**Proiect SGBD. Baza de date pentru o editura**

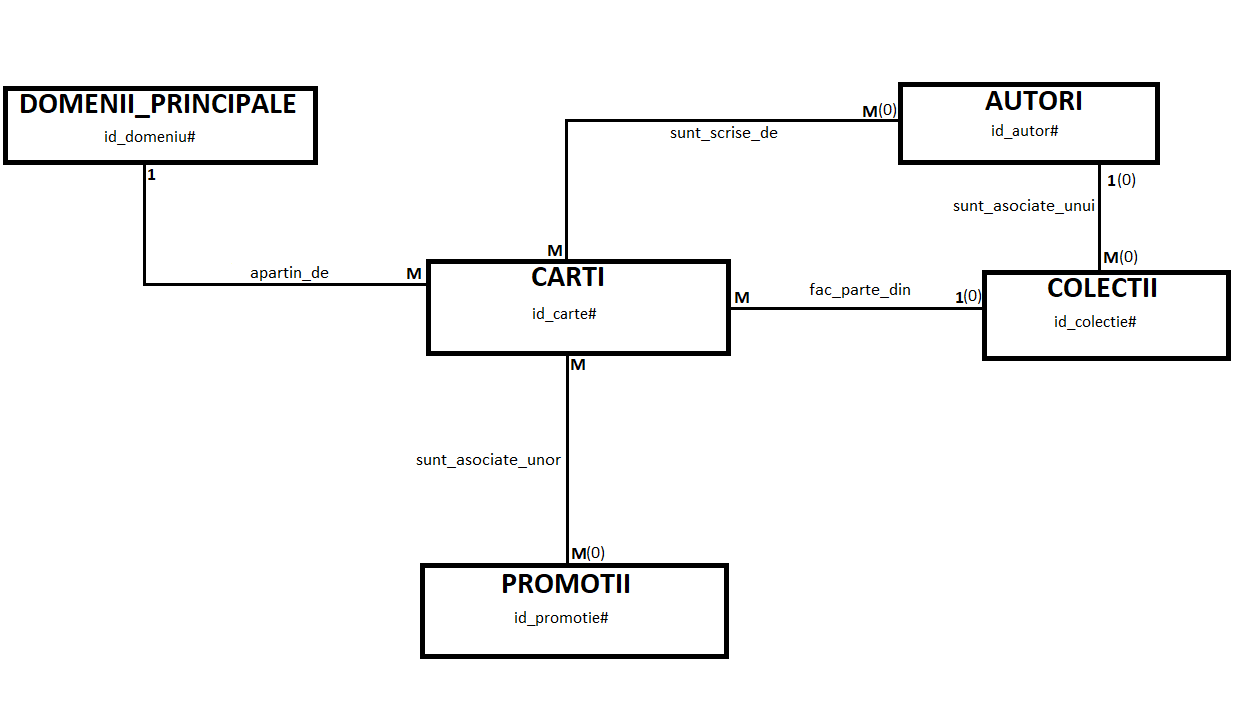
Anghelache Vlad-Alexandru, grupa 232

**1. Prezentați pe scurt baza de date (utilitatea ei).**

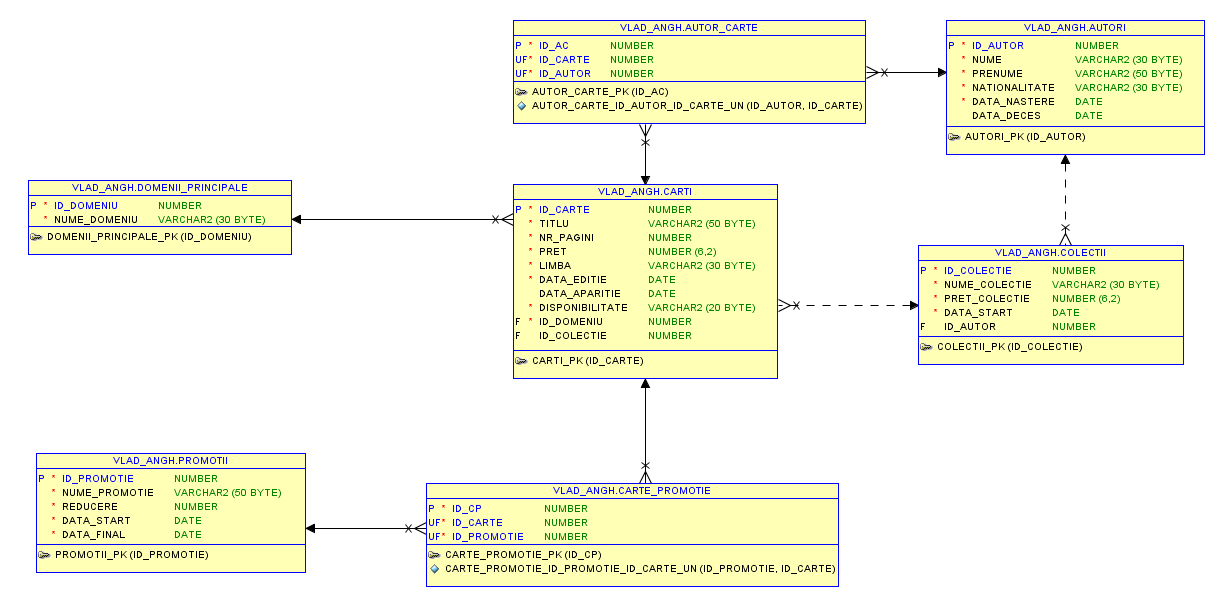
Baza de date din cadrul acestui proiect reprezinta baza de date a unei edituri. In aceasta sunt stocate informatii cu privire la cartile publicate de editura. Pentru fiecare carte se salveaza urmatoarele informatii: titlul, numarul de pagini, pretul, limba in care este scrisa, data aparitiei editiei respective, data in care a fost publicata pentru prima oara si disponibilitatea. Ea poate avea mai multi autori (iar un autor, mai multe carti), pentru fiecare autor stocand numele, prenumele, nationalitatea, data nasterii si data decesului (daca este cazul). Fiecare carte apartine de un domeniu principal. De asemenea, o carte poate fi publicata in cadrul unei colectii pentru care se retine numele, pretul total, reprezentand suma preturilor tuturor cartilor ce apartin colectiei, data in care colectia a fost initializata si id-ul autorului (in cazul in care colectia este o *serie de autor*). Editura poate organiza promotii pentru un anumit interval de timp care sa cuprinda mai multe carti publicate. Fiecarei promotii ii este atribuita o reducere care este aplicata tuturor cartilor din cadrul ei, in intervalul de timp in care promotia este activa. O promotie va avea un nume, un procent care reprezinta reducerea, data de inceput si data de final. O carte poate fi asociata mai multor promotii.

Asadar, baza de date va avea urmatoarele entitati independente: CARTI, AUTORI, COLECTII, DOMENII\_PRINCIPALE si PROMOTII.

**2. Realizați diagrama entitate-relație (ERD).**



**3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.**



**4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).**

Cod:

create table autori(

id\_autor int primary key,

nume varchar(30) not null,

prenume varchar(50) not null,

nationalitate varchar(30) not null,

data\_nastere date not null,

data\_deces date null,

constraint CHL\_data check(data\_deces > data\_nastere)

);

create table colectii(

id\_colectie int primary key,

nume\_colectie varchar(30) not null,

pret\_colectie decimal(6,2) default 0.00 not null check (pret\_colectie >= 0),

data\_start date not null,

id\_autor int null,

foreign key (id\_autor) references autori(id\_autor) on delete cascade

);

create table domenii\_principale(

id\_domeniu int primary key,

nume\_domeniu varchar(30) not null

);

create table promotii(

id\_promotie int primary key,

nume\_promotie varchar(50) not null,

reducere int not null,

data\_start date not null,

data\_final date not null,

constraint CHK\_reducere check (reducere>0 and reducere<= 100)

