****

**-PROIECT SGBD-**

**CUPRINS**

[Cerința 1. 3](#_Toc155640791)

[Cerința 2. 3](#_Toc155640792)

[Cerința 3. 4](#_Toc155640793)

[Cerința 4. 5](#_Toc155640794)

[Cerința 5. 9](#_Toc155640795)

[Cerința 6. 20](#_Toc155640796)

[Cerința 7. 23](#_Toc155640797)

[Cerința 8. 27](#_Toc155640798)

[Cerința 9. 32](#_Toc155640799)

[Cerința 10. 36](#_Toc155640800)

[Cerința 11. 39](#_Toc155640801)

[Cerința 12. 45](#_Toc155640802)

[Cerința 13. 47](#_Toc155640803)

[Cerința 14. 59](#_Toc155640804)

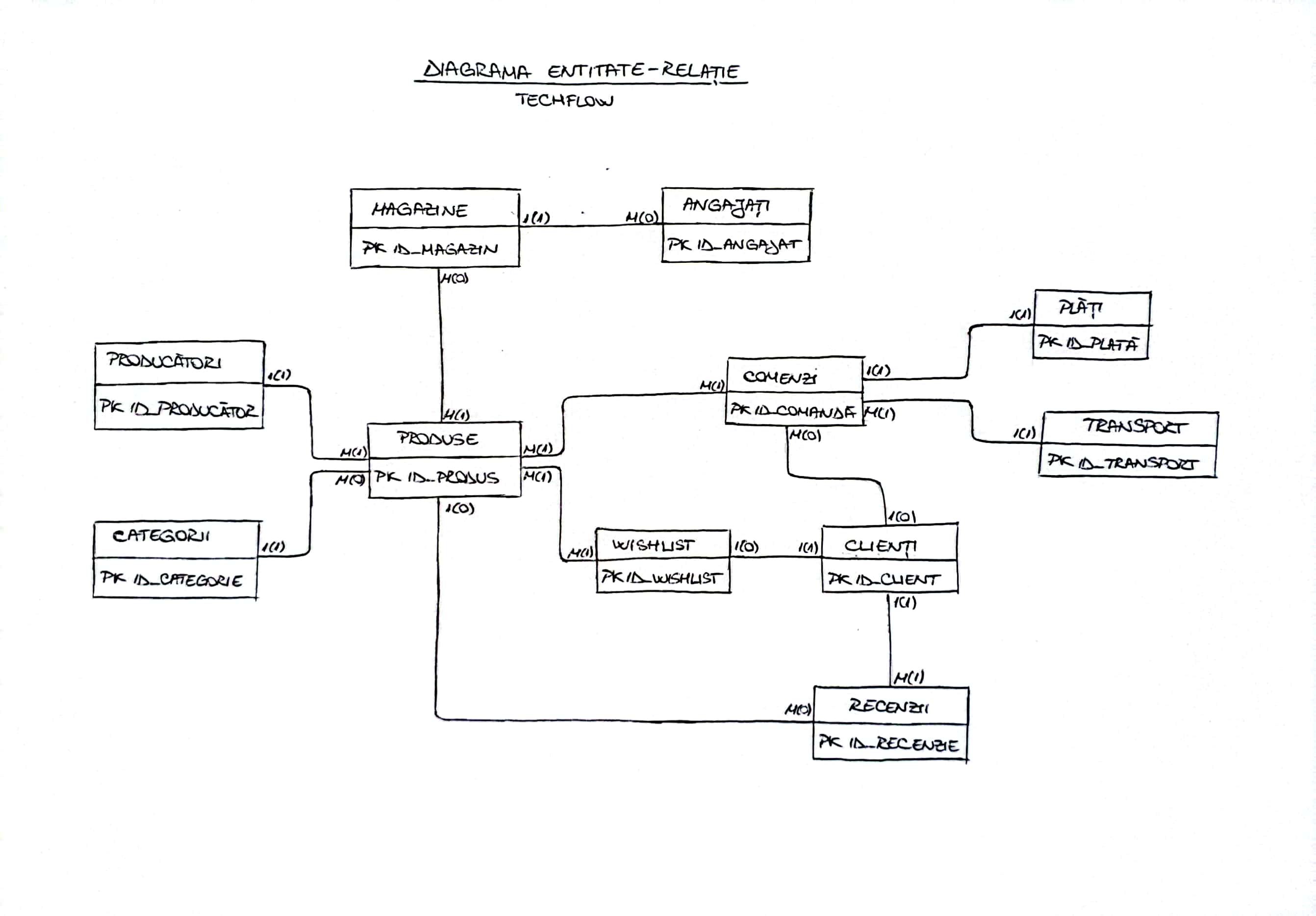
Cerința 1.

Prezentați pe scurt baza de date (utilitatea ei).

**TechFlow** este o afacere specializată în vânzarea de produse IT, cum ar fi componente hardware, accesorii și echipamente IT, prin intermediul magazinelor fizice și a site-ului lor de comerț electronic.

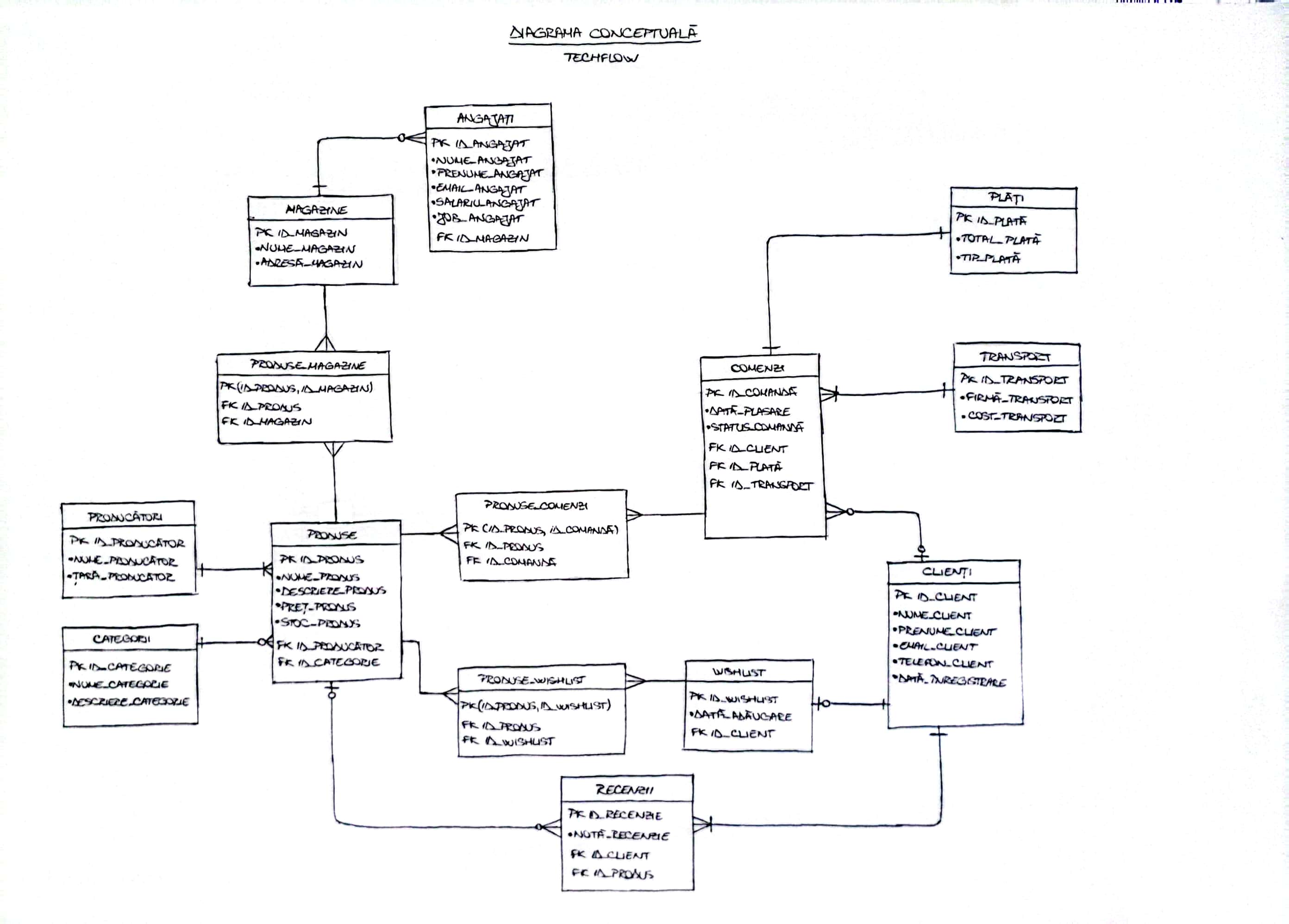
Prin intermediul bazei de date TechFlow, se realizează gestiunea detaliată a produselor, categoriilor de produse, comenzilor , plăților și recenziilor. Aceasta facilitează procesul de vânzare, stocare, gestionare a comenzilor și interacțiune cu clienții. De asemenea, baza de date permite generarea de rapoarte și analize pentru a obține o înțelegere mai profundă a activităților de vânzare și a preferințelor clienților, contribuind astfel la **luarea deciziilor strategice** și la **îmbunătățirea experienței clienților** .

Cerința 2.

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

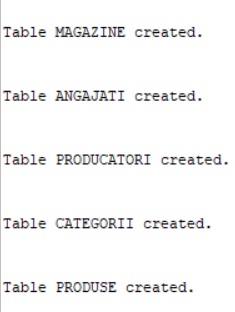
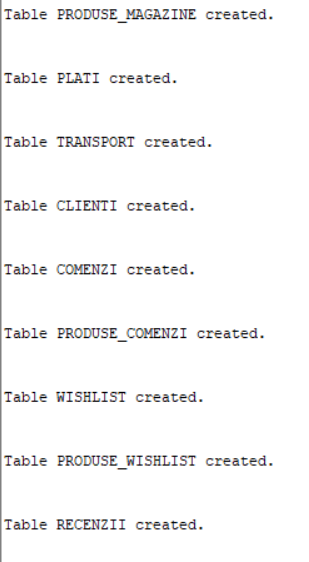
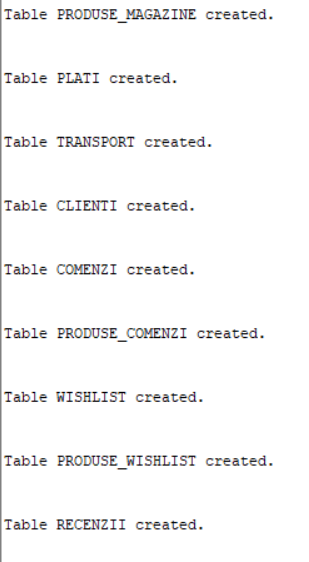
Cerința 3.

Realizați diagrama conceptuală a modelului propus.

****

Cerința 4.

Definiți toate tabelele, definind toate constrângerile de integritate necesare (chei primare, cheile externe etc).



-- Cerinta 4

-- Crearea secventei pentru cheile primare.

create sequence secventa\_techflow

start with 1

increment by 1

maxvalue 10000

minvalue 1

nocycle

nocache;

-- 1 tabel MAGAZINE

create table magazine (

id\_magazin number(4),

nume\_magazin varchar2(30) constraint nn\_nume\_magazin not null,

adresa\_magazin varchar2(40) constraint nn\_adresa\_magazin not null,

constraint pk\_id\_magazin primary key(id\_magazin)

);

-- 2 tabel ANGAJATI

create table angajati (

id\_angajat number(4),

nume\_angajat varchar2(50) constraint nn\_nume\_angajat not null,

prenume\_angajat varchar2(50) constraint nn\_prenume\_angajat not null,

email\_angajat varchar2(50),

salariu\_angajat number(10,2) constraint nn\_salariu\_angajat not null,

job\_angajat varchar2(50),

id\_magazin number(4),

constraint pk\_id\_angajat primary key(id\_angajat),

constraint fk\_id\_magazin foreign key(id\_magazin)

references magazine(id\_magazin)

);

-- 3 tabel PRODUCATORI

create table producatori (

id\_producator number(4),

nume\_producator varchar2(50) constraint nn\_nume\_producator not null,

tara\_producator varchar(50),

constraint pk\_id\_producator primary key(id\_producator)

);

-- 4 tabel CATEGORII

create table categorii (

id\_categorie number(4),

nume\_categorie varchar2(50) constraint nn\_nume\_categorie not null,

descriere\_categorie varchar2(200),

constraint pk\_id\_categorie primary key(id\_categorie)

);

-- 5 tabel PRODUSE

create table produse (

id\_produs number(4),

nume\_produs varchar2(50) constraint nn\_nume\_produs not null,

descriere\_produs varchar2(200),

pret\_produs number(10,2),

stoc\_produs number(3),

id\_producator number(4),

id\_categorie number(4),

constraint pk\_id\_produs primary key(id\_produs),

constraint fk\_id\_producator foreign key(id\_producator)

references producatori(id\_producator),

constraint fk\_id\_categorie foreign key(id\_categorie)

references categorii(id\_categorie)

);

-- 6 tabel PRODUSE\_MAGAZINE

create table produse\_magazine (

id\_produs number(4),

id\_magazin number(4),

constraint pk\_id\_produs\_id\_magazin primary key(id\_produs,id\_magazin),

constraint fk\_id\_produs\_pm foreign key(id\_produs)

references produse(id\_produs),

constraint fk\_id\_magazin\_pm foreign key(id\_magazin)

references magazine(id\_magazin)

);

-- 7 tabel PLATI

create table plati (

id\_plata number(4),

total\_plata number(10,2) constraint nn\_total\_plata not null,

tip\_plata varchar2(50) constraint nn\_tip\_plata not null,

constraint pk\_id\_plata primary key(id\_plata)

);

-- 8 tabel TRANSPORT

create table transport (

id\_transport number(4),

firma\_transport varchar2(50) constraint nn\_firma\_transport not null,

cost\_transport number(10,2) constraint nn\_cost\_transport not null,

constraint pk\_id\_transport primary key (id\_transport)

);

