

# Proiect Elemente Avansate de Baze de Date

## Lanț de magazine

Student: Moț Carina-Elena

Grupa: 1

Am dezvoltat acest proiect pe baza proiectului de anul trecut, din cadrul materiei „Baze de date”. În cadrul acestor proiecte vom putea regăsi activitatea din cadrul unui lanț de magazine. Acest lanț de magazine este compus din magazinele din ERA, Carrefour, Auchan, Lidl, Kaufland, Casa Rusu și Noriel, care își desfășoară activitatea în mai multe județe ale țării. Fiecare magazin are un număr de angajați, care dețin o anumită funcție și un număr de clienți fideli.

În cadrul acestui proiect am modificat tabela magazine, pentru a adăuga un atribut de tip CLOB( Character Large Object), cu ajutorul căruia câmpul denumire din cadrul tabelului magazine va beneficia de o limită mai mare de caractere.

```
CREATE TABLE MAGAZINE(  
Nr_inreg_mag NUMBER(6) PRIMARY KEY,  
Denumire CLOB NOT NULL,  
Adresa VARCHAR2(20),  
Oras VARCHAR2(20),  
Judet VARCHAR2(20),  
Fax NUMBER(6),  
Cod_pds NUMBER(6) REFERENCES PRODUCE(Cod)  
);
```

The screenshot displays the Oracle Application Express interface. A 'Table Finder' window is open, showing the structure of the 'MAGAZINE' table in the 'RO\_A694\_PLSQL\_S18' schema. The table has 15 rows. The columns and their properties are as follows:

Column	Data Type	Length	Precision	Scale	Nullable
NR_INREG_MAG	NUMBER	22	6	0	No
DENUMIRE	CLOB	4000	-	-	No
ADRESA	VARCHAR2	20	-	-	Yes
ORAS	VARCHAR2	20	-	-	Yes
JUDET	VARCHAR2	20	-	-	Yes
FAX	NUMBER	22	6	0	Yes
COD_PDS	NUMBER	22	6	0	Yes

A SQL command window is also visible, showing the following query:

```
select  
NR_INREG_MAG,  
DENUMIRE,  
ADRESA,  
ORAS,  
JUDET,  
FAX,  
COD_PDS  
from MAGAZINE
```

Am creat o tabelă ang care va conține datele angajaților care au fost șterși din baza de date.

```
Create table ang(  
id_angajat INT PRIMARY KEY,  
nume VArchar(50),  
prenume Varchar(50),  
telefon INT,  
email varchar(50),  
magazin varchar(50) );
```

### ***Proceduri:***

1. Procedura numită datePersA este una dintre procedurile publice existente în acest proiect și are ca scop inserarea noilor angajați, iar procedura privată ContributiiStat are ca scop calcularea sumei contribuției pe care fiecare angajat este obligat să o dea.

```
CREATE OR REPLACE PROCEDURE ContributiiStat(p_salar IN angajati.salariu%TYPE) IS-  
-privata
```

```
BEGIN
```

```
    DBMS_OUTPUT.PUT_LINE('Salariul este'|| p_salar-0.25);
```

```
END ContributiiStat;
```

```
CREATE OR REPLACE PROCEDURE datePersA -publica
```

```
(p_id IN angajati.id_angajat%TYPE,
```

```
p_nume IN angajati.nume%TYPE,
```

```
p_prenume IN angajati.prenume%TYPE,
```

```
p_functie IN angajati.functie%TYPE,
```

```
p_salariu IN angajati.salariu%TYPE,
```

```

p_data_angajarii IN angajati.data_angajarii%TYPE,

p_id_drr IN angajati.id_drr%TYPE,

p_nr_inreg_mag_mgn IN angajati.nr_inreg_mag_mgn%TYPE)

IS

BEGIN

    INSERT INTO angajati
VALUES(p_id,p_nume,p_prenume,p functie,p_salariu,p_data_angajarii,p_id_drr,p_nr_inreg_m
ag_mgn);

    DBMS_OUTPUT.PUT_LINE('Angajatul a fost adaugat cu succes!');

    ContributiiStat(p_salariu);

END datePersA;

```

The screenshot shows the Oracle SQL Workshop interface. The 'SQL Commands' tab is active, displaying the following PL/SQL code:

```

CREATE OR REPLACE PROCEDURE ContributiiStat(p_salariu IN angajati.salariu%TYPE) IS--privata
BEGIN
    DBMS_OUTPUT.PUT_LINE('Salariul este'|| p_salariu*0.25);
END ContributiiStat;

CREATE OR REPLACE PROCEDURE datePersA --publica
(p_id IN angajati.id_angajati%TYPE,
p_nume IN angajati.nume%TYPE,
p_prenume IN angajati.prenume%TYPE,
p_functie IN angajati.functie%TYPE,
p_salariu IN angajati.salariu%TYPE,
p_data_angajarii IN angajati.data_angajarii%TYPE,
p_id_drr IN angajati.id_drr%TYPE,
p_nr_inreg_mag_mgn IN angajati.nr_inreg_mag_mgn%TYPE)
IS
BEGIN
    INSERT INTO angajati VALUES(p_id,p_nume,p_prenume,p_functie,p_salariu,p_data_angajarii,p_id_drr,p_nr_inreg_mag_mgn);
    DBMS_OUTPUT.PUT_LINE('Angajatul a fost adaugat cu succes!');
    ContributiiStat(p_salariu);
END datePersA;

```

Below the code editor, the 'Results' tab shows the message 'Procedure created.' and the execution time '0.01 seconds'.

