

# **PROIECT SISTEME DE GESTIUNE A BAZELEOR DE DATE**

**ANUL II**

***Tema proiectului: Magazin de electronice***

***Nume: Petre-Şoldan Adela***

***Grupa: 231***

***Seria: 23***

# CUPRINS

1) Prezentați pe scurt baza de date (utilitatea ei) .....	3
2) Realizați diagrama entitate-relație (ERD): entitățile, relațiile și atributele trebuie definite în limba română .....	4
3) Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare: entitățile, relațiile și atributele trebuie definite în limba română .....	5
4) Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, definind toate constrângările de integritate necesare (chei primare, cheile externe etc) .....	6
5) Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă) .....	19
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 .....	52
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 .....	63
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 .....	68
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 .....	79
10) Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul .....	92
11) Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul .....	101
12) Definiți un trigger de tip LDD. Declanșați trigger-ul .....	122
13) Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului .....	125
14) Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri) .....	149

## 1) Prezentare pe scurt:

Modelul de date urmărește activitatea unui lanț de magazine care se ocupă cu vânzarea electronicelor și care are mai multe **sedii** pe teritoriul țării. Baza de date se concentrează pe entitatea **articol**, care are 2 subentități **produs** și **pachet\_promotional**. Un **pachet** are mai multe **produse**, iar ambele tipuri de **articol** oferă puncte de fidelitate pt cumpărarea lor.

Există mai mulți **furnizori** care alimentează **sediile** firmei, iar un **produs** aparține de un singur **furnizor**. **Pachetele** nu sunt furnizate, sunt împachetate de firmă din **produsele** furnizate. E ținută evidență pentru ce **articole** se află în fiecare **sediu**, cu preț și cantitate. Prețul poate difera de la un **sediu** la altul, iar prețul reținut în entitatea **articol** este cel de pe site, cu care poate fi achiziționat prin **comandă**. O **comandă** are asociat minim un **colet**, și **coletele** pot pleca din **sedii** diferite.

Printre **angajații** firmei se numără **administratorii**, **scenariștii** și **curierii**. Un **administrator** este responsabil de gestionarea și bunăstarea uneia sau mai multor **sedii**, iar într-un sediu lucrează un singur **administrator**. **Scenaristul** este persoana care scrie textul pentru o **reclamă**, iar într-o **reclamă** este prezentat un singur **articol**. **Reclama** poate vorbi și despre magazin în sine, caz în care cheia externă pt **articol** e nulă. La o **reclamă** lucrează doar un **scenarist**. Fiecare **scenarist** are dreptul să își aleagă o zi liberă în cadrul săptămânii, detaliu prevăzut în contractul de angajare. Pentru fiecare **reclamă** reținem și costul final necesar realizării sale, pentru o imagine de ansamblu a pierderilor și câștigurilor magazinului.

