**Tema 8 SGBD**

**Opran Andrei 242**

-- e. o procedură prin care se actualizează cu o valoare dată ca parametru salariul unui angajat al cărui nume este dat ca parametru:

-- - se va verifica dacă valoarea dată pentru salariu respectă limitele impuse pentru acel job;

-- - dacă sunt mai mulţi angajaţi care au acelaşi nume, atunci se va afişa un mesaj corespunzător şi de asemenea se va afişa lista acestora;

-- - dacă nu există angajaţi cu numele dat, atunci se va afişa un mesaj corespunzător;

-- f. un cursor care obţine lista angajaţilor care lucrează pe un job al cărui cod este dat ca

-- parametru;

-- g. un cursor care obţine lista tuturor joburilor din companie;

-- h. o procedură care utilizează cele două cursoare definite anterior şi obţine pentru fiecare job

-- numele acestuia şi lista angajaţilor care lucrează în prezent pe acel job; în plus, pentru

-- fiecare angajat să se specifice dacă în trecut a mai avut sau nu jobul respectiv

CREATE OR REPLACE PACKAGE pachet\_employees\_ao AS

    -- e

    PROCEDURE set\_salary(p\_salary IN EMPLOYEES.SALARY%TYPE, p\_first\_name IN EMPLOYEES.FIRST\_NAME%TYPE, p\_last\_name IN EMPLOYEES.LAST\_NAME%TYPE);

    -- f

    FUNCTION get\_employees\_by\_job(p\_job\_id IN JOBS.JOB\_ID%TYPE) RETURN SYS\_REFCURSOR;

    -- g

    FUNCTION get\_jobs RETURN SYS\_REFCURSOR;

    -- h

    PROCEDURE afisare\_informatii\_job;

END pachet\_employees\_ao;

/

**PACKAGE BODY:**

CREATE OR REPLACE PACKAGE BODY pachet\_employees\_ao AS

    PROCEDURE set\_salary(p\_salary IN EMPLOYEES.SALARY%TYPE, p\_first\_name IN EMPLOYEES.FIRST\_NAME%TYPE, p\_last\_name IN EMPLOYEES.LAST\_NAME%TYPE) IS

    min\_salariu JOBS.MIN\_SALARY%TYPE;

    max\_salariu JOBS.MAX\_SALARY%TYPE;

    cont NUMBER(6);

    salariu\_prea\_mic EXCEPTION;

    salariu\_prea\_mare EXCEPTION;

    no\_job EXCEPTION;

    no\_employee EXCEPTION;

    too\_many\_employees EXCEPTION;

    BEGIN

        -- extragere salariu minim si maxim

        SELECT

            MIN\_SALARY,

            MAX\_SALARY

        INTO min\_salariu, max\_salariu

        FROM JOBS

        WHERE JOB\_ID = (

            SELECT

                JOB\_ID

            FROM EMP\_AO WHERE FIRST\_NAME = 'Alexander' AND LAST\_NAME = 'Hunold'

            AND ROWNUM = 1

        );

        IF min\_salariu = -1 OR max\_salariu = -1 THEN

            RAISE no\_job;

        END IF;

        IF p\_salary < min\_salariu THEN

            RAISE salariu\_prea\_mic;

        END IF;

        IF p\_salary > max\_salariu THEN

            RAISE salariu\_prea\_mare;

        END IF;

        -- verificare numar angajati

        SELECT

            COUNT(\*)

        INTO cont

        FROM EMP\_AO

        WHERE FIRST\_NAME = p\_first\_name AND LAST\_NAME = p\_last\_name;

        IF cont = 0 THEN

            RAISE no\_employee;

        END IF;

        IF cont > 1 THEN

            RAISE too\_many\_employees;

        END IF;

        -- actualizare salariu

        UPDATE EMP\_AO

        SET SALARY = p\_salary

        WHERE FIRST\_NAME = p\_first\_name AND LAST\_NAME = p\_last\_name;