2. Procedura publica strgereA va efectua stergerea angajatului al cărui id corespunde cu parametrul transmis, în timp ce procedura privată log\_angg va insera în tabela ang, id-ul angajatului care a fost șters din baza de date.

```
CREATE OR REPLACE PROCEDURE log_angg(p_id IN angajati.id_angajat%TYPE)--  
privata
```

```
IS BEGIN
```

```
INSERT INTO ang(id_angajat) VALUES(p_id);
```

```
END log_angg;
```

```
CREATE OR REPLACE PROCEDURE stergereA(p_id_angajat in  
angajati.id_angajat%TYPE)--publica
```

```
IS
```

```
BEGIN
```

```
DELETE FROM angajati
```

```
Where id_angajat=p_id_angajat;
```

```
log_angg(p_id_angajat);
```

```
END stergereA;
```

The screenshot displays the Oracle SQL Workshop interface in a web browser. The top navigation bar includes 'ORACLE Application Express', 'Application Builder', 'SQL Workshop' (selected), 'Team Development', and 'Packaged Apps'. The user 'Carina-Elena' is logged in. The 'SQL Commands' tab is active, showing a schema of 'RQ\_A694\_PLSQL\_S18'. The SQL editor contains the following code:

```
CREATE OR REPLACE PROCEDURE log_ang(p_id IN angajati.id_angajat%TYPE)--privata  
IS  
BEGIN  
INSERT INTO ang(id_angajat) VALUES(p_id);  
END log_ang;  
CREATE OR REPLACE PROCEDURE stergereA(p_id_angajat in angajati.id_angajat%TYPE)--publica  
IS  
BEGIN  
DELETE FROM angajati  
Where id_angajat=p_id_angajat;  
log_ang(p_id_angajat);  
END stergereA;
```

Below the editor, the 'Results' tab shows the message 'Procedure created.' and the execution time '0.01 seconds'. The bottom status bar indicates 'Copyright © 1996, 2015, Oracle. All rights reserved.' and the date '1/7/2020'.

## ***Funcții:***

Funcția privată stoc are rolul de a returna stocul fiecărui magazin în parte, în timp ce funcția publică getName, are rol în a regăsi magazinul care are id-ul egal cu id-ul furnizat.

```
CREATE OR REPLACE FUNCTION stoc(f_maga IN magazine.nr_inreg_mag%TYPE)
```

```
RETURN INTEGER
```

```
IS
```

```
CURSOR c_stoc (nr_mag NUMBER) IS
```

```
select COUNT(*)
```

```
from magazine
```

```
where nr_inreg_mag=nr_mag;
```

```
c_magazin NUMBER;
```

```
BEGIN
```

```
OPEN c_stoc(f_maga);
```

```
LOOP
```

```
FETCH c_stoc INTO c_magazin;
```

```
EXIT WHEN c_stoc%NOTFOUND;
```

```
END LOOP;
```

```
RETURN c_magazin;
```

```
CLOSE c_stoc;
```

```
END;
```

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'ORACLE Application Express', 'Application Builder', 'SQL Workshop', 'Team Development', and 'Packaged Apps'. The user is logged in as 'Carina-Elena'. The main area displays the 'SQL Commands' tab with a schema of 'RO\_A694\_PLSQL\_S18'. The command window contains the following PL/SQL code:

```
CREATE OR REPLACE FUNCTION stoc(f_maga IN magazine.nr_inreg_mag%TYPE)
RETURN INTEGER
IS
CURSOR c_stoc (nr_mag NUMBER) IS
select COUNT(*)
from magazine
where nr_inreg_mag=nr_mag;

c_magazin NUMBER;
BEGIN
OPEN c_stoc(f_maga);
LOOP
FETCH c_stoc INTO c_magazin;
EXIT WHEN c_stoc%NOTFOUND;
END LOOP;
RETURN c_magazin;
CLOSE c_stoc;
END;

SELECT nr_inreg_mag, stoc(nr_inreg_mag) AS Stoc
FROM magazine
WHERE oras = 'Oradea';
```

Below the code, the 'Results' tab shows the execution output:

NR_INREG_MAG	STOC
111111	1
333333	1

2 rows returned in 0.01 seconds. The bottom status bar shows the Oracle version '19c\_1904\_plsql\_s18' and the date '1/12/2020'.