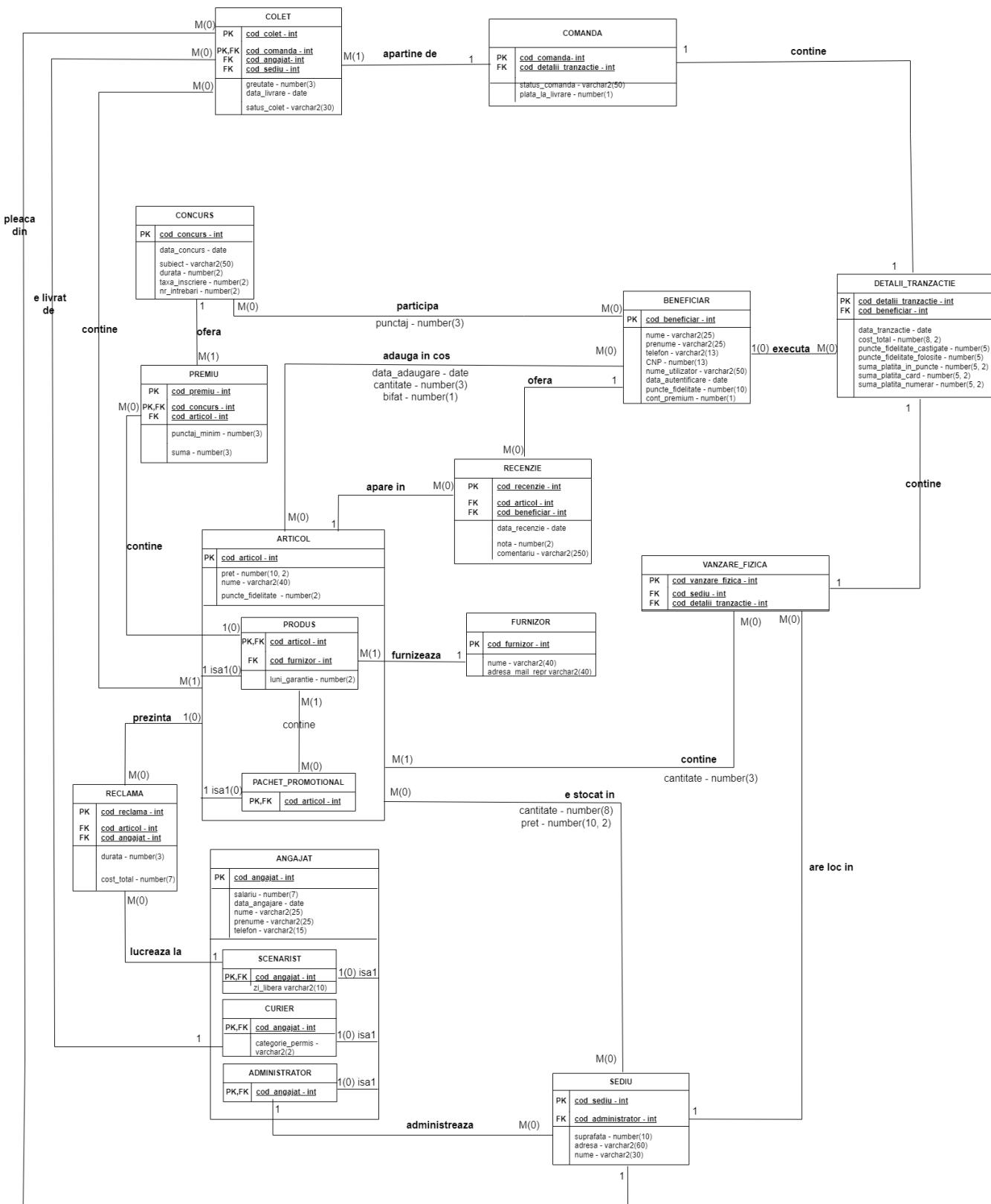
**Curierii** se ocupă de livrarea **coletelor**. Aplicația reține pentru un **colet** cine îl livrează, dar și statusuri pentru **colete** și **comenzi**, ca de exemplu „livrat”, „în procesare”.

Firma apelează la diverse metode de a se promova, printre care se numără organizarea de **concursuri** cu diverse **premii**. **Concursurile** vor avea loc online, au întrebări grilă și nu durează mai mult de 20 de minute. Baza nu reține și întrebările, doar detalii despre **concursuri** cum ar fi data la care are loc. Fiecare **premiu** are un punctaj minim necesar și constă într-un PRODUS din cele furnizate, o sumă de bani sau amândouă. **Beneficiarul** va primi **premiul** cu cel mai mare punctaj necesar care e mai mic sau egal cu punctajul obținut. Un **concurs** are mai multe **premii**, fiecare putând fi câștigat de mai multe persoane în funcție de scorurile obținute de acestea. Modelul oferă posibilitatea de a afla cine a participat și la ce concurs, precum și punctajul obținut. De aici se poate deduce și premiul câștigat de un concurrent, sau dacă nu a câștigat nimic.