);

create table carti(

id\_carte int primary key,

titlu varchar(50) not null,

nr\_pagini int not null,

pret decimal(6,2) not null check(pret >= 0),

limba varchar(30) not null,

data\_editie date not null,

data\_aparitie date,

disponibilitate varchar(20) not null check(disponibilitate = 'Disponibil' or disponibilitate = 'Indisponibil'),

id\_domeniu int not null,

foreign key (id\_domeniu) references domenii\_principale(id\_domeniu) on delete cascade ,

id\_colectie int null,

foreign key (id\_colectie) references colectii(id\_colectie) on delete cascade

);

create table autor\_carte (

id\_ac int primary key,

id\_carte int not null,

id\_autor int not null,

constraint FK\_autor\_carte foreign key (id\_autor) references autori(id\_autor) on delete cascade,

constraint FK\_carte\_autor foreign key (id\_carte) references carti(id\_carte) on delete cascade,

unique (id\_autor, id\_carte)

);

create table carte\_promotie (

id\_cp int primary key,

id\_carte int not null,

id\_promotie int not null,

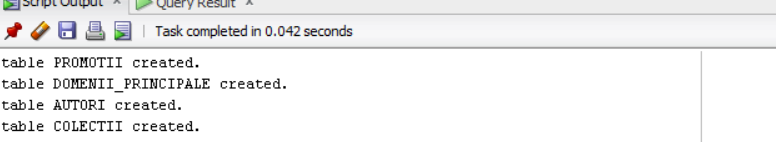
constraint FK\_promotie\_carte foreign key (id\_promotie) references promotii(id\_promotie) on delete cascade,

constraint FK\_carte\_promotie foreign key (id\_carte) references carti(id\_carte) on delete cascade,

unique (id\_promotie, id\_carte)

);

Printscreenuri:









**5. Adăugați informații coerente în tabelele create (minim 3-5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).**

Cod:

-- introducere date:

--autori:

insert into autori

values (1,'Dostoievski', 'Feodor','rusa', to\_date('11/11/1821', 'dd/mm/yyyy'), to\_date('09/02/1881', 'dd/mm/yyyy') );

insert into autori

values (2,'Tolstoi', 'Lev','rusa', to\_date('09/09/1828', 'dd/mm/yyyy'), to\_date('20/11/1910', 'dd/mm/yyyy') );

insert into autori

values (3,'Serghi', 'Cella','romana', to\_date('04/11/1907', 'dd/mm/yyyy'), to\_date('19/09/1992', 'dd/mm/yyyy') );

insert into autori

values (4,'Cartarescu', 'Mircea','romana', to\_date('01/06/1956', 'dd/mm/yyyy'), null );

insert into autori

values (5,'Stancu', 'Elena','romana', to\_date('1982', 'yyyy'), null );

insert into autori

values (6,'Bumbut', 'Cosmin','romana', to\_date('1968', 'yyyy'),null );

insert into autori

values (7,'Kiyosaki', 'Robert','americana', to\_date('08/04/1947', 'dd/mm/yyyy'), null );

insert into autori

values (8,'Petrescu', 'Camil','romana', to\_date('21/04/1894', 'dd/mm/yyyy'), to\_date('14/05/1957', 'dd/mm/yyyy') );

insert into autori

values (9,'Freud', 'Sigmund','austriaca', to\_date('06/05/1856', 'dd/mm/yyyy'), to\_date('23/09/1939', 'dd/mm/yyyy') );

insert into autori

values (10,'Peterson', 'Jordan','canadiana', to\_date('12/06/1962', 'dd/mm/yyyy'), null );

-- domenii:

insert into domenii\_principale

values (1,'Literatura');

insert into domenii\_principale

values (2,'Diverse');

insert into domenii\_principale

values (3,'Psihologie');

insert into domenii\_principale

values (4, 'Dezvoltare personala');

insert into domenii\_principale

values (5,'Economie');

insert into domenii\_principale

values (6,'Filozofie');

insert into domenii\_principale

values(7,'Arte');

insert into domenii\_principale

values (8,'Stiinta');

--colectii:

insert into colectii

values (1,'Serie de autor F. Dostoievski', 0, to\_date('15-08-2016','dd-mm-yyyy'),1);

insert into colectii

values (2,'Serie de autor Tolstoi', 0, to\_date('20-10-2018','dd-mm-yyyy'),2);

insert into colectii

values (3,'Fundamentele Psihologiei', 0, to\_date('15-08-2010','dd-mm-yyyy'), null);

insert into colectii

values (4,'Best of economy', 0, to\_date('15-08-2017','dd-mm-yyyy'),null);

--promotii:

insert into promotii

values (1,'Saptamana literaturii ruse', 50, to\_date('14-02-2021','dd-mm-yyyy'),to\_date('20-02-2021','dd-mm-yyyy'));

insert into promotii

values (2,'Ziua cartilor', 40, to\_date('23-04-2021','dd-mm-yyyy'),to\_date('23-04-2021','dd-mm-yyyy'));

insert into promotii

values (3,'Ziua Nationala a Romaniei', 60, to\_date('01-12-2020','dd-mm-yyyy'),to\_date('01-12-2020','dd-mm-yyyy'));

--carti:

insert into carti

values (1,'Crima si pedeapsa', 592 ,39.99, 'Romana',to\_date('15-08-2016','dd-mm-yyyy'),to\_date('15-01-1866','dd-mm-yyyy'), 'Disponibil',1, 1);

insert into autor\_carte

values(1, 1, 1);

insert into carti

values (2,'Idiotul', 600 ,49.99, 'Romana',to\_date('15-08-2017','dd-mm-yyyy'),to\_date('15-01-1868','dd-mm-yyyy'), 'Disponibil',1, 1);

insert into autor\_carte

values(2, 2, 1);

insert into carti

values (3,'Demonii', 616 ,52.99, 'Romana',to\_date('15-08-2018','dd-mm-yyyy'),to\_date('15-01-1871','dd-mm-yyyy'), 'Disponibil',1, 1);

insert into autor\_carte

values(3, 3, 1);

insert into carti

values (4,'Razboi si pace', 1432 ,88.00, 'Romana',to\_date('20-10-2018','dd-mm-yyyy'),to\_date('15-01-1865','dd-mm-yyyy'), 'Disponibil',1, 2);

insert into autor\_carte

values(4, 4, 2);

insert into carti

values (5,'Anna Karenina', 936 ,58.00, 'Romana',to\_date('20-10-2019','dd-mm-yyyy'),to\_date('15-01-1877','dd-mm-yyyy'), 'Disponibil',1, 2);

insert into autor\_carte

values(5, 5, 2);

insert into carti

values (6,'Moartea lui Ivan Ilici', 124 ,23.00, 'Romana',to\_date('20-10-2000','dd-mm-yyyy'),to\_date('15-01-1886','dd-mm-yyyy'), 'Indisponibil',1, null);

insert into autor\_carte

values(6, 6, 2);

insert into carti

values (7,'Interpretarea viselor', 632 ,49.00, 'Romana',to\_date('15-08-2010','dd-mm-yyyy'),to\_date('04-11-1899','dd-mm-yyyy'), 'Disponibil',3, 3);

insert into autor\_carte

values(7, 7, 9);

insert into carti

values (8,'Despre psihanaliza', 272 ,29.00, 'Romana',to\_date('15-08-2011','dd-mm-yyyy'),to\_date('15-12-1938','dd-mm-yyyy'), 'Disponibil',3, 3);

insert into autor\_carte

values(8, 8, 9);

insert into carti

values (9,'Acasa, pe drum', 388 ,56.00, 'Romana',to\_date('20-10-2017','dd-mm-yyyy'),to\_date('20-10-2017','dd-mm-yyyy'), 'Disponibil',2, null);

insert into autor\_carte

values(9, 9, 5);

insert into autor\_carte

values(10, 9, 6);

insert into carti

values (10,'Rich Dad Poor Dad ', 336 ,89.75, 'Engleza',to\_date('15-08-2017','dd-mm-yyyy'),to\_date('15-04-2017','dd-mm-yyyy'), 'Disponibil',5, 4);

insert into autor\_carte

values(11, 10, 7);

insert into carti

values (11 ,'Rich Dad`s guide to investing', 490 ,93.59, 'Engleza',to\_date('15-08-2018','dd-mm-yyyy'),to\_date('15-04-2018','dd-mm-yyyy'), 'Disponibil',5, 4);