CREATE OR REPLACE FUNCTION getName(f\_nr IN magazine.nr\_inreg\_mag%TYPE)

RETURN magazine.denumire%TYPE

IS

CURSOR c\_mag IS

SELECT m.denumire

FROM magazine m

WHERE m.nr\_inreg\_mag=f\_nr;

c\_denumire magazine.denumire%TYPE;

v\_stoc NUMBER;

BEGIN

OPEN c\_mag;

FETCH c\_mag INTO c\_denumire;

IF stoc(f\_nr)>0 THEN

DBMS\_OUTPUT.PUT\_LINE("DA");

END IF;

IF c\_mag%NOTFOUND THEN

c\_denumire:='error';

END IF;

CLOSE c\_mag;

RETURN c\_denumire;

EXCEPTION

WHEN OTHERS THEN

raise\_application\_error(-20001,'An error was encountered-'||SQLCODE||'-ERROR-'||SQLERRM);

END getName;

The screenshot displays the Oracle SQL Workshop interface. The top navigation bar includes 'ORACLE Application Express', 'Application Builder', 'SQL Workshop', 'Team Development', and 'Packaged Apps'. The 'SQL Commands' tab is active, showing a SQL script for a function named 'getName'. The script defines a cursor 'c\_mag' to select magazine details and includes error handling with 'raise\_application\_error'. Below the script, the 'Results' tab shows the output of the function for two magazine IDs: 111111 (ERA) and 333333 (Auchan). The interface also includes a 'Bottom Splitter' and a status bar at the bottom indicating the application version and date.

```
CREATE OR REPLACE FUNCTION getName(f_nr IN magazine.nr_inreg_mag%TYPE)
RETURN magazine.denumire%TYPE
IS
CURSOR c_mag IS
SELECT m.denumire
FROM magazine m
WHERE m.nr_inreg_mag=f_nr;

c_denumire magazine.denumire%TYPE;
v_stoc NUMBER;
BEGIN
OPEN c_mag;
FETCH c_mag INTO c_denumire;

IF c_mag%NOTFOUND THEN
c_denumire:='error';
END IF;
CLOSE c_mag;
RETURN c_denumire;
EXCEPTION
WHEN OTHERS THEN
raise_application_error(-20001,'An error was encountered-'||SQLCODE||'-ERROR-'||SQLERRM);
END getName;

SELECT nr_inreg_mag, getName(nr_inreg_mag) AS course_id
FROM magazine
WHERE nicks = 'Oracle';
```

NR_INREG_MAG	COURSE_ID
111111	ERA
333333	Auchan

2 rows returned in 0.02 seconds



## ***Declanșatori:***

1. Acest declanșator are rolul de a insera în tabela istoric vechiul preț, înainte ca acesta să fie actualizat.

```
CREATE or REPLACE TRIGGER istoric_pret_trigger
```

```
BEFORE UPDATE OF pret
```

```
ON produse
```

```
FOR EACH ROW
```

```
BEGIN
```

```
INSERT INTO istoric_pret
```

```
VALUES
```

```
(:old.cod,
```

```
:old.denumire,
```

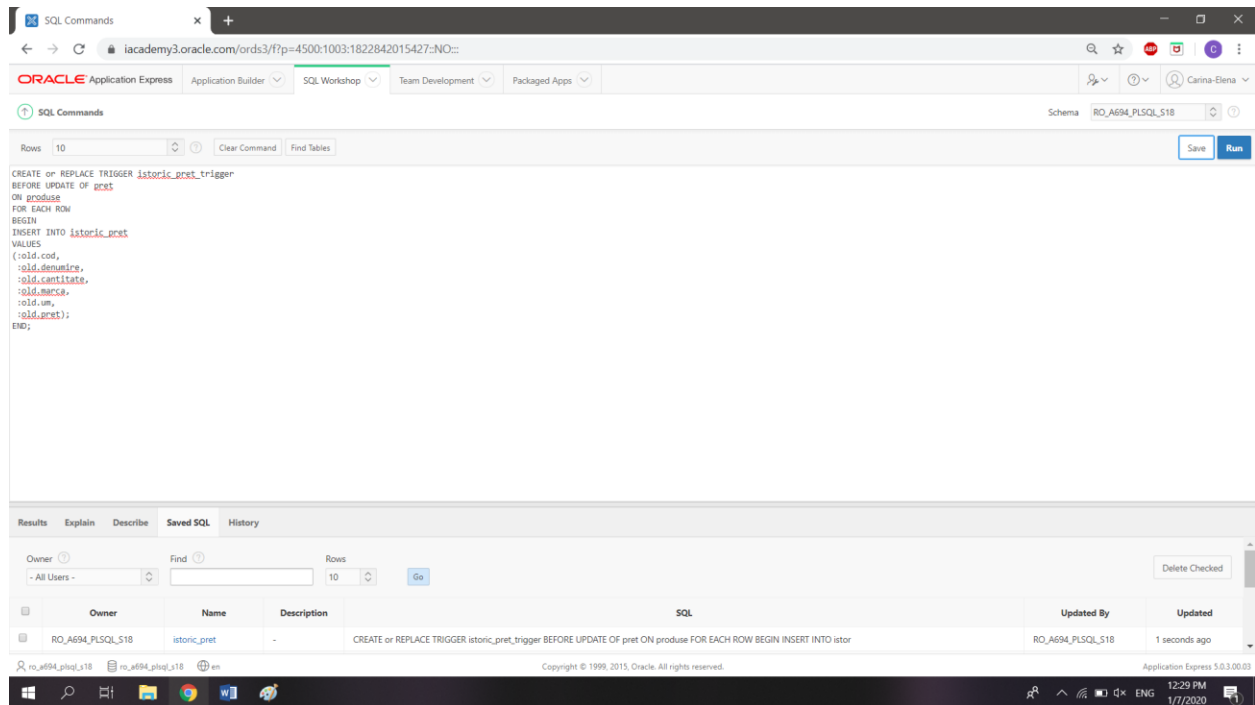
```
:old.cantitate,
```

```
:old.marca,
```

```
:old.um,
```

```
:old.pret);
```

```
END;
```