-- 9 tabel CLIENTI

create table clienti (

id\_client number(4),

nume\_client varchar2(50),

prenume\_client varchar2(50),

email\_client varchar2(35),

telefon\_client varchar2(20),

data\_inregistrare date,

constraint pk\_id\_client primary key (id\_client)

);

-- 10 tabel COMENZI

create table comenzi (

id\_comanda number(4),

data\_plasare date,

status\_comanda varchar2(50),

id\_client number(4),

id\_plata number(4),

id\_transport number(4),

constraint pk\_id\_comanda primary key (id\_comanda),

constraint fk\_id\_client foreign key (id\_client)

references clienti(id\_client),

constraint fk\_id\_plata foreign key (id\_plata)

references plati(id\_plata),

constraint fk\_id\_transport foreign key (id\_transport)

references transport(id\_transport)

);

-- 11 tabel PRODUSE\_COMENZI

create table produse\_comenzi (

id\_produs number(4),

id\_comanda number(4),

constraint pk\_id\_produs\_id\_comanda primary key(id\_produs,id\_comanda),

constraint fk\_id\_produs\_pc foreign key(id\_produs)

references produse(id\_produs),

constraint fk\_id\_comanda\_pc foreign key(id\_comanda)

references comenzi(id\_comanda)

);

-- 12 tabel WISHLIST

create table wishlist (

id\_wishlist number(4),

data\_adaugare date,

id\_client number(4),

constraint pk\_id\_wishlist primary key (id\_wishlist),

constraint fk\_id\_client\_wl foreign key (id\_client)

references clienti(id\_client)

);

-- 13 tabel PRODUSE\_WISHLIST

create table produse\_wishlist (

id\_produs number(4),

id\_wishlist number(4),

constraint pk\_id\_produs\_id\_wishlist primary key(id\_produs,id\_wishlist),

constraint fk\_id\_produs\_pw foreign key(id\_produs)

references produse(id\_produs),

constraint fk\_id\_wishlist\_pw foreign key(id\_wishlist)

references wishlist(id\_wishlist)

);

-- 14 tabel RECENZII

create table recenzii (

id\_recenzie number(4),

nota\_recenzie number(2,1),

id\_client number(4),

id\_produs number(4),

constraint pk\_id\_recenzie primary key (id\_recenzie),

constraint fk\_id\_client\_rz foreign key (id\_client)

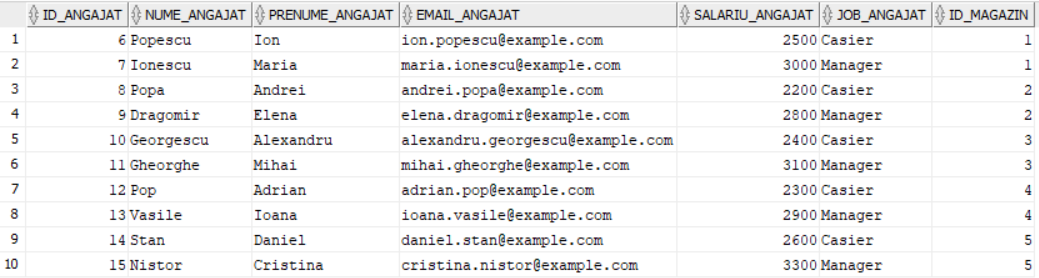
references clienti(id\_client),

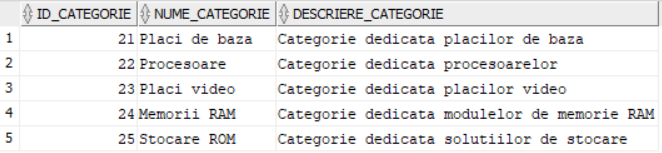
constraint fk\_id\_produs\_rz foreign key (id\_produs)

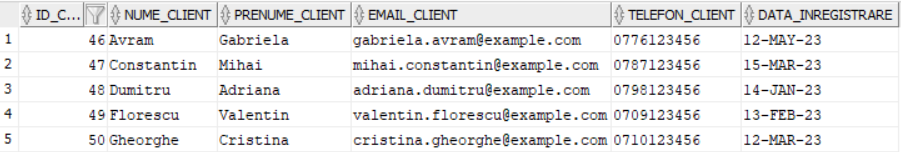
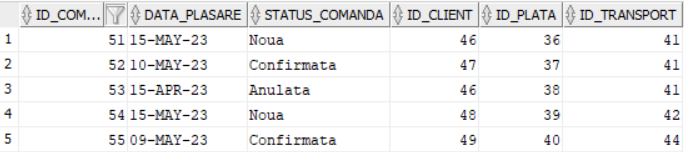
references produse(id\_produs)

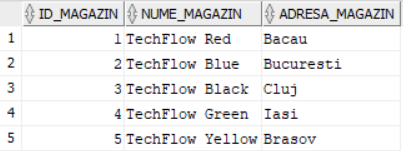
);

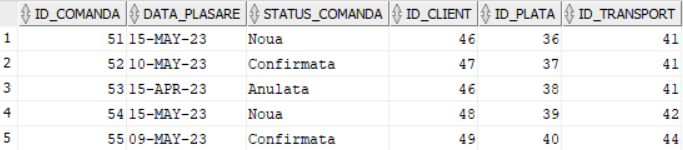
Cerința 5.

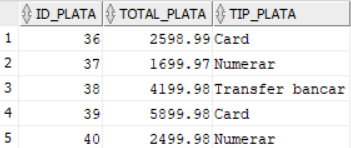
Adăugați informații coerente în tabelele create.

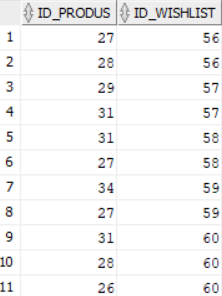
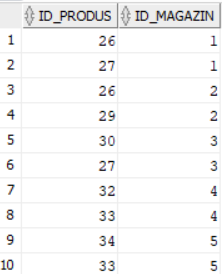
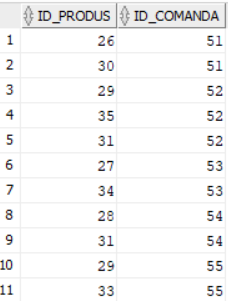


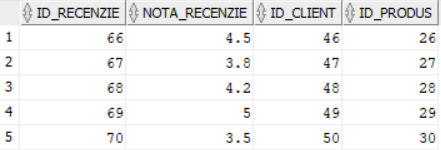
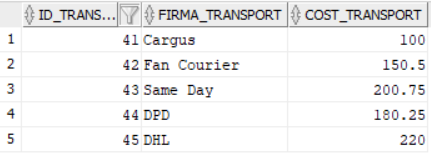


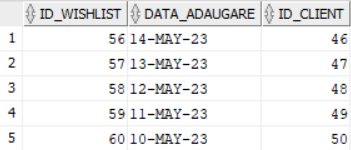












-- 1 tabel MAGAZINE

insert into magazine

values (secventa\_techflow.nextval, 'TechFlow Red', 'Bacau');

insert into magazine

values (secventa\_techflow.nextval, 'TechFlow Blue', 'Bucuresti');

insert into magazine

values (secventa\_techflow.nextval, 'TechFlow Black', 'Cluj');

insert into magazine

values (secventa\_techflow.nextval, 'TechFlow Green', 'Iasi');

insert into magazine

values (secventa\_techflow.nextval, 'TechFlow Yellow', 'Brasov');

select \* from magazine;

-- 2 tabel ANGAJATI

insert into angajati

values (secventa\_techflow.nextval, 'Popescu', 'Ion', 'ion.popescu@example.com', 2500.00, 'Casier', 1);

insert into angajati

values (secventa\_techflow.nextval, 'Ionescu', 'Maria', 'maria.ionescu@example.com', 3000.00, 'Manager', 1);

insert into angajati

values (secventa\_techflow.nextval, 'Popa', 'Andrei', 'andrei.popa@example.com', 2200.00, 'Casier', 2);

insert into angajati

values (secventa\_techflow.nextval, 'Dragomir', 'Elena', 'elena.dragomir@example.com', 2800.00, 'Manager', 2);

insert into angajati

values (secventa\_techflow.nextval, 'Georgescu', 'Alexandru', 'alexandru.georgescu@example.com', 2400.00, 'Casier', 3);

insert into angajati

values (secventa\_techflow.nextval, 'Gheorghe', 'Mihai', 'mihai.gheorghe@example.com', 3100.00, 'Manager', 3);

insert into angajati

values (secventa\_techflow.nextval, 'Pop', 'Adrian', 'adrian.pop@example.com', 2300.00, 'Casier', 4);

insert into angajati

values (secventa\_techflow.nextval, 'Vasile', 'Ioana', 'ioana.vasile@example.com', 2900.00, 'Manager', 4);

insert into angajati

values (secventa\_techflow.nextval, 'Stan', 'Daniel', 'daniel.stan@example.com', 2600.00, 'Casier', 5);

insert into angajati

values (secventa\_techflow.nextval, 'Nistor', 'Cristina', 'cristina.nistor@example.com', 3300.00, 'Manager', 5);

select \* from angajati;

-- 3 tabel PRODUCATORI (inserare explicita)

insert into producatori (id\_producator, nume\_producator, tara\_producator)

values (secventa\_techflow.nextval, 'Stentor', 'Romania');

insert into producatori (id\_producator, nume\_producator, tara\_producator)

values (secventa\_techflow.nextval, 'PC Master', 'Germania');

insert into producatori (id\_producator, nume\_producator, tara\_producator)

values (secventa\_techflow.nextval, 'HardwareCO', 'Franta');

insert into producatori (id\_producator, nume\_producator, tara\_producator)

values (secventa\_techflow.nextval, 'VideoMaster', 'Italia');

insert into producatori (id\_producator, nume\_producator, tara\_producator)

values (secventa\_techflow.nextval, 'SiSoft', 'Spania');

select \* from producatori;

-- 4 tabel CATEGORII

insert into categorii

values (secventa\_techflow.nextval, 'Placi de baza', 'Categorie dedicata placilor de baza');

insert into categorii

values (secventa\_techflow.nextval, 'Procesoare', 'Categorie dedicata procesoarelor');

insert into categorii

values (secventa\_techflow.nextval, 'Placi video', 'Categorie dedicata placilor video');

insert into categorii

values (secventa\_techflow.nextval, 'Memorii RAM', 'Categorie dedicata modulelor de memorie RAM');

insert into categorii

values (secventa\_techflow.nextval, 'Stocare ROM', 'Categorie dedicata solutiilor de stocare');

select \* from categorii;

-- 5 tabel PRODUSE

insert into produse

values (secventa\_techflow.nextval, 'ASUS ROG Strix X570-E Gaming', 'Placă de bază ATX pentru gaming, compatibilă cu procesoare AMD Ryzen', 1599.99, 10, 16, 21);

insert into produse

values (secventa\_techflow.nextval, 'Intel Core i9-10900K', 'Procesor Intel de ultimă generație cu 10 nuclee și 20 fire de execuție', 2899.99, 5, 16, 22);

insert into produse

values (secventa\_techflow.nextval, 'NVIDIA GeForce RTX 3080', 'Placă video high-end pentru gaming, cu 10 GB memorie GDDR6X', 5299.99, 3, 17, 23);

insert into produse

values (secventa\_techflow.nextval, 'Corsair Vengeance RGB Pro', 'Modul de memorie DDR4 cu iluminare RGB și capacitate de 16 GB', 699.99, 20, 17, 24);

insert into produse

values (secventa\_techflow.nextval, 'Samsung 970 EVO Plus', 'Solid State Drive NVMe cu capacitate de 1 TB și viteză de transfer ridicată', 999.99, 8, 17, 25);

insert into produse

VALUES (secventa\_techflow.nextval, 'MSI B450 TOMAHAWK MAX', 'Placă de bază ATX pentru procesoare AMD Ryzen, suportă memorie DDR4', 599.99, 15, 18, 21);

insert into produse

VALUES (secventa\_techflow.nextval, 'AMD Ryzen 7 5800X', 'Procesor AMD cu 8 nuclee și 16 fire de execuție, frecvență de bază 3.8 GHz', 1999.99, 7, 18, 22);

insert into produse

VALUES (secventa\_techflow.nextval, 'Gigabyte GeForce GTX 1660 SUPER OC', 'Placă video performantă pentru gaming, 6 GB memorie GDDR6', 1799.99, 10, 19, 23);

insert into produse

VALUES (secventa\_techflow.nextval, 'RAM Kingston HyperX Fury RGB', 'Modul de memorie DDR4 cu iluminare RGB și capacitate de 32 GB', 1299.99, 12, 19, 24);

insert into produse

VALUES (secventa\_techflow.nextval, 'SSD Western Digital Blue', 'Solid State Drive SATA cu capacitate de 500 GB și viteză de transfer mare', 399.99, 25, 20, 25);

select \* from produse;

-- 6 tabel PRODUSE\_MAGAZINE

-- Magazinul - TechFlow Red

-- Produs - ASUS ROG Strix X570-E Gaming

insert into produse\_magazine

values (26, 1);

-- Produs - Intel Core i9-10900K

insert into produse\_magazine

values (27, 1);

-- Magazinul - TechFlow Blue

-- Produs - ASUS ROG Strix X570-E Gaming

insert into produse\_magazine

values (26,2);

-- Produs - Corsair Vengeance RGB Pro

insert into produse\_magazine

values (29,2);

-- Magazinul - TechFlow Black

-- Produs - Samsung 970 EVO Plus

insert into produse\_magazine

values (30,3);

-- Produs - Intel Core i9-10900K

insert into produse\_magazine

values (27,3);

-- Magazinul - TechFlow Green

-- Produs - AMD Ryzen 7 5800X

insert into produse\_magazine

values (32,4);

-- Produs - Gigabyte GeForce GTX 1660 SUPER OC

insert into produse\_magazine

values (33,4);

-- Magazinul - TechFlow Yellow

-- Produs - RAM Kingston HyperX Fury RGB

insert into produse\_magazine

values (34,5);

-- Produs - Gigabyte GeForce GTX 1660 SUPER OC

insert into produse\_magazine

values (33,5);

select \* from produse\_magazine;

