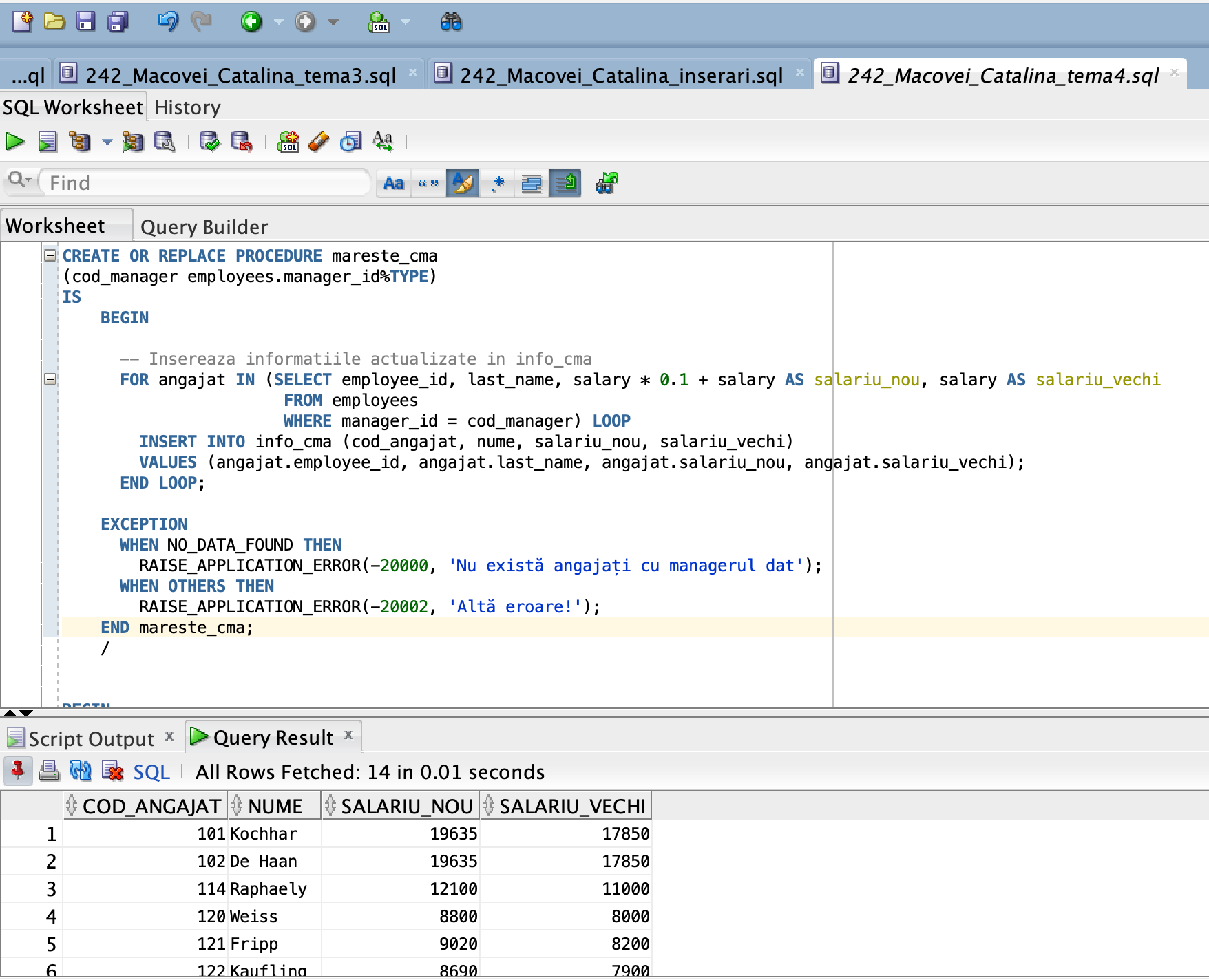
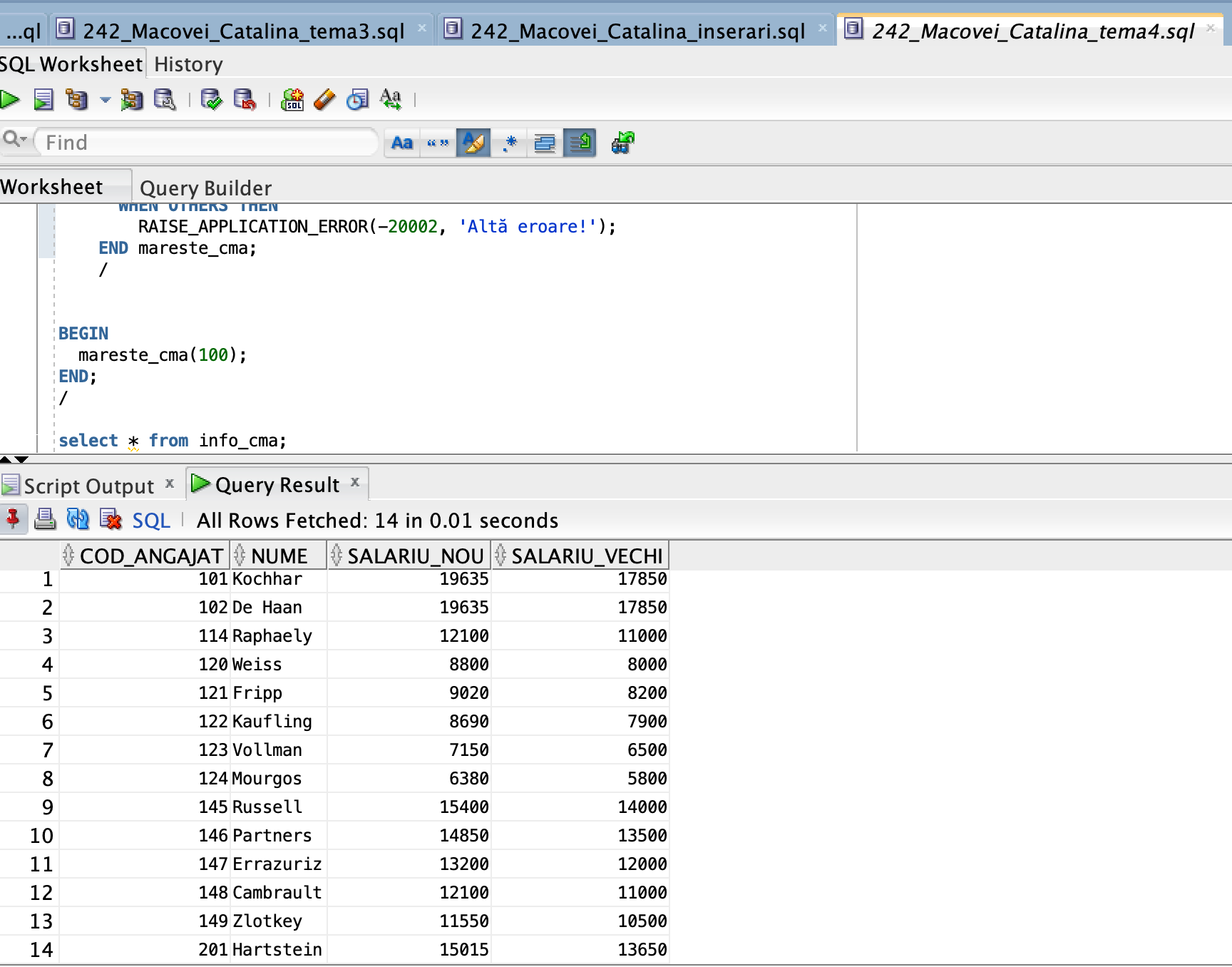
SGBD **Tema 4**