2. Acest declanșator va afișa vechiul salariu, noul salariu și diferența dintre acestea înainte ca în tabela angajați să se facă modificările DML.

CREATE OR REPLACE TRIGGER salarii

BEFORE DELETE OR INSERT OR UPDATE ON angajati

FOR EACH ROW

WHEN (new.id\_angajat > 0)

DECLARE

sal\_diff number;

BEGIN

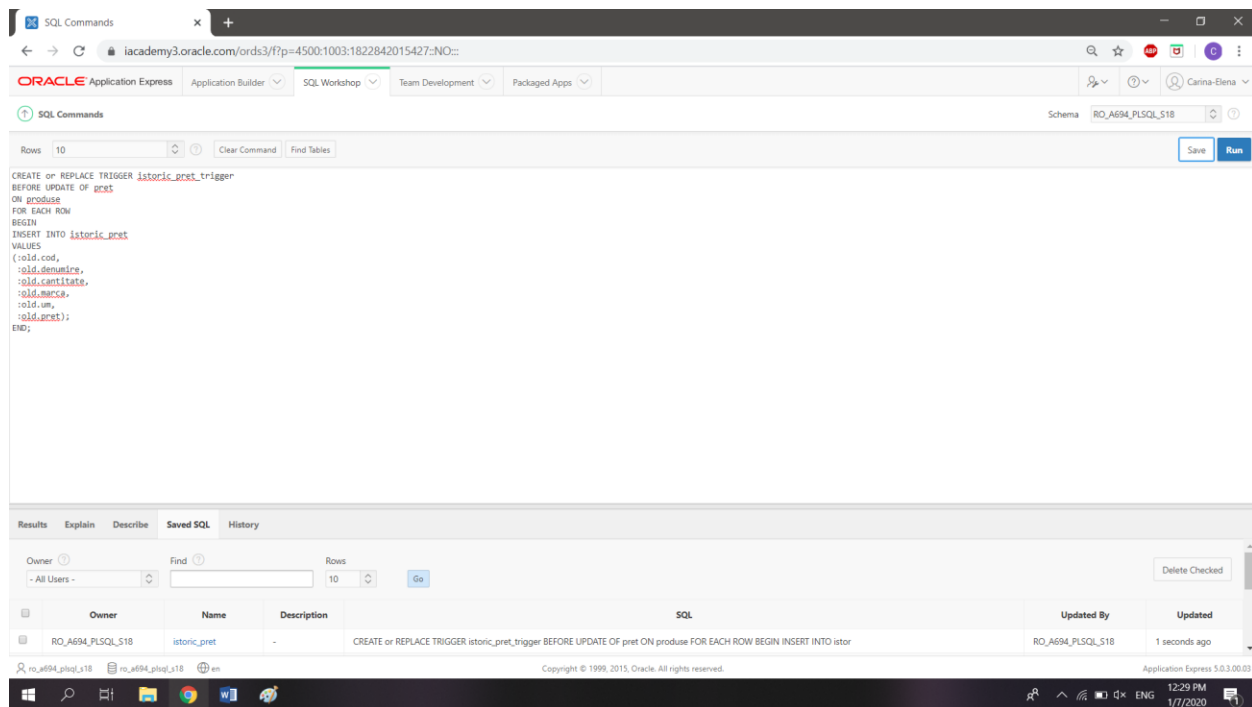
sal\_diff := :new.salariu - :old.salariu;

dbms\_output.put('Old salary: ' || :old.salariu);

dbms\_output.put(' New salary: ' || :new.salariu);

dbms\_output.put\_line(' Difference ' || sal\_diff);

END;



- Acest declanșator face ca limita inferioară a salariului să fie de 2000 de lei, iar dacă această condiție nu este îndeplinită, se va declanșa o eroare.