insert into autor\_carte

values(12, 11, 7);

insert into carti

values (12 ,'12 Rules for Life: An Antidote to Chaos', 409 ,60.00, 'Engleza',to\_date('15-08-2018','dd-mm-yyyy'),to\_date('15-04-2018','dd-mm-yyyy'), 'Disponibil',4, null);

insert into autor\_carte

values(13, 12, 10);

insert into carti

values (13 ,'Patul lui Procust', 330 ,30.00, 'Romana',to\_date('23-07-2015','dd-mm-yyyy'),to\_date('15-02-1933','dd-mm-yyyy'), 'Disponibil',1, null);

insert into autor\_carte

values(14, 13, 8);

insert into carti

values (14 ,'Panza de paianjen', 400 ,40.00, 'Romana',to\_date('23-07-2011','dd-mm-yyyy'),to\_date('15-02-1937','dd-mm-yyyy'), 'Disponibil',1, null);

insert into autor\_carte

values(15, 14, 3);

insert into carti

values (15 ,'Melancolia', 256 ,49.99, 'Romana',to\_date('23-07-2019','dd-mm-yyyy'),to\_date('23-07-2019','dd-mm-yyyy'), 'Disponibil',1, null);

insert into autor\_carte

values(16, 15, 4);

--aplicarea promotiilor:

declare

nr number :=0;

nation varchar(30);

ok\_rus number;

ok\_roman number;

begin

for carte in (select id\_carte, id\_domeniu from carti) loop

ok\_rus:=0;

ok\_roman:=0;

nr:=nr+1;

insert into carte\_promotie

values(nr,carte.id\_carte,2);

for autor in (select id\_autor from autor\_carte where id\_carte = carte.id\_carte)loop

select nationalitate

into nation

from autori

where id\_autor = autor.id\_autor;

if nation = 'rusa' then

ok\_rus := 1;

elsif nation ='romana' then

ok\_roman := 1;

end if;

end loop;

if ok\_rus = 1 and carte.id\_domeniu = 1 then

nr:=nr+1;

insert into carte\_promotie

values(nr,carte.id\_carte,1);

end if;

if ok\_roman = 1 then

nr:=nr+1;

insert into carte\_promotie

values(nr,carte.id\_carte,3);

end if;

end loop;

end;

/

-- update pret colectii (se va face in aceasta maniera numai la inceput, urmand ca acest lucru sa fie facut cu ajutorul unui trigger)

begin

for carte in (select pret, id\_colectie from carti) loop

if carte.id\_colectie is not null then

update colectii

set pret\_colectie = pret\_colectie + carte.pret

where id\_colectie = carte.id\_colectie;

end if;

end loop;

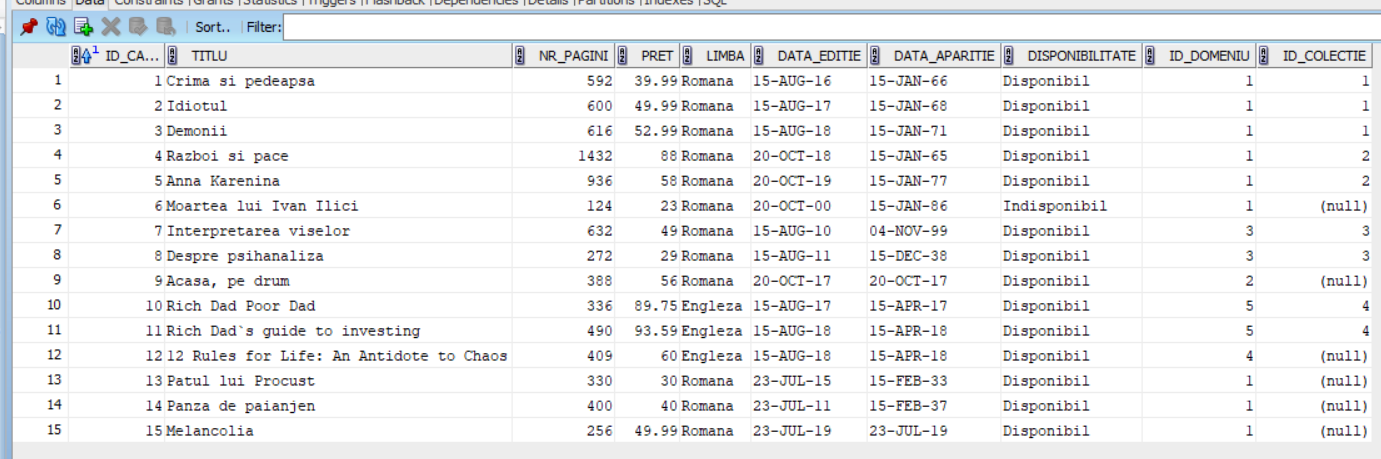
end;

/

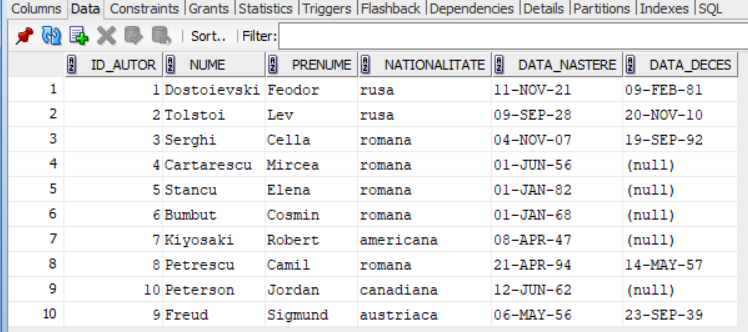
commit;

Informatiile introduse initial in tabele:

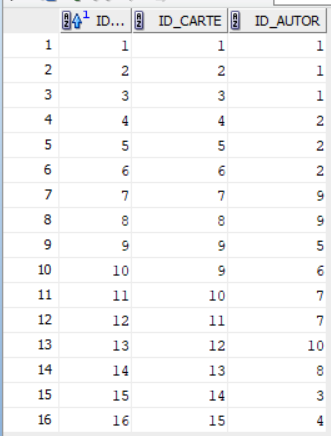
-tabelul CARTI:



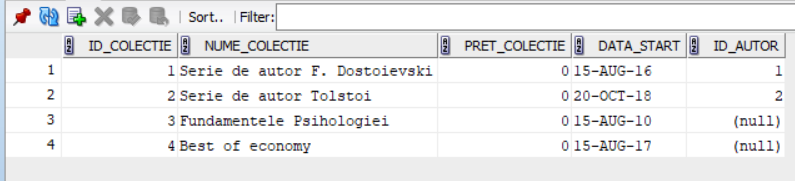
-tabelul AUTORI:



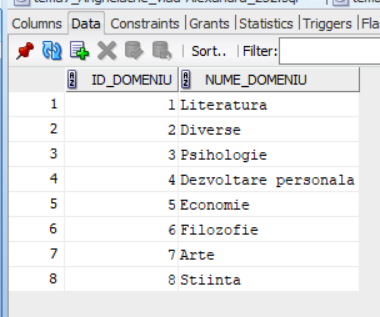
-tabelul AUTOR\_CARTE:



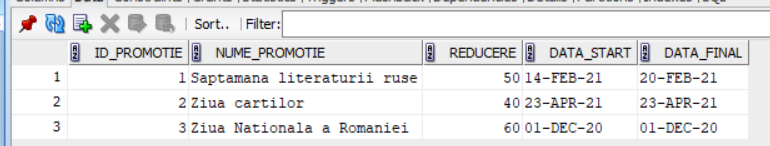
-tabelul COLECTII:



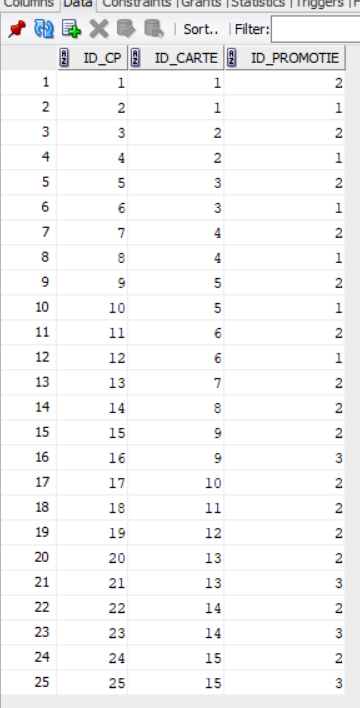
-tabelul DOMENII\_PRINCIPALE:



-tabelul PROMOTII:



-tabelul CARTE\_PROMOTIE:



**6. Definiți un subprogram stocat care să utilizeze un tip de colecție studiat. Apelați subprogramul.**

Procedura lista\_Autori\_Carti primeste ca parametru o nationalitate si afiseaza lista autorilor de nationalitatea respectiva in ordine descrescatoare dupa numarul de carti publicate la editura. Daca la apelare procedura nu primeste valoare pentru parametru, atunci se va afisa lista autorilor in general. De asemenea, pentru fiecare autor se va afisa afisa lista cartilor publicate la editura.