short line

Macovei Catalina  
Grupa 242

EX E4 PLSQL:





**Descriere:** Această procedură mareste\_cma primește un cod de manager și actualizează salariile angajaților care sunt subordonati acelui manager cu 10%. Apoi, inserează într-o tabelă numită info\_cma informații despre fiecare angajat actualizat, inclusiv codul angajatului, numele, salariul vechi și noul salariu. Dacă nu există angajați cu managerul dat sau apare o altă eroare, procedura gestionează aceste situații și afișează un mesaj corespunzător.

**COD SURSA:**

CREATE OR REPLACE PROCEDURE mareste\_cma

(cod\_manager employees.manager\_id%TYPE)

IS

BEGIN

-- Insereaza informatiile actualizate in info\_cma

FOR angajat IN (SELECT employee\_id, last\_name, salary \* 0.1 + salary AS salariu\_nou, salary AS salariu\_vechi

FROM employees

WHERE manager\_id = cod\_manager) LOOP

INSERT INTO info\_cma (cod\_angajat, nume, salariu\_nou, salariu\_vechi)

VALUES (angajat.employee\_id, angajat.last\_name, angajat.salariu\_nou, angajat.salariu\_vechi);

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Nu există angajați cu managerul dat');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Altă eroare!');

END mareste\_cma;

/

BEGIN

mareste\_cma(100);

