**PROIECT SGBD ORACLE**

**-GESTIUNEA UNUI LANT DE BIBLIOTECI-**

Fusneica Florentin-Cristian, grupa 241

**1. DESCRIEREA PROBLEMEI ECONOMICE ALESE**

**OBIECTIVUL PROIECTULUI**

Prin prisma acestui proiect, am ales sa prezint demersul unui lant de biblioteci, amplasate in trei locatii.

Astfel, proiectul are ca obiectiv principal analiza activitatii bibliotecilor, gestionand relatia dintre angajati, carti si clienti.

**DESCRIEREA BAZEI DE DATE**

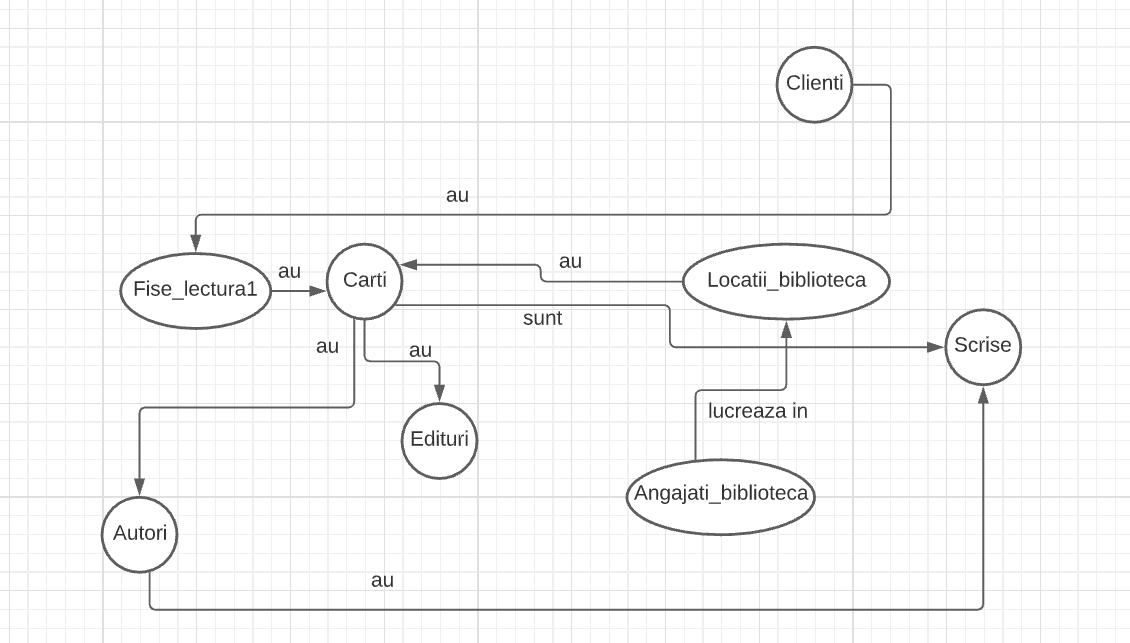
Baza de date consta in opt tabele:

1. CARTI: contine cartile care apar in fiecare biblioteca; tabela ofera informatii despre acestea; fiecare carte are cod, nume,autor si apartine unui domeniu.
2. FISE\_LECTURA1: constituie fisa fiecarui cititor, continand date atat despre cel ce imprumuta cartea, cat si despre carte.
3. LOCATII\_BIBLIOTECA: arata toate locatiile bibliotecilor; unde sunt amplasate acestea
4. CITITORI: tabela contine informatii despre persoana care imprumuta o carte (nume, prenume, adresa,…)
5. ANGAJATI\_BIBLIOTECA:ofera informatii cu legatura la angajatii bibliotecii
6. SCRISE: tabela arata cartile scrise de un anumit autor
7. EDITURI: editurile cartilor de la biblioteca
8. AUTORI: tabela contine informatii referitoare la autorii cartilor (nume, prenume,..)

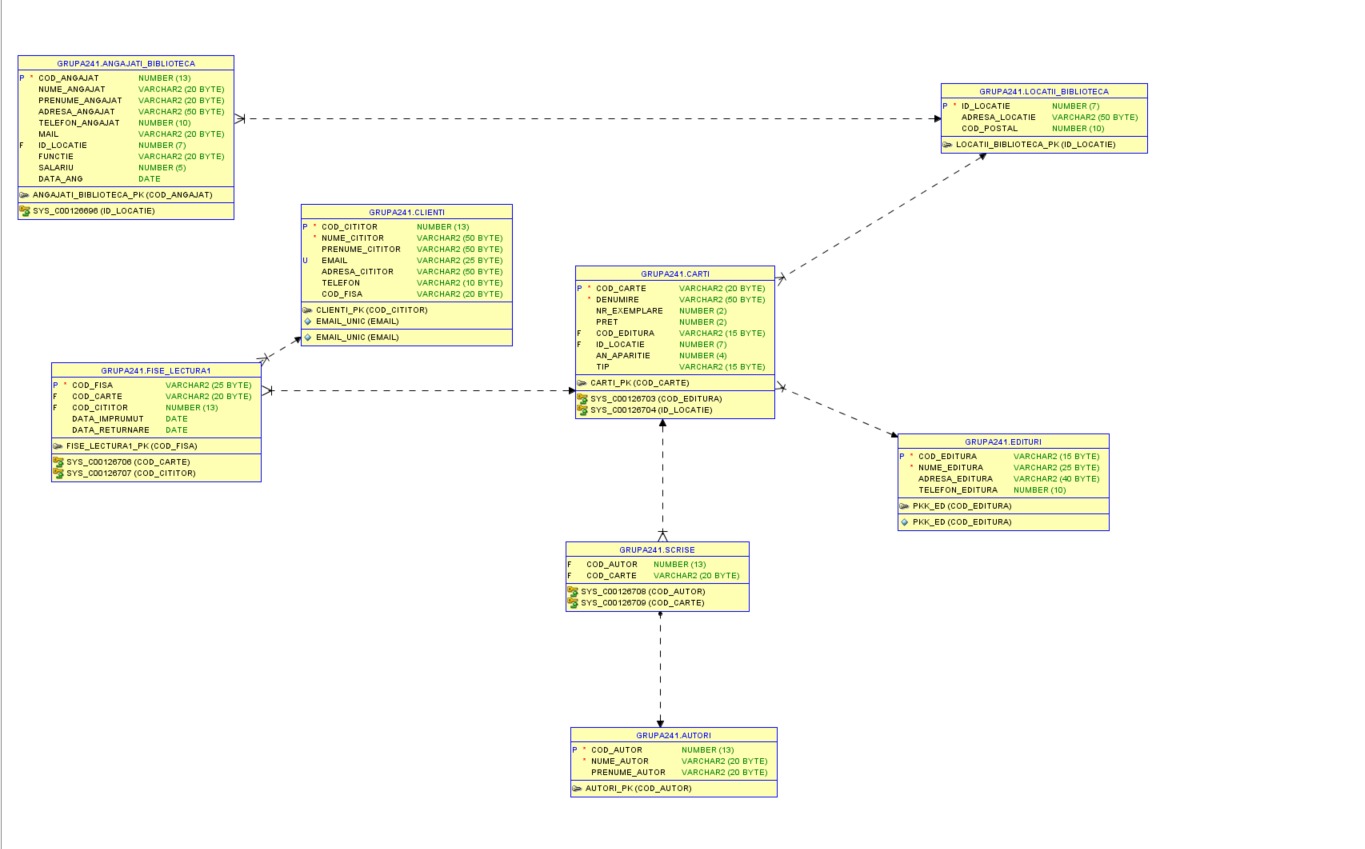
|  |  |  |  |
| --- | --- | --- | --- |
| TABELA | ATRIBUTE | TIP DATA | RESTRICTII |