        COMMIT;

        DBMS\_OUTPUT.PUT\_LINE('Salariul a fost actualizat cu succes');

        EXCEPTION

            WHEN salariu\_prea\_mic THEN

                DBMS\_OUTPUT.PUT\_LINE('Salariul introdus este prea mic');

            WHEN salariu\_prea\_mare THEN

                DBMS\_OUTPUT.PUT\_LINE('Salariul introdus este prea mare');

            WHEN no\_job THEN

                DBMS\_OUTPUT.PUT\_LINE('Nu exista job pentru angajatul ' || p\_first\_name || ' ' || p\_last\_name);

            WHEN no\_employee THEN

                DBMS\_OUTPUT.PUT\_LINE('Nu exista angajatul ' || p\_first\_name || ' ' || p\_last\_name);

            WHEN too\_many\_employees THEN

                DBMS\_OUTPUT.PUT\_LINE('Exista mai multi angajati cu numele ' || p\_first\_name || ' ' || p\_last\_name);

                FOR inregistrare IN (

                    SELECT

                        EMPLOYEE\_ID,

                        FIRST\_NAME,

                        LAST\_NAME

                    FROM EMPLOYEES

                    WHERE FIRST\_NAME = p\_first\_name AND LAST\_NAME = p\_last\_name

                )

                LOOP

                    DBMS\_OUTPUT.PUT\_LINE('ID: ' || inregistrare.EMPLOYEE\_ID || ', Nume: ' || inregistrare.FIRST\_NAME || ' ' || inregistrare.LAST\_NAME);

                END LOOP;

            WHEN OTHERS THEN

                DBMS\_OUTPUT.PUT\_LINE('OTHER');

    END set\_salary;

    FUNCTION get\_employees\_by\_job(p\_job\_id IN JOBS.JOB\_ID%TYPE) RETURN SYS\_REFCURSOR IS

    rez SYS\_REFCURSOR;

    BEGIN

        OPEN rez FOR

        SELECT \* FROM EMPLOYEES

        WHERE JOB\_ID = p\_job\_id;

        RETURN rez;

    END get\_employees\_by\_job;

    FUNCTION get\_jobs RETURN SYS\_REFCURSOR IS

    rez SYS\_REFCURSOR;

    BEGIN

        OPEN rez FOR

        SELECT \* FROM JOBS;

        RETURN rez;

    END get\_jobs;

    PROCEDURE afisare\_informatii\_job IS

    job\_cursor SYS\_REFCURSOR;

    emp\_cursor SYS\_REFCURSOR;

    job JOBS%ROWTYPE;

    employee EMPLOYEES%ROWTYPE;

    BEGIN

        job\_cursor := get\_jobs;

        LOOP

            FETCH job\_cursor INTO job;

            EXIT WHEN job\_cursor%NOTFOUND;

            DBMS\_OUTPUT.PUT\_LINE('---------------------------------------');

            DBMS\_OUTPUT.PUT\_LINE('JOB TITLE: ' || job.JOB\_TITLE);

            DBMS\_OUTPUT.PUT\_LINE('---------------------------------------');

            emp\_cursor := get\_employees\_by\_job(job.JOB\_ID);

            LOOP

                FETCH emp\_cursor INTO employee;

                EXIT WHEN emp\_cursor%NOTFOUND;

                DBMS\_OUTPUT.PUT\_LINE(employee.FIRST\_NAME || ' ' || employee.LAST\_NAME);

            END LOOP;

            CLOSE emp\_cursor;

        END LOOP;

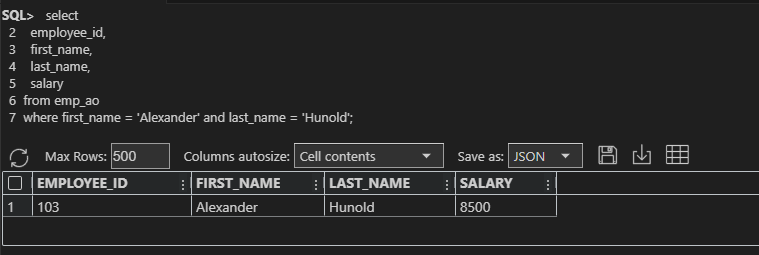
        CLOSE job\_cursor;