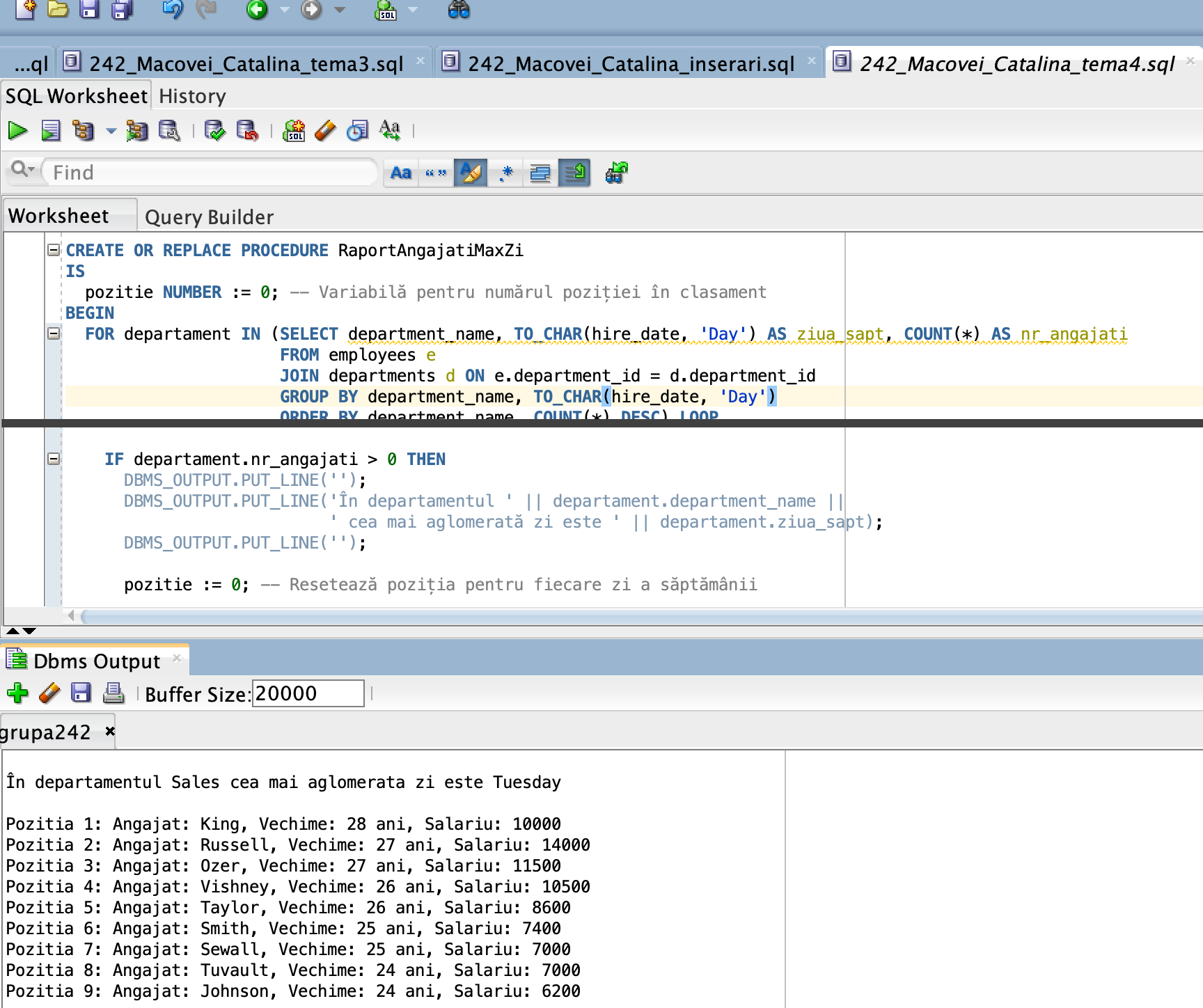
END;

/

**Ex E6 PLSQL :**

**Descriere:**

Această procedură RaportAngajatiMaxZi generează un raport pentru fiecare departament, prezentând cea mai aglomerată zi a săptămânii pentru angajarea noilor persoane în acel departament. Afisează o listă de angajați în ordinea descrescătoare a vechimii lor în departamentul respectiv, indicând poziția fiecărui angajat în clasament, numele lor, vechimea și salariul. În cazul în care nu există informații despre angajați într-un anumit departament sau apare o altă eroare în procesare, raportul indică aceste situații.

****

**Cod sursa:**

CREATE OR REPLACE PROCEDURE RaportAngajatiMaxZi

IS

pozitie NUMBER := 0; -- Variabilă pentru numărul poziției în clasament

BEGIN

FOR departament IN (SELECT department\_name, TO\_CHAR(hire\_date, 'Day') AS ziua\_sapt, COUNT(\*) AS nr\_angajati

FROM employees e

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

GROUP BY department\_name, TO\_CHAR(hire\_date, 'Day')

ORDER BY department\_name, COUNT(\*) DESC) LOOP

IF departament.nr\_angajati > 0 THEN

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

DBMS\_OUTPUT.PUT\_LINE('În departamentul ' || departament.department\_name ||

' cea mai aglomerată zi este ' || departament.ziua\_sapt);

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

pozitie := 0; -- Resetează poziția pentru fiecare zi a săptămânii

FOR angajat IN (SELECT employee\_id, last\_name, ROUND(MONTHS\_BETWEEN(SYSDATE, hire\_date) / 12) AS vechime, salary

FROM employees

WHERE department\_id = (SELECT department\_id FROM departments WHERE department\_name = departament.department\_name)

