4-