Procedura foloseste tablouri imbricate.

Cod:

-- cerinta 6:

create or replace PROCEDURE lista\_Autori\_Carti ( v\_nationalitate autori.nationalitate%TYPE:= 'international')

as

type tablou\_imbricat\_autori is table of varchar(80);

autori tablou\_imbricat\_autori;

type tablou\_imbricat\_id is table of int;

id\_autori tablou\_imbricat\_id;

type tablou\_imbricat\_carti is table of varchar(50);

carti tablou\_imbricat\_carti;

verif number;

i integer;

j integer;

no\_author exception;

begin

if v\_nationalitate = 'international' then

select count(\*)

into verif

from (select id\_autor

from autori join autor\_carte using (id\_autor)

group by id\_autor);

if verif = 0 then

raise no\_author;

end if;

select id\_autor, (prenume||' '||nume)

bulk collect into id\_autori, autori

from ( select id\_autor, prenume, nume, count(\*) nr

from autori join autor\_carte using (id\_autor)

group by id\_autor, prenume, nume

order by nr desc,nume, prenume );

else

select count(\*)

into verif

from (select id\_autor

from autori join autor\_carte using (id\_autor)

where nationalitate = lower(v\_nationalitate)

group by id\_autor);

if verif = 0 then

raise no\_author;

end if;

select id\_autor, (prenume||' '||nume)

bulk collect into id\_autori, autori

from ( select id\_autor, prenume, nume, count(\*) nr

from autori join autor\_carte using (id\_autor)

where nationalitate = lower(v\_nationalitate)

group by id\_autor, prenume, nume

order by nr desc,nume, prenume );

end if;

i := autori.first;

while i <= autori.last loop

dbms\_output.put\_line('============================');

dbms\_output.put\_line(autori(i));

dbms\_output.put\_line('----------------------------');

select count(\*)

into verif

from autor\_carte

where id\_autor = id\_autori(i);

if verif = 0 then

dbms\_output.put\_line('Autorul nu are nicio carte');

else

select titlu

bulk collect into carti

from carti join autor\_carte using (id\_carte)

where id\_autor = id\_autori(i);

j:=carti.first;

while j <= carti.last loop

dbms\_output.put\_line(carti(j));

j := carti.next(j);

end loop;

end if;

i:=autori.next(i);

end loop;

exception

when no\_author then

dbms\_output.put\_line('Nu s-a gasit niciun autor');

end lista\_Autori\_Carti;

/

set serveroutput on

begin

lista\_autori\_carti();

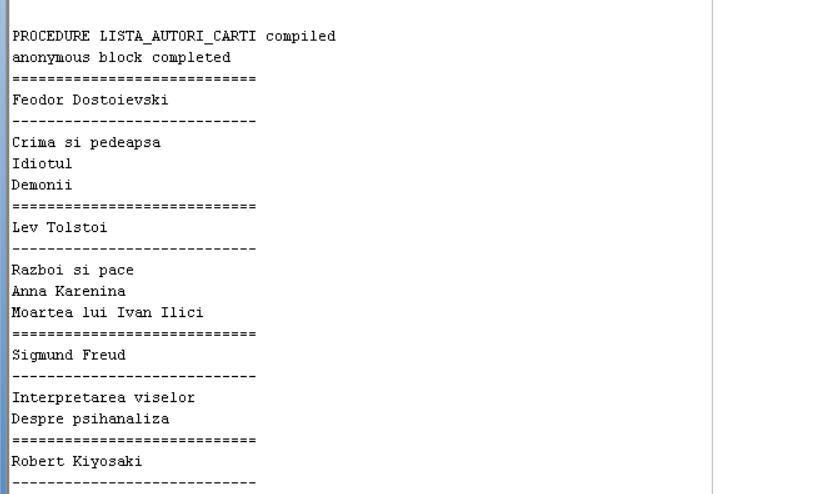
dbms\_output.put\_line('+++++++++++++++++++++++++++++++++++');

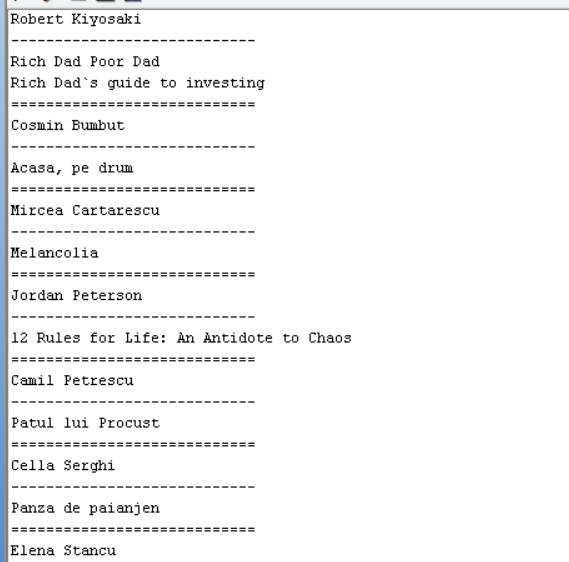
lista\_autori\_carti('rusa');

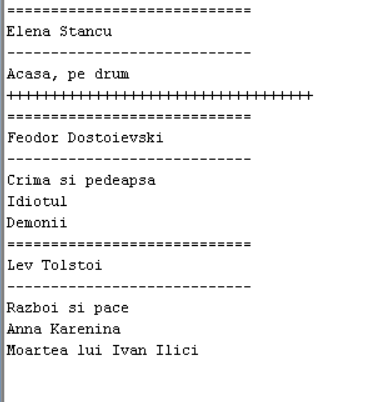
end;

/

Printscreenuri:







**7. Definiți un subprogram stocat care să utilizeze un tip de cursor studiat. Apelați subprogramul.**

Procedura afisare\_promotii primeste ca paramentru o data si afiseaza toate promotiile care vor urma, incepand cu data respectiva. Afiseaza datele promotiei si datele cartilor care sunt asociate ei.

Procedura foloseste cursor parametrizat si expresii cursor.