AND TO\_CHAR(hire\_date, 'Day') = departament.ziua\_sapt

ORDER BY vechime DESC) LOOP

pozitie := pozitie + 1; -- Incrementarea poziției pentru fiecare angajat

DBMS\_OUTPUT.PUT\_LINE('Pozitia ' || pozitie || ': Angajat: ' || angajat.last\_name || ', Vechime: ' || angajat.vechime || ' ani, Salariu: ' || angajat.salary);

END LOOP;

ELSE

DBMS\_OUTPUT.PUT\_LINE('În departamentul ' || departament.department\_name || ' nu lucrează niciun angajat.');

END IF;

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu există informații despre angajați.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Eroare în procesare.');

END RaportAngajatiMaxZi;

/

–apel:

BEGIN

RaportAngajatiMaxZi;

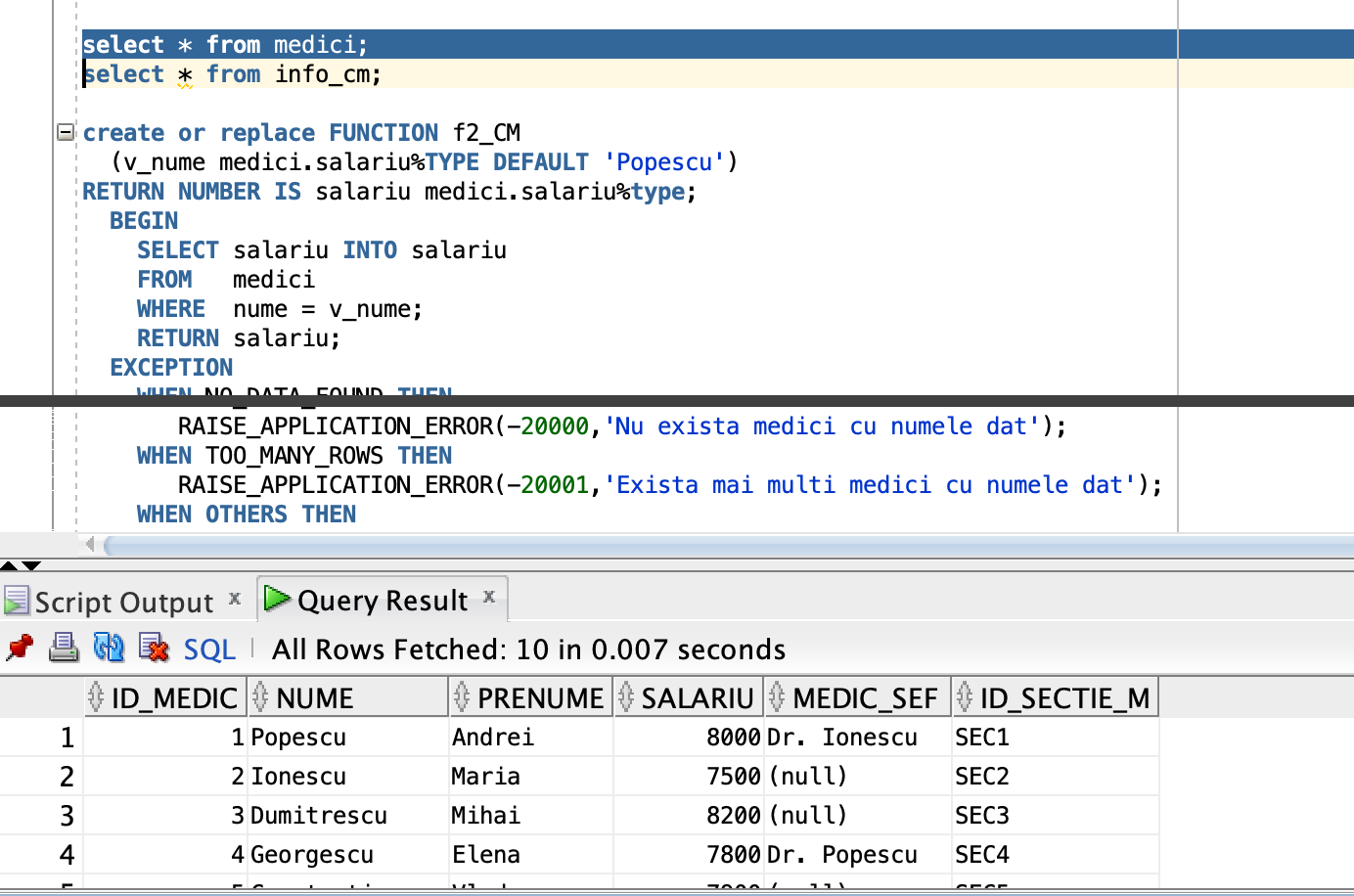
END;

