ACADEMIA DE STUDII ECONOMICE DIN BUCUREȘTI

Facultatea de Cibernetică, Statistică și Informatică Economică

**PROIECT SGBD**

**GESTIUNEA UNUI MAGAZIN DE PRODUSE ELECTRONICE**

**BERDEI IOANA-ANDREEA PROFESOR COORDONATOR**

**Grupa 1040 PREDA ȘTEFAN**

**CUPRINS**

[**1. Descrierea problemei și prezentarea schemei conceptuale a tabelelor**](#_tqc933ylw7af) **2**

[**2. Folosirea elementelor limbajului PL/SQL**](#_yexl2j6d8k66) **3**

[2.1 IF..THEN..END IF](#_ek7zo380nwnd) 3

[2.2 CASE ... WHEN... THEN...](#_vs0yediqv5xi) 4

[2.3 Una din instrucțiunile “LOOP……END LOOP “ sau “WHILE…..LOOP….END LOOP” sau “FOR…..LOOP….END LOOP”](#_ivu8hnuwmsyd) 6

[2.4 Tipul RECORD sau NUME\_RECORD tabela%ROWTYPE](#_k81mlt2rp06k) 7

[2.5 Tipuri de tabele INDEX BY](#_tr5tt3sio0wn) 8

[2.6 Un cursor implicit](#_4b5g5tjc58ll) 9

[2.7 Un cursor explicit](#_2za9rgtajcjz) 10

[2.8 Un cursor cu parametru](#_udbpfxwp2opq) 12

[2.9 Tratarea unei excepții predefinite](#_7wit67bmo6ac) 13

[2.10 Tratarea unei excepții non predefinite](#_wq470ffctke) 14

[2.11 O procedură](#_wz9eaqmaxg7p) 15

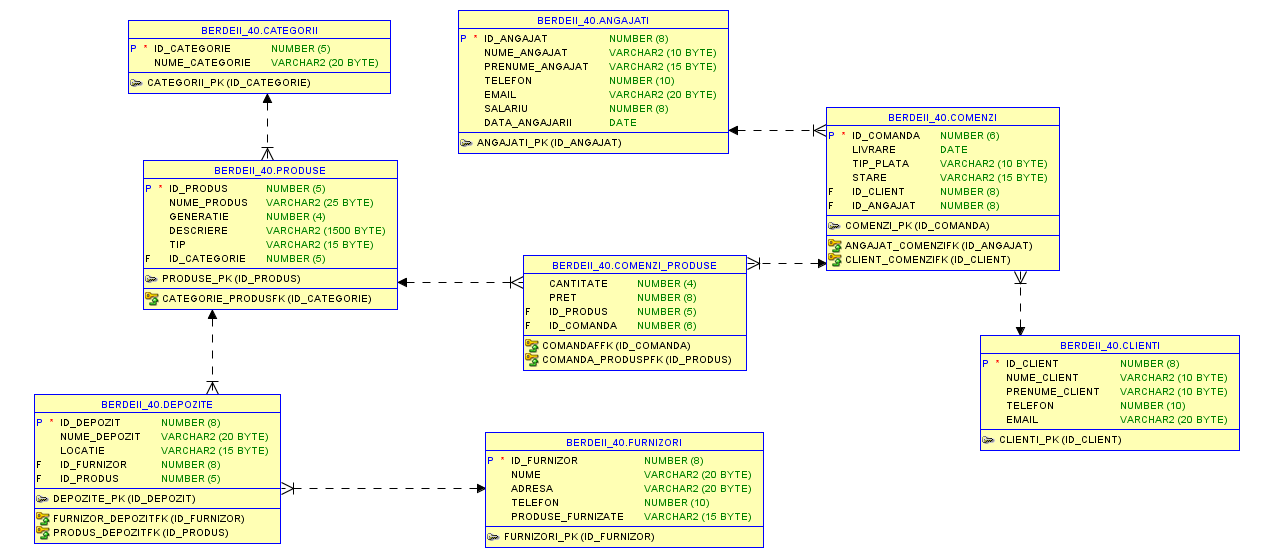
[2.12 O funcție](#_9td6jceqo0ba) 16

[2.13 Un trigger](#_gjdgxs) 17

[2.14 Un pachet](#_bn31g4g2bew4) 19

# 1. Descrierea problemei și prezentarea schemei conceptuale a tabelelor

Proiectul are ca scop realizarea gestiunii unui magazin de produse electronice, a evidenței activității din cadrul acestuia. Utilizatorul bazei de date poate vizualiza elementele ce țin ce activitatea economică, precum și de cea organizatorică a firmei. Datele stocate în această bază de date sunt referitoare la produsele comercializate, angajații ce lucrează în cadrul magazinului, furnizori, precum și la clienții care achiziționează produsele și comenzile date de aceștia.