Cod:

create or replace procedure afisare\_promotii( v\_data promotii.data\_start%type)

as

type refcursor is ref cursor;

cursor c\_prom (v\_prom promotii.nume\_promotie%type) is

select nume\_promotie , reducere , data\_start, data\_final,

cursor (

select c.titlu, c.pret, a.nume, a.prenume

from carti c join carte\_promotie cp using (id\_carte)

join autor\_carte ac using (id\_carte)

join autori a using (id\_autor)

where cp.id\_promotie = p.id\_promotie

and c.disponibilitate = 'Disponibil'

order by titlu)

from promotii p

where p.data\_start > v\_data;

c\_carti refcursor;

p\_nume promotii.nume\_promotie%type;

red promotii.reducere%type;

st promotii.data\_start%type;

fin promotii.data\_final%type;

titlu carti.titlu%type;

pret carti.pret%type;

nume autori.nume%type;

prenume autori.prenume%type;

pret\_red number;

verif number;

exc\_promotie exception;

begin

select count(\*)

into verif

from promotii

where data\_start > v\_data;

if verif = 0 then

raise exc\_promotie;

end if;

open c\_prom(v\_data);

loop

fetch c\_prom INTO p\_nume , red , st ,fin , c\_carti;

exit when c\_prom%NOTFOUND;

dbms\_output.put\_line('===========================');

dbms\_output.put\_line('Nume promotie: '||p\_nume);

dbms\_output.put\_line('Reducere '||red||'%');

dbms\_output.put\_line('Incepand din: '||st||' pana in '||fin);

dbms\_output.put\_line('Cartile incluse: ');

verif:=0;

loop

fetch c\_carti into titlu, pret, nume, prenume;

exit when c\_carti%NOTFOUND;

verif := 1;

pret\_red := pret-pret\*red/100;

dbms\_output.put\_line('----- '||titlu||' de '||prenume||' '||nume||' pret fara reducere: '||pret||' pret redus: '||pret\_red);

end loop;

if(verif = 0) then

dbms\_output.put\_line('----- Acestei promotii nu ii este asociata nicio carte!');

end if;

end loop;

exception

when exc\_promotie then

dbms\_output.put\_line('Nu exista nicio promotie dupa data respectiva!');

end afisare\_promotii;

/

begin

afisare\_promotii(sysdate);

dbms\_output.put\_line(' ');

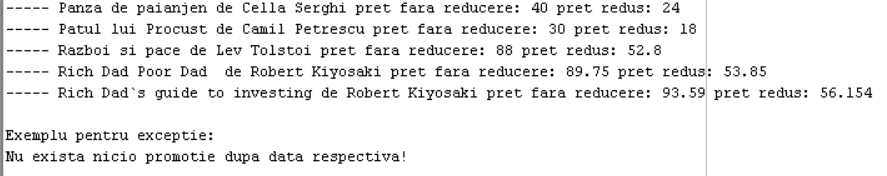
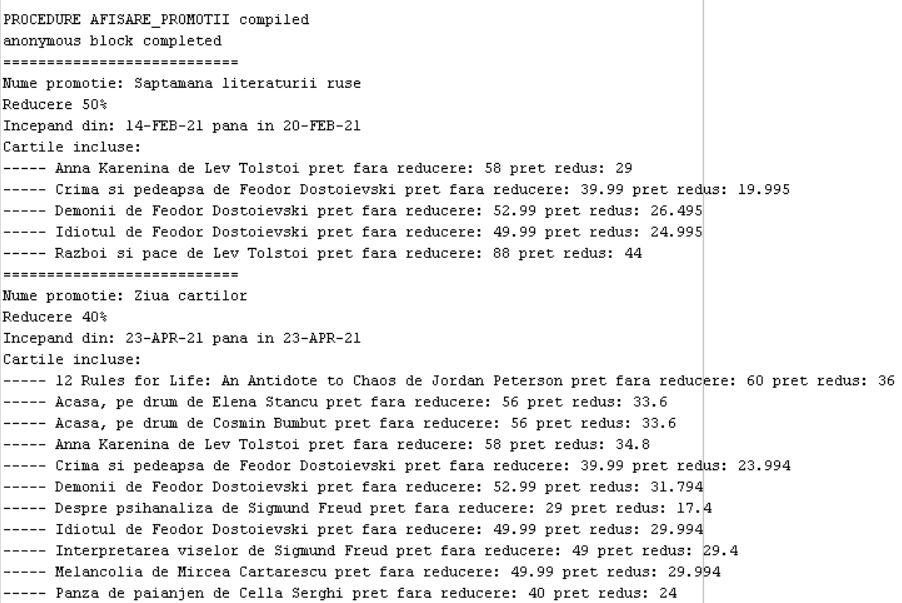
dbms\_output.put\_line('Exemplu pentru exceptie:');

afisare\_promotii(to\_date('01-01-2022','dd-mm-yyyy'));

end;

/

Printscreenuri:



**8. Definiți un subprogram stocat de tip funcție care să utilizeze 3 dintre tabelele definite. Tratați toate excepțiile care pot apăr** **ea. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.**

Functia reducere\_colectie primeste ca paramentru un nume de colectie si returneaza pretul total al comenzii aplicand toate reducerile specifice promotiilor valabile in prezent asociate cartilor din colectia respectiva. Daca nu exista colectie cu numele dat se va returna –2, daca sunt mai multe colectii cu numele dat se va returna –1, daca colectia nu are nicio carte se va returna –3, iar pentru alte exceptii –4. Functia utilizeaza tabelele carti, colectii, promotii si carte\_promotie. Daca pentru o carte se aplica mai multe promotii se va alege cea cu reducerea cea mai mare.

Cod:

create or replace function reducere\_colectie(v\_colectie colectii.nume\_colectie%TYPE)

return number is

suma number := 0;

verif\_colectie number:=0;

no\_colectie exception;

no\_carte exception;

begin

select count(\*)

into verif\_colectie

from colectii

where lower(nume\_colectie) = lower(v\_colectie);

if verif\_colectie = 0 then

raise no\_colectie;

elsif verif\_colectie > 1 then

raise too\_many\_rows;

end if;

select count(\*)

into verif\_colectie

from carti join colectii using(id\_colectie)

where lower(nume\_colectie) = lower(v\_colectie);

if verif\_colectie = 0 then

raise no\_carte;

end if;

for carte in (select id\_carte, pret, max(reducere) red

from carti join colectii using(id\_colectie)

left outer join carte\_promotie using (id\_carte)

left outer join (select id\_promotie, reducere from promotii where sysdate >= data\_start and sysdate <=data\_final) p using (id\_promotie)

where lower(nume\_colectie) = lower(v\_colectie)

group by id\_carte, pret)

loop

if carte.red is not null then

suma := suma + carte.pret - (carte.pret\*carte.red/100);

else

suma := suma + carte.pret;

end if;

end loop;

return suma;

exception

when too\_many\_rows then

return -1;

when no\_colectie then

return -2;

when no\_carte then

return -3;

when others then

return -4;

end reducere\_colectie;

/

insert into colectii

values (5, 'Fara carti', 0, sysdate, null);

insert into colectii

values (6, 'Duplicat', 0, sysdate, null);

insert into colectii

values (7, 'Duplicat', 0, sysdate, null);

insert into colectii

values (8, 'Carti de desenat', 0, sysdate, null);

insert into carti

values(17, 'Animale', 35, 20, 'romana', sysdate,sysdate, 'Disponibil', 2, 8 );

insert into carti

values(18, 'Peisaje', 35, 29, 'romana', sysdate,sysdate, 'Disponibil', 2, 8 );

insert into carti

values(19, 'Planete', 40, 30, 'romana', sysdate,sysdate, 'Disponibil', 2, 8 );

insert into promotii

values(4, 'Promotie Planete', 30, to\_date('01-01-2019','dd-mm-yyyy'), to\_date('01-01-2022','dd-mm-yyyy'));

insert into carte\_promotie

values(26,19,4);

begin

dbms\_output.put\_line(reducere\_colectie('Fara carti'));

dbms\_output.put\_line(reducere\_colectie('Aceasta colectie nu exista'));

dbms\_output.put\_line(reducere\_colectie('Duplicat'));

dbms\_output.put\_line(reducere\_colectie('Carti de desenat'));

dbms\_output.put\_line(reducere\_colectie('Serie de autor F. Dostoievski'));

end;

/

PrintScreenuri:



**9. Definiți un subprogram stocat de tip procedură care să utilizeze 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.**

Procedura afisare\_carti primeste ca paramentru un titlu de carte si afiseaza pe ecran informatii despre cartile care au titlul respectiv. Procedura afiseaza informatii din toate tabelele (7). Daca nu se gaseste nicio carte cu titlul dat atunci se va afisa mesajul “Nu a fost gasita nicio carte cu titlul ***v\_titlu***” . De asemenea, pentru fiecare carte se trateaza cazurile in care nu ii este asociat niciun autor sau nicio promotie.