CREATE OR REPLACE TRIGGER restrictionare\_salar

BEFORE INSERT OR UPDATE OF salariu ON angajati FOR EACH ROW

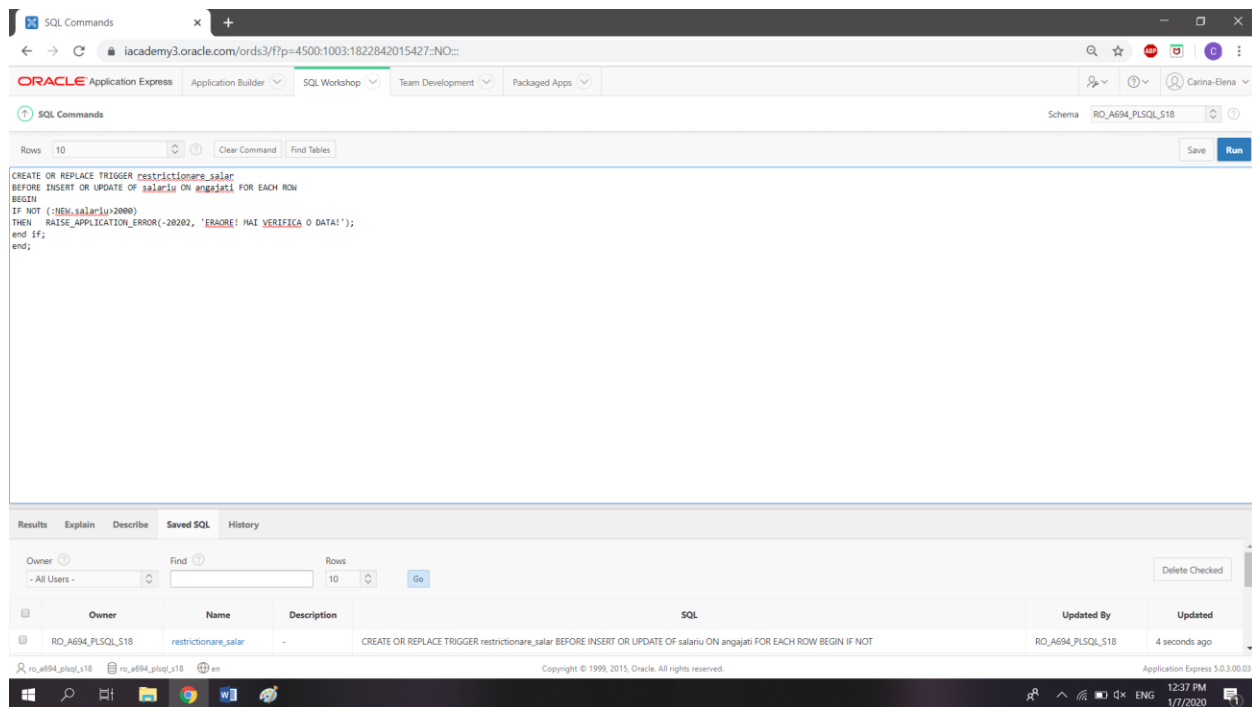
BEGIN

IF NOT (:NEW.salariu>2000)

THEN RAISE\_APPLICATION\_ERROR(-20202, 'ERAORE! MAI VERIFICA O DATA!');

end if;

end;



4. Acest declanșator va insera în tabela log\_clienti id-ul, numele, prenumele și telefonul fiecărui client după fiecare inserare a unui client nou în tabela clienti.