# 2. Folosirea elementelor limbajului PL/SQL

## 2.1 IF..THEN..END IF

Dacă prețul facturii comenzii cu id-ul 3 este de peste 5000 de lei se va afișa prețul cu o reducere de 20%, iar dacă acesta depășește 6000 de lei se va aplica o reducere de 25%. Toate celelalte comezi primesc o reducere de 5%.

SET SERVEROUTPUT ON

DECLARE

v\_pret\_nou number(4);

BEGIN

SELECT pret INTO v\_pret\_nou FROM comenzi\_produse WHERE id\_comanda = 3;

DBMS\_OUTPUT.PUT\_LINE('Pretul initial este: ' || v\_pret\_nou);

IF v\_pret\_nou > 5000 THEN v\_pret\_nou :=v\_pret\_nou\*0.8;

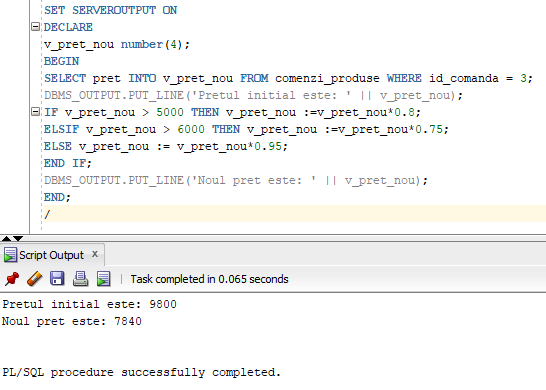
ELSIF v\_pret\_nou > 6000 THEN v\_pret\_nou :=v\_pret\_nou\*0.75;

ELSE v\_pret\_nou := v\_pret\_nou\*0.95;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Noul pret este: ' || v\_pret\_nou);

END;

/

## 2.2 CASE ... WHEN... THEN...

Pentru angajatul alcărui id se va prelua de la tastatură, prin intermediul unei variabile de substituție, dacă salariul său este mai mic de 2600 de lei, se va aplica o mărire de 20%, în timp ce, dacă salariul său este cuprins între 2600 și 3000 de lei, acesta va primi o mărire de 10%. Salariul pentru orice altă sumă va rămâne neschimbat.

SET SERVEROUTPUT ON

DECLARE

v\_salariu angajati.salariu%type;

BEGIN

SELECT salariu INTO v\_salariu FROM angajati WHERE id\_angajat=&id;

DBMS\_OUTPUT.PUT\_LINE ('Salariul initial este: '||v\_salariu);

CASE

WHEN v\_salariu < 2600 THEN

v\_salariu:=1.2 \* v\_salariu;

WHEN v\_salariu BETWEEN 2600 AND 3000 THEN

v\_salariu:=1.1 \* v\_salariu;

ELSE

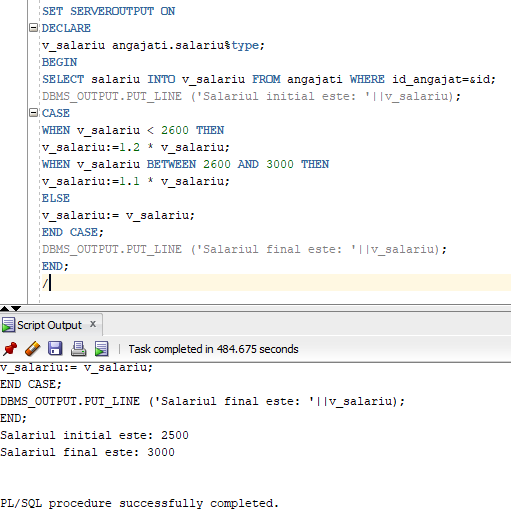
v\_salariu:= v\_salariu;

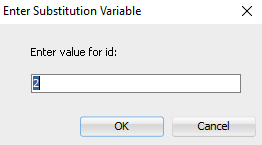
END CASE;

DBMS\_OUTPUT.PUT\_LINE ('Salariul final este: '||v\_salariu);

END;

/





## 2.3 Una din instrucțiunile “LOOP……END LOOP “ sau “WHILE…..LOOP….END LOOP” sau “FOR…..LOOP….END LOOP”

Se afișează toți angajații care au salariul peste medie.