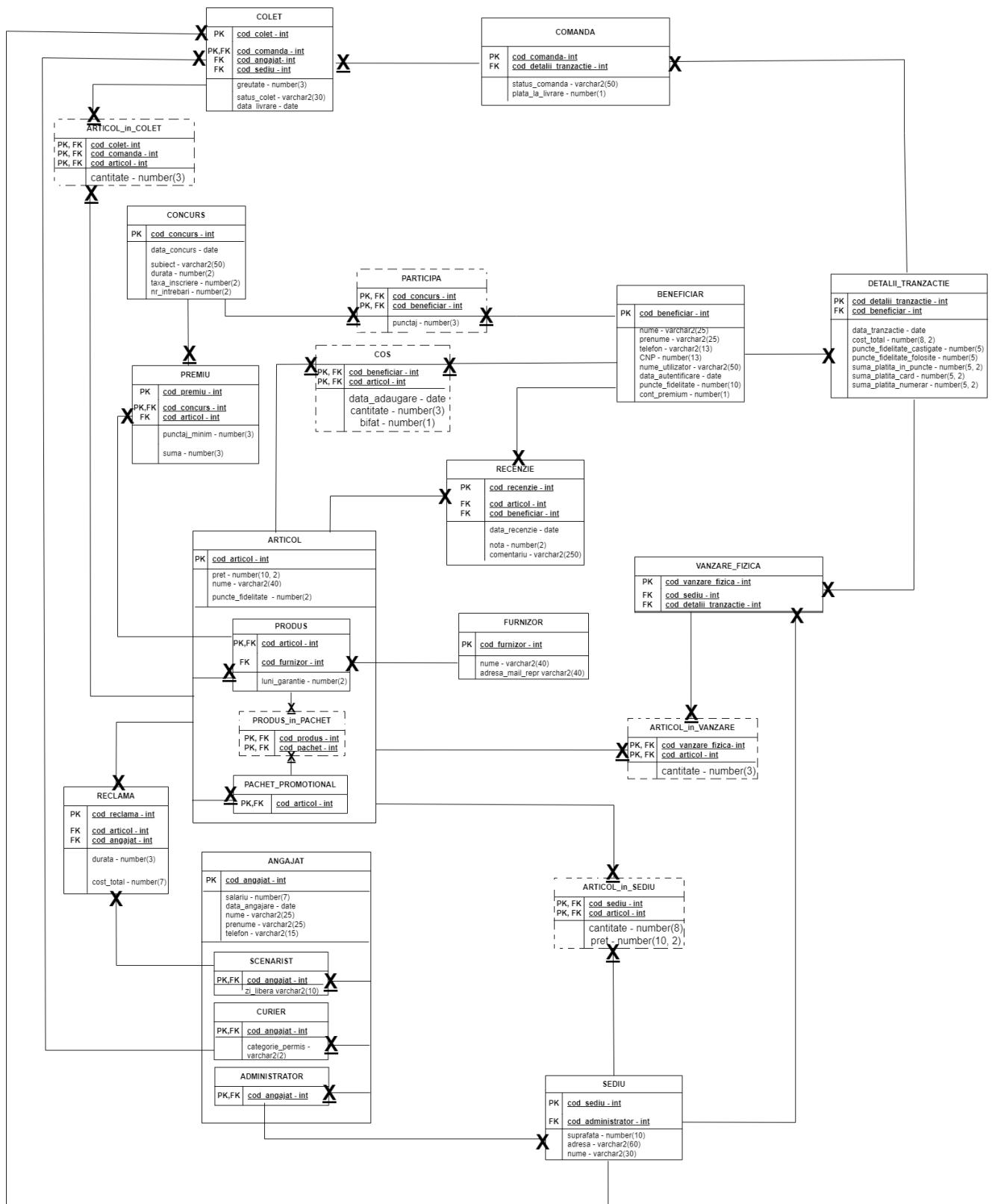
**Beneficiarii** pot scrie **recenzii** online pentru fiecare **articol** și pot de asemenea să le adauge în **cos**, unde vor rămâne până le vor elibera. Când vor finaliza **comanda**, **articolele** se vor debifa, dar vor rămâne în **cos**.

Fiecare tranzacție, fie vanzare\_fizica sau **comandă**, este legată de un tabel **detalii\_tranzactie** care oferă detalii precum prețul total, suma plătită cu cardul și punctele de fidelitate câștigate. De acesta e legat și **beneficiarul** care a realizat tranzacția. Acesta poate fi null în cazul **vânzărilor fizice** unde cumpărătorul nu avea card de fidelitate cu care să fi fost introdus în baza de date. Pentru acest tip de tranzacție reținem și **sediu** din care **beneficiarul** a achiziționat **articole**. Astfel, putem afla pentru un **beneficiar** tot ce a comandat online și tot ce a cumpărat fizic conectat la cardul său de fidelitate. Reținem și **articolele** din fiecare **colet** și **vanzare\_fizica**.