-- 7 tabel PLATI

-- total\_plata id\_produs : 26 + 30

insert into plati (id\_plata, total\_plata, tip\_plata)

values (secventa\_techflow.nextval, 2598.99, 'Card');

-- total\_plata id\_produs : 29 + 35 + 31

insert into plati (id\_plata, total\_plata, tip\_plata)

values (secventa\_techflow.nextval, 1699.97, 'Numerar');

-- total\_plata id\_produs : 27 + 34

insert into plati (id\_plata, total\_plata, tip\_plata)

values (secventa\_techflow.nextval, 4199.98, 'Transfer bancar');

-- total\_plata id\_produs : 28 + 31

insert into plati (id\_plata, total\_plata, tip\_plata)

values (secventa\_techflow.nextval, 5899.98, 'Card');

-- total\_plata id\_produs : 29 + 33

insert into plati (id\_plata, total\_plata, tip\_plata)

values (secventa\_techflow.nextval, 2499.98, 'Numerar');

select \* from plati;

-- 8 tabel TRANSPORT

insert into transport

values (secventa\_techflow.nextval, 'Cargus', 100.00);

insert into transport

values (secventa\_techflow.nextval, 'Fan Courier', 150.50);

insert into transport

values (secventa\_techflow.nextval, 'Same Day', 200.75);

insert into transport

values (secventa\_techflow.nextval, 'DPD', 180.25);

insert into transport

values (secventa\_techflow.nextval, 'DHL', 220.00);

select \* from transport;

-- 9 tabel CLIENTI

insert into clienti

VALUES (secventa\_techflow.nextval, 'Avram', 'Gabriela', 'gabriela.avram@example.com', '0776123456', TO\_DATE('2023-05-12', 'YYYY-MM-DD'));

insert into clienti

VALUES (secventa\_techflow.nextval, 'Constantin', 'Mihai', 'mihai.constantin@example.com', '0787123456', TO\_DATE('2023-03-15', 'YYYY-MM-DD'));

insert into clienti

VALUES (secventa\_techflow.nextval, 'Dumitru', 'Adriana', 'adriana.dumitru@example.com', '0798123456', TO\_DATE('2023-01-14', 'YYYY-MM-DD'));

insert into clienti

VALUES (secventa\_techflow.nextval, 'Florescu', 'Valentin', 'valentin.florescu@example.com', '0709123456', TO\_DATE('2023-02-13', 'YYYY-MM-DD'));

insert into clienti

VALUES (secventa\_techflow.nextval, 'Gheorghe', 'Cristina', 'cristina.gheorghe@example.com', '0710123456', TO\_DATE('2023-03-12', 'YYYY-MM-DD'));

insert into clienti

VALUES (secventa\_techflow.nextval, 'Popescu', 'Mihai', 'popescu.mihai@example.com', '0710121245', TO\_DATE('2024-01-02', 'YYYY-MM-DD'));

insert into clienti

VALUES (secventa\_techflow.nextval, 'Pop', 'Alexandra', 'pop.alexandra@example.com', '0710131245', TO\_DATE('2024-01-04', 'YYYY-MM-DD'));

select \* from clienti;

-- 10 tabel COMENZI

insert into comenzi

values (secventa\_techflow.nextval, to\_date('2023-05-15', 'YYYY-MM-DD'), 'Noua', 46, 36, 41);

insert into comenzi

values (secventa\_techflow.nextval, to\_date('2023-05-10', 'YYYY-MM-DD'), 'Confirmata', 47, 37, 41);

insert into comenzi

values (secventa\_techflow.nextval, to\_date('2023-04-15', 'YYYY-MM-DD'), 'Anulata', 46, 38, 41);

insert into comenzi

values (secventa\_techflow.nextval, to\_date('2023-05-15', 'YYYY-MM-DD'), 'Noua', 48, 39, 42);

insert into comenzi

values (secventa\_techflow.nextval, to\_date('2023-05-09', 'YYYY-MM-DD'), 'Confirmata', 49, 40, 44);

select \* from comenzi;

-- 11 tabel PRODUSE\_COMENZI

insert into produse\_comenzi

values (26,51);

insert into produse\_comenzi

values (30,51);

insert into produse\_comenzi

values (29,52);

insert into produse\_comenzi

values (35,52);

insert into produse\_comenzi

values (31,52);

insert into produse\_comenzi

values (27,53);

insert into produse\_comenzi

values (34,53);

insert into produse\_comenzi

values (28,54);

insert into produse\_comenzi

values (31,54);

insert into produse\_comenzi

values (29,55);

insert into produse\_comenzi

values (33,55);

select \* from produse\_comenzi;

-- 12 tabel WISHLIST

insert into wishlist

values (secventa\_techflow.nextval, to\_date('2023-05-14', 'YYYY-MM-DD'), 46);

insert into wishlist

values (secventa\_techflow.nextval, to\_date('2023-05-13', 'YYYY-MM-DD'), 47);

insert into wishlist

values (secventa\_techflow.nextval, to\_date('2023-05-12', 'YYYY-MM-DD'), 48);

insert into wishlist

values (secventa\_techflow.nextval, to\_date('2023-05-11', 'YYYY-MM-DD'), 49);

insert into wishlist

values (secventa\_techflow.nextval, to\_date('2023-05-10', 'YYYY-MM-DD'), 50);

select \* from wishlist;

-- 13 tabel PRODUSE\_WISHLIST

insert into produse\_wishlist

values (27,56);

insert into produse\_wishlist

values (28,56);

insert into produse\_wishlist

values (29,57);

insert into produse\_wishlist

values (31,57);

insert into produse\_wishlist

values (31,58);

insert into produse\_wishlist

values (27,58);

insert into produse\_wishlist

values (34,59);

insert into produse\_wishlist

values (27,59);

insert into produse\_wishlist

values (31,60);

insert into produse\_wishlist

values (28,60);

insert into produse\_wishlist

values (26,60);

select \* from produse\_wishlist;

-- 14 tabel RECENZII

insert into recenzii

values (secventa\_techflow.nextval, 4.5, 46, 26);

insert into recenzii

values (secventa\_techflow.nextval, 3.8, 47, 27);

insert into recenzii

values (secventa\_techflow.nextval, 4.2, 48, 28);

insert into recenzii

values (secventa\_techflow.nextval, 5.0, 49, 29);

insert into recenzii

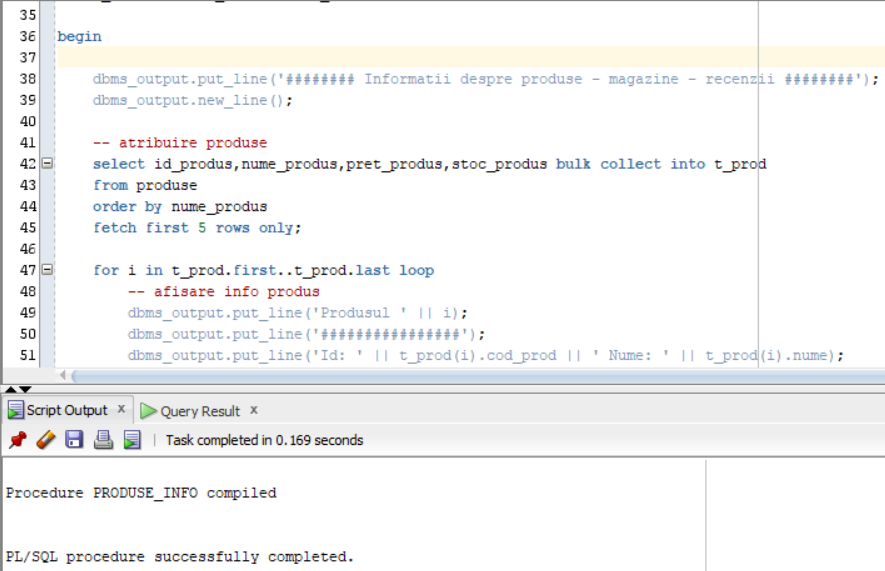
values (secventa\_techflow.nextval, 3.5, 50, 30);

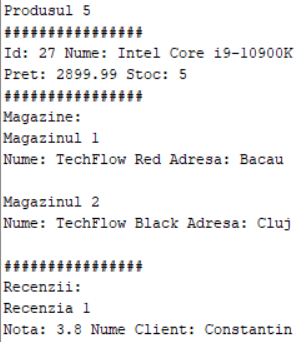
select \* from recenzii;

Cerința 6.

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate. Apelați subprogramul.

Pentru primele 5 produse (varray) in ordine alfabetica dupa nume sa se afiseze codul, numele, pretul, stocul, in ce magazine se afla (t indexat) si recenziile lasate de clienti (t imbr).



create or replace procedure produse\_info as

-- varray de record-uri pentru produse

type prod\_rec is record (

cod\_prod produse.id\_produs%type,

nume produse.nume\_produs%type,

pret produse.pret\_produs%type,

stoc produse.stoc\_produs%type

);

type prod\_vec is varray(5) of prod\_rec;

t\_prod prod\_vec := prod\_vec();

-- tablou indexat pentru magazine

type mag\_rec is record (

nume magazine.nume\_magazin%type,

adresa magazine.adresa\_magazin%type

);

type mag\_ind is table of mag\_rec index by binary\_integer;

t\_mag mag\_ind;

-- tablou imbricat pentru recenzii

type recenzii\_rec is record (

nota recenzii.nota\_recenzie%type,

nume\_client clienti.nume\_client%type

);

type rec\_imbr is table of recenzii\_rec;

t\_recenzii rec\_imbr := rec\_imbr();

begin

dbms\_output.put\_line('######## Informatii despre produse - magazine - recenzii ########');

dbms\_output.new\_line();

-- atribuire produse

select id\_produs,nume\_produs,pret\_produs,stoc\_produs bulk collect into t\_prod

from produse

order by nume\_produs

fetch first 5 rows only;

for i in t\_prod.first..t\_prod.last loop

-- afisare info produs

dbms\_output.put\_line('Produsul ' || i);

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

dbms\_output.put\_line('Id: ' || t\_prod(i).cod\_prod || ' Nume: ' || t\_prod(i).nume);

dbms\_output.put\_line('Pret: ' || t\_prod(i).pret || ' Stoc: ' || t\_prod(i).stoc);

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

-- atribuire magazine

select m.nume\_magazin,m.adresa\_magazin bulk collect into t\_mag

from magazine m

join produse\_magazine pm on (m.id\_magazin = pm.id\_magazin)

where pm.id\_produs = t\_prod(i).cod\_prod;

-- afisare info magazin

if t\_mag.count = 0 then

dbms\_output.put\_line('Produsul nu se afla intr-un magazin!');

dbms\_output.new\_line;

else

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

for j in t\_mag.first..t\_mag.last loop

dbms\_output.put\_line('Magazinul ' || j);

dbms\_output.put\_line('Nume: ' || t\_mag(j).nume || ' Adresa: ' || t\_mag(j).adresa);

dbms\_output.new\_line();

end loop;

end if;

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

-- atribuire recenzii

select r.nota\_recenzie,c.nume\_client bulk collect into t\_recenzii

from recenzii r

join clienti c on (r.id\_client = c.id\_client)

where r.id\_produs = t\_prod(i).cod\_prod;

-- afisare recenzii

if t\_recenzii.count = 0 then

dbms\_output.put\_line('Produsul nu are recenzii!');

dbms\_output.new\_line;

else

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

for k in t\_recenzii.first..t\_recenzii.last loop

dbms\_output.put\_line('Recenzia ' || k);

dbms\_output.put\_line('Nota: ' || t\_recenzii(k).nota || ' Nume Client: ' || t\_recenzii(k).nume\_client);

dbms\_output.new\_line();

end loop;

end if;

dbms\_output.new\_line();

end loop;

end produse\_info;

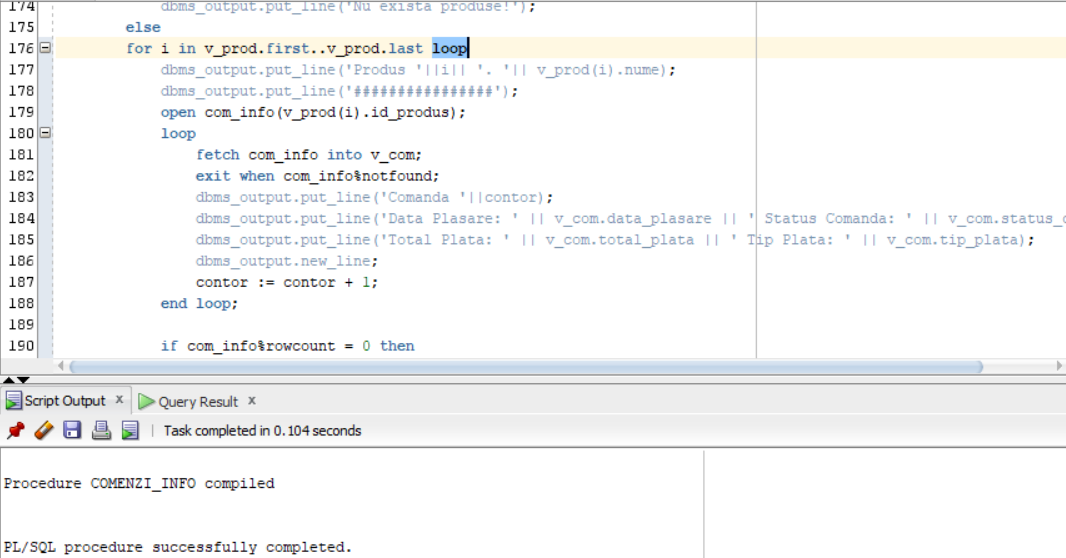
/

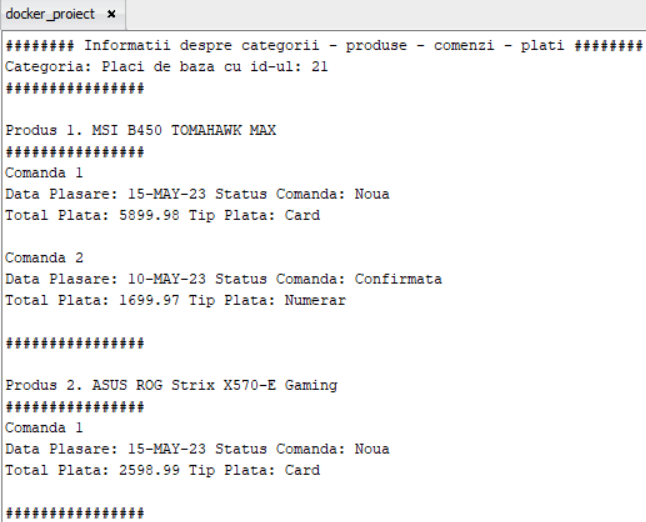
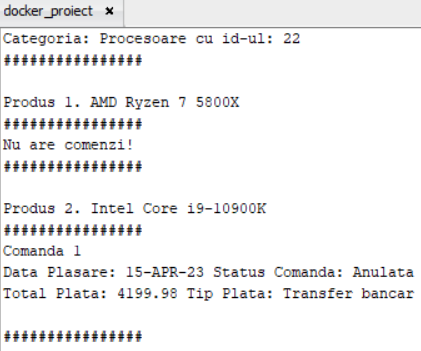
execute produse\_info;

Cerința 7.