SET SERVEROUTPUT ON

DECLARE

v\_sal angajati.salariu%type;

v\_salmed v\_sal%type;

BEGIN

SELECT avg(salariu) into v\_salmed from angajati;

DBMS\_OUTPUT.PUT\_LINE('Salariul mediu este: '||v\_salmed);

FOR i IN 1..6 LOOP

SELECT salariu INTO v\_sal FROM angajati WHERE id\_angajat=i;

IF v\_sal > v\_salmed THEN

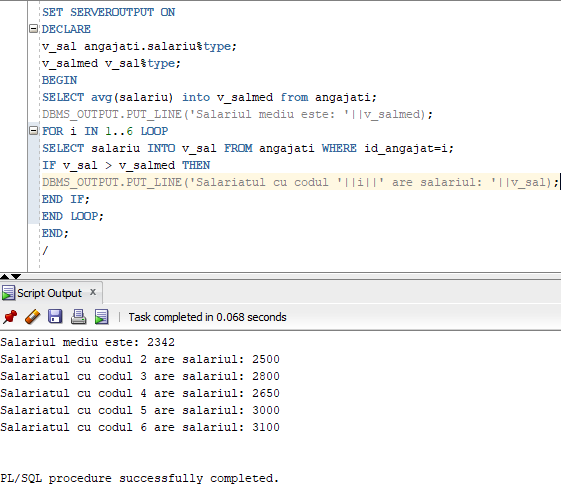
DBMS\_OUTPUT.PUT\_LINE('Salariatul cu codul '||i||' are salariul: '||v\_sal);

END IF;

END LOOP;

END;

/



## 2.4 Tipul RECORD sau NUME\_RECORD tabela%ROWTYPE

Utilizând un tip de dată record de același tip cu tabela clienți se afișează datele (nume, prenume, email) pentru clientul cu id-ul 8.

SET SERVEROUTPUT ON

DECLARE

vrec\_clienti clienti%rowtype;

BEGIN

SELECT \* into vrec\_clienti from clienti where id\_client=8;

dbms\_output.put\_line('Clientul: '|| vrec\_clienti.nume\_client||' '||vrec\_clienti.prenume\_client|| ' are adresa de mail: '||vrec\_clienti.email);

END;

/



## 2.5 Tipuri de tabele INDEX BY

Folosind acest tip de date compuse, se afișează numele, prenumele și data angajării pentru fiecare salariat, precum și numărul total de angajați.

SET SERVEROUTPUT ON

DECLARE

type ang\_table is table of angajati%rowtype index by pls\_integer;

v\_tab ang\_table;

BEGIN

for i in 1..6 loop

SELECT \* into v\_tab(i) from angajati where id\_angajat=i;

end loop;

for i in v\_tab.first..v\_tab.last loop

DBMS\_OUTPUT.PUT\_LINE('Angajatul: '|| v\_tab(i).nume\_angajat||' '||v\_tab(i).prenume\_angajat||' a fost angajat in data de : '||v\_tab(i).data\_angajarii);

end loop;

DBMS\_OUTPUT.PUT\_LINE('Total angajati in tabela: '|| v\_tab.count);

END;

/

## 2.6 Un cursor implicit

Se modifică salariul angajatului cu id-ul introdus prin intermediul unei variabile de substituție.

SET SERVEROUTPUT ON

BEGIN

UPDATE angajati

SET salariu = 2000

WHERE id\_angajat = &id;

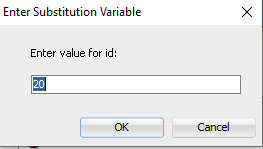
IF SQL%NOTFOUND THEN

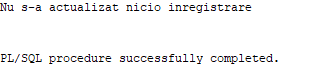
DBMS\_OUTPUT.PUT\_LINE('Nu s-a actualizat nicio inregistrare');

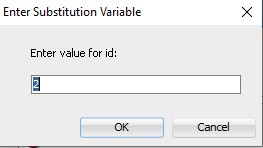
END IF;

END;

/









## 2.7 Un cursor explicit

Folosind un cursor explicit se afișează toate categoriile de produse.

