LISTA 10 - RESPOSTAS

STORED PROCEDURES

1 - Elabore uma Stored Procedure para inserir valores automáticos na tabela "job", passando como parâmetro a quantidade registros a serem inseridas, além do salário mínimo e máximo a ser atribuído para cada registro.

```
CREATE OR REPLACE FUNCTION insertJobs (qtd INTEGER, salMin NUMERIC(8,2), salMax
NUMERIC(8,2))
    RETURNS INTEGER
    AS $$
    DECLARE
        contJob INTEGER;
        sal NUMERIC(8,2);
    BEGIN
        contJob := 1;
        WHILE contJob <= qtd LOOP
            sal := CAST (RANDOM() * (salMax - salMin) + salMin AS NUMERIC(8,2));
            INSERT INTO job (name, minsalary, maxsalary)
            VALUES ('Job test ' || contJob, sal, salMax);
            contJob := contJob + 1;
        END LOOP;
        RETURN 0;
    END;
    $$ LANGUAGE plpgsql;

SELECT insertJobs (5, 1000, 3000);
```

2 - Elabore uma Stored Procedure para excluir um intervalo desejado de "IDs" na tabela "job".

```
CREATE OR REPLACE FUNCTION deleteJobs (jobIni INTEGER, jobEnd INTEGER)
    RETURNS INTEGER
    AS $$
    DECLARE
        contJob INTEGER;
    BEGIN
        contJob := jobIni;
        WHILE contJob <= jobEnd LOOP
            DELETE FROM job
            WHERE id = contJob;
            contJob := contJob + 2;
        END LOOP;
        RETURN 0;
    END;
    $$ LANGUAGE plpgsql;

SELECT deleteJobs (23, 31);
```

3 - Implemente uma Stored Procedure utilizando o tipo CURSOR. O procedimento deverá atualizar o salário mínimo de todos os cargos da tabela "job", baseando-se no percentual de aumento (inteiro) recebido como parâmetro.

```
CREATE OR REPLACE FUNCTION updatejob (perc INTEGER)
    RETURNS INTEGER
    AS $$
    DECLARE
        curJob CURSOR FOR
            SELECT id, minsalary
            FROM job;
        jobId INTEGER;
        minSal NUMERIC(8,2);
    BEGIN
        OPEN curJob;
        FETCH curJob INTO jobId, minSal;
        WHILE FOUND LOOP
            UPDATE job SET minsalary = (minsalary * ((perc / 100.00) + 1.00))
            WHERE id = jobId;
            FETCH curJob INTO jobId, minSal;
        END LOOP;
        CLOSE curJob;
        RETURN 0;
    END;
    $$ LANGUAGE plpgsql;

SELECT updatejob (10);
```

4 - Implemente uma Stored Procedure utilizando o tipo %ROWTYPE. A função deverá retornar o maior salário entre um intervalo de "IDs", passadas como parâmetros genéricos.

```
CREATE OR REPLACE FUNCTION higherSalary (INT, INT)
    RETURNS NUMERIC(8,2)
    AS $$
    DECLARE
        linJob job%ROWTYPE;
        id1 ALIAS FOR $1;
        id2 ALIAS FOR $2;
        maxSal NUMERIC(8,2);
    BEGIN
        IF EXISTS (SELECT *
                   FROM job
                   WHERE id BETWEEN id1 AND id2) THEN
            SELECT *
            INTO linJob
            FROM job
            WHERE maxsalary >= ALL (SELECT maxsalary
                                    FROM job
                                    WHERE id BETWEEN id1 AND id2);
            maxSal := linJob.maxsalary;
        ELSE
            maxSal := null;
        END IF;
        RETURN maxSal;
    END;
    $$ LANGUAGE plpgsql;

SELECT higherSalary (1, 11);
```

5 - Implemente uma Stored Procedure utilizando o tipo RECORD. A função deverá retornar todas as informações da tabela "job" que possuam salário mínimo igual ao passado como parâmetro.

```
CREATE OR REPLACE FUNCTION selectJob (minSal NUMERIC(8,2))
    RETURNS SETOF job
    AS $$
    DECLARE selJob RECORD;
    BEGIN
        FOR selJob IN (SELECT *
                       FROM job
                       WHERE minsalary = minSal) LOOP
            RETURN NEXT selJob;
        END LOOP;
    END;
    $$ LANGUAGE plpgsql;

SELECT * from selectJob (1100);
```

6 - Crie um tipo para armazenar os seguintes dados da tabela "employee": código do funcionário, nome completo e data de contratação. Utilize este tipo em uma função que recebe como parâmetro duas datas e retorna as informações do tipo criado, dos funcionários que foram contratados entre o intervalo solicitado (datas).

```
CREATE TYPE t_employee AS
    (id INTEGER, name VARCHAR(91), hire DATE);

CREATE OR REPLACE FUNCTION vwEmployee (DATE, DATE)
    RETURNS SETOF t_employee
    AS $$
    SELECT id, firstname || ' ' || lastname, hire
    FROM employee
    WHERE hire BETWEEN $1 AND $2
    $$ LANGUAGE sql;

SELECT * FROM vwEmployee ('2010-01-01', '2015-01-01');
```