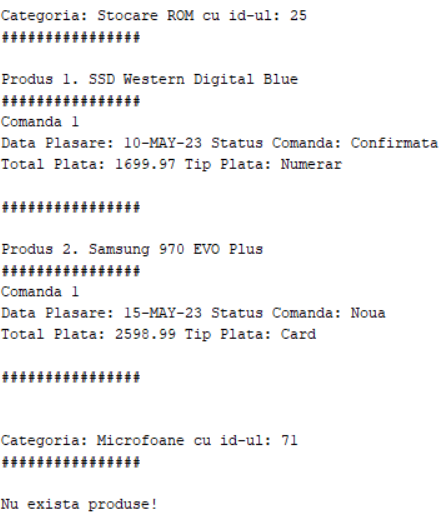
Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat, dependent de celălalt cursor. Apelați subprogramul.

Pentru fiecare categorie sa se determine primele 2 produse ordonate descrescator dupa stoc (cursor dinamic). Sa se afiseze detalii despre comenzile care contin produsele determinate anterior (cursor parametrizat).







create or replace procedure comenzi\_info as

type com\_rec is record (

data\_plasare comenzi.data\_plasare%type,

status\_comanda comenzi.status\_comanda%type,

total\_plata plati.total\_plata%type,

tip\_plata plati.tip\_plata%type

);

type prod\_rec is record (

nume produse.nume\_produs%type,

id\_produs produse.id\_produs%type

);

type t\_prod is table of prod\_rec

index by binary\_integer;

v\_prod t\_prod;

v\_com com\_rec;

-- cursor dinamic/ expresie cursor

-- pentru fiecare categorie se determina produsul cu cel mai mare stoc

cursor categorii\_info is

select id\_categorie,nume\_categorie,

cursor(select max(nume\_produs),max(id\_produs)

from produse p

where c.id\_categorie = p.id\_categorie

group by p.id\_produs

order by max(stoc\_produs) desc)

from categorii c;

-- cursor parametrizat pentru comenzi

cursor com\_info(v\_produs produse.id\_produs%type) is

select c.data\_plasare,c.status\_comanda,p.total\_plata,p.tip\_plata

from comenzi c

join plati p on (c.id\_plata = p.id\_plata)

join produse\_comenzi pc on (c.id\_comanda = pc.id\_comanda)

where pc.id\_produs = v\_produs

order by c.data\_plasare desc;

v\_cursor sys\_refcursor;

v\_id\_categorie categorii.id\_categorie%type;

v\_nume\_categorie categorii.nume\_categorie%type;

contor number := 1;

begin

dbms\_output.put\_line('######## Informatii despre categorii - produse - comenzi - plati ########');

open categorii\_info;

loop

fetch categorii\_info into v\_id\_categorie,v\_nume\_categorie,v\_cursor;

exit when categorii\_info%notfound;

dbms\_output.put\_line('Categoria: ' || v\_nume\_categorie || ' cu id-ul: ' || v\_id\_categorie);

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

fetch v\_cursor bulk collect into v\_prod limit 2;

dbms\_output.new\_line;

if v\_prod.count = 0 then

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

else

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

dbms\_output.put\_line('Produs '||i|| '. '|| v\_prod(i).nume);

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

open com\_info(v\_prod(i).id\_produs);

loop

fetch com\_info into v\_com;

exit when com\_info%notfound;

dbms\_output.put\_line('Comanda '||contor);

dbms\_output.put\_line('Data Plasare: ' || v\_com.data\_plasare || ' Status Comanda: ' || v\_com.status\_comanda);

dbms\_output.put\_line('Total Plata: ' || v\_com.total\_plata || ' Tip Plata: ' || v\_com.tip\_plata);

dbms\_output.new\_line;

contor := contor + 1;

end loop;

if com\_info%rowcount = 0 then

dbms\_output.put\_line('Nu are comenzi!');

end if;

close com\_info;

contor := 1;

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

dbms\_output.new\_line;

end loop;

end if;

dbms\_output.new\_line;

end loop;

close categorii\_info;

end comenzi\_info;

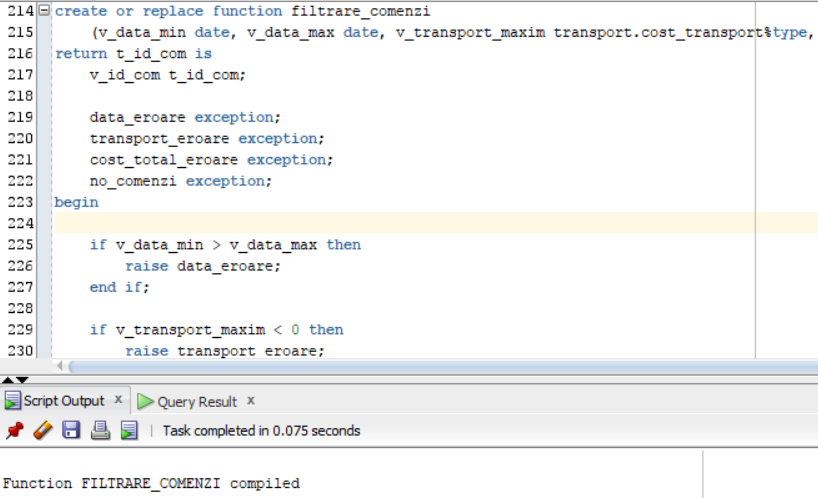
/

execute comenzi\_info;

Cerința 8.

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate

Sa se returneze comenzile plasate intr-o anumita perioada care nu depasesc un anumit cost de transport si un anumit cost total.



create or replace type t\_id\_com is table of number;

/

-- returneaza un tablou de id-uri

create or replace function filtrare\_comenzi

(v\_data\_min date, v\_data\_max date, v\_transport\_maxim transport.cost\_transport%type, v\_cost\_total plati.total\_plata%type)

return t\_id\_com is

v\_id\_com t\_id\_com;

data\_eroare exception;

transport\_eroare exception;

cost\_total\_eroare exception;

no\_comenzi exception;

begin

if v\_data\_min > v\_data\_max then

raise data\_eroare;

end if;

if v\_transport\_maxim < 0 then

raise transport\_eroare;

end if;

if v\_cost\_total < 0 then

raise cost\_total\_eroare;

end if;

-- folosirea a 3 tabele

select distinct c.id\_comanda bulk collect into v\_id\_com

from comenzi c

join plati p on (c.id\_plata = p.id\_plata)

join transport t on (c.id\_transport = c.id\_transport)

where c.data\_plasare >= v\_data\_min and c.data\_plasare <= v\_data\_max

and t.cost\_transport <= v\_transport\_maxim

and p.total\_plata <= v\_cost\_total;

if v\_id\_com.count = 0 then

raise no\_comenzi;

end if;

return v\_id\_com;

exception

when no\_comenzi then

raise\_application\_error(-20000,'Nu exista comanda cu aceste conditii!');

when data\_eroare then

raise\_application\_error(-20001,'Intervalul nu este introdus corect!');

when transport\_eroare then

raise\_application\_error(-20002,'Costul transportului nu poate fi negativ!');

when cost\_total\_eroare then

raise\_application\_error(-20003,'Costul total nu poate fi negativ!');

when others then

raise\_application\_error(-20004,'Alta eroare!');

end filtrare\_comenzi;

/

**-- caz 1 apel corect**

declare

solutie t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2023 cu totatul de plata mai mic de 5000');

dbms\_output.put\_line('si costul de transport mai mic de 190.');

solutie := filtrare\_comenzi(to\_date('10-may-2023', 'dd-mon-yyyy'),to\_date('15-may-2023', 'dd-mon-yyyy'),190,5000);

dbms\_output.new\_line;

dbms\_output.put\_line('Id-uri comenzi care respecta criteriul:');

for i in solutie.first..solutie.last loop

dbms\_output.put\_line(solutie(i)|| ' ');

end loop;

end;

/

**-- caz2 exceptie no\_comenzi**

**-- caz in care nu exista comenzi care sa respecte conditiile**

declare

solutie t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2022 cu totatul de plata mai mic de 3000');

dbms\_output.put\_line('si costul de transport mai mic de 100.');

dbms\_output.new\_line;

solutie := filtrare\_comenzi(to\_date('10-may-2022', 'dd-mon-yyyy'),to\_date('15-may-2022', 'dd-mon-yyyy'),100,3000);

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

**-- caz3 exceptie data\_eroare**

**-- caz in care data minima este mai mare decat data maxima**

declare

solutie t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 15-10 mai 2022 cu totatul de plata mai mic de 3000');

dbms\_output.put\_line('si costul de transport mai mic de 100.');

dbms\_output.new\_line;

solutie := filtrare\_comenzi(to\_date('15-may-2022', 'dd-mon-yyyy'),to\_date('10-may-2022', 'dd-mon-yyyy'),100,3000);

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

**-- caz4 exceptie transport\_eroare**

**-- caz in care costul pentru transport maximal este negativ**

declare

solutie t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2023 cu totatul de plata mai mic de 3000');

dbms\_output.put\_line('si costul de transport mai mic de -10.');

dbms\_output.new\_line;

solutie := filtrare\_comenzi(to\_date('10-may-2022', 'dd-mon-yyyy'),to\_date('15-may-2022', 'dd-mon-yyyy'),-10,3000);

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

**-- caz5 exceptie cost\_total\_eroare**

**-- caz in care costul total maximal este negativ**

declare

solutie t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2023 cu totatul de plata mai mic de -50');

dbms\_output.put\_line('si costul de transport mai mic de 200.');

dbms\_output.new\_line;

solutie := filtrare\_comenzi(to\_date('10-may-2022', 'dd-mon-yyyy'),to\_date('15-may-2022', 'dd-mon-yyyy'),200,-50);

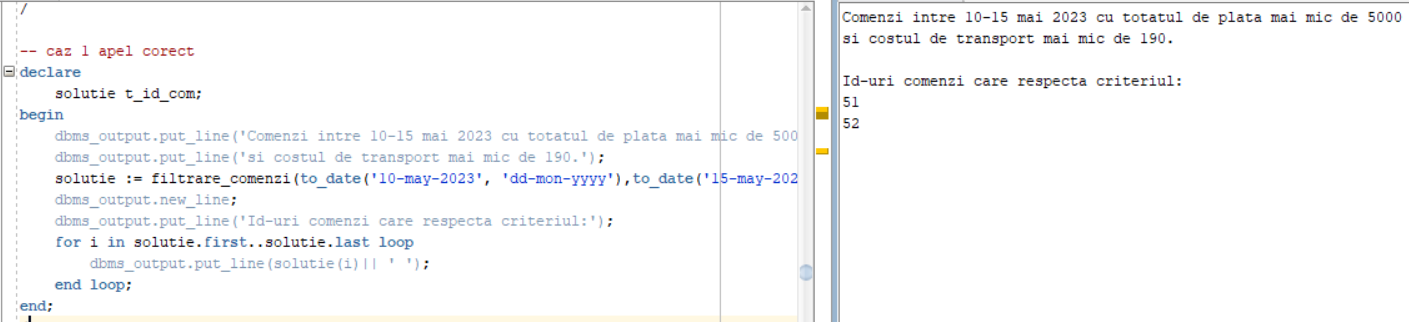
exception

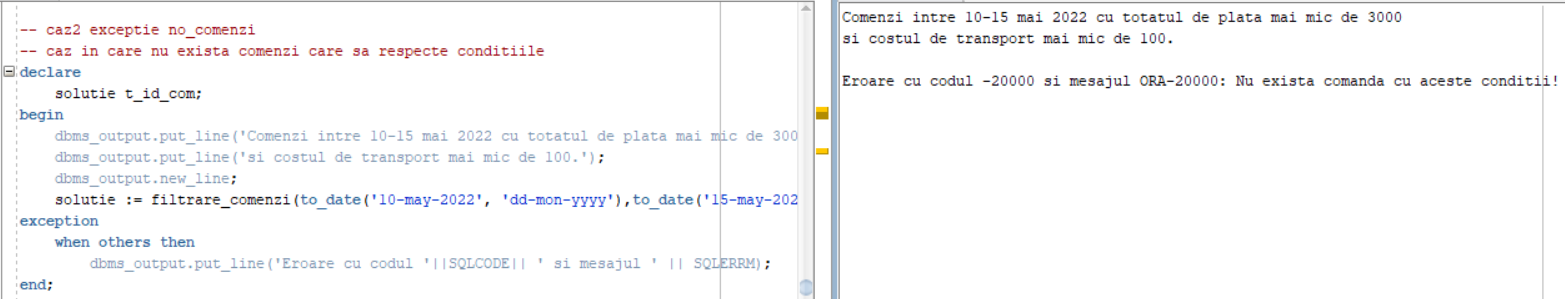
when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