SET SERVEROUTPUT ON

DECLARE

CURSOR c is SELECT nume\_categorie FROM categorii;

r c%ROWTYPE;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Lista cu categoriile de produse disponibile este: ');

OPEN c;

LOOP

FETCH c INTO r;

EXIT WHEN c%notfound;

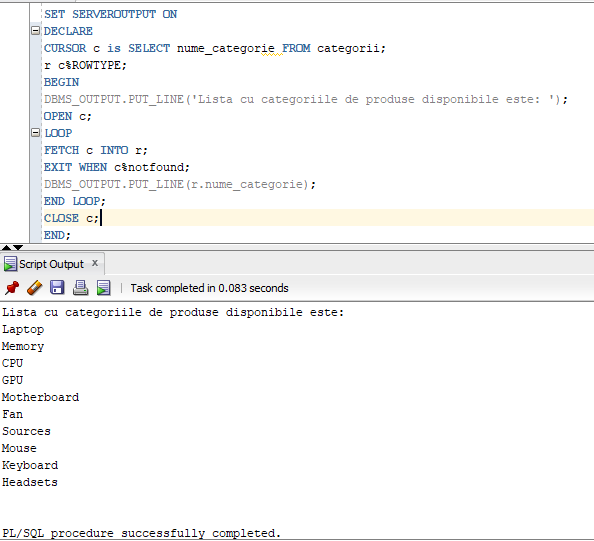
DBMS\_OUTPUT.PUT\_LINE(r.nume\_categorie);

END LOOP;

CLOSE c;

END;

/



## 2.8 Un cursor cu parametru

Folosind un cursor ce primește drept parametru tipul plății comenzilor, se vor afișa comenzile ce au fost plătite folosind plata cash și prin card.

SET SERVEROUTPUT ON

DECLARE

CURSOR c(plata VARCHAR2) IS SELECT \* FROM comenzi WHERE tip\_plata=plata;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Comenzi cu plata cash: ');

FOR r IN c('cash') LOOP

DBMS\_OUTPUT.PUT\_LINE('Comanda cu id-ul '||r.id\_comanda||', livrata in data de '||r.livrare);

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Comenzi cu plata card: ');

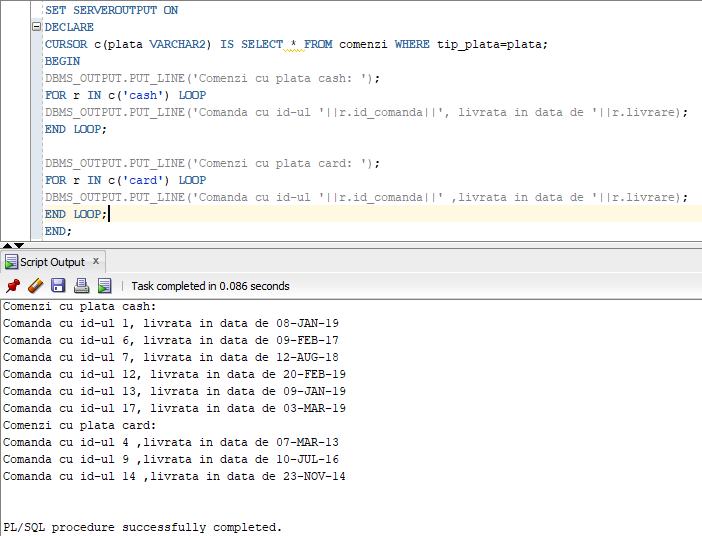
FOR r IN c('card') LOOP

DBMS\_OUTPUT.PUT\_LINE('Comanda cu id-ul '||r.id\_comanda||' ,livrata in data de '||r.livrare);

END LOOP;

END;

/



## 2.9 Tratarea unei excepții predefinite

Se încearcă afișarea adresei de mail a clientului cu numele de familie Ionescu. În cazul în care există mai multe persoane ce au acest nume, se va trata excepția predefinită apărută din această cauză.