/

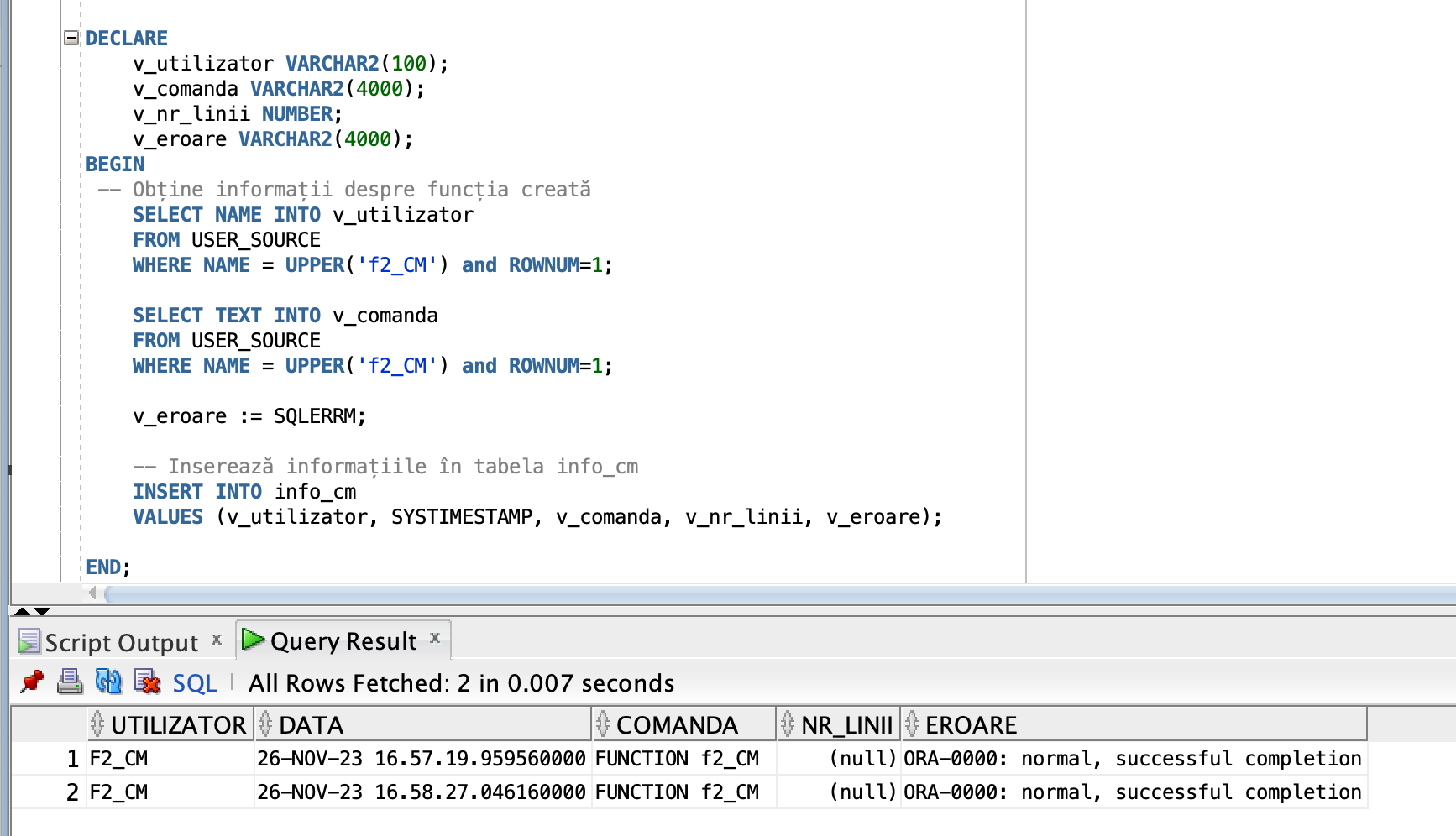
**Ex 7.2 PLSQL - PROIECT**

**Cerința a fost adaptată în baza tabelului MEDICI.**

**Descriere funcție:** Este definită o funcție numită f2\_CM, care primește un parametru cu numele v\_nume și returnează un salariu. Funcția caută un medic cu numele specificat în tabelul medici și returnează salariul acestuia. Se gestionează trei tipuri de excepții: NO\_DATA\_FOUND, TOO\_MANY\_ROWS și OTHERS, fiecare generând o eroare specifică.

****

**Descriere bloc:** Acest bloc PL/SQL încearcă să extragă informații despre o funcție numită 'f2\_CM' din USER\_SOURCE și să le înregistreze în tabela info\_cm.