## 2) Diagrama entitate-relație:



### 3) Diagrama conceptuală:



#### 4) Creare tabele:

```
create table FURNIZOR (
    cod_furnizor int constraint cod_furnizor_pk primary key,
    nume varchar2(40) constraint nume_furnizor_nn NOT NULL,
    adresa_mail_repr varchar2(40),
    constraint nume_furnizor_uq unique(nume)
);

create table CONCURS (
    cod_concurs int constraint cod_concurs_pk primary key,
    data_concurs date default sysdate constraint data_concurs_nn NOT NULL,
    subiect varchar2(50) constraint subiect_nn NOT NULL,
    durata number(2) constraint durata_concurs NOT NULL,
    nr_intrebari number(2) default 20 constraint nr_intrebari_nn NOT NULL,
    taxa_inscriere number(2) constraint taxa_inscriere_nn NOT NULL,
    constraint durata_concurs_ck check(durata <= 20),
    constraint subiect_uq unique(subiect)
);

create table BENEFICIAR (
    codBeneficiar int,
    nume varchar2(25) constraint nume_beneficiar_nn NOT NULL,
    prenume varchar2(25) constraint prenume_beneficiar_nn NOT NULL,
    telefon varchar2(13),
    CNP number(13) constraint cnp_beneficiar_nn NOT NULL constraint
    cnp_beneficiar_uq unique,
    numeUtilizator varchar2(50) constraint nume_utilizator_nn NOT NULL,
    cont_premium number(1) default 0 constraint cont_premium_nn NOT NULL,
    data_autentificare date default sysdate constraint data_autentificare_nn NOT NULL,
    puncte_fidelitate number(10) default 0,
```

```
constraint puncte_fidelitate_ck check(puncte_fidelitate <= 10000),
constraint nume_utilizator_uq unique(nume_utilizator),
constraint cod_beneficiar_pk primary key(cod_beneficiar)
);
```

```
create table ARTICOL (
    cod_articol int,
    pret number(10, 2) constraint pret_articol_nn NOT NULL,
    nume varchar2(40) constraint nume_articol_nn NOT NULL,
    puncte_fidelitate number(2),
    constraint puncte_fidelitate_articol_ck check(puncte_fidelitate <= 30),
    constraint nume_articol_uq unique(nume),
    constraint cod_articol_pk primary key(cod_articol)
);
```

```
create table PACHET_PROMOTIONAL (
    cod_articol int constraint cod_pachet_pk primary key references
ARTICOL(cod_articol)
);
```

```
create table PRODUS (
    cod_articol int constraint cod_produs_pk primary key references
ARTICOL(cod_articol),
    cod_furnizor int,
    luni_garantie number(2) default 12,
    constraint produs_furnizor_fk foreign key(cod_furnizor) references
FURNIZOR(cod_furnizor)
);
```

```
create table PRODUS_in_PACHET (
```

```

cod_produs int,
cod_pachet int,
constraint produs_in_pachet_pk primary key(cod_produs, cod_pachet),
constraint produs_in_pachet_produs_fk foreign key(cod_produs) references
PRODUS(cod_articol),
constraint produs_in_pachet_pachet_fk foreign key(cod_pachet) references
PACHET_PROMOTIONAL(cod_articol)
);

```

```

create table PARTICIPA (
cod_concurs int,
cod_beneficiar int,
punctaj number(3) constraint punctaj_nn NOT NULL,
constraint participa_pk primary key(cod_concurs, cod_beneficiar),
constraint participa_concurs_fk foreign key(cod_concurs) references
CONCURS(cod_concurs),
constraint participa_beneficiar_fk foreign key(cod_beneficiar) references
BENEFICIAR(cod_beneficiar)
);

```

```

create table PREMIU (
cod_premiu int,
cod_concurs int,
cod_articol int,
punctaj_minim number(3) constraint punctaj_minim_nn NOT NULL,
suma number(6, 2) default 0,
constraint premiu_pk primary key(cod_premiu, cod_concurs),
constraint premiu_concurs_fk foreign key(cod_concurs) references
CONCURS(cod_concurs),
constraint premiu_produs_fk foreign key(cod_articol) references
PRODUS(cod_articol)
);

```

```
create table ANGAJAT (
    cod_angajat int,
    nume varchar2(25) constraint nume_angajat_nn NOT NULL,
    prenume varchar2(25) constraint prenume_angajat_nn NOT NULL,
    salariu number(7) default 3500,
    telefon varchar2(15),
    data_angajare date default sysdate constraint data_angajare_nn NOT NULL,
    constraint cod_angajat_pk primary key(cod_angajat)
);
```

```
create table SCENARIST (
    cod_angajat int constraint cod_scenarist_pk primary key references
ANGAJAT(cod_angajat),
    zi_libera varchar2(10)
);
```

```
create table ADMINISTRATOR (
    cod_angajat int constraint cod_administrator_pk primary key references
ANGAJAT(cod_angajat)
);
```

```
create table CURIER (
    cod_angajat int constraint cod_curier_pk primary key references
ANGAJAT(cod_angajat),
    categorie_permis varchar2(2) constraint categorie_permis_nn NOT NULL
);
```

```
create table RECLAMA (
    cod_reclama int constraint cod_reclama_pk primary key,
    cod_articol int,
```

```

cod_angajat int constraint cod_reclama_angajat_nn NOT NULL,
durata number(3) constraint durata_reclama_nn NOT NULL,
cost_total number(9, 2) constraint cost_total_nn NOT NULL,
constraint reclama_articol_fk foreign key(cod_articol) references
ARTICOL(cod_articol),
constraint reclama_scenarist_fk foreign key(cod_angajat) references
SCENARIST(cod_angajat)
);