Cod:

create or replace procedure afisare\_carti(v\_titlu carti.titlu%TYPE)

as

v\_nr number:=0;

v\_nr\_autori number;

v\_nr\_promotii number;

begin

for carte in (select c.id\_carte, c.titlu, c.nr\_pagini, c.pret, c.limba, c.data\_editie, c.data\_aparitie, c.disponibilitate, cl.nume\_colectie, dp.nume\_domeniu

from carti c left outer join colectii cl using (id\_colectie)

join domenii\_principale dp using (id\_domeniu)

where c.titlu = v\_titlu)

loop

v\_nr:=v\_nr + 1;

dbms\_output.put\_line('------------------------------------------------------');

dbms\_output.put\_line(v\_nr||'. '||carte.titlu);

v\_nr\_autori :=0;

dbms\_output.put\_line('Autori:');

for autor in (select nume, prenume from autori join autor\_carte using (id\_autor) where id\_carte = carte.id\_carte)

loop

v\_nr\_autori:=v\_nr\_autori + 1;

dbms\_output.put\_line(autor.prenume||' '||autor.nume);

end loop;

if v\_nr\_autori = 0 then

dbms\_output.put\_line('Acestei carti nu ii este asociat niciun autor');

end if;

dbms\_output.put\_line(' ');

dbms\_output.put\_line('Pret: '|| carte.pret||' lei');

dbms\_output.put\_line('Domeniu: '|| carte.nume\_domeniu);

dbms\_output.put\_line('Numar de pagini: '|| carte.nr\_pagini);

dbms\_output.put\_line('Limba: '|| carte.limba);

dbms\_output.put\_line('An editie: '|| extract(year from carte.data\_editie));

dbms\_output.put\_line('An publicare: '|| extract(year from carte.data\_aparitie));

dbms\_output.put\_line('Disponibilitate: '|| carte.disponibilitate);

if carte.nume\_colectie is not null then

dbms\_output.put\_line('Face parte din colectia: '|| carte.nume\_colectie);

end if;

dbms\_output.put\_line(' ');

v\_nr\_promotii := 0;

dbms\_output.put\_line('Promotii: ');

for promotie in (select nume\_promotie, reducere, data\_start, data\_final from promotii join carte\_promotie using (id\_promotie) where id\_carte = carte.id\_carte)

loop

v\_nr\_promotii := v\_nr\_promotii + 1;

dbms\_output.put\_line(' '||promotie.nume\_promotie||' -- Reducere: '||promotie.reducere||'% -- Interval: '||promotie.data\_start||' - '||promotie.data\_final);

end loop;

if v\_nr\_promotii = 0 then

dbms\_output.put\_line('Acestei carti nu ii este asociata nicio promotie');

end if;

end loop;

if v\_nr = 0 then

raise no\_data\_found;

end if;

exception

when no\_data\_found then

dbms\_output.put\_line('Nu a fost gasita nicio carte cu titlul: '||v\_titlu);

when others then

dbms\_output.put\_line (SQLCODE || ' - ' || SQLERRM);

end afisare\_carti;

/

insert into carti

values(16, 'Idiotul', 109, 30, 'romana', to\_date('20/01/2020','dd/mm/yyyy'),to\_date('20/01/2020','dd/mm/yyyy'), 'Indisponibil', 1, null );

set serveroutput on

begin

dbms\_output.put\_line('Exemplul nr.1:');

afisare\_carti('Iditul');

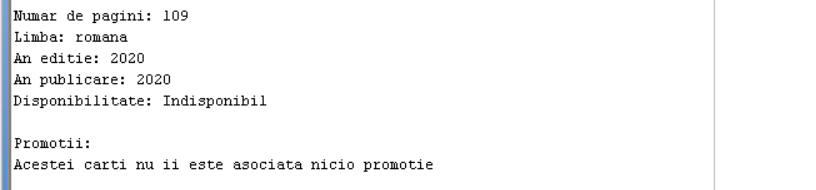
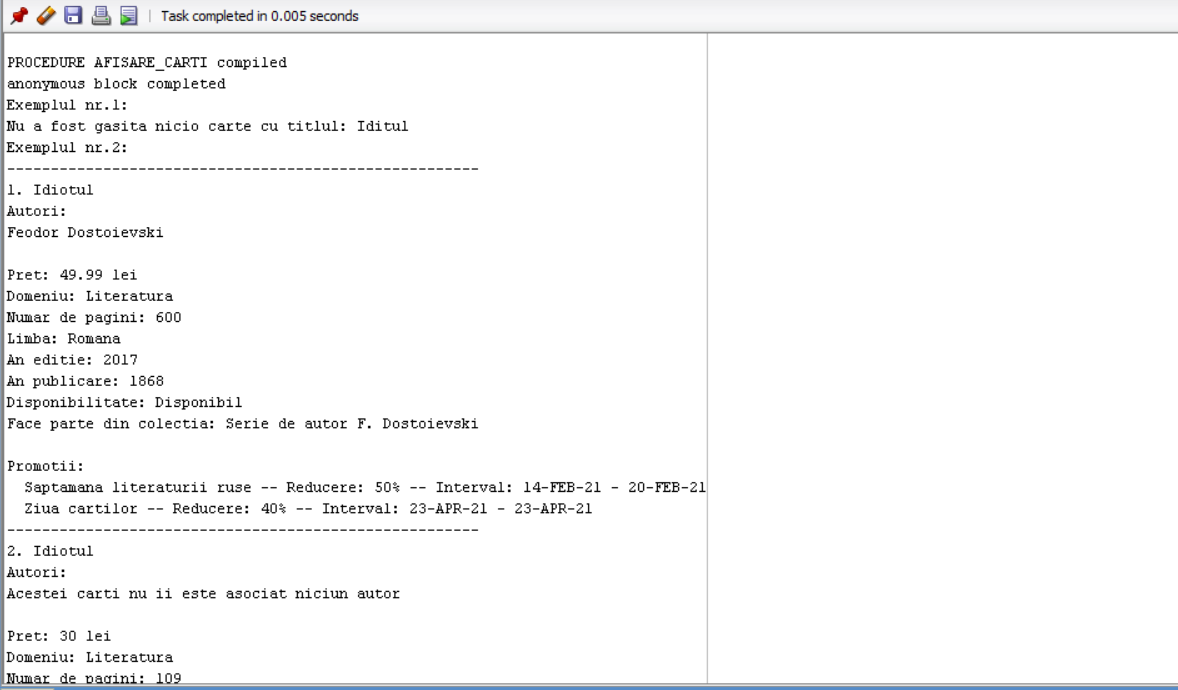
dbms\_output.put\_line('Exemplul nr.2:');

afisare\_carti('Idiotul');

end;

/

Printscreenuri:



**10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.**

Trigger-ul trig\_domenii impiedica pe stergerea, inserarea si modificarea informatiei din tabelul DOMENII\_PRINCIPALE in afara programului de lucru.

Cod:

create or replace trigger trig\_domenii

before insert or delete or update on domenii\_principale

begin

if (to\_char(sysdate, 'D') = 1) or (to\_char(sysdate, 'D') = 7)

or (to\_char(sysdate, 'HH24') not between 9 and 17) then

if DELETING then

RAISE\_APPLICATION\_ERROR(-20001, 'Nu se pot efectiua stergeri pe tabel in afara programului de lucru! ');

elsif INSERTING then

RAISE\_APPLICATION\_ERROR(-20002, 'Nu se pot efectiua inserari pe tabel in afara programului de lucru! ');

elsif UPDATING then

RAISE\_APPLICATION\_ERROR(-20003, 'Nu se pot efectiua modificari pe tabel in afara programului de lucru! ');

end if;

end if;

end;