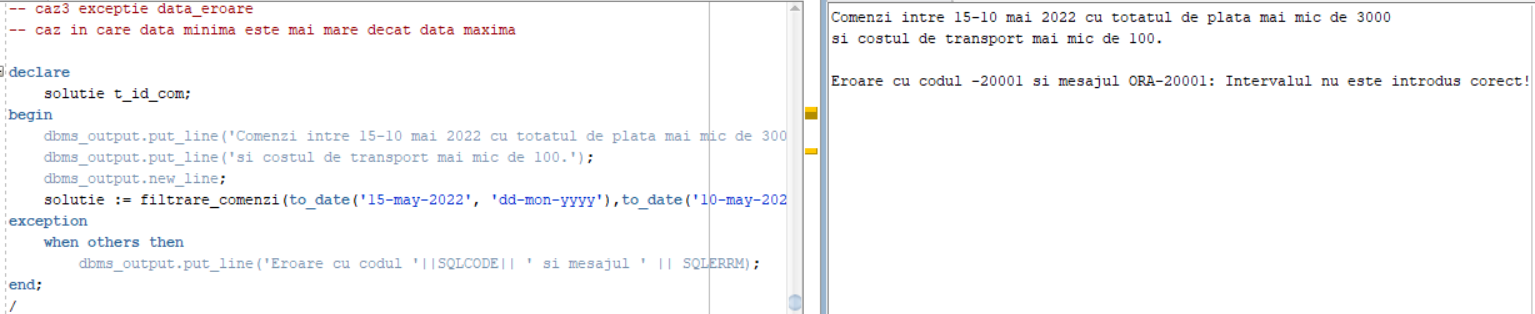
end;

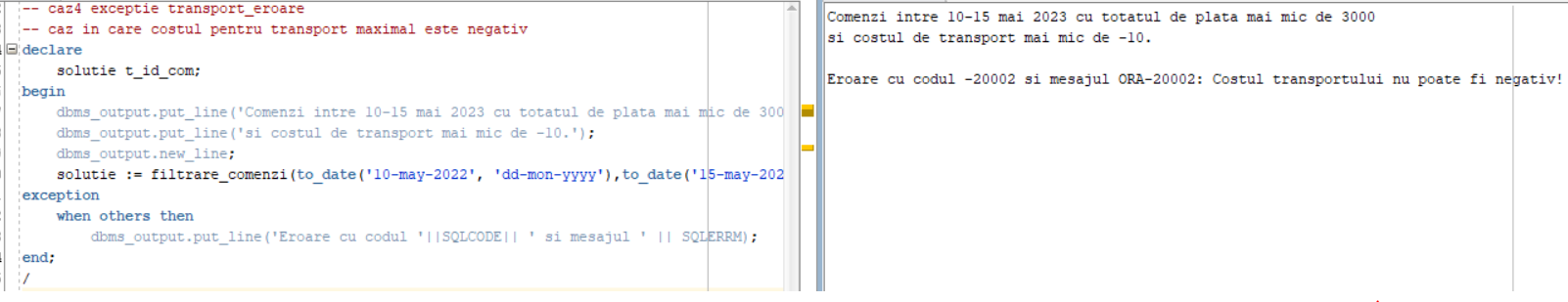
/

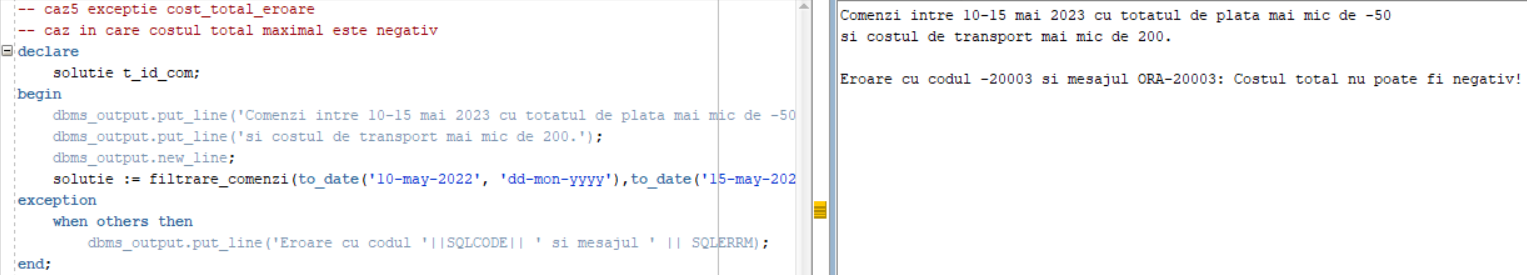




e





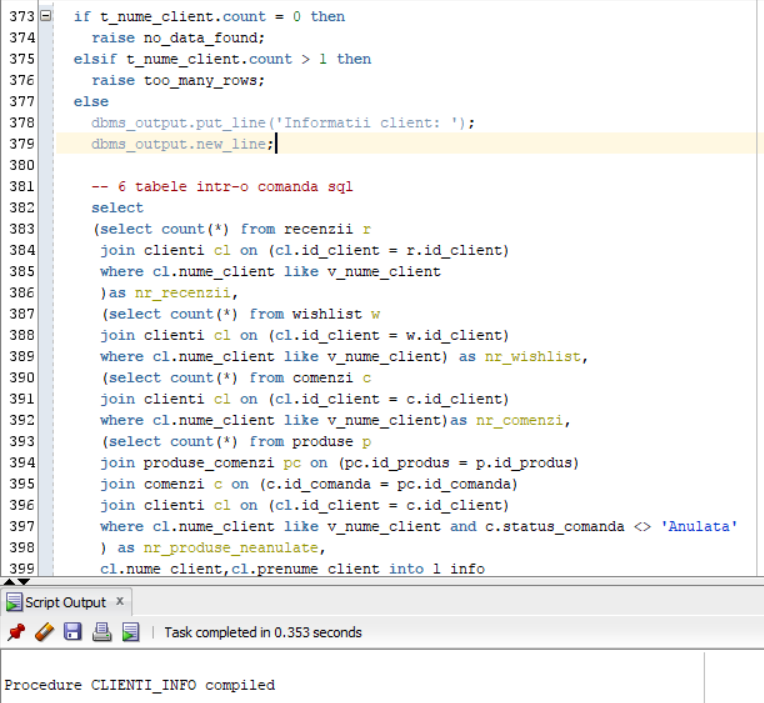


Cerința 9.

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Definiti un subprogram stocat de tip procedura care sa afiseze numarul de recenzii, numarul de produse in wishlist, numarul de comenzi si numarul de produse din comenzile neanulate ale clientului al caror nume este dat.

De asemenea, sa se afiseze numele si prenumele clientului.



create or replace procedure clienti\_info(v\_nume\_client clienti.nume\_client%type)

is

type info\_rec is record (

nr\_recenzii integer,

nr\_wishlist integer,

nr\_comenzi integer,

nr\_produse\_neanulate integer,

nume\_client clienti.nume\_client%type,

prenume\_client clienti.prenume\_client%type

);

l\_info info\_rec;

type nume\_client\_imbr is table of clienti.nume\_client%type;

t\_nume\_client nume\_client\_imbr := nume\_client\_imbr();

begin

select nume\_client bulk collect into t\_nume\_client

from clienti

where nume\_client like v\_nume\_client || '%';

if t\_nume\_client.count = 0 then

raise no\_data\_found;

elsif t\_nume\_client.count > 1 then

raise too\_many\_rows;

else

dbms\_output.put\_line('Informatii client: ');

dbms\_output.new\_line;

-- 6 tabele intr-o comanda sql

select

(select count(\*) from recenzii r

join clienti cl on (cl.id\_client = r.id\_client)

where cl.nume\_client like v\_nume\_client

)as nr\_recenzii,

(select count(\*) from wishlist w

join clienti cl on (cl.id\_client = w.id\_client)

where cl.nume\_client like v\_nume\_client) as nr\_wishlist,

(select count(\*) from comenzi c

join clienti cl on (cl.id\_client = c.id\_client)

where cl.nume\_client like v\_nume\_client)as nr\_comenzi,

(select count(\*) from produse p

join produse\_comenzi pc on (pc.id\_produs = p.id\_produs)

join comenzi c on (c.id\_comanda = pc.id\_comanda)

join clienti cl on (cl.id\_client = c.id\_client)

where cl.nume\_client like v\_nume\_client and c.status\_comanda <> 'Anulata'

) as nr\_produse\_neanulate,

cl.nume\_client,cl.prenume\_client into l\_info

from clienti cl

where cl.nume\_client like v\_nume\_client;

dbms\_output.put\_line('Nume: ' || l\_info.nume\_client);

dbms\_output.put\_line('Prenume: ' || l\_info.prenume\_client);

dbms\_output.new\_line;

dbms\_output.put\_line('Numar recenzii: ' || l\_info.nr\_recenzii);

dbms\_output.put\_line('Numar produse wishlist: ' || l\_info.nr\_wishlist);

dbms\_output.put\_line('Numar comenzi: ' || l\_info.nr\_comenzi);

dbms\_output.put\_line('Numar produse din comenzile neanulate: ' || l\_info.nr\_produse\_neanulate); end if;

exception

when no\_data\_found then

raise\_application\_error(-20000,'Nu exista client cu acest nume!');

when too\_many\_rows then

raise\_application\_error(-20001,'Prea multi clienti cu acest string!');

when others then

raise\_application\_error(-20002,'Alta eroare!');

end clienti\_info;

/

**-- caz 1 apel corect**

begin

clienti\_info('Avram');

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

**-- caz 2 no\_data\_found (nu exista cu client cu acest nume)**

begin

clienti\_info('Huma');

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

**-- caz 3 too\_many\_rows (pentru Pop => exista Popa si Popescu)**

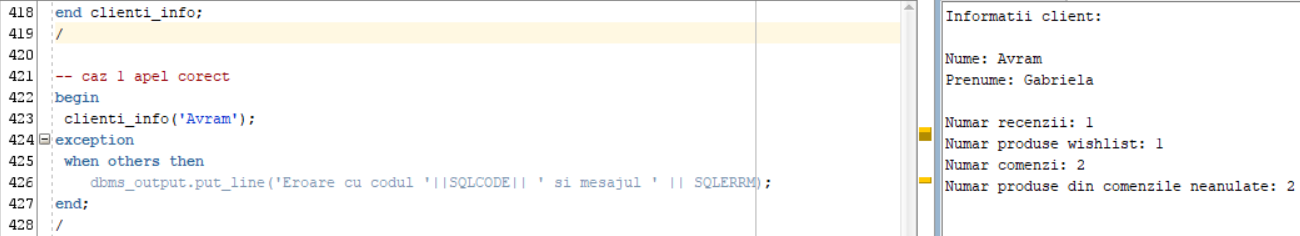
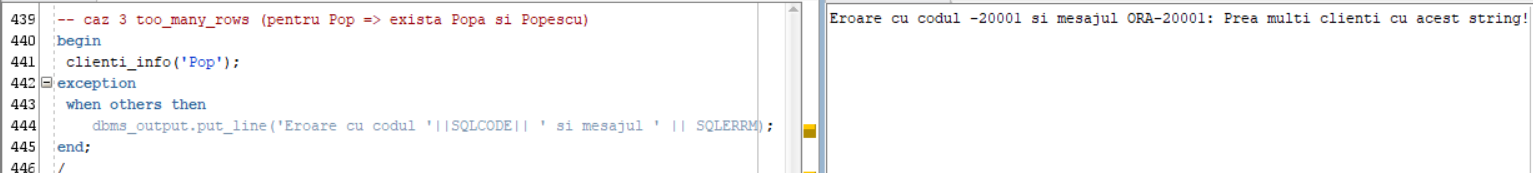
begin clienti\_info('Pop');

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

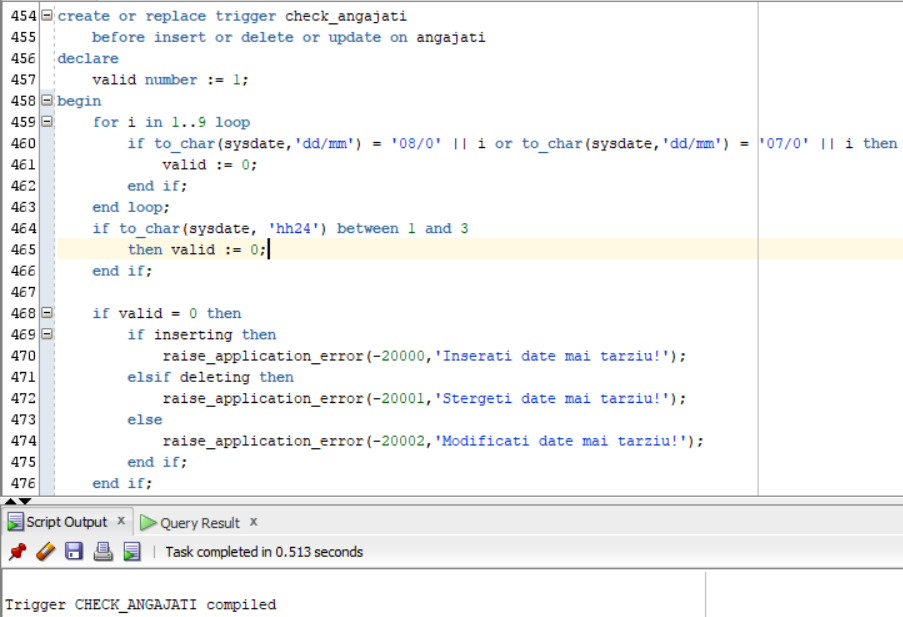
end;



Cerința 10.