****

**Cod sursa**

–User INfo

DECLARE

v\_utilizator VARCHAR2(100);

v\_comanda VARCHAR2(4000);

v\_nr\_linii NUMBER;

v\_eroare VARCHAR2(4000);

BEGIN

-- Obține informații despre funcția creată

SELECT NAME INTO v\_utilizator

FROM USER\_SOURCE

WHERE NAME = UPPER('f2\_CM') and ROWNUM=1;

SELECT TEXT INTO v\_comanda

FROM USER\_SOURCE

WHERE NAME = UPPER('f2\_CM') and ROWNUM=1;

v\_eroare := SQLERRM;

-- Inserează informațiile în tabela info\_cm

INSERT INTO info\_cm

VALUES (v\_utilizator, SYSTIMESTAMP, v\_comanda, v\_nr\_linii, v\_eroare);

END;

/

create or replace FUNCTION f2\_CM

(v\_nume medici.salariu%TYPE DEFAULT 'Popescu')

RETURN NUMBER IS salariu medici.salariu%type;

BEGIN

SELECT salariu INTO salariu

FROM medici

WHERE nume = v\_nume;

RETURN salariu;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista medici cu numele dat');

WHEN TOO\_MANY\_ROWS THEN

RAISE\_APPLICATION\_ERROR(-20001,'Exista mai multi medici cu numele dat');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20002,'Alta eroare!');

END f2\_CM;