/

insert into domenii\_principale

values(9, 'Sport');

delete from domenii\_principale

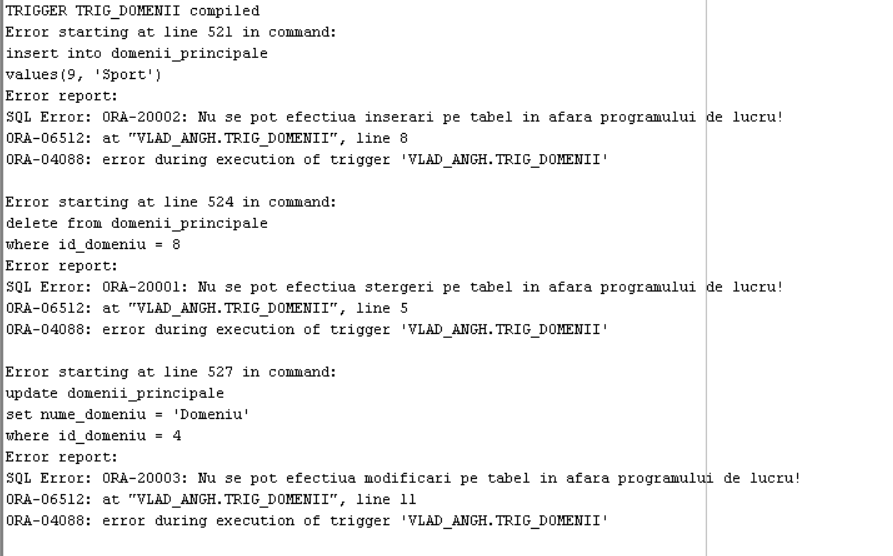
where id\_domeniu = 8;

update domenii\_principale

set nume\_domeniu = 'Domeniu'

where id\_domeniu = 4;

Printscreenuri:



**11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.**

Trigger-ul trig\_pret\_colectii actualizeaza coloana pret\_colectie din tabelul colectii atunci cand se insereaza sau sterge o carte, sau se modifica pretul unei carti.

Cod:

create or replace trigger trig\_pret\_colectii

after insert or delete or update of pret on carti

for each row

begin

if DELETING then

if :OLD.id\_colectie is not null then

update colectii

set pret\_colectie = pret\_colectie - :OLD.pret

where id\_colectie = :OLD.id\_colectie;

end if;

elsif UPDATING then

update colectii

set pret\_colectie = pret\_colectie - :OLD.pret + :NEW.pret

where id\_colectie = :OLD.id\_colectie;

else

update colectii

set pret\_colectie = pret\_colectie + :NEW.pret

where id\_colectie = :NEW.id\_colectie;

end if;

end;

/

insert into carti

values (16,'Adolescentul', 400 ,42.99, 'Romana',to\_date('15-08-2018','dd-mm-yyyy'),to\_date('15-01-1871','dd-mm-yyyy'), 'Disponibil',1, 1);

insert into autor\_carte

values(17, 16, 1);

update carti

set pret = 30.00

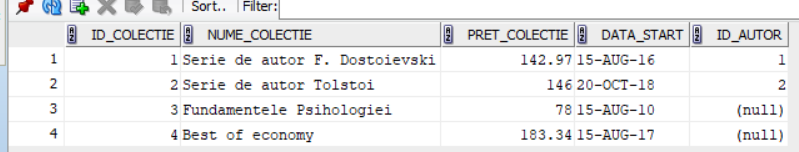
where titlu = 'Adolescentul';

delete from carti

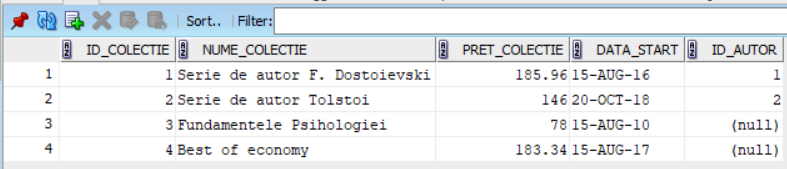
where titlu = 'Adolescentul';

Printscreenuri:

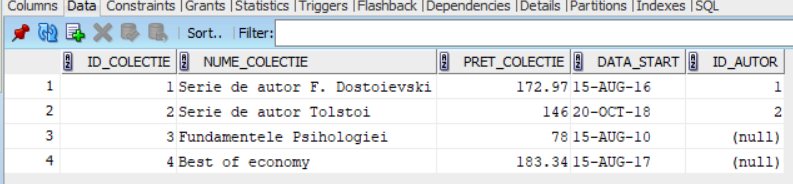
Inainte de inserare carte (Seria de autor F. Dostoievski)



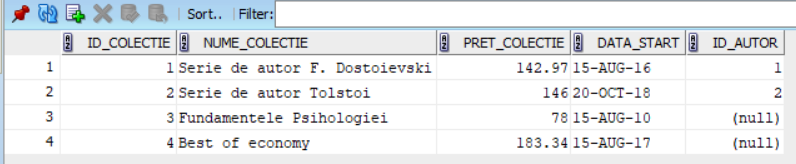
Dupa inserare:



Dupa update:



Dupa delete:



**12. Definiți un trigger de tip LDD. Declanșați trigger-ul.**

Triggerul trig\_info insereaza in tabelul info userul, numele actiunii, obiectul vizat si data de fiecare data cand se face drop, create sau alter.

Cod:

create table info(

username varchar(50),

actiune varchar(30),

nume\_obiect varchar(30),

data date

);

create or replace trigger trig\_info

after create or drop or alter on schema

begin

insert into info

values (SYS.login\_user, sys.sysevent, SYS.dictionary\_obj\_name, sysdate);

end;

/

create table test\_tabel (

col1 int,

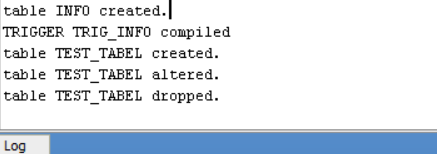
col2 varchar(45));

alter table test\_tabel

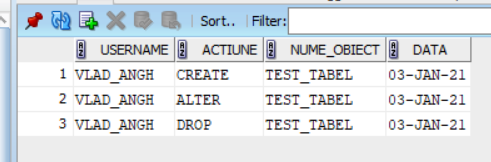
drop column col2;

drop table test\_tabel;

Printscreenuri:



-tabelul INFO:



**13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.**

Cod:

create or replace package pachet\_editura as

PROCEDURE lista\_Autori\_Carti ( v\_nationalitate autori.nationalitate%TYPE:= 'international');

procedure afisare\_promotii( v\_data promotii.data\_start%type);

function reducere\_colectie(v\_colectie colectii.nume\_colectie%TYPE)

return number;

procedure afisare\_carti(v\_titlu carti.titlu%TYPE);

end pachet\_editura;

/

create or replace package body pachet\_editura as

PROCEDURE lista\_Autori\_Carti ( v\_nationalitate autori.nationalitate%TYPE:= 'international')

as

type tablou\_imbricat\_autori is table of varchar(80);

autori tablou\_imbricat\_autori;

type tablou\_imbricat\_id is table of int;

id\_autori tablou\_imbricat\_id;

type tablou\_imbricat\_carti is table of varchar(50);

carti tablou\_imbricat\_carti;

verif number;

i integer;

j integer;

no\_author exception;

begin

if v\_nationalitate = 'international' then

select count(\*)

into verif

from (select id\_autor

from autori join autor\_carte using (id\_autor)

group by id\_autor);

if verif = 0 then

raise no\_author;

end if;

select id\_autor, (prenume||' '||nume)

bulk collect into id\_autori, autori

from ( select id\_autor, prenume, nume, count(\*) nr

from autori join autor\_carte using (id\_autor)

group by id\_autor, prenume, nume

order by nr desc,nume, prenume );

else

select count(\*)

into verif

from (select id\_autor

from autori join autor\_carte using (id\_autor)

where nationalitate = lower(v\_nationalitate)

group by id\_autor);

if verif = 0 then

raise no\_author;

end if;

select id\_autor, (prenume||' '||nume)

bulk collect into id\_autori, autori

from ( select id\_autor, prenume, nume, count(\*) nr

from autori join autor\_carte using (id\_autor)

where nationalitate = lower(v\_nationalitate)

group by id\_autor, prenume, nume

order by nr desc,nume, prenume );

end if;

i := autori.first;

while i <= autori.last loop

dbms\_output.put\_line('============================');

dbms\_output.put\_line(autori(i));

dbms\_output.put\_line('----------------------------');

select count(\*)

into verif

from autor\_carte

where id\_autor = id\_autori(i);

if verif = 0 then

dbms\_output.put\_line('Autorul nu are nicio carte');

else

select titlu

bulk collect into carti

from carti join autor\_carte using (id\_carte)

where id\_autor = id\_autori(i);

j:=carti.first;

while j <= carti.last loop

dbms\_output.put\_line(carti(j));