|  |  |  |  |
| --- | --- | --- | --- |
| EDITURI | cod\_editura  nume\_editura  adresa\_editura  telefon\_editura | VARCHAR2(25)  VARCHAR(15)  VARCHAR2(40)  NUMBER(10) | PRIMARY KEY  NOT NULL |
| CARTI | cod\_carte  denumire  nr\_exemplare  pret  an\_aparitie  tip  cod\_editura  id\_locatie | NUMBER(20)  VARCHAR2(50)  NUMBER(2)  NUMBER(2)  NUMBER(4)  VARCHAR2(25)  VARCHAR2(25)  NUMBER(7) | PRIMARY KEY  NOT NULL  FOREIGN KEY  FOREIGN KEY |
| CLIENTI | cod\_cititor  nume\_cititor  prenume\_cititor  adresa\_mail  telefon  adresa  cod\_fisa | NUMBER(13)  VARCHAR2(20)  VARCHAR2(20)  VARCHAR2(25),  NUMBER(10)  VARCHAR2(10)  VARCHAR(25) | PRIMARY  Key  NOT NULL  NOT NULL  CHECK UNIQUE  FOREIGN KEY |
| FISE\_LECTURA | cod\_fisa  cod\_carte  data\_imprumut  data\_returnare | VARCHAR(25)  NUMBER(20)  DATE  DATE | PRIMARY KEY  FOREIGN KEY  NOT NULL |
| AUTORI | id\_autor  nume\_autor  prenume\_autor | NUMBER(13)  VARCHAR2(20)  VARCHAR2(20) | PRIMARY KEY |
| SCRISE | id\_autor  cod\_carte | NUMBER(13)  NUMBER(20) | FOREIGN KEYS |
| LOCATII\_BIBLIOTECA | id\_locatie  adresa\_locatie  cod\_postal | NUMBER(7)  VARCHAR2(50)  NUMBER(10) | PRIMARY KEY |
| ANGAJATI | cod\_angajat  nume\_angajat  prenume\_angajat  adresa\_angajat  telefon  mail\_angajat  salariu  id\_locatie  functie | NUMBER(13)  VARCHAR2(20)  VARCHAR2(20)  VARCHAR2(50)  NUMBER(10)  VARCHAR2(20)  NUMBER(4)  NUMBER(3) | PRIMARY KEY  NOT NULL  NOT NULL  CHECK UNIQUE  CHECK |

**2.DIAGRAMA CONCEPTUALA**

****

**3.DIAGRAMA ERD**

****

**4.IMPLEMENTAREA DIAGRAMEI CONCEPTUALE**

* Crearea tabelelor mentionate anterior in care sa fie precizate restrictiile de integritate.

CREATE TABLE EDITURI

(cod\_editura VARCHAR2(15) CONSTRAINT pkk\_ed PRIMARY KEY,

nume\_editura VARCHAR(25) NOT NULL,

adresa\_editura VARCHAR2(40),

telefon\_editura NUMBER(10));

ALTER TABLE EDITURI

ADD CONSTRAINT ED\_NUME\_NN1 CHECK("NUME\_EDITURA" IS NOT NULL);

CREATE TABLE LOCATII\_BIBLIOTECA

(id\_locatie NUMBER(7) PRIMARY KEY,

adresa\_locatie VARCHAR2(50) ,

cod\_postal NUMBER(10));

CREATE TABLE CARTI

(cod\_carte VARCHAR2(20) primary key,

denumire VARCHAR2(50) NOT NULL,

nr\_exemplare NUMBER(2),

pret NUMBER(2),

cod\_editura REFERENCES EDITURI(cod\_editura),

id\_locatie references locatii\_biblioteca(id\_locatie),

an\_aparitie number(4));

ALTER TABLE CARTI

ADD CONSTRAINT CARTE\_NUME\_N1 CHECK("DENUMIRE" IS NOT NULL);

ALTER TABLE CARTI ADD TIP VARCHAR(15);

CREATE TABLE AUTORI

(cod\_autor NUMBER(13) PRIMARY KEY,

nume\_autor VARCHAR2(20) ,

prenume\_autor VARCHAR2(20));

ALTER TABLE AUTORI

ADD CONSTRAINT AUT\_NUME\_N1 CHECK("NUME\_AUTOR" IS NOT NULL);

CREATE TABLE SCRISE

(cod\_autor REFERENCES autori(cod\_autor),

cod\_carte REFERENCES carti(cod\_carte));

CREATE TABLE ANGAJATI\_BIBLIOTECA

(cod\_angajat NUMBER(13) PRIMARY KEY,

nume\_angajat VARCHAR2(20) ,

prenume\_angajat VARCHAR2(20),

adresa\_angajat VARCHAR2(50),

telefon\_angajat NUMBER(10),

mail VARCHAR2(20),

id\_locatie REFERENCES locatii\_biblioteca(id\_locatie));

alter table angajati\_biblioteca add functie varchar(20);

alter table angajati\_biblioteca add salariu number(5);

alter table angajati\_biblioteca add data\_ang date;

CREATE TABLE CLIENTI

(cod\_cititor NUMBER(13) PRIMARY KEY,

nume\_cititor VARCHAR2(50) ,

prenume\_cititor VARCHAR2(50),

email VARCHAR2(25),

adresa\_cititor VARCHAR2(50),

telefon VARCHAR2(10));

ALTER TABLE CLIENTI

ADD CONSTRAINT CIT\_NUME\_N1 CHECK("NUME\_CITITOR" IS NOT NULL);

ALTER TABLE CLIENTI

ADD CONSTRAINT format\_email CHECK(email LIKE '%@%.\_\_');

ALTER TABLE CLIENTI

ADD CONSTRAINT email\_unic UNIQUE(email);

CREATE TABLE FISE\_LECTURA1

(cod\_fisa VARCHAR(25) PRIMARY KEY,

cod\_carte REFERENCES CARTI(cod\_carte),

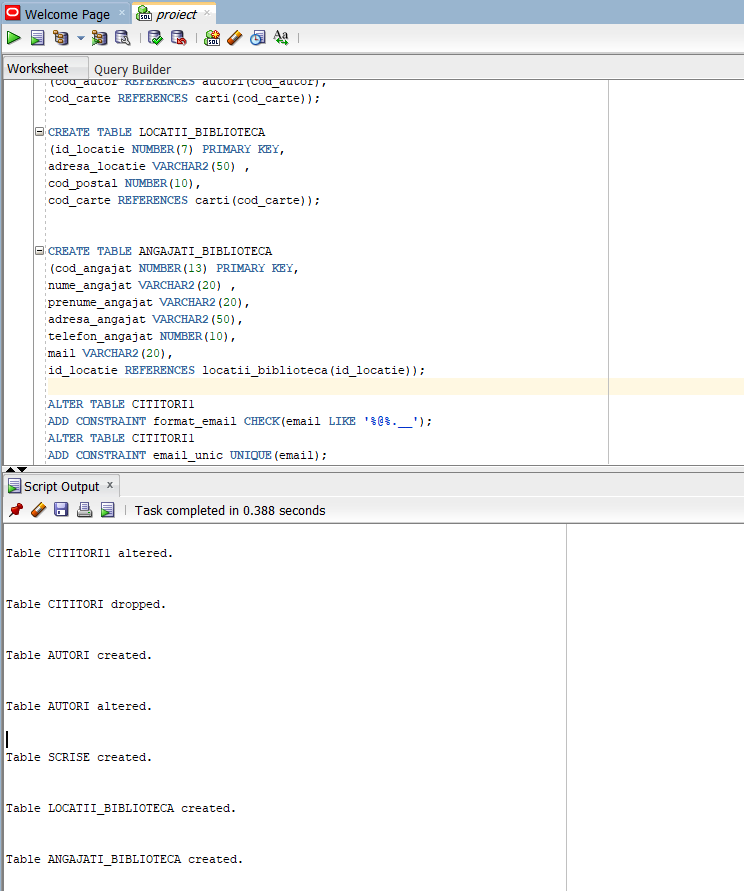
cod\_cititor references clienti(cod\_cititor),

data\_imprumut DATE,

data\_returnare DATE DEFAULT SYSDATE);