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

Trigger LMD la nivel de comanda care nu permite executarea unui comenzi pe tabelul angajati intre orele 01:00 - 03:00 in ziua de 8 si 9 a lunilor din intervalul ianuarie - septembrie.



create or replace trigger check\_angajati

before insert or delete or update on angajati

declare

valid number := 1;

begin

for i in 1..9 loop

if to\_char(sysdate,'dd/mm') = '08/0' || i or to\_char(sysdate,'dd/mm') = '07/0' || i then

valid := 0;

end if;

end loop;

if to\_char(sysdate, 'hh24') between 1 and 3

then valid := 0;

end if;

if valid = 0 then

if inserting then

raise\_application\_error(-20000,'Inserati date mai tarziu!');

elsif deleting then

raise\_application\_error(-20001,'Stergeti date mai tarziu!');

else

raise\_application\_error(-20002,'Modificati date mai tarziu!');

end if;

end if;

end;

/

**-- inserare**

insert into angajati

values (secventa\_techflow.nextval, 'Gabroveanu', 'Ion', 'gabroveanu.ion@example.com', 2500.00, 'Casier', 1);

**-- stergere**

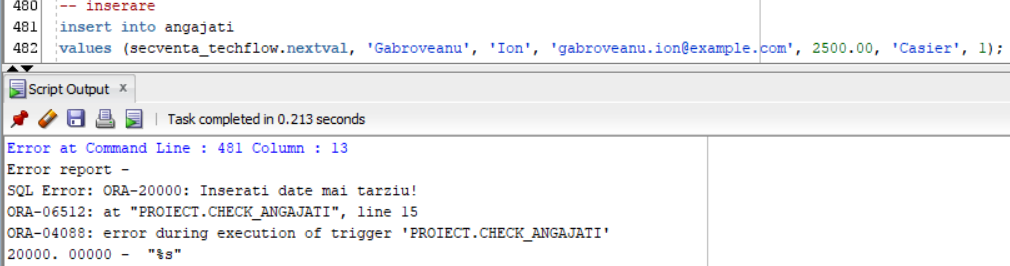
delete from angajati where id\_angajat = 6;

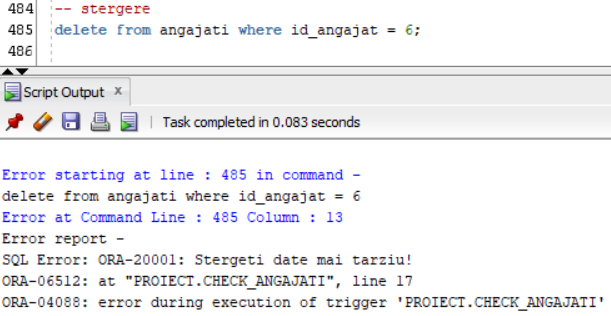
**-- modificare**

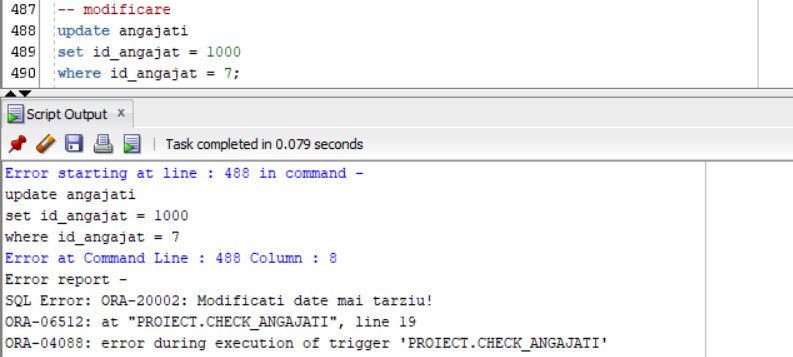
update angajati

set id\_angajat = 1000

where id\_angajat = 7;







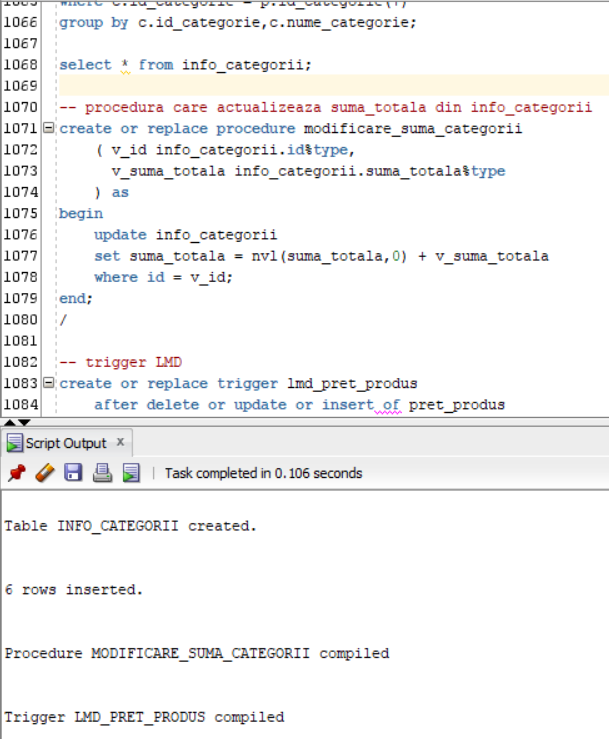
Cerința 11.

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

* Varianta 1

(se va folosi o procedura de actualizare a sumei totale in trigger)

Trigger LMD la nivel de linie care va actualiza automat campul suma\_totala din tabelul info\_categorii cand se introduce un nou produs, se sterge un produs sau se modifica pretul produsului.



**-- Tabel info\_categorii in care suma\_totala reprezinta suma preturilor**

**-- produselor din categoria respectiva**

create table info\_categorii

(id number(3) primary key,

nume\_categorie varchar2(50),

suma\_totala number);

**-- adaugare date in tabel**

insert into info\_categorii

select c.id\_categorie,c.nume\_categorie,nvl(sum(p.pret\_produs),0)

from categorii c, produse p

where c.id\_categorie = p.id\_categorie(+)

group by c.id\_categorie,c.nume\_categorie;

select \* from info\_categorii;

**-- procedura care actualizeaza suma\_totala din info\_categorii**

create or replace procedure modificare\_suma\_categorii

( v\_id info\_categorii.id%type,

v\_suma\_totala info\_categorii.suma\_totala%type

) as

begin

update info\_categorii

set suma\_totala = nvl(suma\_totala,0) + v\_suma\_totala

where id = v\_id;

end;

/

**-- trigger LMD**

create or replace trigger lmd\_pret\_produs

after delete or update or insert of pret\_produs

on produse

for each row – la nivel de linie

begin

if deleting then

-- stergerea unui produs

modificare\_suma\_categorii(:old.id\_categorie,-1\*:old.pret\_produs);

elsif updating then

-- modificarea unui produs

modificare\_suma\_categorii(:old.id\_categorie,:new.pret\_produs-:old.pret\_produs);

else

-- adaugarea unui produs

modificare\_suma\_categorii(:new.id\_categorie,:new.pret\_produs);

end if;

end;

/

select \* from info\_categorii;

select \* from produse;

**-- stergere produs**

delete from produse where id\_produs = 75;

**-- modificare produs**

update produse

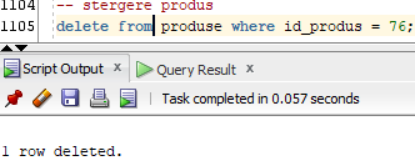
set pret\_produs = 1999.99

where id\_produs = 27;

**-- adaugare produs**

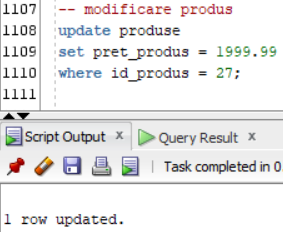
insert into produse

values (secventa\_techflow.nextval, ‘Microfon Test’, ‘-‘, 1799.99, 25, 16, 71);

**stergere**

inainte de trigger dupa trigger





**modificare**

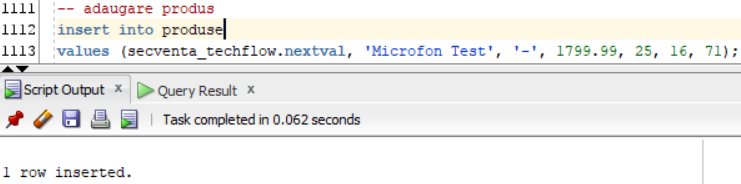
inainte de trigger dupa trigger



**inserare**

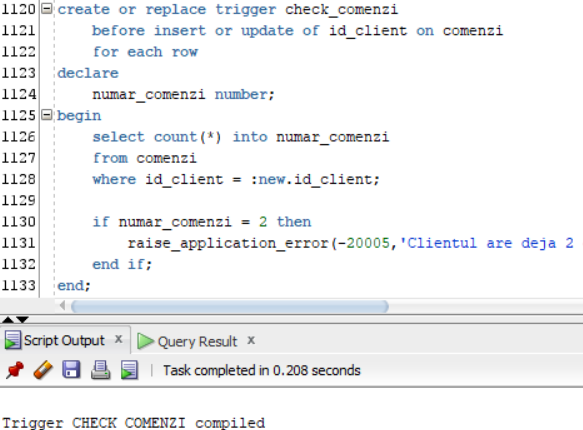
inainte de trigger dupa trigger





* Varianta 2

(trigger pe tabela mutating) Trigger LMD la nivel de linie care sa nu permita ca un client sa aiba mai mult de 2 comenzi.



-- Varianta 2

-- (trigger pe tabela mutating)

-- Trigger LMD la nivel de linie care sa nu permita ca un client sa aiba mai mult

-- de 2 comenzi.

create or replace trigger check\_comenzi

before insert or update of id\_client on comenzi

for each row

declare

numar\_comenzi number;

begin

select count(\*) into numar\_comenzi

from comenzi

where id\_client = :new.id\_client;

if numar\_comenzi = 2 then

raise\_application\_error(-20005,'Clientul are deja 2 comenzi in desfasurare!');

end if;

end;

/

select \* from clienti;

select \* from comenzi;

-- nu se declanseaza triggerul deoarece clientul are 0 comenzi

insert into comenzi

values (secventa\_techflow.nextval, sysdate, 'Noua', 72, 36, 42);

-- clientul are numarul maxim de comenzi

insert into comenzi

values (secventa\_techflow.nextval, sysdate, 'Noua', 46, 38, 44);

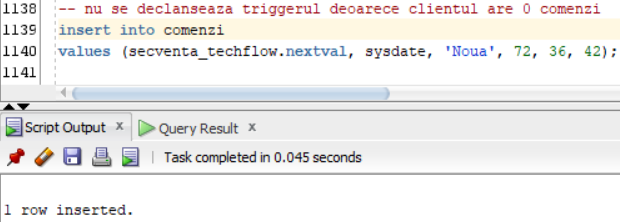
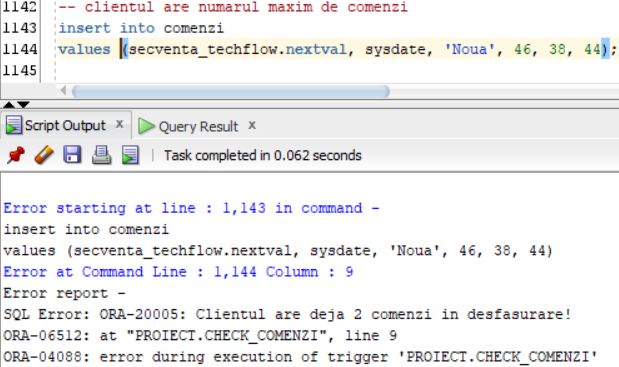
-- eroare mutating

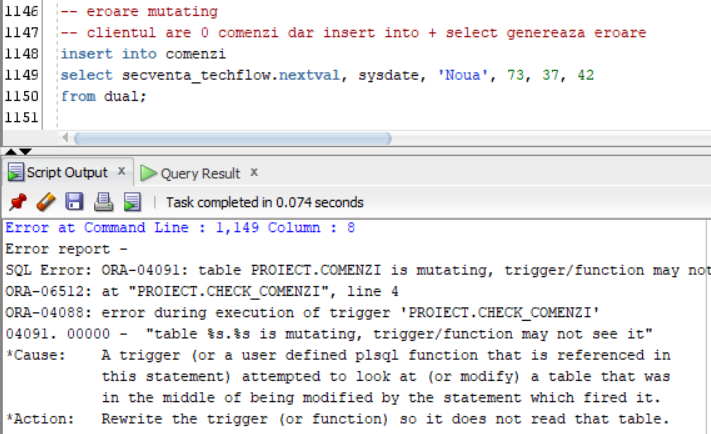
-- clientul are 0 comenzi dar insert into + select genereaza eroare

insert into comenzi

select secventa\_techflow.nextval, sysdate, 'Noua', 73, 37, 42

from dual;





-- rezolvarea erorii de tabel mutating

create or replace package pachet\_check\_comenzi

as

type com\_rec\_pachet is record ( id\_client clienti.id\_client%type,

numar\_comenzi number(1));

type com\_idx is table of com\_rec\_pachet index by pls\_integer;

comenzi com\_idx;

contor number(1) :=0;

end;

/

create or replace trigger check\_comenzi\_comanda

before insert or update of id\_client on comenzi

begin

pachet\_check\_comenzi.contor := 0;

select id\_client, count(\*) bulk collect into pachet\_check\_comenzi.comenzi

from comenzi

group by id\_client;

end;

/

create or replace trigger check\_comenzi\_linie

before insert or update of id\_client on comenzi

for each row

begin

for i in 1..pachet\_check\_comenzi.comenzi.last loop

if pachet\_check\_comenzi.comenzi(i).numar\_comenzi +

pachet\_check\_comenzi.contor = 2

and pachet\_check\_comenzi.comenzi(i).id\_client = :new.id\_client

then

raise\_application\_error(-20005,'Clientul are deja 2 comenzi in desfasurare!');

end if;

end loop;

pachet\_check\_comenzi.contor := pachet\_check\_comenzi.contor + 1;

end;

/

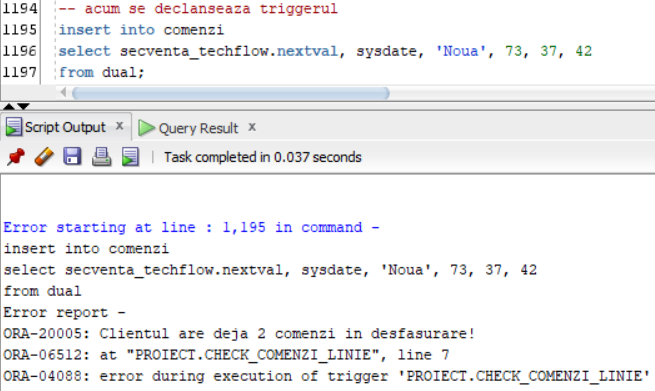
alter trigger proiect.check\_comenzi disable;

-- acum se declanseaza triggerul

insert into comenzi

select secventa\_techflow.nextval, sysdate, 'Noua', 73, 37, 42

from dual;



Cerința 12.