```

```

create table SEDIU (
cod_sediu int constraint cod_sediu_pk primary key,
cod_administrator int constraint sediu_administrator_nn NOT NULL,
nume varchar2(30) constraint nume_sediu_nn NOT NULL,
adresa varchar2(60) constraint adresa_sediu_nn NOT NULL,
suprafata number(10) default 100000 constraint suprafata_sediu_nn NOT NULL,
constraint nume_sediu_uq unique(nume),
constraint sediu_administrator_fk foreign key(cod_administrator) references
ADMINISTRATOR(cod_angajat)
);

```

```

create table ARTICOL_in_SEDIU (
cod_sediu int,
cod_articol int,
cantitate number(8) constraint sediu_cantitate_nn NOT NULL,
pret number(10, 2) constraint sediu_pret_nn NOT NULL,
constraint articol_in_sediu_pk primary key(cod_sediu, cod_articol),
constraint articol_in_sediu_sediu_fk foreign key(cod_sediu) references
SEDIU(cod_sediu),
constraint articol_in_sediu_articol_fk foreign key(cod_articol) references
ARTICOL(cod_articol)
);

```

```

create table DETALII_TRANZACTIE (

```

```

cod_detalii_tranzactie int constraint cod_detalii_tranzactie_pk primary key,
cod_beneficiar int,
data_tranzactie date default sysdate constraint data_tranzactie_nn NOT NULL,
cost_total number(10, 2) constraint cost_tranzactie_nn NOT NULL,
puncte_fidelitate_castigate number(8) default 0 constraint
tranzactie_puncte_fidelitate_castigate_nn NOT NULL,
puncte_fidelitate_folosite number(8) default 0 constraint
tranzactie_puncte_fidelitate_folosite_nn NOT NULL,
suma_platita_in_puncte number(10, 2) default 0 constraint
tranzactie_plata_puncte_nn NOT NULL,
suma_platita_card number(10, 2) default 0 constraint tranzactie_plata_card_nn NOT
NULL,
suma_platita_numerar number(10, 2) default 0 constraint
tranzactie_plata_numerar_nn NOT NULL,
constraint tranzactie_beneficiar_fk foreign key(cod_beneficiar) references
BENEFICIAR(cod_beneficiar)
);

```

```

create table VANZARE_FIZICA (
cod_vanzare_fizica int constraint cod_vanzare_pk primary key,
cod_sediu constraint vanzare_sediu_nn NOT NULL,
cod_detalii_tranzactie constraint vanzare_detalii_tranzactie_nn NOT NULL,
constraint vanzare_detalii_fk foreign key(cod_detalii_tranzactie) references
DETALII_TRANZACTIE(cod_detalii_tranzactie),
constraint vanzare_sediu_fk foreign key(cod_sediu) references SEDIU(cod_sediu)
);

```

```

create table ARTICOL_in_VANZARE (
cod_vanzare_fizica int,
cod_articol int,
cantitate number(3) constraint vanzare_cantitate_nn NOT NULL,
constraint articol_in_vanzare_pk primary key(cod_vanzare_fizica, cod_articol),

```

```
constraint articol_in_vanzare_vanzare_fk foreign key(cod_vanzare_fizica) references
VANZARE_FIZICA(cod_vanzare_fizica),
```

```
constraint articol_in_vanzare_articol_fk foreign key(cod_articol) references
ARTICOL(cod_articol)
```

```
);
```

```
create table COMANDA (
```

```
cod_comanda int constraint cod_comanda_pk primary key,
```

```
cod_detalii_tranzactie constraint comanda_detalii_tranzactie_nn NOT NULL,
```

```
status_comanda varchar2(20) default 'se proceseaza',
```

```
tip_comanda varchar2(30) default 'comanda generala' constraint tip_comanda_nn
NOT NULL,
```

```
plata_la_livrare number(1) constraint comanda_plata_la_livrare_nn NOT NULL,
```

```
constraint comanda_detalii_fk foreign key(cod_detalii_tranzactie) references
DETALII_TRANZACTIE(cod_detalii_tranzactie)
```

```
);
```

```
create table COLET (
```

```
cod_colet int,
```

```
cod_comanda int,
```

```
cod_angajat int,
```

```
cod_sediu int,
```

```
greutate number(3) constraint greutate_nn NOT NULL,
```

```
data_livrare date,
```

```
status_colet varchar2(30) default 'se proceseaza' constraint status_colet_nn NOT
NULL,
```

```
constraint greutate_ck check(greutate > 0),
```

```
constraint colet_comanda_fk foreign key(cod_comanda) references
COMANDA(cod_comanda),
```

```
constraint colet_angajat_fk foreign key(cod_angajat) references
CURIER(cod_angajat),
```

```
constraint colet_sediu_fk foreign key(cod_sediu) references SEDIU(cod_sediu),
```

```

constraint colet_pk primary key(cod_colet, cod_comanda)
);

create table RECENZIE (
    cod_recenzie int constraint cod_recenzie_pk primary key,
    cod_articol int constraint cod_recenzie_articol_nn NOT NULL,
    cod_beneficiar int constraint cod_recenzie_beneficiar_nn NOT NULL,
    data_recenzie date default sysdate,
    nota number(2) constraint nota_recenzie_nn NOT NULL,
    comentariu varchar2(250),
    constraint recenzie_articol_fk foreign key(cod_articol) references
ARTICOL(cod_articol),
    constraint recenzie_beneficiar_fk foreign key(cod_beneficiar) references
BENEFICIAR(cod_beneficiar),
    constraint nota_recenzie_ck check(nota <= 10)
);

create table ARTICOL_in_COLET (
    cod_articol int,
    cod_colet int,
    cod_comanda int,
    cantitate number(3) default 1 constraint canitate_articol_colet_nn NOT NULL,
    constraint cantitate_articol_colet_ck check(cantitate >= 0),
    constraint articol_in_colet_pk primary key(cod_articol, cod_colet, cod_comanda),
    constraint articol_in_colet_articol_fk foreign key(cod_articol) references
ARTICOL(cod_articol),
    constraint articol_in_colet_colet_fk foreign key(cod_colet, cod_comanda)
REFERENCES COLET(cod_colet, cod_comanda)
);

create table COS(

```

```

cod_articol int,
cod_beneficiar int,
cantitate number(3) default 1 constraint canitate_produs_cos_nn NOT NULL,
data_adaugare date default sysdate,
bifat number(1) default 1,
constraint cantitate_cos_produs_ck check(cantitate > 0),
constraint adauga_in_cos_pk primary key(cod_articol, cod_beneficiar),
constraint cos_articol_fk foreign key(cod_articol) references ARTICOL(cod_articol),
constraint cos_beneficiar_fk foreign key(cod_beneficiar) references
BENEFICIAR(cod_beneficiar)
);

```

The screenshot shows the Oracle SQL Developer interface with a query worksheet open. The code in the worksheet creates three tables: FURNIZOR, CONCURS, and BENEFICIAR. It also drops four sequences: seq\_detalii\_tranzactie, seq\_vanzare\_fizica, seq\_comanda, and seq\_colet. The FURNIZOR table has columns cod\_furnizor, nume, and adresa\_mail\_repr, with cod\_furnizor as the primary key and nume as a unique constraint. The CONCURS table has columns cod\_concurs, data\_concurs, subiect, durata, nr\_intrebari, taxa\_inscriere, and two constraints: a primary key on cod\_concurs and a check constraint on durata. The BENEFICIAR table has a single column cod\_beneficiar. The bottom of the screen shows a task status bar indicating "Task completed in 0.851 seconds".

```

drop sequence seq_detalii_tranzactie;
drop sequence seq_vanzare_fizica;
drop sequence seq_comanda;
drop sequence seq_colet;

--creare table;
create table FURNIZOR (
    cod_furnizor int constraint cod_furnizor_pk primary key,
    nume varchar2(40) constraint nume_furnizor_nn NOT NULL,
    adresa_mail_repr varchar2(40),
    constraint nume_furnizor_uq unique(nume)
);

create table CONCURS (
    cod_concurs int constraint cod_concurs_pk primary key,
    data_concurs date default sysdate constraint data_concurs_nn NOT NULL,
    subiect varchar2(50) constraint subiect_nn NOT NULL,
    durata number(2) constraint durata_concurs NOT NULL,
    nr_intrebari number(2) default 20 constraint nr_intrebari_nn NOT NULL,
    taxa_inscriere number(2) constraint taxa_inscriere_nn NOT NULL,
    constraint durata_concurs_ck check(durata <= 20),
    constraint subiect_uq unique(subiect)
);

create table BENEFICIAR (
    cod_beneficiar int,

```

Table FURNIZOR created.

Table CONCURS created.

SQL Worksheet History

Worksheet Query Builder

```

...ge [ create_inserare.sql ] * exercitul9.sql * exercitul13.sql * exercitul11.sql * exercitul100.sql * exercitul8.sql * exercitul7.sql...
Baza | Worksheet | Query Builder

[+] create table BENEFICIAR (
    cod_beneficiar int,
    nume varchar2(25) constraint nume_beneficiar_nn NOT NULL,
    prenume varchar2(25) constraint prenume_beneficiar_nn NOT NULL,
    telefon varchar2(13),
    CNP number(13) constraint cnp_beneficiar_nn NOT NULL constraint cnp_beneficiar_uq unique,
    nume_utilizator varchar2(50) constraint nume_utilizator_nn NOT NULL,
    cont_premium number(1) default 0 constraint cont_premium_nn NOT NULL,
    data_autentificare date default sysdate constraint data_autentificare_nn NOT NULL,
    puncte_fidelitate number(10) default 0,
    constraint puncte_fidelitate_ck check(puncte_fidelitate <= 10000),
    constraint nume_utilizator_uq unique(nume_utilizator),
    constraint cod_beneficiar_pk primary key(cod_beneficiar)
);

[+] create table ARTICOL (
    cod_articol int,
    pret number(10, 2) constraint pret_articol_nn NOT NULL,
    nume varchar2(40) constraint nume_articol_nn NOT NULL,
    puncte_fidelitate number(2),
    constraint puncte_fidelitate_articol_ck check(puncte_fidelitate <= 30),
    constraint nume_articol_uq unique(nume),
    constraint cod_articol_pk primary key(cod_articol)
);
[+] create table PACHET_PROMOTIONAL (
    cod_articol int constraint cod_pachet_pk primary key references ARTICOL(cod_articol)
);

```

Task completed in 0.218 seconds

Table BENEFICIAR created.

Table ARTICOL created.

Compiler - Log

SQL Worksheet History

Worksheet Query Builder

```

...ge [ create_inserare.sql ] * exercitul9.sql * exercitul13.sql * exercitul11.sql * exercitul100.sql * exercitul8.sql * exercitul7.sql...
Baza | Worksheet | Query Builder
0.223 seconds

[+] create table ARTICOL (
    nume varchar2(40) constraint nume_articol_nn NOT NULL,
    puncte_fidelitate number(2),
    constraint puncte_fidelitate_articol_ck check(puncte_fidelitate <= 30),
    constraint nume_articol_uq unique(nume),
    constraint cod_articol_pk primary key(cod_articol)
);
[+] create table PACHET_PROMOTIONAL (
    cod_articol int constraint cod_pachet_pk primary key references ARTICOL(cod_articol)
);

[+] create table PRODUS (
    cod_articol int constraint cod_produs_pk primary key references ARTICOL(cod_articol),
    cod_furnizor int,
    luni_garantie number(2) default 12,
    constraint produs_furnizor_fk foreign key(cod_furnizor) references FURNIZOR(cod_furnizor)
);
[+] create table PRODUS_in_PACHET (
    cod_produs int,
    cod_pachet int,
    constraint produs_in_pachet_pk primary key(cod_produs, cod_pachet),
    constraint produs_in_pachet_produs_fk foreign key(cod_produs) references PRODUS(cod_articol),
    constraint produs_in_pachet_pachet_fk foreign key(cod_pachet) references PACHET_PROMOTIONAL(cod_articol)
);

```

Task completed in 0.223 seconds

Table PACHET\_PROMOTIONAL created.

Table PRODUS created.

Table PRODUS\_IN\_PACHET created.

Compiler - Log

SQL Worksheet History

Worksheet Query Builder

```

);
);

create table PARTICIPA (
    cod_concurs int,
    cod_beneficiar int,
    punctaj number(3) constraint punctaj_nn NOT NULL,
    constraint participa_pk primary key(cod_concurs, cod_beneficiar),
    constraint participa_concurs_fk foreign key(cod_concurs) references CONCURS(cod_concurs),
    constraint participa_beneficiar_fk foreign key(cod_beneficiar) references BENEFICIAR(cod_beneficiar)
);

create table PREMIU (
    cod_premiu int,
    cod_concurs int,
    cod_articol int,
    punctaj_minim number(3) constraint punctaj_minim_nn NOT NULL,
    suma number(6, 2) default 0,
    constraint premiu_pk primary key(cod_premiu, cod_concurs),
    constraint premiu_concurs_fk foreign key(cod_concurs) references CONCURS(cod_concurs),
    constraint premiu_produs_fk foreign key(cod_articol) references PRODUS(cod_articol)
);

create table ANGAJAT (
    cod_angajat int,
    nume varchar2(25) constraint nume_angajat_nn NOT NULL,
    prenume varchar2(25) constraint prenume_angajat_nn NOT NULL,
    salariu number(7) default 3500,
    telefon varchar2(15)
);

```

Task completed in 0.167 seconds

Table PARTICIPA created.

Table PREMIU created.