ALTER TABLE CLIENTI

ADD (COD\_FISA VARCHAR2(25)REFERENCES FISE\_LECTURA1(COD\_FISA));



**5.ADAUGAREA INFORMATIILOR IN TABELE**

INSERT INTO AUTORI VALUES(25678543,'Verne','Jules');

INSERT INTO AUTORI VALUES(34747584,'Eliade','Mircea');

INSERT INTO AUTORI VALUES(27583623,'Eminescu','Mihai');

INSERT INTO AUTORI VALUES(37353327,'Cartarescu','Mircea');

INSERT INTO AUTORI VALUES(37387637,'Creanga','Ion');

INSERT INTO AUTORI VALUES(35363627,'Nabokov','Vladimir');

INSERT INTO AUTORI VALUES(37459027,'Proust','Marcel');

INSERT INTO AUTORI VALUES(30593827,'Joyce','James');

INSERT INTO AUTORI VALUES(30483927,'Woolf','Virginia');

INSERT INTO AUTORI VALUES(36083627,'Marquez','Gabriel');

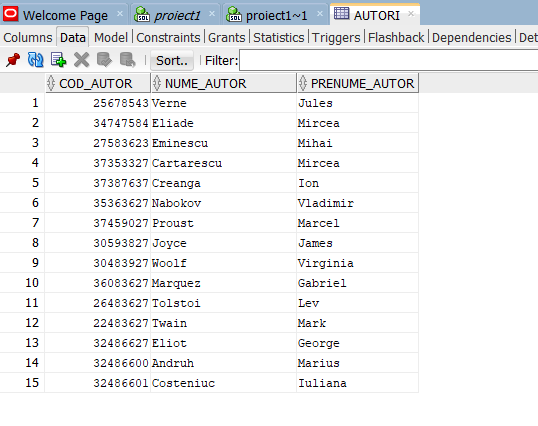
INSERT INTO AUTORI VALUES(26483627,'Tolstoi','Lev');

INSERT INTO AUTORI VALUES(22483627,'Twain','Mark');

INSERT INTO AUTORI VALUES(32486627,'Eliot','George');

INSERT INTO AUTORI VALUES(32486600,'Andruh','Marius');

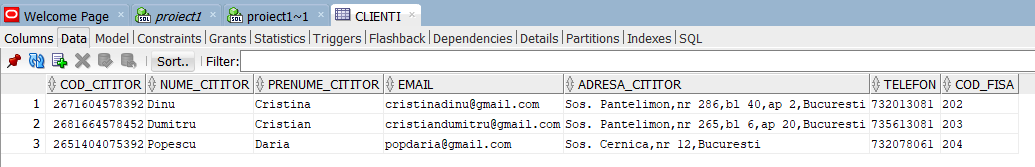
INSERT INTO AUTORI VALUES(32486601,'Costeniuc','Iuliana');



INSERT INTO CLIENTI VALUES(2671604578392,'Dinu','Cristina','cristinadinu@gmail.com','Sos. Pantelimon,nr 286,bl 40,ap 2,Bucuresti',0732013081,202);

INSERT INTO CLIENTI VALUES(2681664578452,'Dumitru','Cristian','cristiandumitru@gmail.com','Sos. Pantelimon,nr 265,bl 6,ap 20,Bucuresti',0735613081,203);

INSERT INTO CLIENTI VALUES(2651404075392,'Popescu','Daria','popdaria@gmail.com','Sos. Cernica,nr 12,Bucuresti',0732078061,204);



INSERT INTO EDITURI VALUES('101','Accent','Str. Dimitrie Cantemir,nr. 20,Bucuresti',021745634);

INSERT INTO EDITURI VALUES('102','ALL','Str. Aleea Lotrului,nr. 25,Bucuresti',031745658);

INSERT INTO EDITURI VALUES('103','Art','Str. Aleea Alexandru,nr. 80,Bucuresti',021485694);

INSERT INTO EDITURI VALUES('104','Egmont','Str. Aleea Pajurei,nr. 33,Bucuresti',021215634);

INSERT INTO EDITURI VALUES('105','IDEA','Str. Aleea Teisani,nr. 10,Bucuresti',021747024);

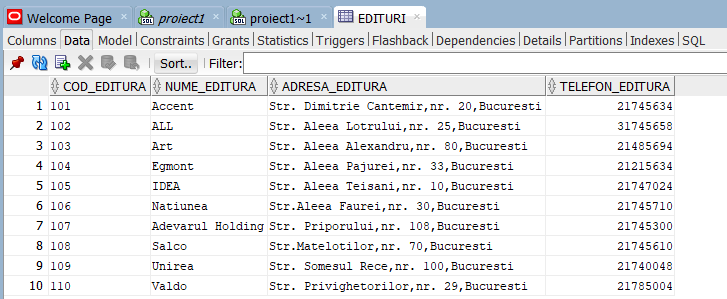
INSERT INTO EDITURI VALUES('106','Natiunea','Str.Aleea Faurei,nr. 30,Bucuresti',021745710);

INSERT INTO EDITURI VALUES('107','Adevarul Holding','Str. Priporului,nr. 108,Bucuresti',021745300);

INSERT INTO EDITURI VALUES('108','Salco','Str.Matelotilor,nr. 70,Bucuresti',021745610);

INSERT INTO EDITURI VALUES('109','Unirea','Str. Somesul Rece,nr. 100,Bucuresti',021740048);

INSERT INTO EDITURI VALUES('110','Valdo','Str. Privighetorilor,nr. 29,Bucuresti',021785004);



INSERT INTO CARTI VALUES('ISBN 120-4590','Ocolul Pamantului in 80 de zile',3,23,107,1873,'FICTIUNE');

INSERT INTO CARTI VALUES('ISBN 103-4570','Copiii Capitanului Grant',5,28,107,186,'FICTIUNE');

INSERT INTO CARTI VALUES('ISBN 193-4767','Amintiri din copilarie',2,30,109,1959,'BIOGRAFIE');

INSERT INTO CARTI VALUES('ISBN 193-0567','Maitreyi',5,26,110,1933,'FICTIUNE');

INSERT INTO CARTI VALUES('ISBN 123-9577','Zece Negri Mititei',3,25,105,1939,'POLITIST');

INSERT INTO CARTI VALUES('ISBN 129-4568','Crima din Orient Express',5,24,105,1934,'POLITIST');

INSERT INTO CARTI VALUES('ISBN 125-9067','Orbitor',2,30,102,2007,'ISTORIC');

INSERT INTO CARTI VALUES('ISBN 123-4576','Nostalgia',6,36,103,1989,'FICTIUNE');

INSERT INTO CARTI VALUES('ISBN 120-4997','Lolita',4,21,108,1955,'FICTIUNE');

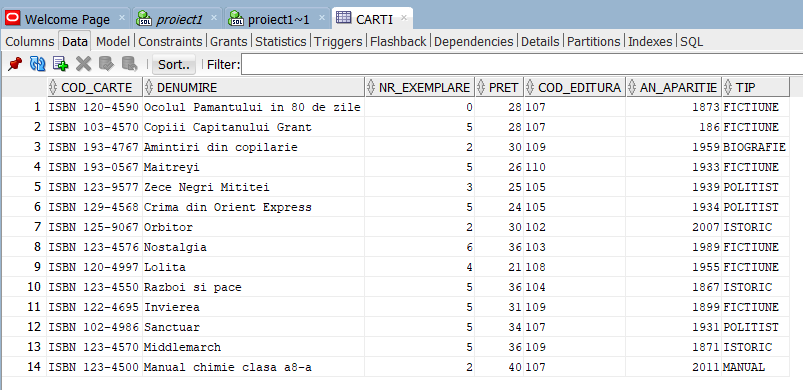
INSERT INTO CARTI VALUES('ISBN 123-4550','Razboi si pace',5,36,104,1867,'ISTORIC');