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

Trigger LDD care blocheaza adaugarea/stergerea/modificarea tabelelor in zilele de sambata si duminica. Daca se realizeaza o operatie atunci informatiile sunt inserate in tabelul audit\_user

create table audit\_user

(nume\_bd varchar2(50),

user\_logat varchar2(30),

eveniment varchar2(20),

tip\_obiect\_referit varchar2(30),

nume\_obiect\_referit varchar2(30),

data timestamp(3));

create or replace trigger check\_schema

after create or drop or alter on schema

begin

if to\_char(sysdate,'day') = 'sunday' or to\_char(sysdate,'day') = 'saturday'

then

raise\_application\_error(-20000,'Nu se pot face operatii LDD in weekend!');

end if;

insert into audit\_user

values (sys.database\_name, sys.login\_user,

sys.sysevent, sys.dictionary\_obj\_type,

sys.dictionary\_obj\_name, systimestamp(3));

end;

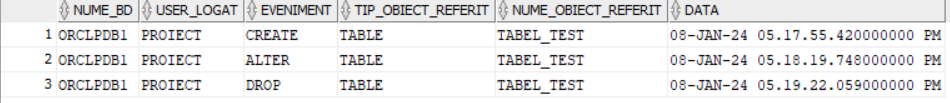
/

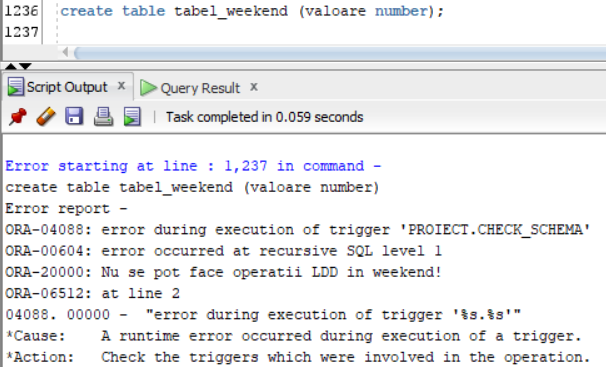
create table tabel\_test (valoare number);

alter table tabel\_test add (alta\_valoare number);

drop table tabel\_test;

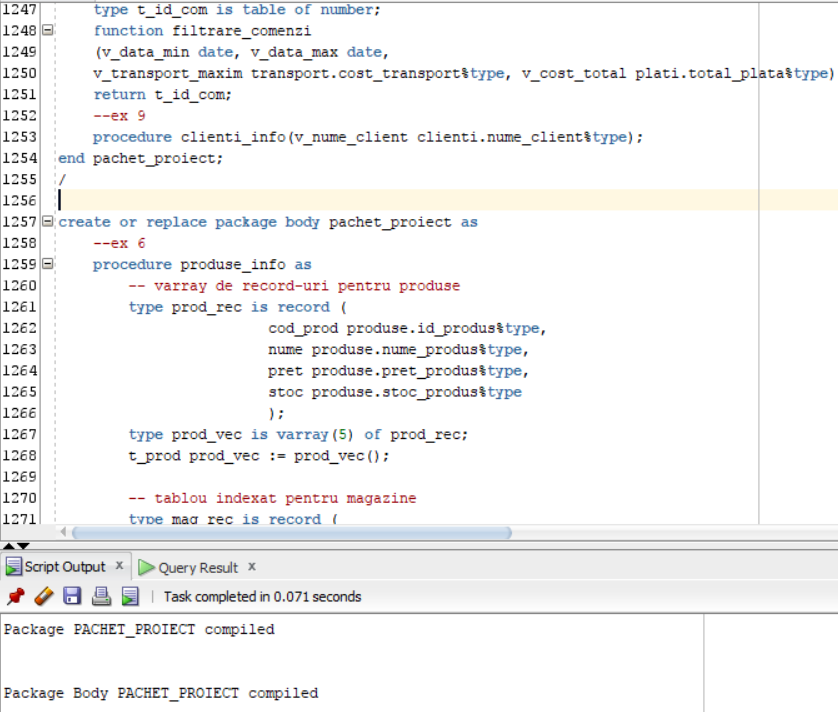
select \* from audit\_user;

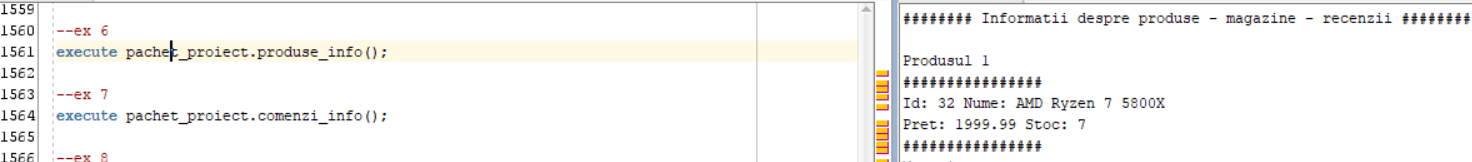




Cerința 13.

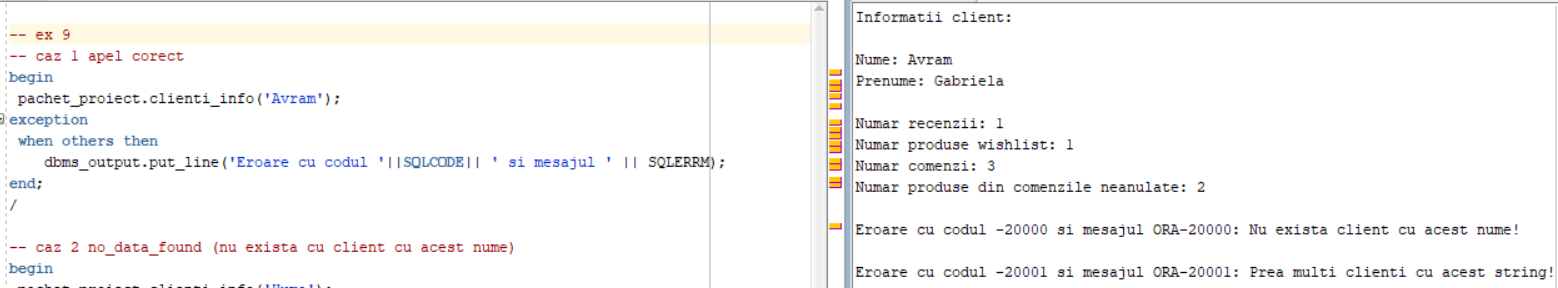
Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.











create or replace package pachet\_proiect as

--ex 6

procedure produse\_info;

--ex 7

procedure comenzi\_info;

--ex 8

type t\_id\_com is table of number;

function filtrare\_comenzi

(v\_data\_min date, v\_data\_max date,

v\_transport\_maxim transport.cost\_transport%type, v\_cost\_total plati.total\_plata%type)

return t\_id\_com;

--ex 9

procedure clienti\_info(v\_nume\_client clienti.nume\_client%type);

end pachet\_proiect;

/

create or replace package body pachet\_proiect as

--ex 6

procedure produse\_info as

-- varray de record-uri pentru produse

type prod\_rec is record (

cod\_prod produse.id\_produs%type,

nume produse.nume\_produs%type,

pret produse.pret\_produs%type,

stoc produse.stoc\_produs%type

);

type prod\_vec is varray(5) of prod\_rec;

t\_prod prod\_vec := prod\_vec();

-- tablou indexat pentru magazine

type mag\_rec is record (

nume magazine.nume\_magazin%type,

adresa magazine.adresa\_magazin%type

);

type mag\_ind is table of mag\_rec index by binary\_integer;

t\_mag mag\_ind;

-- tablou imbricat pentru recenzii

type recenzii\_rec is record (

nota recenzii.nota\_recenzie%type,

nume\_client clienti.nume\_client%type

);

type rec\_imbr is table of recenzii\_rec;

t\_recenzii rec\_imbr := rec\_imbr();

begin

dbms\_output.put\_line('######## Informatii despre produse - magazine - recenzii ########');

dbms\_output.new\_line();

-- atribuire produse

select id\_produs,nume\_produs,pret\_produs,stoc\_produs bulk collect into t\_prod

from produse

order by nume\_produs

fetch first 5 rows only;

for i in t\_prod.first..t\_prod.last loop

-- afisare info produs

dbms\_output.put\_line('Produsul ' || i);

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

dbms\_output.put\_line('Id: ' || t\_prod(i).cod\_prod || ' Nume: ' || t\_prod(i).nume);

dbms\_output.put\_line('Pret: ' || t\_prod(i).pret || ' Stoc: ' || t\_prod(i).stoc);

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

-- atribuire magazine

select m.nume\_magazin,m.adresa\_magazin bulk collect into t\_mag

from magazine m

join produse\_magazine pm on (m.id\_magazin = pm.id\_magazin)

where pm.id\_produs = t\_prod(i).cod\_prod;

-- afisare info magazin

if t\_mag.count = 0 then

dbms\_output.put\_line('Produsul nu se afla intr-un magazin!');

dbms\_output.new\_line;

else

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

for j in t\_mag.first..t\_mag.last loop

dbms\_output.put\_line('Magazinul ' || j);

dbms\_output.put\_line('Nume: ' || t\_mag(j).nume || ' Adresa: ' || t\_mag(j).adresa);

dbms\_output.new\_line();

end loop;

end if;

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

-- atribuire recenzii

select r.nota\_recenzie,c.nume\_client bulk collect into t\_recenzii

from recenzii r

join clienti c on (r.id\_client = c.id\_client)

where r.id\_produs = t\_prod(i).cod\_prod;

-- afisare recenzii

if t\_recenzii.count = 0 then

dbms\_output.put\_line('Produsul nu are recenzii!');

dbms\_output.new\_line;

else

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

for k in t\_recenzii.first..t\_recenzii.last loop

dbms\_output.put\_line('Recenzia ' || k);

dbms\_output.put\_line('Nota: ' || t\_recenzii(k).nota || ' Nume Client: ' || t\_recenzii(k).nume\_client);

dbms\_output.new\_line();

end loop;

end if;

dbms\_output.new\_line();

end loop;

end produse\_info;

--ex 7

procedure comenzi\_info as

type com\_rec is record (

data\_plasare comenzi.data\_plasare%type,

status\_comanda comenzi.status\_comanda%type,

total\_plata plati.total\_plata%type,

tip\_plata plati.tip\_plata%type

);

type prod\_rec is record (

nume produse.nume\_produs%type,

id\_produs produse.id\_produs%type

);

type t\_prod is table of prod\_rec

index by binary\_integer;

v\_prod t\_prod;

v\_com com\_rec;

-- cursor dinamic/ expresie cursor

-- pentru fiecare categorie se determina produsul cu cel mai mare stoc

cursor categorii\_info is

select id\_categorie,nume\_categorie,

cursor(select max(nume\_produs),max(id\_produs)

from produse p

where c.id\_categorie = p.id\_categorie

group by p.id\_produs

order by max(stoc\_produs) desc)

from categorii c;

-- cursor parametrizat pentru comenzi

cursor com\_info(v\_produs produse.id\_produs%type) is

select c.data\_plasare,c.status\_comanda,p.total\_plata,p.tip\_plata

from comenzi c

join plati p on (c.id\_plata = p.id\_plata)

join produse\_comenzi pc on (c.id\_comanda = pc.id\_comanda)

where pc.id\_produs = v\_produs

order by c.data\_plasare desc;

v\_cursor sys\_refcursor;

v\_id\_categorie categorii.id\_categorie%type;

v\_nume\_categorie categorii.nume\_categorie%type;

contor number := 1;

begin

dbms\_output.put\_line('######## Informatii despre categorii - produse - comenzi - plati ########');

open categorii\_info;

loop

fetch categorii\_info into v\_id\_categorie,v\_nume\_categorie,v\_cursor;

exit when categorii\_info%notfound;

dbms\_output.put\_line('Categoria: ' || v\_nume\_categorie || ' cu id-ul: ' || v\_id\_categorie);

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

fetch v\_cursor bulk collect into v\_prod limit 2;

dbms\_output.new\_line;

if v\_prod.count = 0 then

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

else

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

dbms\_output.put\_line('Produs '||i|| '. '|| v\_prod(i).nume);

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

open com\_info(v\_prod(i).id\_produs);

loop

fetch com\_info into v\_com;

exit when com\_info%notfound;

dbms\_output.put\_line('Comanda '||contor);

dbms\_output.put\_line('Data Plasare: ' || v\_com.data\_plasare || ' Status Comanda: ' || v\_com.status\_comanda);

dbms\_output.put\_line('Total Plata: ' || v\_com.total\_plata || ' Tip Plata: ' || v\_com.tip\_plata);

dbms\_output.new\_line;

contor := contor + 1;

end loop;

if com\_info%rowcount = 0 then

dbms\_output.put\_line('Nu are comenzi!');

end if;

close com\_info;

contor := 1;

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

dbms\_output.new\_line;

end loop;

end if;

dbms\_output.new\_line;

end loop;

close categorii\_info;

end comenzi\_info;

--ex 8

function filtrare\_comenzi

(v\_data\_min date, v\_data\_max date, v\_transport\_maxim transport.cost\_transport%type, v\_cost\_total plati.total\_plata%type)

return t\_id\_com is

v\_id\_com t\_id\_com;

data\_eroare exception;

transport\_eroare exception;

cost\_total\_eroare exception;

no\_comenzi exception;

begin

if v\_data\_min > v\_data\_max then

raise data\_eroare;

end if;

if v\_transport\_maxim < 0 then

raise transport\_eroare;

end if;

if v\_cost\_total < 0 then

raise cost\_total\_eroare;

end if;