    END afisare\_informatii\_job;

END pachet\_employees\_ao;

/

**Testare subpunctul e. :  
  
inainte de rulare:**

****

**Apelare procedura (primele 2 cazuri sunt evidentierea exceptiilor):**

BEGIN

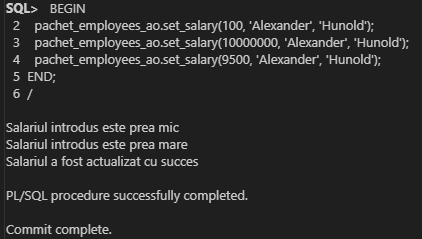
    pachet\_employees\_ao.set\_salary(100, 'Alexander', 'Hunold');

    pachet\_employees\_ao.set\_salary(10000000, 'Alexander', 'Hunold');

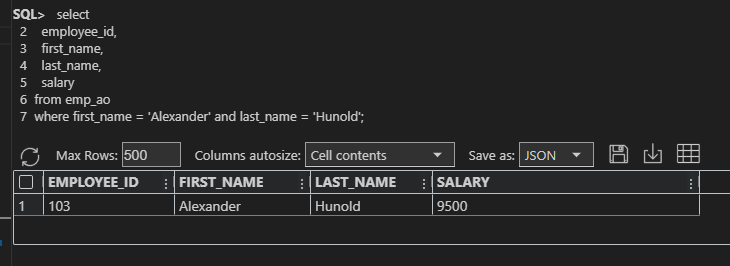
    pachet\_employees\_ao.set\_salary(9500, 'Alexander', 'Hunold');

END;

/

****

**Dupa rulare:**

****

**Rularea cursoarelor de la subpunctele f. si g. va fi aratata la subpunctul h. (h. foloseste cursoarele de la f. si g.)**

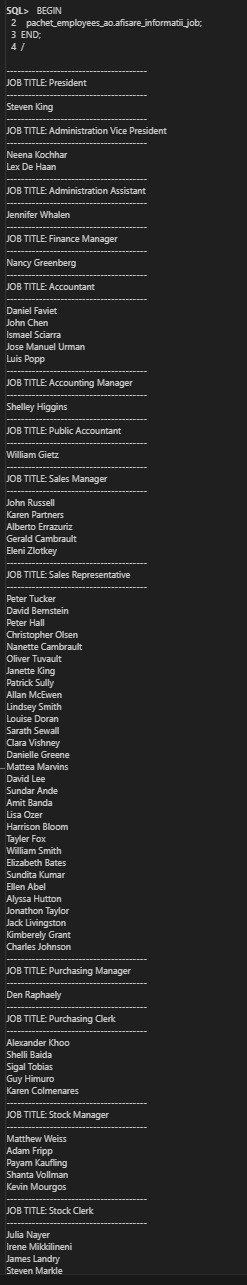
**Subpunctul h.**

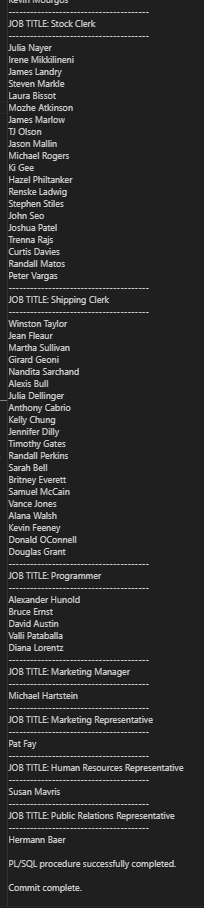
BEGIN

    pachet\_employees\_ao.afisare\_informatii\_job;

END;

/

****

****