INSERT INTO CARTI VALUES('ISBN 122-4695','Invierea',5,31,109,1899,'FICTIUNE');

INSERT INTO CARTI VALUES('ISBN 102-4986','Sanctuar',5,34,107,1931,'POLITIST');

INSERT INTO CARTI VALUES('ISBN 123-4570','Middlemarch',5,36,109,1871,'ISTORIC');

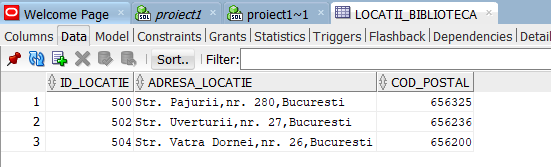
INSERT INTO CARTI VALUES('ISBN 123-4500','Manual chimie clasa a8-a',2,40,107,2011,'MANUAL');



INSERT INTO LOCATII\_BIBLIOTECA VALUES(500,'Str. Pajurii,nr. 280,Bucuresti',656325);

INSERT INTO LOCATII\_BIBLIOTECA VALUES(502,'Str. Uverturii,nr. 27,Bucuresti',656236);

INSERT INTO LOCATII\_BIBLIOTECA VALUES(504,'Str. Vatra Dornei,nr. 26,Bucuresti',656200);



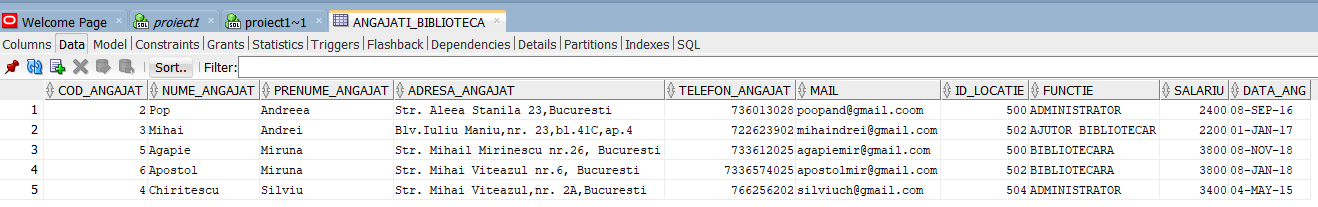
INSERT INTO ANGAJATI\_BIBLIOTECA VALUES(002,'Pop','Andreea','Str. Aleea Stanila 23,Bucuresti',0736013028,'poopand@gmail.coom',500,'ADMINISTRATOR',2400,to\_date('08-09-2016','dd-mm-yyyy'));

INSERT INTO ANGAJATI\_BIBLIOTECA VALUES(003,'Mihai','Andrei','Blv.Iuliu Maniu,nr. 23,bl.41C,ap.4',0722623902,'mihaindrei@gmail.com',502,'AJUTOR BIBLIOTECAR',2000,to\_date('01-01-2017','dd-mm-yyyy'));

INSERT INTO ANGAJATI\_BIBLIOTECA VALUES(004,'Chiritescu','Silviu','Str. Mihai Viteazul,nr. 2A,Bucuresti',0766256202,'silviuch@gmail.com',504,'ADMINISTRATOR',3400,to\_date('04-05-2015','dd-mm-yyyy'));

INSERT INTO ANGAJATI\_BIBLIOTECA VALUES(005,'Agapie','Miruna','Str. Mihail Mirinescu nr.26, Bucuresti',0733612025,'agapiemir@gmail.com',500,'BIBLIOTECARA',1500,to\_date('08-11-2018','dd-mm-yyyy'));

INSERT INTO ANGAJATI\_BIBLIOTECA VALUES(006,'Apostol','Miruna','Str. Mihai Viteazul nr.6, Bucuresti',07336574025,'apostolmir@gmail.com',502,'BIBLIOTECARA',1800,to\_date('08-01-2018','dd-mm-yyyy'));



INSERT INTO SCRISE VALUES(25678543,'ISBN 120-4590');

INSERT INTO SCRISE VALUES(25678543,'ISBN 123-4567');

INSERT INTO SCRISE VALUES(25678543,'ISBN 103-4570');

INSERT INTO SCRISE VALUES(34747584,'ISBN 193-0567');

INSERT INTO SCRISE VALUES(37353327,'ISBN 125-9067');

INSERT INTO SCRISE VALUES(37353327,'ISBN 123-4576');

INSERT INTO SCRISE VALUES(35363627,'ISBN 120-4997');

INSERT INTO SCRISE VALUES(35363627,'ISBN 123-4907');

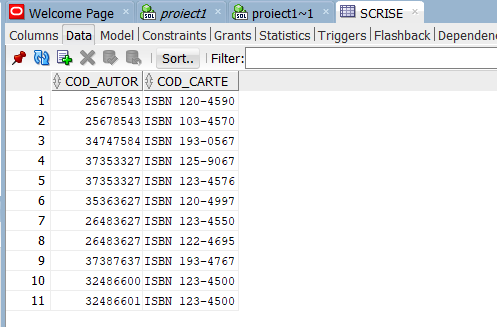
INSERT INTO SCRISE VALUES(32483657,'ISBN 123-4907');

INSERT INTO SCRISE VALUES(36483627,'ISBN 196-4789');

INSERT INTO SCRISE VALUES(32883627,'ISBN 120-4570');

INSERT INTO SCRISE VALUES(32486600,'ISBN 123-4500');

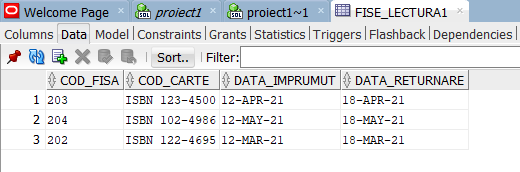
INSERT INTO SCRISE VALUES(32486601,'ISBN 123-4500');



INSERT INTO FISE\_LECTURA1 VALUES('202','ISBN 122-4695',TO\_DATE('12-03-2021','DD-MM-YYYY'),TO\_DATE('18-03-2021','DD-MM-YYYY'));

INSERT INTO FISE\_LECTURA1 VALUES('203','ISBN 123-4500',TO\_DATE('12-04-2021','DD-MM-YYYY'),TO\_DATE('18-04-2021','DD-MM-YYYY'));

INSERT INTO FISE\_LECTURA1 VALUES('204','ISBN 102-4986',TO\_DATE('12-05-2021','DD-MM-YYYY'),TO\_DATE('18-05-2021','DD-MM-YYYY'));



**6.SUBPROGRAM CU INDEX-BY TABLE**

* Creati un subprogram de tip procedura care sa afiseze toate editurile din cadrul bibliotecilor si sa calculeze numarul lor.

set serveroutput on

create or replace procedure nr\_ed

as

TYPE ed\_tab IS TABLE OF edituri%ROWTYPE INDEX BY

PLS\_INTEGER;

v\_ed ed\_tab;

BEGIN

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

FOR i IN 101..110 LOOP

SELECT \* INTO v\_ed(i) FROM edituri WHERE

cod\_editura=i;

END LOOP;

FOR i IN v\_ed.FIRST..v\_ed.LAST LOOP

DBMS\_OUTPUT.PUT\_LINE(v\_ed(i).nume\_editura);