-- folosirea a 3 tabele

select distinct c.id\_comanda bulk collect into v\_id\_com

from comenzi c

join plati p on (c.id\_plata = p.id\_plata)

join transport t on (c.id\_transport = c.id\_transport)

where c.data\_plasare >= v\_data\_min and c.data\_plasare <= v\_data\_max

and t.cost\_transport <= v\_transport\_maxim

and p.total\_plata <= v\_cost\_total;

if v\_id\_com.count = 0 then

raise no\_comenzi;

end if;

return v\_id\_com;

exception

when no\_comenzi then

raise\_application\_error(-20000,'Nu exista comanda cu aceste conditii!');

when data\_eroare then

raise\_application\_error(-20001,'Intervalul nu este introdus corect!');

when transport\_eroare then

raise\_application\_error(-20002,'Costul transportului nu poate fi negativ!');

when cost\_total\_eroare then

raise\_application\_error(-20003,'Costul total nu poate fi negativ!');

when others then

raise\_application\_error(-20004,'Alta eroare!');

end filtrare\_comenzi;

--ex 9

procedure clienti\_info(v\_nume\_client clienti.nume\_client%type)

is

type info\_rec is record (

nr\_recenzii integer,

nr\_wishlist integer,

nr\_comenzi integer,

nr\_produse\_neanulate integer,

nume\_client clienti.nume\_client%type,

prenume\_client clienti.prenume\_client%type

);

l\_info info\_rec;

type nume\_client\_imbr is table of clienti.nume\_client%type;

t\_nume\_client nume\_client\_imbr := nume\_client\_imbr();

begin

select nume\_client bulk collect into t\_nume\_client

from clienti

where nume\_client like v\_nume\_client || '%';

if t\_nume\_client.count = 0 then

raise no\_data\_found;

elsif t\_nume\_client.count > 1 then

raise too\_many\_rows;

else

dbms\_output.put\_line('Informatii client: ');

dbms\_output.new\_line;

-- 6 tabele intr-o comanda sql

select

(select count(\*) from recenzii r

join clienti cl on (cl.id\_client = r.id\_client)

where cl.nume\_client like v\_nume\_client

)as nr\_recenzii,

(select count(\*) from wishlist w

join clienti cl on (cl.id\_client = w.id\_client)

where cl.nume\_client like v\_nume\_client) as nr\_wishlist,

(select count(\*) from comenzi c

join clienti cl on (cl.id\_client = c.id\_client)

where cl.nume\_client like v\_nume\_client)as nr\_comenzi,

(select count(\*) from produse p

join produse\_comenzi pc on (pc.id\_produs = p.id\_produs)

join comenzi c on (c.id\_comanda = pc.id\_comanda)

join clienti cl on (cl.id\_client = c.id\_client)

where cl.nume\_client like v\_nume\_client and c.status\_comanda <> 'Anulata'

) as nr\_produse\_neanulate,

cl.nume\_client,cl.prenume\_client into l\_info

from clienti cl

where cl.nume\_client like v\_nume\_client

dbms\_output.put\_line('Nume: ' || l\_info.nume\_client);

dbms\_output.put\_line('Prenume: ' || l\_info.prenume\_client);

dbms\_output.new\_line;

dbms\_output.put\_line('Numar recenzii: ' || l\_info.nr\_recenzii);

dbms\_output.put\_line('Numar produse wishlist: ' || l\_info.nr\_wishlist);

dbms\_output.put\_line('Numar comenzi: ' || l\_info.nr\_comenzi);

dbms\_output.put\_line('Numar produse din comenzile neanulate: ' || l\_info.nr\_produse\_neanulate);

end if;

exception

when no\_data\_found then

raise\_application\_error(-20000,'Nu exista client cu acest nume!');

when too\_many\_rows then

raise\_application\_error(-20001,'Prea multi clienti cu acest string!');

when others then

raise\_application\_error(-20002,'Alta eroare!');

end clienti\_info;

end pachet\_proiect;

/

--ex 6

execute pachet\_proiect.produse\_info();

--ex 7

execute pachet\_proiect.comenzi\_info();

--ex 8

-- caz 1 apel corect

declare

solutie pachet\_proiect.t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2023 cu totatul de plata mai mic de 5000');

dbms\_output.put\_line('si costul de transport mai mic de 190.');

solutie := pachet\_proiect.filtrare\_comenzi(to\_date('10-may-2023', 'dd-mon-yyyy'),to\_date('15-may-2023', 'dd-mon-yyyy'),190,5000);

dbms\_output.new\_line;

dbms\_output.put\_line('Id-uri comenzi care respecta criteriul:');

for i in solutie.first..solutie.last loop

dbms\_output.put\_line(solutie(i)|| ' ');

end loop;

end;

/

-- caz2 exceptie no\_comenzi

-- caz in care nu exista comenzi care sa respecte conditiile

declare

solutie pachet\_proiect.t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2022 cu totatul de plata mai mic de 3000');

dbms\_output.put\_line('si costul de transport mai mic de 100.');

dbms\_output.new\_line;

solutie := pachet\_proiect.filtrare\_comenzi(to\_date('10-may-2022', 'dd-mon-yyyy'),to\_date('15-may-2022', 'dd-mon-yyyy'),100,3000);

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

-- caz3 exceptie data\_eroare

-- caz in care data minima este mai mare decat data maxima

declare

solutie pachet\_proiect.t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 15-10 mai 2022 cu totatul de plata mai mic de 3000');

dbms\_output.put\_line('si costul de transport mai mic de 100.');

dbms\_output.new\_line;

solutie := pachet\_proiect.filtrare\_comenzi(to\_date('15-may-2022', 'dd-mon-yyyy'),to\_date('10-may-2022', 'dd-mon-yyyy'),100,3000);

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

-- caz4 exceptie transport\_eroare

-- caz in care costul pentru transport maximal este negativ

declare

solutie pachet\_proiect.t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2023 cu totatul de plata mai mic de 3000');

dbms\_output.put\_line('si costul de transport mai mic de -10.');

dbms\_output.new\_line;

solutie := pachet\_proiect.filtrare\_comenzi(to\_date('10-may-2022', 'dd-mon-yyyy'),to\_date('15-may-2022', 'dd-mon-yyyy'),-10,3000);

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

-- caz5 exceptie cost\_total\_eroare

-- caz in care costul total maximal este negativ

declare

solutie pachet\_proiect.t\_id\_com;

begin

dbms\_output.put\_line('Comenzi intre 10-15 mai 2023 cu totatul de plata mai mic de -50');

dbms\_output.put\_line('si costul de transport mai mic de 200.');

dbms\_output.new\_line;

solutie := pachet\_proiect.filtrare\_comenzi(to\_date('10-may-2022', 'dd-mon-yyyy'),to\_date('15-may-2022', 'dd-mon-yyyy'),200,-50);

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

-- ex 9

-- caz 1 apel corect

begin

pachet\_proiect.clienti\_info('Avram');

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

-- caz 2 no\_data\_found (nu exista cu client cu acest nume)

begin

pachet\_proiect.clienti\_info('Huma');

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

/

-- caz 3 too\_many\_rows (pentru Pop => exista Popa si Popescu)

begin

pachet\_proiect.clienti\_info('Pop');

exception

when others then

dbms\_output.put\_line('Eroare cu codul '||SQLCODE|| ' si mesajul ' || SQLERRM);

end;

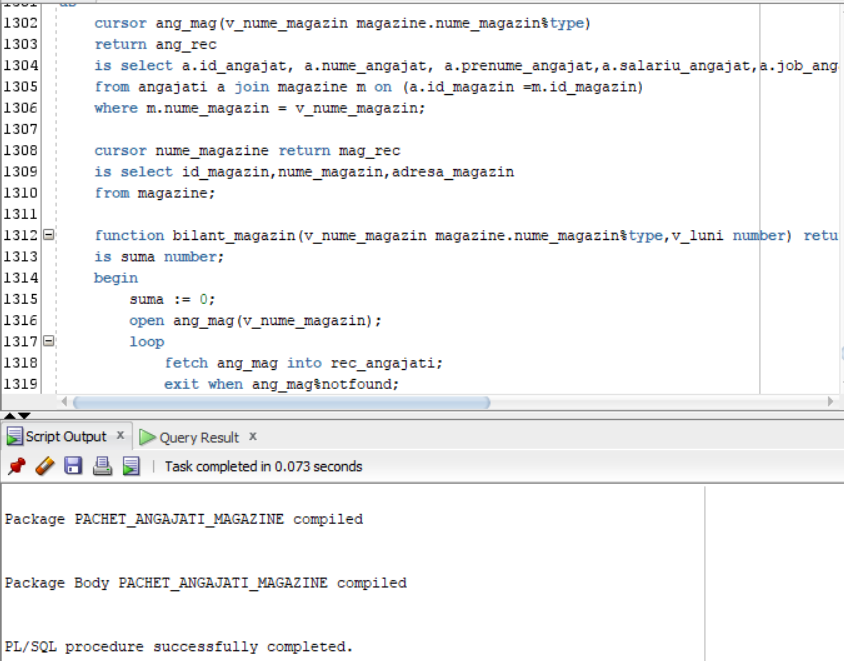
/

Cerința 14.

Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite.

Pachetul angajati\_magazine se ocupa cu realizarea unui bilant (suma salariilor angajatilor) pe un anumit magazin sau pe toata compania (overload de functii si metode) pe un anumit numar de luni. De asemenea pachetul cuprinde proceduri standard care determina bilantul anual si semestrial.

* 4 functii, 4 proceduri, 2 cursoare



create or replace package pachet\_angajati\_magazine

as

-- record info angajati

type ang\_rec is record

(id angajati.id\_angajat%type,

nume angajati.nume\_angajat%type,

prenume angajati.prenume\_angajat%type,

salariu angajati.salariu\_angajat%type,

job angajati.job\_angajat%type);

-- record info magazine

type mag\_rec is record

(id magazine.id\_magazin%type,

nume magazine.nume\_magazin%type,

adresa magazine.adresa\_magazin%type

);

-- tablou angajati

type t\_ang is table of ang\_rec

index by binary\_integer;

vec\_angajati t\_ang;

rec\_angajati ang\_rec;

rec\_magazine mag\_rec;

-- cursor pentru angajati

cursor ang\_mag(v\_nume\_magazin magazine.nume\_magazin%type) return ang\_rec;

-- cursor magazine

cursor nume\_magazine return mag\_rec;

-- returneaza suma salariilor angajatilor dintr-un magazin dat

-- pe un numar de luni dat

function bilant\_magazin(v\_nume\_magazin magazine.nume\_magazin%type,v\_luni number)

return number;

-- returneaza suma salariilor angajatilor din toate magazinele

-- pe un numar de luni dat

function bilant\_magazin(v\_luni number)

return number;

-- afiseaza suma salariilor angajatilor din toate magazinele

-- pe 12 luni

procedure afisare\_bilant\_anual;

-- afiseaza suma salariilor angajatilor dintr-un magazin dat

-- pe 12 luni

procedure afisare\_bilant\_anual(v\_nume\_magazin magazine.nume\_magazin%type);

-- afiseaza suma salariilor angajatilor din toate magazinele

-- pe 6 luni

procedure afisare\_bilant\_semestrial;

-- afiseaza suma salariilor angajatilor dintr-un magazin dat

-- pe 6 luni

procedure afisare\_bilant\_semestrial(v\_nume\_magazin magazine.nume\_magazin%type);

end pachet\_angajati\_magazine;

/

create or replace package body pachet\_angajati\_magazine

as

cursor ang\_mag(v\_nume\_magazin magazine.nume\_magazin%type)

return ang\_rec

is select a.id\_angajat, a.nume\_angajat, a.prenume\_angajat,a.salariu\_angajat,a.job\_angajat

from angajati a join magazine m on (a.id\_magazin =m.id\_magazin)

where m.nume\_magazin = v\_nume\_magazin;

cursor nume\_magazine return mag\_rec

is select id\_magazin,nume\_magazin,adresa\_magazin

from magazine;

function bilant\_magazin(v\_nume\_magazin magazine.nume\_magazin%type,v\_luni number) return number

is suma number;

begin

suma := 0;

open ang\_mag(v\_nume\_magazin);

loop

fetch ang\_mag into rec\_angajati;

exit when ang\_mag%notfound;

suma := suma + rec\_angajati.salariu \* v\_luni;

end loop;

close ang\_mag;

return suma;

end bilant\_magazin;

function bilant\_magazin(v\_luni number) return number

is suma number;

begin

suma := 0;

open nume\_magazine;

loop

fetch nume\_magazine into rec\_magazine;

exit when nume\_magazine%notfound;

suma := suma + bilant\_magazin(rec\_magazine.nume,v\_luni);

end loop;

close nume\_magazine;

return suma;

end bilant\_magazin;

procedure afisare\_bilant\_anual is

bilant number := bilant\_magazin(12);

begin

dbms\_output.put\_line('Bilantul anual pentru Compania TechFlow este: ');

dbms\_output.put\_line(bilant || ' $');

end afisare\_bilant\_anual;

procedure afisare\_bilant\_semestrial is

bilant number := bilant\_magazin(6);

begin

dbms\_output.put\_line('Bilantul semestrial pentru Compania TechFlow este: ');

dbms\_output.put\_line(bilant || ' $');

end afisare\_bilant\_semestrial;

procedure afisare\_bilant\_anual(v\_nume\_magazin magazine.nume\_magazin%type) is

bilant number := bilant\_magazin(v\_nume\_magazin,12);

begin

dbms\_output.put\_line('Bilantul anual pentru magazinul ' || v\_nume\_magazin || ' este: ');

dbms\_output.put\_line(bilant || ' $');

end afisare\_bilant\_anual;

procedure afisare\_bilant\_semestrial(v\_nume\_magazin magazine.nume\_magazin%type) is

bilant number := bilant\_magazin(v\_nume\_magazin,6);

begin

dbms\_output.put\_line('Bilantul semestrial pentru magazinul ' || v\_nume\_magazin || ' este: ');

dbms\_output.put\_line(bilant || ' $');

end afisare\_bilant\_semestrial;

end pachet\_angajati\_magazine;

/

begin

pachet\_angajati\_magazine.afisare\_bilant\_anual();

pachet\_angajati\_magazine.afisare\_bilant\_semestrial();

for i in pachet\_angajati\_magazine.nume\_magazine loop

pachet\_angajati\_magazine.afisare\_bilant\_anual(i.nume);

pachet\_angajati\_magazine.afisare\_bilant\_semestrial(i.nume);

end loop;

end;