SET SERVEROUTPUT ON

DECLARE

v\_email clienti.email%type;

BEGIN

SELECT email INTO v\_email FROM clienti WHERE nume\_client='Ionescu';

DBMS\_OUTPUT.PUT\_LINE('Adresa de mail a clientului cu numele Ionescu este: '||v\_email);

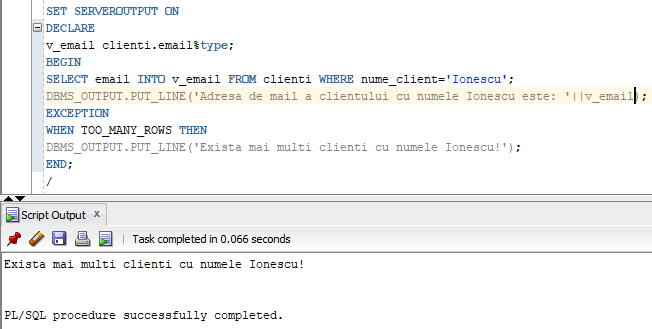
EXCEPTION

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Exista mai multi clienti cu numele Ionescu!');

END;

/



## 2.10 Tratarea unei excepții non predefinite

Se încearcă mărirea salariului angajatului cu id-ul 15. Dacă acest lucru nu reușește se tratează excepția.

SET SERVEROUTPUT ON

DECLARE

excep EXCEPTION;

BEGIN

UPDATE angajati

SET salariu=salariu+100

WHERE id\_angajat=70;

IF sql%rowcount=0 THEN

RAISE excep;

END IF;

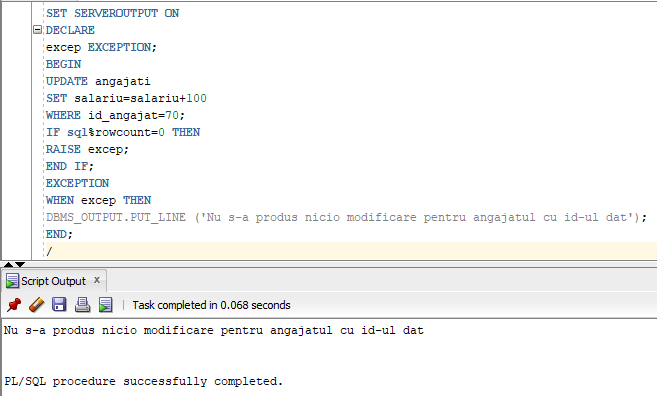
EXCEPTION

WHEN excep THEN

DBMS\_OUTPUT.PUT\_LINE ('Nu s-a produs nicio modificare pentru angajatul cu id-ul dat');

END;

/



## 2.11 O procedură

Se creează o procedură ce permite prin majorarea cu un anumit procent salariul unui angajat, în funcție de id-ul său (id-ul angajatului și procentul sunt parametri de tip IN).

CREATE OR REPLACE PROCEDURE majorare\_salariu

(p\_id IN angajati.id\_angajat%type, procent IN number)

IS

v\_sal angajati.salariu%type;

BEGIN

SELECT salariu INTO v\_sal FROM angajati WHERE id\_angajat=p\_id;

DBMS\_OUTPUT.PUT\_LINE ('Angajatul are salariul: '||v\_sal);

UPDATE angajati

SET salariu=salariu\*(1+procent/100)

WHERE id\_angajat=p\_id;

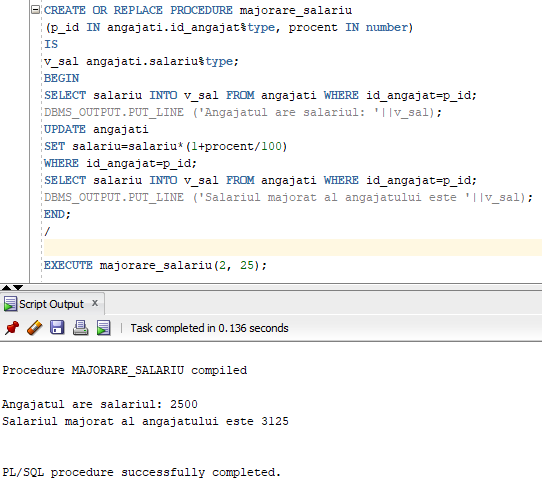
SELECT salariu INTO v\_sal FROM angajati WHERE id\_angajat=p\_id;

DBMS\_OUTPUT.PUT\_LINE ('Salariul majorat al angajatului este '||v\_sal);

END;

/

EXECUTE majorare\_salariu(2, 25);



## 2.12 O funcție

Se creează o funcție care va returna generația corespunzătoare unui produs, în funcție de tipul său.

CREATE OR REPLACE FUNCTION afla\_generatie(p\_tip produse.tip%type)

RETURN NUMBER

IS

v\_gen produse.generatie%type;

BEGIN

SELECT generatie INTO v\_gen FROM produse

WHERE tip = p\_tip;

RETURN v\_gen;

END;

/

DECLARE

v\_genprod NUMBER(3);

BEGIN

v\_genprod:=afla\_generatie('Notebook');

DBMS\_OUTPUT.PUT\_LINE('Produsul de tip notebook este generatia: '|| v\_genprod);

END;

/

## 

## 2.13 Un trigger

Se creează un trigger ce se declanșează înainte de actualizarea tabelei angajați și care nu permite setarea salariului sub nivelul salariului minim (presupunem în acest exemplu 1500).

CREATE OR REPLACE TRIGGER verifica\_sal

BEFORE UPDATE ON angajati

FOR EACH ROW

BEGIN

IF :new.salariu<1500 THEN

:new.salariu := 1500;

END IF;

END;

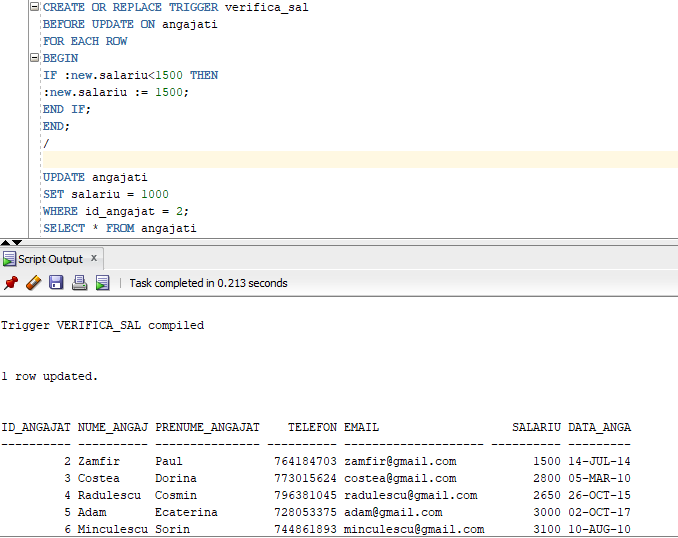
/

UPDATE angajati

SET salariu = 1000

WHERE id\_angajat = 2;

SELECT \* FROM angajați



## 2.14 Un pachet

Se creează un pachet ce conține o funcție ce returnează vechimea unui angajat și o procedură ce calculează salariul mediu al angajaților.

CREATE OR REPLACE PACKAGE pachet\_angajati

AS

PROCEDURE salariu\_mediu

(sal\_mediu OUT NUMBER);

FUNCTION vechime

(p\_id angajati.id\_angajat%type)

RETURN NUMBER;

END;

/

CREATE OR REPLACE PACKAGE BODY pachet\_angajati

AS

PROCEDURE salariu\_mediu

(sal\_mediu OUT NUMBER)

AS

BEGIN

SELECT AVG(salariu) INTO sal\_mediu FROM angajati;

END;

FUNCTION vechime

(p\_id angajati.id\_angajat%type)

RETURN NUMBER

AS

vechime NUMBER;

BEGIN

SELECT(sysdate-data\_angajarii)/365 INTO vechime FROM angajati WHERE id\_angajat=p\_id;

RETURN vechime;

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE ('Nu exista angajatul cu acest id!');

RETURN 0;

END;

END;

/

VARIABLE v\_sal\_mediu NUMBER;

EXECUTE pachet\_angajati.salariu\_mediu(:v\_sal\_mediu);

PRINT v\_sal\_mediu;

SELECT nume\_angajat, prenume\_angajat, pachet\_angajati.vechime(6) AS vechime FROM angajati WHERE id\_angajat=6;