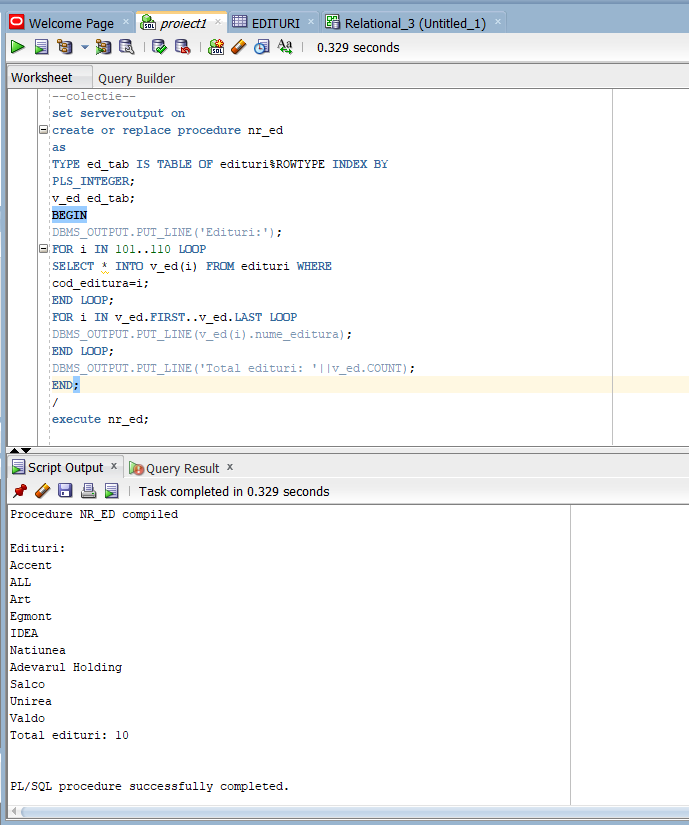
END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Total edituri: '||v\_ed.COUNT);

END;

/

execute nr\_ed;



**7. SUBPROGRAM CARE SA UTILIZEZE UN CURSOR**

* Creati un subprogram de tip procedura care sa afiseze codul si denumirea cartilor de tipul *ISTORIC*, utilizand un cursor explicit.

CREATE OR REPLACE PROCEDURE CARTI\_IST

AS

cursor c is select cod\_carte,denumire from carti

where tip='ISTORIC';

v\_cod carti.cod\_carte%type;

v\_den carti.denumire%type;

begin

open c;

loop

fetch c into v\_cod,v\_den;

exit when c%notfound;

dbms\_output.put\_line('Denumire: ' ||v\_den||';Cod: '||v\_cod);

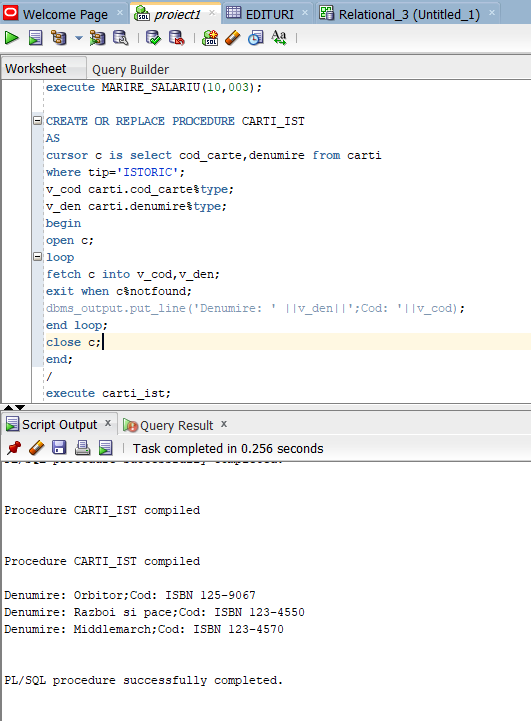
end loop;

close c;

end;

/

execute carti\_ist;



**8.SUBPROGRAM STOCAT DE TIP FUNCTIE CARE SA UTILIZEZE 3 TABELE**

* Creati o functie care sa afiseze codul, denumirea, tipul, informatii despre autor si valoarea totala (pret\*nr\_exemplare) pentru o carte data prin parametru.

set serveroutput on

create or replace function val\_tot(nrc in varchar) return number

as

ex exception;

nre number;

v\_codc scrise.cod\_carte%type;

v\_coda scrise.cod\_autor%type;

v\_den carti.denumire%type;

v\_aut autori.nume\_autor%type;

v\_autp autori.prenume\_autor%type;

v\_tip carti.tip%type;

begin

select s.cod\_carte,c.denumire,s.cod\_autor,a.nume\_autor,a.prenume\_autor,c.tip,sum(c.pret\*c.nr\_exemplare) into v\_codc,v\_den,v\_coda,v\_aut,v\_autp,v\_tip,nre from carti c,autori a, scrise s where

s.cod\_carte=c.cod\_carte and s.cod\_autor=a.cod\_autor and c.cod\_carte=nrc

group by s.cod\_carte,c.denumire,s.cod\_autor,a.nume\_autor,a.prenume\_autor,c.tip ;

dbms\_output.put\_line('Carte:'||v\_den||';valoare totala'||nre||';nume autor:'||v\_aut||';prenume autor:'||v\_autp);

return nre;

if(nre=0)

then raise ex;

end if;

exception

when ex then dbms\_output.put\_line ('Nu sunt exemplare');

end;

/

Apel1:

DECLARE

var NUMBER;

BEGIN

var:=val\_tot('ISBN 120-4590');

IF var>0 THEN

DBMS\_OUTPUT.PUT\_LINE(var);

ELSIF var=0 THEN

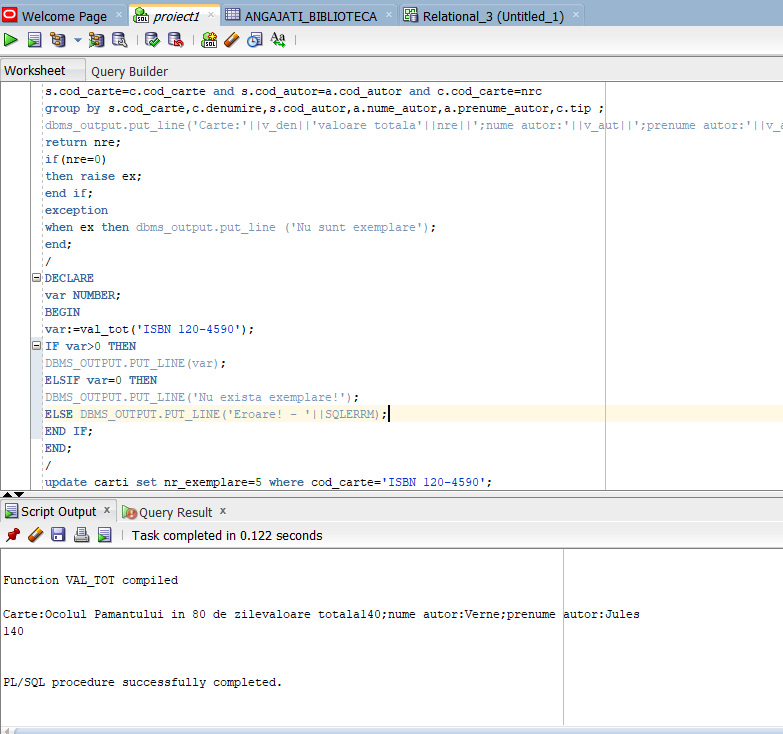
DBMS\_OUTPUT.PUT\_LINE('Nu exista exemplare!');

ELSE DBMS\_OUTPUT.PUT\_LINE('Eroare! - '||SQLERRM);

END IF;

END;

/



Apel2 (numarul de exemplare pentru cartea data prin parametru este setat la 0):

update carti set nr\_exemplare=0 where cod\_carte='ISBN 120-4590';