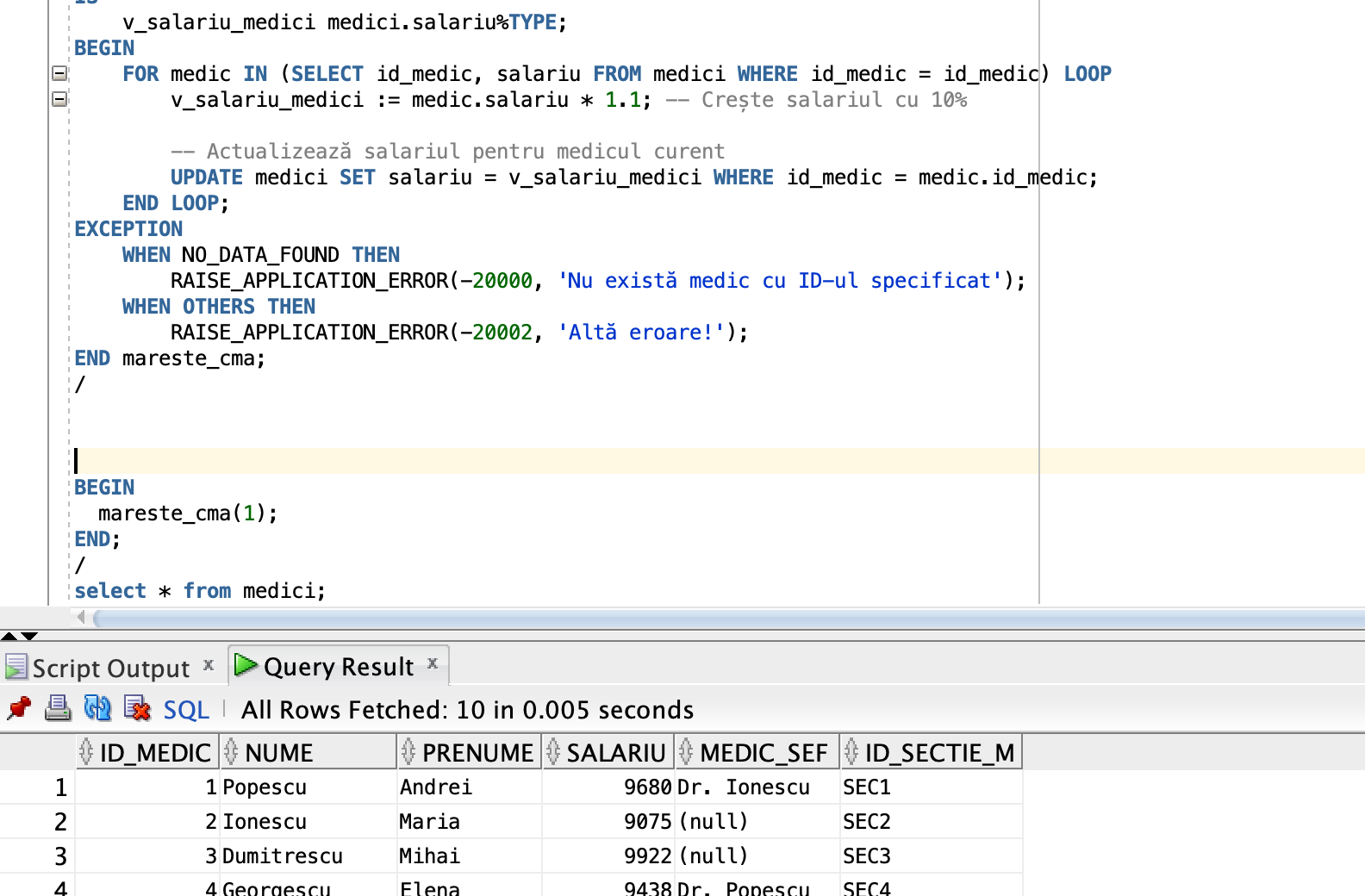
/

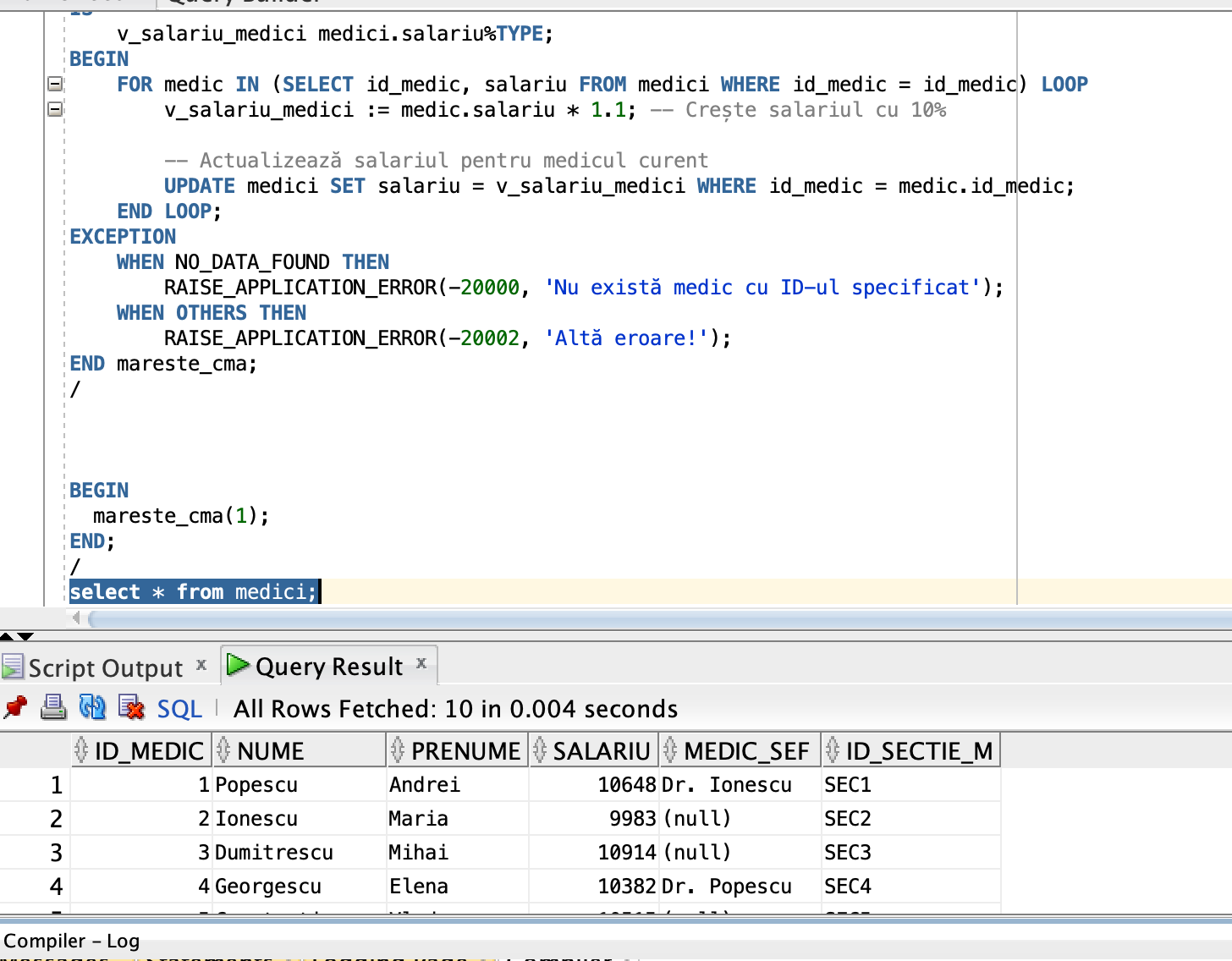
**Ex 7. 4**

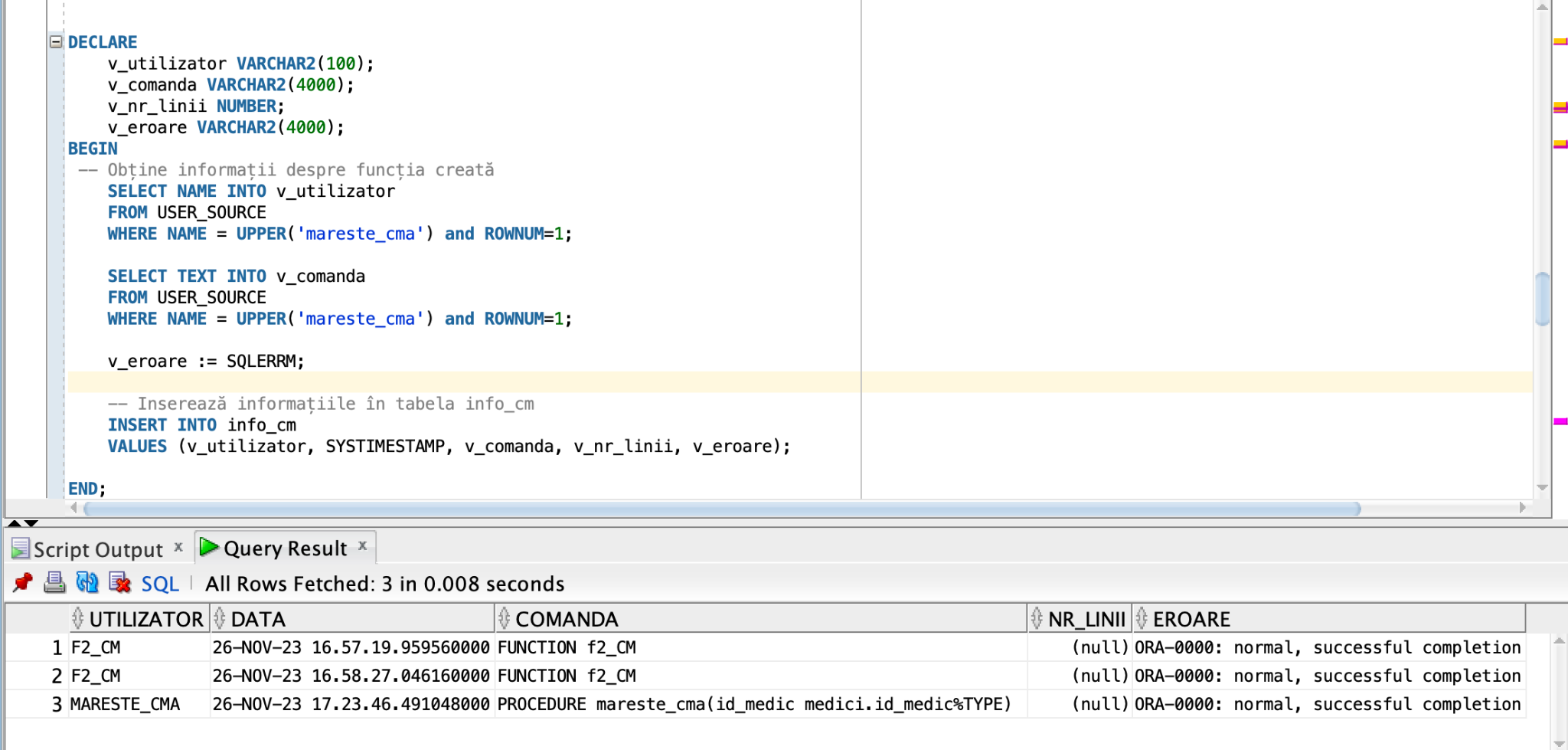
**Descriere:**Procedura mareste\_cma mărește salariul medicului al cărui id este dat ca parametru procedurii. Procedura tratează cazurile de excepție, dacă există medicul cu ID-ul specificat sau nu.

**OBSERVAȚIE**: Din compilare, în pozele de mai jos, observam ca am schimbat salariul medicului cu ID = 1 - Popescu Andrei. Astfel, din 9680 , salariul a fost marit cu 10%, dupa compilare fiind 10 640 .

**USER\_INFO**: De asemenea în a 3-a poza atașată acestui ex, vedem completarea tabelei **info\_cm** cu rezultatele compilarii procedurii **mareste\_cma**.

****

****

****

**COD SURSA:**

–User INfo

DECLARE

v\_utilizator VARCHAR2(100);

v\_comanda VARCHAR2(4000);

v\_nr\_linii NUMBER;

v\_eroare VARCHAR2(4000);

BEGIN

-- Obține informații despre funcția creată

SELECT NAME INTO v\_utilizator

FROM USER\_SOURCE

WHERE NAME = UPPER('mareste\_cma') and ROWNUM=1;

SELECT TEXT INTO v\_comanda

FROM USER\_SOURCE

WHERE NAME = UPPER('mareste\_cma') and ROWNUM=1;

v\_eroare := SQLERRM;

-- Inserează informațiile în tabela info\_cm

INSERT INTO info\_cm

VALUES (v\_utilizator, SYSTIMESTAMP, v\_comanda, v\_nr\_linii, v\_eroare);

END;

/

–Procedura

CREATE OR REPLACE PROCEDURE mareste\_cma

(id\_medic medici.id\_medic%TYPE)

IS

BEGIN

FOR medic IN (select id\_medic, salariu from medici);

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Nu există angajați cu managerul dat');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Altă eroare!');

END mareste\_cma;

/

CREATE OR REPLACE PROCEDURE mareste\_cma(id\_medic medici.id\_medic%TYPE)

IS

v\_salariu\_medici medici.salariu%TYPE;

BEGIN

FOR medic IN (SELECT id\_medic, salariu FROM medici WHERE id\_medic = id\_medic) LOOP

v\_salariu\_medici := medic.salariu \* 1.1; -- Crește salariul cu 10%

-- Actualizează salariul pentru medicul curent

UPDATE medici SET salariu = v\_salariu\_medici WHERE id\_medic = medic.id\_medic;

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Nu există medic cu ID-ul specificat');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Altă eroare!');

END mareste\_cma;

/

BEGIN

mareste\_cma(1);

END;

/

select \* from medici;