CREATE OR REPLACE TRIGGER log\_clienti\_trigger

AFTER INSERT ON clienti

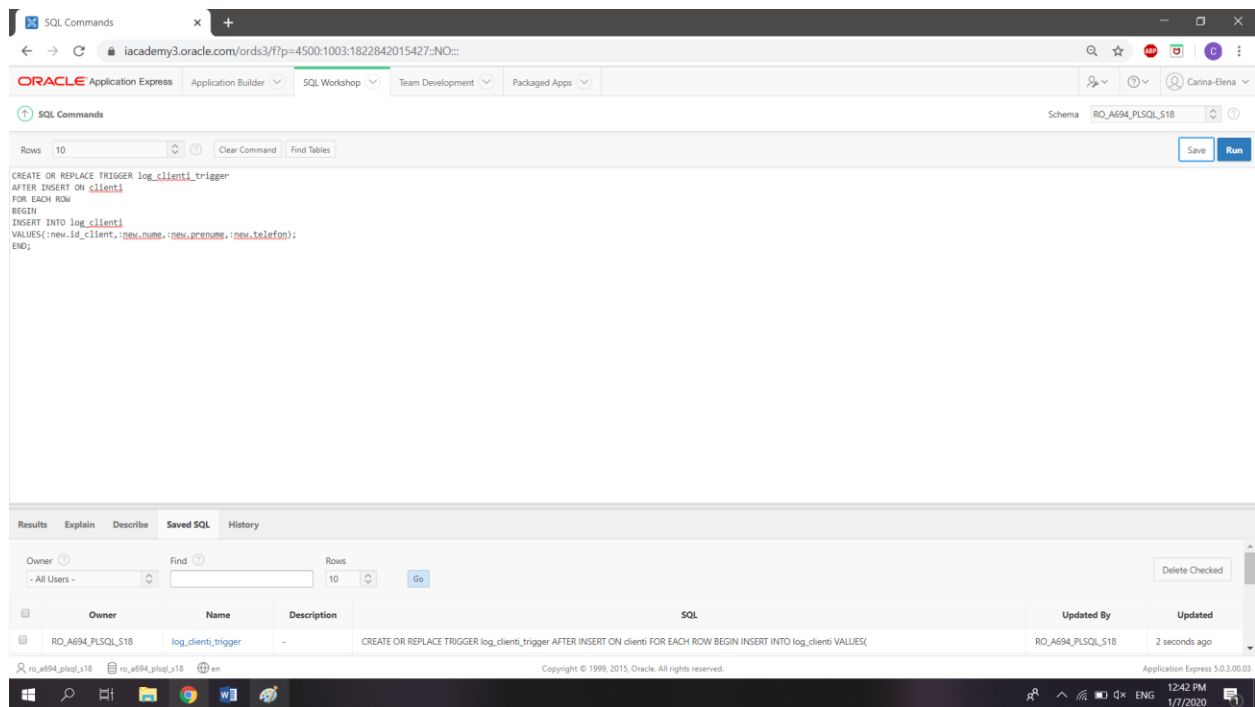
FOR EACH ROW

BEGIN

INSERT INTO log\_clienti

VALUES(:new.id\_client,:new.nume,:new.prenume,:new.telefon);

END;



## Secvențe:

1.

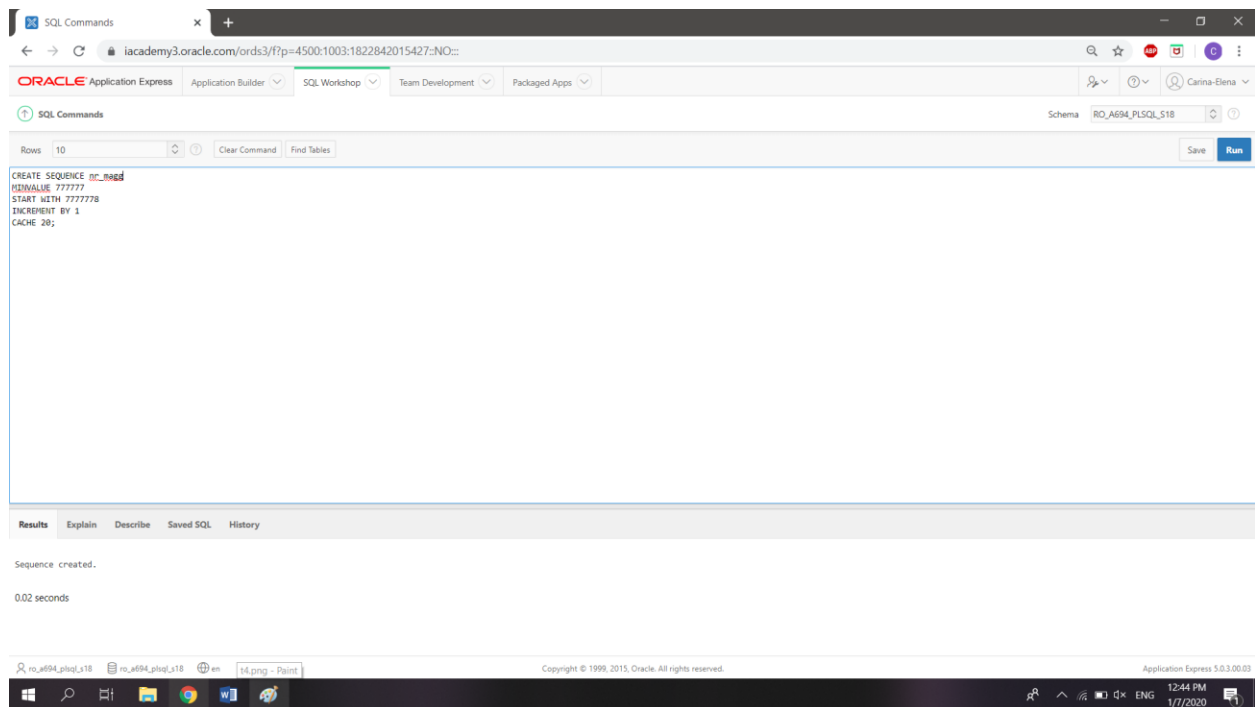
CREATE SEQUENCE nr\_magg

MINVALUE 777777

START WITH 7777778

INCREMENT BY 1

CACHE 20;



2.

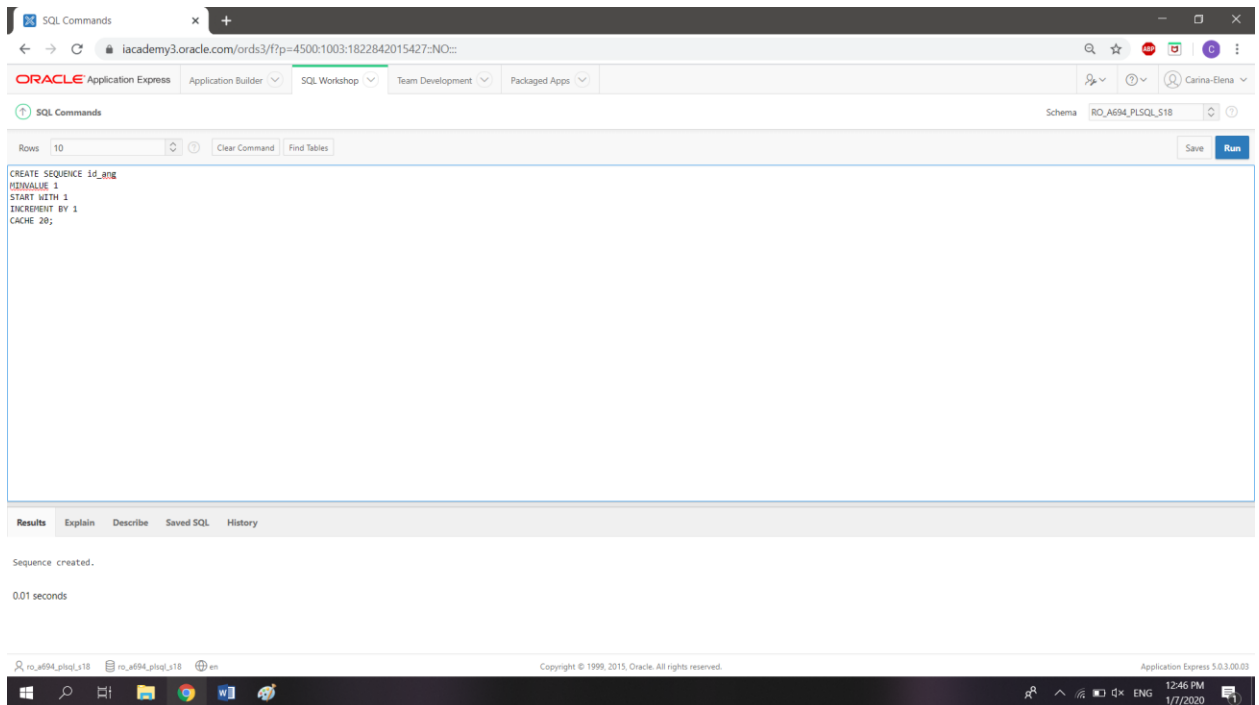
CREATE SEQUENCE id\_ang

MINVALUE 1

START WITH 1

INCREMENT BY 1

CACHE 20;



3.

CREATE SEQUENCE id\_clienti

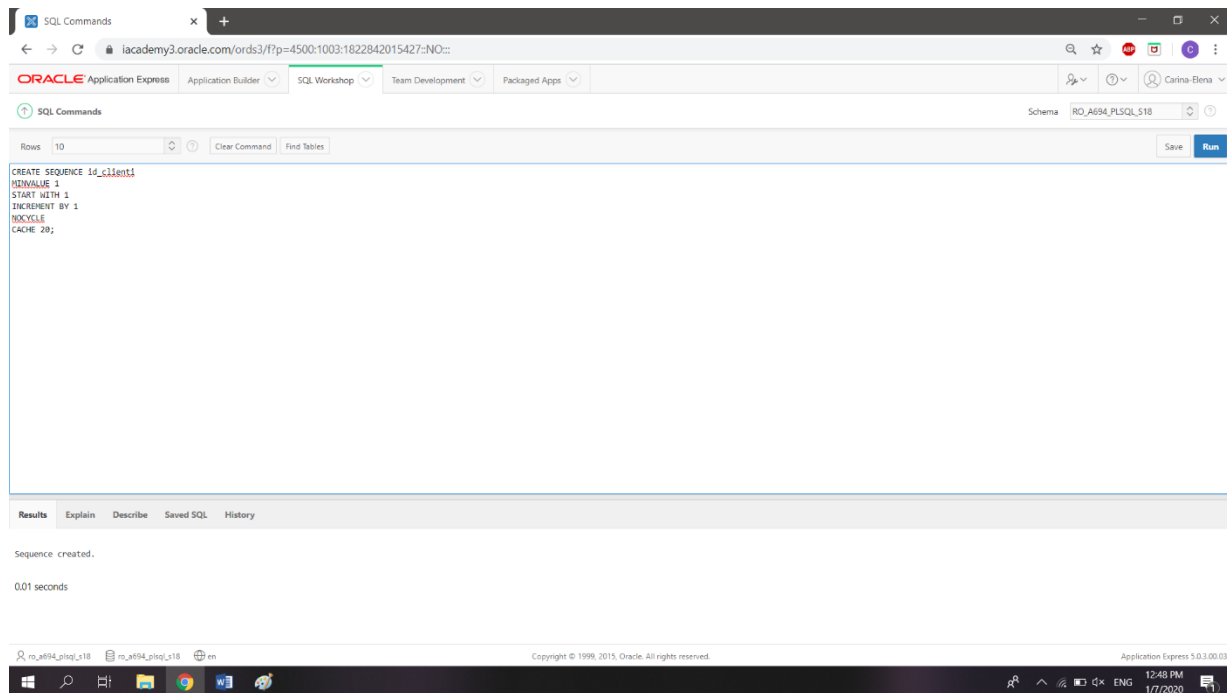
MINVALUE 1

START WITH 1

INCREMENT BY 1

NOCYCLE

CACHE 20;



## ***Package:***

CREATE OR REPLACE PACKAGE proiect AS

PROCEDURE datePersA(p\_id IN angajati.id\_angajat%TYPE, p\_nume IN angajati.nume%TYPE, p\_prename IN angajati.prename%TYPE, p\_functie IN angajati.functie%TYPE, p\_salariu IN angajati.salariu%TYPE, p\_data\_angajarii IN angajati.data\_angajarii%TYPE, p\_id\_drr IN angajati.id\_drr%TYPE, p\_nr\_inreg\_mag\_mgn IN angajati.nr\_inreg\_mag\_mgn%TYPE);

PROCEDURE stergereA(p\_id\_angajat in angajati.id\_angajat%TYPE);

FUNCTION getName(f\_nr IN magazine.nr\_inreg\_mag%TYPE) RETURN magazine.denumire%TYPE;

END proiect;

CREATE OR REPLACE PACKAGE BODY proiect AS

PROCEDURE ContributiiStat(p\_salar IN angajati.salariu%TYPE) IS

BEGIN



```

        DBMS_OUTPUT.PUT_LINE('Salariul este'|| p_salar-0.25);

END ContributiiStat;


PROCEDURE datePersA(p_id IN  angajati.id_angajat%TYPE, p_nume IN
angajati.numename%TYPE, p_prename IN  angajati.prename%TYPE, p_functie IN
angajati.functie%TYPE, p_salariu IN  angajati.salariu%TYPE, p_data_angajarii IN
angajati.data_angajarii%TYPE, p_id_drr IN  angajati.id_drr%TYPE, p_nr_inreg_mag_mgn IN
angajati.nr_inreg_mag_mgn%TYPE)

IS

BEGIN

    INSERT INTO angajati
VALUES(p_id,p_nume,p_prename,p_functie,p_salariu,p_data_angajarii,p_id_drr,p_nr_inreg_m
ag_mgn);

    DBMS_OUTPUT.PUT_LINE('Angajatul a fost adaugat cu succes!');

    ContributiiStat(p_salariu);

END datePersA;


PROCEDURE log_angg(p_id IN angajati.id_angajat%TYPE)

IS BEGIN

INSERT INTO ang(id_angajat) VALUES(p_id);

END log_angg;

PROCEDURE stergereA(p_id_angajat in angajati.id_angajat%TYPE)

IS

BEGIN

DELETE FROM angajati

Where id_angajat=p_id_angajat;

log_angg(p_id_angajat);

```

```

END stergereA;

CREATE OR REPLACE FUNCTION stoc(f_maga IN magazine.nr_inreg_mag%TYPE)
RETURN INTEGER
IS
CURSOR c_stoc (nr_mag NUMBER) IS
select COUNT(*)
from magazine
where nr_inreg_mag=nr_mag;

c_magazin NUMBER;
BEGIN
OPEN c_stoc(f_maga);
LOOP
FETCH c_stoc INTO c_magazin;
EXIT WHEN c_stoc%NOTFOUND;
END LOOP;
RETURN c_magazin;
CLOSE c_stoc;
END;

FUNCTION getName(f_nr IN magazine.nr_inreg_mag%TYPE)
RETURN magazine.denumire%TYPE
IS
CURSOR c_mag IS
SELECT m.denumire

```

FROM magazine m

WHERE m.nr\_inreg\_mag=f\_nr;

c\_denumire magazine.denumire%TYPE;

v\_stoc NUMBER;

BEGIN

OPEN c\_mag;

LOOP

FETCH c\_mag INTO c\_denumire;

EXIT WHEN c\_mag%NOTFOUND;

END LOOP;

CLOSE c\_mag;

RETURN c\_denumire;

END getName;

END proiect;