DECLARE

var NUMBER;

BEGIN

var:=val\_tot('ISBN 120-4590');

IF var>0 THEN

DBMS\_OUTPUT.PUT\_LINE(var);

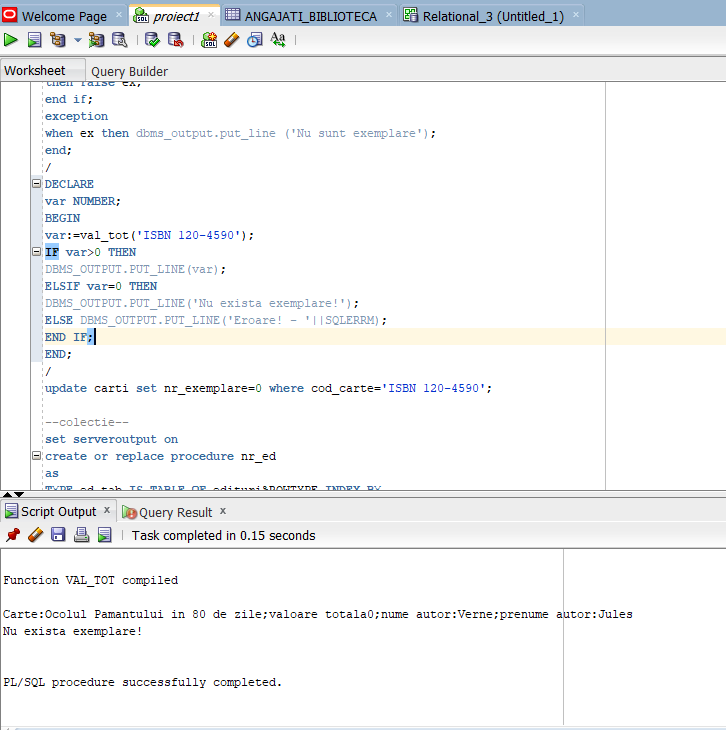
ELSIF var=0 THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista exemplare!');

ELSE DBMS\_OUTPUT.PUT\_LINE('Eroare! - '||SQLERRM);

END IF;

END;

/

**9.SUBPROGRAM DE TIP PROCEDURA CARE SA UTILIZEZE 5 TABELE**

* Creati o procedura care sa afiseze codul, denumirea, tipul, informatii despre autor si editura pentru o carte data prin parametru.Sa se modifice pretul cu un procent dat daca valoarea totala (pret\*nr\_exemplare) a cartii date prin parametru depaseste 150 de lei.

set serveroutput on

create or replace procedure modif\_pret (nrc in varchar,proc in number)

as

ex exception;

nre number;

v\_cod carti.cod\_carte%type;

v\_coda scrise.cod\_autor%type;

v\_den carti.denumire%type;

v\_aut autori.nume\_autor%type;

v\_autp autori.prenume\_autor%type;

v\_tip carti.tip%type;

v\_ed edituri.nume\_editura%type;

begin

select s.cod\_carte,c.denumire,s.cod\_autor,a.nume\_autor,a.prenume\_autor,c.tip,e.nume\_editura,sum(c.pret\*c.nr\_exemplare) into v\_cod,v\_den,v\_coda,v\_aut,v\_autp,v\_tip,v\_ed,nre from carti c,autori a, scrise s,edituri e where

s.cod\_carte=c.cod\_carte and s.cod\_autor=a.cod\_autor and c.cod\_editura=e.cod\_editura and c.cod\_carte=v\_cod

group by s.cod\_carte,c.denumire,s.cod\_autor,a.nume\_autor,a.prenume\_autor,c.tip,e.nume\_editura ;

update carti

set pret=pret\*(1+proc/100)

where cod\_carte=v\_cod and pret\*nr\_exemplare>150;

dbms\_output.put\_line('Carte:'||v\_den||';valoare totala'||nre||';nume autor:'||v\_aut||';prenume autor:'||v\_autp||';editura:'||v\_ed);

if(nre=0)

then raise ex;

end if;

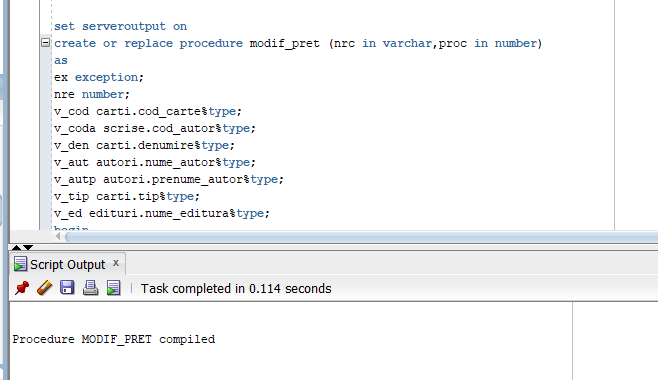
exception

when ex then dbms\_output.put\_line ('Nu exista exemplare');

end;

/

execute modif\_pret('ISBN 123-4567',10);



**10. TRIGGER LMD LA NIVEL DE COMANDA**

* Creati un trigger care sa nu permita stergerea din tabela *CARTI.*

set serveroutput on

create or replace trigger trigger\_stergere

before delete on carti

begin

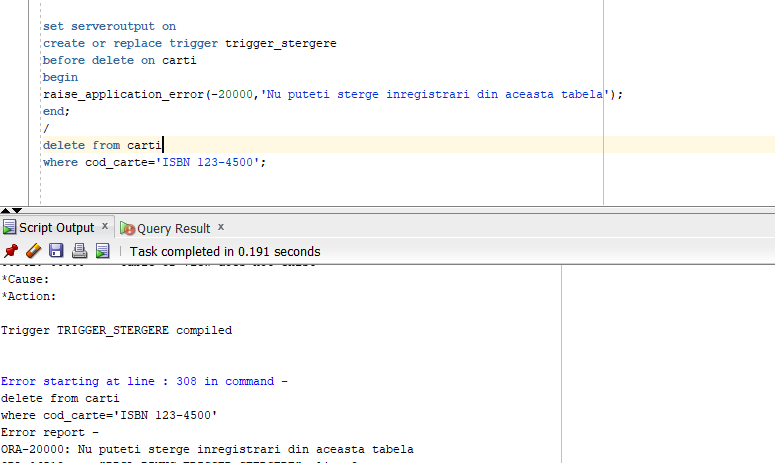
raise\_application\_error(-20000,'Nu puteti sterge inregistrari din aceasta tabela');

end;

/

delete from carti

where cod\_carte='ISBN 123-4500';



* Creati un trigger care sa afiseze modificarile aparute la un update pe tabela ANGAJATI\_BIBLIOTECA referitor la salariul angajatilor cu functia ‘BIBLIOTECARA’.

create or replace trigger trigger\_angajati

BEFORE DELETE OR INSERT OR UPDATE ON angajati\_biblioteca

FOR EACH ROW

WHEN (NEW.functie <> 'BIBILIOTECARA')

DECLARE

diferenta NUMBER;

BEGIN

diferenta := :NEW.salariu - :OLD.salariu;

DBMS\_OUTPUT.PUT(:NEW.nume\_angajat || ': ');

DBMS\_OUTPUT.PUT('Salariul vechi = ' || :OLD.salariu || ', ');

DBMS\_OUTPUT.PUT('Salariul nou = ' || :NEW.salariu || ', ');

DBMS\_OUTPUT.PUT\_LINE('Diferenta: ' || diferenta);

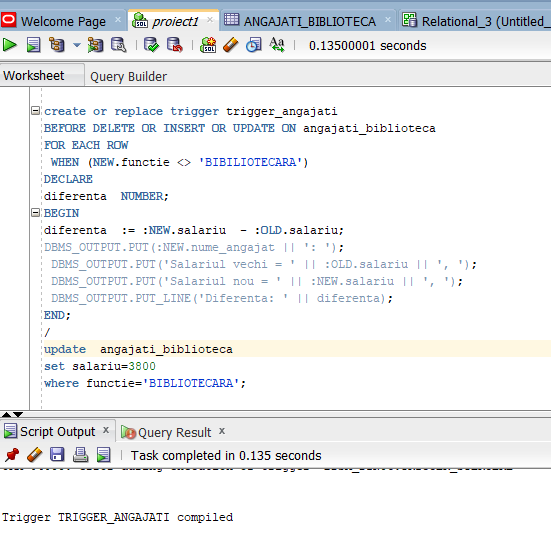
END;

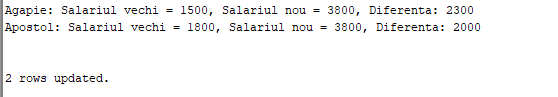
/

update angajati\_biblioteca

set salariu=3800

where functie='BIBLIOTECARA';





**12.TRIGGER LDD**

* Creati un trigger care să introducă date în tabela *Useri* după ce utilizatorul a folosit o comandă LDD.

CREATE TABLE Useri

(user VARCHAR2(20),

nume\_bd VARCHAR2(50),

eveniment VARCHAR2(20),

data DATE);

CREATE OR REPLACE TRIGGER trigger\_ldd

AFTER CREATE OR DROP OR ALTER ON SCHEMA

BEGIN

INSERT INTO useri

VALUES (SYS.LOGIN\_USER, SYS.DATABASE\_NAME, SYS.SYSEVENT,

SYS.DICTIONARY\_OBJ\_NAME, SYSDATE);

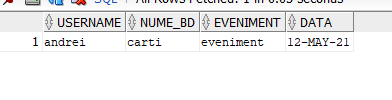
END;

/

SELECT \* FROM useri;



insert into useri values ('andrei','carti','eveniment',TO\_DATE('12-05-2021','DD-MM-YYYY'));



**13.PACHET CARE SA INCLUDA TOATE TABELELE**

* Creati un pachet care sa contina urmatoarele functii/proceduri:
* o procedura de inserare in tabela CARTI
* o functie care modifica numarul de exemplare al fiecarei carti cu 2 exemplare mai mult
* o functie de modificare a pretului unei carti cu 5%.
* o procedura de marire a salariului pentru angajatii care lucreaza intr-o locatie din Bucuresti
* o procedura care afiseaza informatii despre clienti si fisele de lectura
* o functie care sa afiseze codul, denumirea, tipul, informatii despre autor si valoarea totala (pret\*nr\_exemplare) pentru o carte data prin parametru

Procedurile/ functiile trateaza si exceptiile ce pot aparea.

create or replace package carti\_up is

procedure ins\_carti

(cod\_c in carti.cod\_carte%type,

nume out carti.denumire%type,

pr out carti.pret%type);

function ex\_cod(cod\_c carti.cod\_carte%type)

return boolean;

function exp (a\_nr\_exemplare in number) return number;

function pretul(a\_pret in number) return number;

function val\_tot(nrc in varchar) return number;

PROCEDURE MARIRE\_SALARIU(MAR IN NUMBER, ID\_ANG IN NUMBER);

procedure inf\_clienti;

exc exception;

end;

create or replace package body carti\_up is

procedure ins\_carti

(cod\_c in carti.cod\_carte%type,

nume out carti.denumire%type,

pr out carti.pret%type)

is

begin

if ex\_cod(cod\_c) then

raise exc;

else

insert into carti(cod\_carte,denumire,pret) values (cod\_c,nume,pr);

end if;

exception

when exc then

dbms\_output.put\_line('Nu exista cartea');

end;

function ex\_cod

(cod\_c carti.cod\_carte%type)

return boolean

is

v\_adv number;

begin

select 1 into v\_adv

from carti

where cod\_carte=cod\_c;

return true;

exception

when no\_data\_found then

return false;

end;

function exp (a\_nr\_exemplare in number) return number

is

begin

return(a\_nr\_exemplare+2);

end;

function pretul(a\_pret in number) return number

is

begin

return (a\_pret+a\_pret\*0.05);

end;

function val\_tot(nrc in varchar) return number

as

ex exception;

nre number;

v\_codc scrise.cod\_carte%type;

v\_coda scrise.cod\_autor%type;

v\_den carti.denumire%type;

v\_aut autori.nume\_autor%type;

v\_autp autori.prenume\_autor%type;

v\_tip carti.tip%type;

begin

select s.cod\_carte,c.denumire,s.cod\_autor,a.nume\_autor,a.prenume\_autor,c.tip,sum(c.pret\*c.nr\_exemplare) into v\_codc,v\_den,v\_coda,v\_aut,v\_autp,v\_tip,nre from carti c,autori a, scrise s where

s.cod\_carte=c.cod\_carte and s.cod\_autor=a.cod\_autor and c.cod\_carte=nrc

group by s.cod\_carte,c.denumire,s.cod\_autor,a.nume\_autor,a.prenume\_autor,c.tip ;

dbms\_output.put\_line('Carte:'||v\_den||';valoare totala'||nre||';nume autor:'||v\_aut||';prenume autor:'||v\_autp);

return nre;

if(nre=0)

then raise ex;

end if;

exception

when ex then dbms\_output.put\_line ('Nu sunt exemplare');

end;

PROCEDURE MARIRE\_SALARIU(MAR IN NUMBER, ID\_ANG IN NUMBER)

AS

EXC EXCEPTION;

BEGIN

UPDATE ANGAJATI\_BIBLIOTECA

SET SALARIU= SALARIU\*(1+MAR\*SALARIU)

WHERE id\_locatie=(select id\_locatie from locatii\_biblioteca where adresa\_locatie='%Bucuresti') and COD\_ANGAJAT= ID\_ANG;

IF(SQL%ROWCOUNT=0)

THEN RAISE EXC;

ELSE

DBMS\_OUTPUT.PUT\_LINE('S-a modificat SALARIUL.');

END IF;

EXCEPTION WHEN EXC

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista angajatul');

when others then dbms\_output.put\_line(sqlerrm);

end MARIRE\_SALARIU;

procedure INF\_CLIENTI as

tel clienti.telefon%type;

nume clienti.nume\_cititor%type;

fis clienti.cod\_fisa%type;

id\_c clienti.cod\_cititor%type:=&cod\_cititor;

begin

begin

select nume\_cititor into nume

from clienti

where cod\_cititor=id\_c;

dbms\_output.put\_line ('Cititorul'||id\_c);

exception

when no\_data\_found then

dbms\_output.put\_line ('Nu sunt informatii');

end;

begin

select cod\_fisa into fis

from fise\_lectura\_1

where cod\_cititor=id\_c;

dbms\_output.put\_line ('Cititorul'||nume||'are fisa:'||fis);

end;

exception

when no\_data\_found then

dbms\_output.put\_line ('Nu exista informatii despre fisa de lectura a acestui cititor');

end;

end;

/

Apel:

select cod\_carte,denumire,an\_aparitie,

carti\_up.pretul(pret) PretActualizat

from carti;

select cod\_carte,denumire,an\_aparitie,

carti\_up.exp(nr\_exemplare) ExemplareActualizate

from carti;