j := carti.next(j);

end loop;

end if;

i:=autori.next(i);

end loop;

exception

when no\_author then

dbms\_output.put\_line('Nu s-a gasit niciun autor');

end lista\_Autori\_Carti;

procedure afisare\_promotii( v\_data promotii.data\_start%type)

as

type refcursor is ref cursor;

cursor c\_prom (v\_prom promotii.nume\_promotie%type) is

select nume\_promotie , reducere , data\_start, data\_final,

cursor (

select c.titlu, c.pret, a.nume, a.prenume

from carti c join carte\_promotie cp using (id\_carte)

join autor\_carte ac using (id\_carte)

join autori a using (id\_autor)

where cp.id\_promotie = p.id\_promotie

and c.disponibilitate = 'Disponibil'

order by titlu)

from promotii p

where p.data\_start > v\_data;

c\_carti refcursor;

p\_nume promotii.nume\_promotie%type;

red promotii.reducere%type;

st promotii.data\_start%type;

fin promotii.data\_final%type;

titlu carti.titlu%type;

pret carti.pret%type;

nume autori.nume%type;

prenume autori.prenume%type;

pret\_red number;

verif number;

exc\_promotie exception;

begin

select count(\*)

into verif

from promotii

where data\_start > v\_data;

if verif = 0 then

raise exc\_promotie;

end if;

open c\_prom(v\_data);

loop

fetch c\_prom INTO p\_nume , red , st ,fin , c\_carti;

exit when c\_prom%NOTFOUND;

dbms\_output.put\_line('===========================');

dbms\_output.put\_line('Nume promotie: '||p\_nume);

dbms\_output.put\_line('Reducere '||red||'%');

dbms\_output.put\_line('Incepand din: '||st||' pana in '||fin);

dbms\_output.put\_line('Cartile incluse: ');

verif:=0;

loop

fetch c\_carti into titlu, pret, nume, prenume;

exit when c\_carti%NOTFOUND;

verif := 1;

pret\_red := pret-(pret\*red/100);

dbms\_output.put\_line('----- '||titlu||' de '||prenume||' '||nume||' pret fara reducere: '||pret||' pret redus: '||pret\_red);

end loop;

if(verif = 0) then

dbms\_output.put\_line('----- Acestei promotii nu ii este asociata nicio carte!');

end if;

end loop;

exception

when exc\_promotie then

dbms\_output.put\_line('Nu exista nicio promotie dupa data respectiva!');

end afisare\_promotii;

function reducere\_colectie(v\_colectie colectii.nume\_colectie%TYPE)

return number is

suma number := 0;

verif\_colectie number:=0;

no\_colectie exception;

no\_carte exception;

begin

select count(\*)

into verif\_colectie

from colectii

where lower(nume\_colectie) = lower(v\_colectie);

if verif\_colectie = 0 then

raise no\_colectie;

elsif verif\_colectie > 1 then

raise too\_many\_rows;

end if;

select count(\*)

into verif\_colectie

from carti join colectii using(id\_colectie)

where lower(nume\_colectie) = lower(v\_colectie);

if verif\_colectie = 0 then

raise no\_carte;

end if;

for carte in (select id\_carte, pret, max(reducere) red

from carti join colectii using(id\_colectie)

left outer join carte\_promotie using (id\_carte)

left outer join (select id\_promotie, reducere from promotii where sysdate >= data\_start and sysdate <=data\_final) p using (id\_promotie)

where lower(nume\_colectie) = lower(v\_colectie)

group by id\_carte, pret)

loop

if carte.red is not null then

suma := suma + carte.pret - (carte.pret\*carte.red/100);

else

suma := suma + carte.pret;

end if;

end loop;

return suma;

exception

when too\_many\_rows then

return -1;

when no\_colectie then

return -2;

when no\_carte then

return -3;

when others then

return -4;

end reducere\_colectie;

procedure afisare\_carti(v\_titlu carti.titlu%TYPE)

as

v\_nr number:=0;

v\_nr\_autori number;

v\_nr\_promotii number;

begin

for carte in (select c.id\_carte, c.titlu, c.nr\_pagini, c.pret, c.limba, c.data\_editie, c.data\_aparitie, c.disponibilitate, cl.nume\_colectie, dp.nume\_domeniu

from carti c left outer join colectii cl using (id\_colectie)

join domenii\_principale dp using (id\_domeniu)

where c.titlu = v\_titlu)

loop

v\_nr:=v\_nr + 1;

dbms\_output.put\_line('------------------------------------------------------');

dbms\_output.put\_line(v\_nr||'. '||carte.titlu);

v\_nr\_autori :=0;

dbms\_output.put\_line('Autori:');

for autor in (select nume, prenume from autori join autor\_carte using (id\_autor) where id\_carte = carte.id\_carte)

loop

v\_nr\_autori:=v\_nr\_autori + 1;

dbms\_output.put\_line(autor.prenume||' '||autor.nume);

end loop;

if v\_nr\_autori = 0 then

dbms\_output.put\_line('Acestei carti nu ii este asociat niciun autor');

end if;

dbms\_output.put\_line(' ');

dbms\_output.put\_line('Pret: '|| carte.pret||' lei');

dbms\_output.put\_line('Domeniu: '|| carte.nume\_domeniu);

dbms\_output.put\_line('Numar de pagini: '|| carte.nr\_pagini);

dbms\_output.put\_line('Limba: '|| carte.limba);

dbms\_output.put\_line('An editie: '|| extract(year from carte.data\_editie));

dbms\_output.put\_line('An publicare: '|| extract(year from carte.data\_aparitie));

dbms\_output.put\_line('Disponibilitate: '|| carte.disponibilitate);

if carte.nume\_colectie is not null then

dbms\_output.put\_line('Face parte din colectia: '|| carte.nume\_colectie);

end if;

dbms\_output.put\_line(' ');

v\_nr\_promotii := 0;

dbms\_output.put\_line('Promotii: ');

for promotie in (select nume\_promotie, reducere, data\_start, data\_final from promotii join carte\_promotie using (id\_promotie) where id\_carte = carte.id\_carte)

loop

v\_nr\_promotii := v\_nr\_promotii + 1;

dbms\_output.put\_line(' '||promotie.nume\_promotie||' -- Reducere: '||promotie.reducere||'% -- Interval: '||promotie.data\_start||' - '||promotie.data\_final);

end loop;

if v\_nr\_promotii = 0 then

dbms\_output.put\_line('Acestei carti nu ii este asociata nicio promotie');

end if;

end loop;

if v\_nr = 0 then

raise no\_data\_found;

end if;

exception

when no\_data\_found then

dbms\_output.put\_line('Nu a fost gasita nicio carte cu titlul: '||v\_titlu);

when others then

dbms\_output.put\_line (SQLCODE || ' - ' || SQLERRM);

end afisare\_carti;

end pachet\_editura;

/

begin

pachet\_editura.lista\_autori\_carti();

dbms\_output.put\_line('+++++++++++++++++++++++++++++++++++');

pachet\_editura.lista\_autori\_carti('rusa');

end;

/

begin

pachet\_editura.afisare\_promotii(sysdate);

dbms\_output.put\_line(' ');

dbms\_output.put\_line('Exemplu pentru exceptie:');

pachet\_editura.afisare\_promotii(to\_date('01-01-2022','dd-mm-yyyy'));

end;

/

begin

dbms\_output.put\_line(pachet\_editura.reducere\_colectie('Fara carti'));

dbms\_output.put\_line(pachet\_editura.reducere\_colectie('Aceasta colectie nu exista'));

dbms\_output.put\_line(pachet\_editura.reducere\_colectie('Duplicat'));

dbms\_output.put\_line(pachet\_editura.reducere\_colectie('Carti de desenat'));

dbms\_output.put\_line(pachet\_editura.reducere\_colectie('Serie de autor F. Dostoievski'));

end;

/

begin

dbms\_output.put\_line('Exemplul nr.1:');

pachet\_editura.afisare\_carti('Iditul');

dbms\_output.put\_line('Exemplul nr.2:');

pachet\_editura.afisare\_carti('Idiotul');

end;

/

Printscreenuri:

