**PROIECT SGBD**

**STUDENT: Ciobanu Alexandra**

**GRUPA**

**SERIA**

**CUPRINS**

[1. STRUCTURI DE CONTROL 3](#_Toc124115543)

[2. CURSORI ȘI EXCEPȚII 7](#_Toc124115544)

[3. SUBPROGRAME 10](#_Toc124115545)

[3.1. FUNCȚII 10](#_Toc124115546)

[3.2. PROCEDURI 13](#_Toc124115547)

# STRUCTURI DE CONTROL

--1. Sa se determine nr de parteneriate pentru angajatul al carui id se introduce de la tastatura

declare

v\_nr number;

v\_id number;

begin

select count(id\_parteneriat1),id\_angajat1 into v\_nr ,v\_id--se incarca o singura valoare in variabilele scalare

from parteneriat1

where id\_angajat1=&id\_angajat1

group by id\_angajat1;

dbms\_output.put\_line('Nr parteneriate pentru angajatul cu id '||v\_id ||': '||v\_nr);

end;

Graphical user interface, text, chat or text message

Description automatically generated

--2. Cu ocazia sarbatorilor, preturile se vor reduce astfel:

-- pentru preturile mai mici de 500 => o reducere de 5%

-- pentru cele intre 500 si 2500 => o reducere de 15%

-- pentru cele peste 2500=> o reducere de 30%

DECLARE

v\_pret lista\_preturi1.pret%TYPE;

v\_min NUMBER;

v\_max NUMBER;

BEGIN

SELECT MIN(id\_pret1), MAX(id\_pret1)

INTO v\_min, v\_max

FROM lista\_preturi1;

WHILE v\_min <= v\_max LOOP

SELECT pret INTO v\_pret

FROM lista\_preturi1

WHERE id\_pret1 = v\_min;

CASE

WHEN v\_pret <500 THEN v\_pret:=v\_pret\*0.95;

WHEN v\_pret BETWEEN 500 AND 2500 THEN v\_pret:=v\_pret\*0.85;

ELSE v\_pret:=v\_pret\*0.7;

END CASE;

DBMS\_OUTPUT.PUT\_LINE ('Pretul cu id '|| v\_min || ' va avea pretul '|| v\_pret);

v\_min:= v\_min+1;

END LOOP;

END;WHEN v\_sal BETWEEN 1500 AND 2000 THEN v\_sal:=v\_sal\*1.15;

ELSE v\_sal:=v\_sal\*1.1;

END CASE;

DBMS\_OUTPUT.PUT\_LINE ('Angajatul '|| v\_min || ' va avea salariul '|| v\_sal);

v\_min:= v\_min+1;

END LOOP;

END;

Graphical user interface, text, application

Description automatically generated

-- 3. Dat fiind id-ul unui angajat de la tastatura, sa se afiseze daca acesta este angajat primavara sau nu.

DECLARE

v\_nume angajat1.nume%TYPE;

v\_prenume angajat1.prenume%TYPE;

v\_data DATE;

v\_c angajat1.id\_angajat1%TYPE:= &c;

BEGIN

SELECT nume, prenume, data\_angajare

INTO v\_nume, v\_prenume, v\_data

FROM angajat1

WHERE id\_angajat1= v\_c;

IF (extract(month from v\_data) BETWEEN 3 AND 5)

THEN DBMS\_OUTPUT.PUT\_LINE ('Angajatul '||v\_nume||' '|| v\_prenume || ' a fost angajat primavara.');

ELSE DBMS\_OUTPUT.PUT\_LINE ('Angajatul '||v\_nume||' '|| v\_prenume || ' nu a fost angajat primavara.');

END IF;

END;

Graphical user interface, text, application, email

Description automatically generated

--4. Sa se afiseze preturile din lista mai mari decat 2500 lei, folosind structura repetitiva FOR.

DECLARE

v\_pret NUMBER;

v\_min NUMBER;

v\_max NUMBER;

BEGIN

SELECT MIN(id\_pret1), MAX(id\_pret1)

INTO v\_min, v\_max

FROM lista\_preturi1;

FOR i IN v\_min..v\_max LOOP

SELECT pret

INTO v\_pret

FROM lista\_preturi1

WHERE id\_pret1=i;

IF( v\_pret > 2500 ) THEN

DBMS\_OUTPUT.PUT\_LINE ('Pretul '||i|| ' este - '||v\_pret);

END IF;

END LOOP;

END;

Graphical user interface, text, application, email

Description automatically generated

# CURSORI ȘI EXCEPȚII

Graphical user interface, text, application, email

Description automatically generated

--1. Sa se afiseze primele 5 preturi, ordonate descrescator.

declare

cursor c is select id\_pret1,pret from lista\_preturi1 order by pret desc;

v\_pret\_row c%rowtype;

begin

open c;

loop

fetch c into v\_pret\_row;

exit when c%rowcount=6;

dbms\_output.put\_line('id '||v\_pret\_row.id\_pret1|| ' = '||v\_pret\_row.pret);

end loop;

close c;

end;

/

-- 2. Sa se modifice locatiile din strainatate din tabela locatie1 ca fiind 'EXTERN'. Se trateaza cazul in care nu exista nicio locatie.

begin

for v\_loc in (select id\_locatie1, locatie,zona from locatie1 where zona='strainatate')

loop

update locatie1 set locatie = 'EXTERN'

where id\_locatie1=v\_loc.id\_locatie1;

dbms\_output.put\_line('Locatia '|| v\_loc.locatie || ' are locatia modificata.');

end loop;

exception

when no\_data\_found then dbms\_output.put\_line('Nu exista nicio astfel de locatie.');

end;

rollback; -- pentru a nu pastra modificarile

Graphical user interface, text, application, email

Description automatically generated

-- 3. Sa se afiseze parteneriatele din anul dat de la tastatura, cu un cursor explicit si sa se trateze cazul in care nu exista.

declare

cursor c\_parteneriat is select id\_parteneriat1, data from parteneriat1;

begin

for i in c\_parteneriat loop

if (extract(year from i.data) = &data ) then

dbms\_output.put\_line(i.id\_parteneriat1|| ' ' || i.data);

end if;

end loop;

exception

when NO\_DATA\_FOUND then

dbms\_output.put\_line('Nu exista parteneriat in acest an!');

end;

Graphical user interface, text, application, email

Description automatically generated

-- 4. Sa se construiasca un bloc PLSQL prin care sa se afiseze pentru fiecare elev: numele complet, cnp, si datele parintelui.

declare

cursor c is select nume||prenume as nume\_complet, cnp, telefon\_parinte, mail\_parinte from elev1;

begin

for v\_compusa in c

loop

exit when c%notfound;

dbms\_output.put\_line(v\_compusa.nume\_complet||' '||v\_compusa.cnp||' '||v\_compusa.telefon\_parinte||' '||v\_compusa.mail\_parinte);

end loop;

end;

Graphical user interface, text, application

Description automatically generated

# SUBPROGRAME

## 3.1. FUNCȚII

-- 1. Sa se genereze o functie care afiseaza suma preturilor pentru locatiile cu id-ul introdus de la tastatura.

-- Sa se trateze cazul in care nu exista pret pentru locatia cu id ul precizat.

create or replace function pret\_total

(p\_id in number)

return number

is

cursor c is select SUM(pret) from lista\_preturi1 where id\_locatie1=p\_id;

v\_pret NUMBER;

begin

open c;

fetch c into v\_pret;

close c;

RETURN v\_pret;

exception

when NO\_DATA\_FOUND then RETURN 0;

end;

DECLARE

v\_pret NUMBER;

v\_id NUMBER:=&id;

BEGIN

v\_pret:= pret\_total(v\_id);

if v\_pret =0 then DBMS\_OUTPUT.PUT\_LINE('Nu exista pret pentru locatia ceruta.' );

else DBMS\_OUTPUT.PUT\_LINE('Pretul total pentru locatia '|| v\_id || ' este '|| v\_pret );

end if;

END;

Graphical user interface, text, application, email

Description automatically generated

--2. Generati o functie care se calculeze numarul de ani impliniti de la data angajarii unui angajat al carui id este dat ca parametru de intrare. Tratati cazul în care angajatul indicat nu exista.

create or replace function calcul\_vechime(p\_id in angajat1.id\_angajat1%type)

return number

as

vechime number;

begin

select round((sysdate-data\_angajare)/365,0) into vechime from angajat1 where id\_angajat1=p\_id;

return vechime;

exception

when no\_data\_found then return -1;

end;

/

execute dbms\_output.put\_line(calcul\_vechime(3));

Graphical user interface, text, application, email

Description automatically generated

-- 3. Sa se construiasca functia care verifica daca o locatie este autohtona sau nu.

create or replace function autohton

(p\_id IN number)

RETURN boolean

IS

v\_autohton locatie1.zona%type;

begin

select zona into v\_autohton from locatie1 where id\_locatie1=p\_id;

if v\_autohton = 'tara' then return true;

else return false;

end if;

exception

when no\_data\_found then return NULL;

end;

--apel

declare

v\_id number := &id;

begin

if (autohton(v\_id)) IS null then dbms\_output.put\_line('Locatia nu exista!');

elsif (autohton(v\_id)) then dbms\_output.put\_line('Locatia este autohtona.');

else dbms\_output.put\_line('Locatia nu este autohtona');

end if;

end;

Graphical user interface, text, application, email

Description automatically generated

## 3.2. PROCEDURI

-- 1. Sa se creeze o procedura care sa insereze un elev.

create or replace procedure adauga\_elev

(

p\_nume elev1.nume%type,

p\_prenume elev1.prenume%type,

p\_cnp elev1.cnp%type,

p\_tel elev1.telefon\_parinte%type,

p\_email elev1.mail\_parinte%type)

IS

p\_id number := 11;

begin

select max(id\_elev1) into p\_id from elev1;

insert into elev1

values (p\_id+1, p\_nume, p\_prenume, p\_cnp, p\_tel, p\_email);

end;

begin

adauga\_elev('Anghel', 'Ion', '5010320300222', '0723356665', 'ang.marius@gmail.com');

end;

**Graphical user interface, text, application, email

Description automatically generated**

--2. Sa se genereze o procedura de marire a preturilor prin care sa se mareasca cu o anumita valoare pretul din lista. Sa se trateze printr-o exceptie cazul în care nu exista pretul.

create or replace procedure marire\_pret(x number, p\_id number)

as

v\_pret lista\_preturi1.pret%TYPE;

begin

select pret into v\_pret from lista\_preturi1 where id\_pret1=p\_id;

dbms\_output.put\_line('Pretul vechi, cu id '||p\_id||' :' ||v\_pret);

update lista\_preturi1

set pret=pret+x

where id\_pret1=p\_id;

select pret into v\_pret from lista\_preturi1 where id\_pret1=p\_id;

dbms\_output.put\_line('Pretul nou, cu id '||p\_id||' ' ||v\_pret);

exception

when no\_data\_found then dbms\_output.put\_line('Pretul nu exista!');

end;

/

Graphical user interface, text, application, email

Description automatically generatedexecute marire\_pret(1000,1);

-- 3. Sa se scrie o procedura care sa afiseze urmatorul id valabil pentru tabela angajati. Aceasta va fi utila la inserarea unei noi inregistrari.

create or replace procedure next\_id\_ang

IS

p\_id number;

begin

select max(id\_angajat1) into p\_id from angajat1;

p\_id := p\_id+1;

dbms\_output.put\_line('Urmatorul id poate fi: '||p\_id);

end;

begin

next\_val();

end;

Graphical user interface, text, application, email

Description automatically generated

-- 4. Parintele unui elev si-a schimbat numarul de telefon. Sa se scrie o procedura care modifica numarul de telefon al parintelui unui elev cu id primit ca parametru.

create or replace procedure actualizare\_telefon

(p\_id number, p\_telefon elev1.telefon\_parinte%type)

is

v\_tel elev1.telefon\_parinte%type;

begin

select telefon\_parinte into v\_tel from elev1 where id\_elev1=p\_id;

dbms\_output.put\_line('Telefonul vechi, al parintelui elevului cu id '||p\_id||' :' ||v\_tel);

update elev1

set telefon\_parinte=p\_telefon

where id\_elev1=p\_id;

select telefon\_parinte into v\_tel from elev1 where id\_elev1=p\_id;

dbms\_output.put\_line('Telefonul nou, al parintelui elevului cu id '||p\_id||' :' ||v\_tel);

exception

when no\_data\_found then dbms\_output.put\_line('Elevul nu exista!');

end;

/

execute actualizare\_telefon(&id, '0799456333');

Graphical user interface, text, application

Description automatically generated