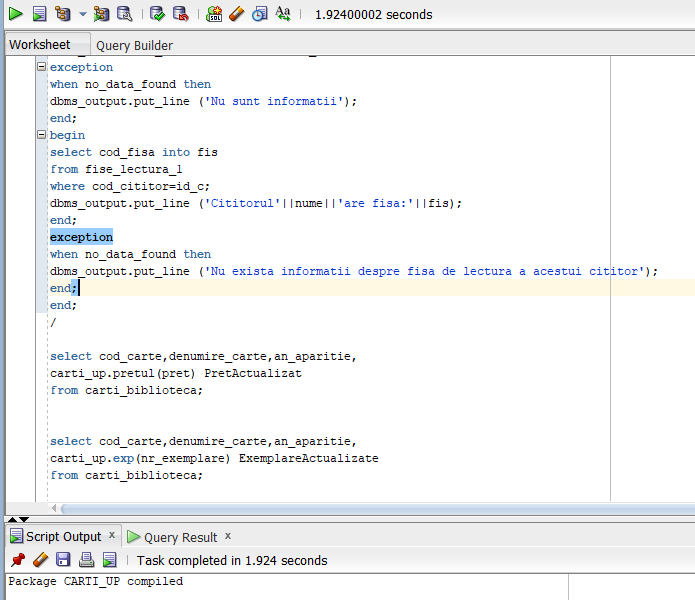
--apel prin bloc anonim--

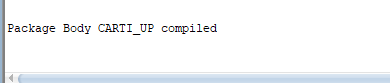
begin

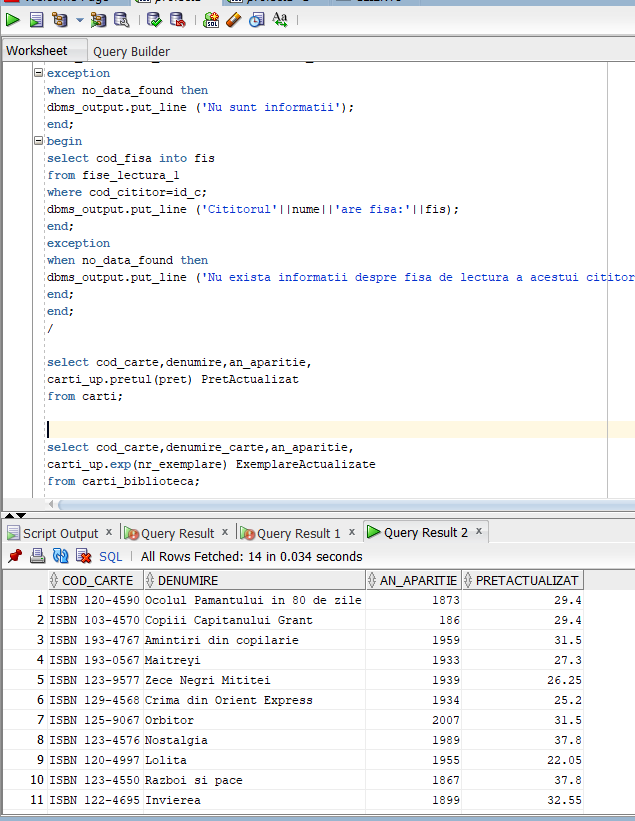
carti\_up.INF\_CLIENTI;

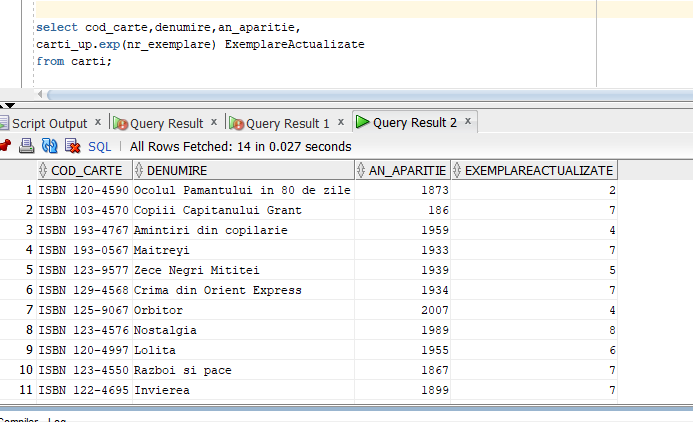
end;

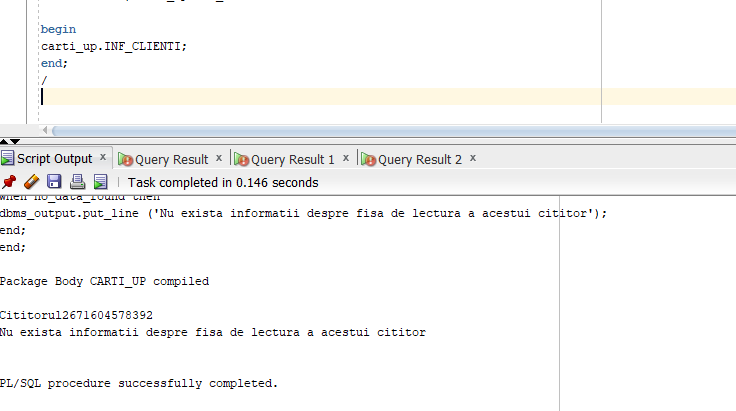
/











**14.PACHET**

* Construiti un pachet care sa contina functii si proceduri referitoare la editurile cartilor din biblioteca:
* un cursor LISTA\_ED care sa parcurga editurile
* o functie care sa gaseasca o editura al carei cod este dat prin parametru
* o procedura care afiseaza editura
* o procedura care afiseaza informatii despre editura data prin parametru si despre cartile care au editura respectiva.

CREATE SEQUENCE sec\_cod\_editura START WITH 101;

CREATE OR REPLACE PACKAGE pachet\_edituri

IS

CURSOR lista\_ed(

v\_cod edituri.cod\_editura%type)

IS

SELECT \* FROM edituri

WHERE cod\_editura = v\_cod

;

FUNCTION gaseste\_editura(

v\_cod edituri.cod\_editura%type)

RETURN BOOLEAN;

PROCEDURE afiseaza\_editura(

v\_cod edituri.cod\_editura%type);

PROCEDURE carti\_cu\_editura\_data(

v\_cod in out edituri.cod\_editura%type);

END pachet\_edituri;

/

CREATE OR REPLACE PACKAGE BODY pachet\_edituri

IS

FUNCTION gaseste\_editura(

v\_cod edituri.cod\_editura%type)

RETURN BOOLEAN

IS

rez NUMBER;

BEGIN

SELECT COUNT(\*) INTO rez

FROM edituri

WHERE cod\_editura=v\_cod;

IF rez>0 THEN RETURN TRUE;

ELSE RETURN FALSE;

END IF;

END gaseste\_editura;

PROCEDURE afiseaza\_editura(

v\_cod edituri.cod\_editura%type)

IS

BEGIN

FOR i IN lista\_ed(v\_cod) LOOP

IF lista\_ed%ROWCOUNT=1 THEN

DBMS\_OUTPUT.PUT\_LINE('Editura: '||

i.nume\_editura || ' cu codul '||i.cod\_editura );

ELSE

DBMS\_OUTPUT.PUT\_LINE('Nu s-a gasit editura');

END IF;

END LOOP;

END afiseaza\_editura;

PROCEDURE carti\_cu\_editura\_data(

v\_cod in out edituri.cod\_editura%type)

IS

v\_cod\_carte carti.cod\_carte%type;

v\_den carti.denumire%type;

BEGIN

IF gaseste\_editura(v\_cod)

THEN

DBMS\_OUTPUT.PUT\_LINE('Editura gasita!');

afiseaza\_editura(v\_cod);

select c.cod\_carte, c.denumire, e.nume\_editura into v\_cod\_carte,v\_den,v\_cod from carti c, edituri e

where c.cod\_editura=e.cod\_editura and c.cod\_editura=v\_cod;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Nu s-a gasit editura');

END IF;

END carti\_cu\_editura\_data;

END pachet\_edituri;

set serveroutput on

begin

pachet\_edituri.carti\_cu\_editura\_data(101);

end;

/

begin

pachet\_edituri.afiseaza\_editura(101);

end;

/