SQL Worksheet History

Worksheet Query Builder

```

0.25400001 seconds
;
;

create table ANGAJAT (
    cod_angajat int,
    nume varchar2(25) constraint nume_angajat_nn NOT NULL,
    prenume varchar2(25) constraint prenume_angajat_nn NOT NULL,
    salariu number(7) default 3500,
    telefon varchar2(15),
    data_angajare date default sysdate constraint data_angajare_nn NOT NULL,
    constraint cod_angajat_pk primary key(cod_angajat)
);

create table SCENARIST (
    cod_angajat int constraint cod_scenarist_pk primary key references ANGAJAT(cod_angajat),
    zi_libera varchar2(10)
);

create table ADMINISTRATOR (
    cod_angajat int constraint cod_administrator_pk primary key references ANGAJAT(cod_angajat)
);

create table CURIER (
    cod_angajat int constraint cod_curier_pk primary key references ANGAJAT(cod_angajat),
    categorie_permis varchar2(2) constraint categorie_permis_nn NOT NULL
);

```

Task completed in 0.254 seconds

Table ANGAJAT created.

Table SCENARIST created.

Table ADMINISTRATOR created.

Table CURIER created.

Compiler - Log

SQL Worksheet | History

```

Worksheet Query Builder Baza
...ge [create_inserare.sql] [exercitul9.sql] [exercitul13.sql] [exercitul11.sql] [exercitul100.sql] [exercitul8.sql] [exercitul7.sql...]
;
CREATE TABLE RECLAMA (
    cod_reclama INT CONSTRAINT cod_reclama_pk PRIMARY KEY,
    cod_articol INT,
    cod_angajat INT CONSTRAINT cod_reclama_angajat_nn NOT NULL,
    durata NUMBER(3) CONSTRAINT durata_reclama_nn NOT NULL,
    cost_total NUMBER(9, 2) CONSTRAINT cost_total_nn NOT NULL,
    constraint reclama_articol_fk FOREIGN KEY(cod_articol) REFERENCES ARTICOL(cod_articol),
    constraint reclama_scenarist_fk FOREIGN KEY(cod_angajat) REFERENCES SCENARIST(cod_angajat)
);
;

CREATE TABLE SEDIU (
    cod_sediu INT CONSTRAINT cod_sediu_pk PRIMARY KEY,
    cod_administrator INT CONSTRAINT sediu_administrator_nn NOT NULL,
    nume VARCHAR(30) CONSTRAINT nume_sediu_nn NOT NULL,
    adresa VARCHAR(60) CONSTRAINT adresa_sediu_nn NOT NULL,
    suprafata NUMBER(10) DEFAULT 100000 CONSTRAINT suprafata_sediu_nn NOT NULL,
    constraint nume_sediu_uq UNIQUE(nume),
    constraint sediu_administrator_fk FOREIGN KEY(cod_administrator) REFERENCES ADMINISTRATOR(cod_angajat)
);
;

CREATE TABLE ARTICOL_IN_SEDIU (
    cod_sediu INT,
    cod_articol INT,
    cantitate NUMBER(8) CONSTRAINT sediu_cantitate_nn NOT NULL,
    pret NUMBER(10, 2) CONSTRAINT sediu_pret_nn NOT NULL,
    constraint articol_in_sediu_pk PRIMARY KEY(cod_sediu, cod_articol),
);
;

Table RECLAMA created.

Table SEDIU created.

Task completed in 0.225 seconds

```

Compiler - Log

SQL Worksheet | History

```

Worksheet Query Builder Baza
...ge [create_inserare.sql] [exercitul9.sql] [exercitul13.sql] [exercitul11.sql] [exercitul100.sql] [exercitul8.sql] [exercitul7.sql...]
;
ALTER TABLE SEDIU
    ADD CONSTRAINT suprafata_nn CHECK (suprafata >= 0),
    ADD CONSTRAINT nume_sediu_uq UNIQUE(nume),
    ADD CONSTRAINT sediu_administrator_fk FOREIGN KEY(cod_administrator) REFERENCES ADMINISTRATOR(cod_angajat)
;
;

CREATE TABLE ARTICOL_IN_SEDIU (
    cod_sediu INT,
    cod_articol INT,
    cantitate NUMBER(8) CONSTRAINT sediu_cantitate_nn NOT NULL,
    pret NUMBER(10, 2) CONSTRAINT sediu_pret_nn NOT NULL,
    constraint articol_in_sediu_pk PRIMARY KEY(cod_sediu, cod_articol),
    constraint articol_in_sediu_sediu_fk FOREIGN KEY(cod_sediu) REFERENCES SEDIU(cod_sediu),
    constraint articol_in_sediu_articol_fk FOREIGN KEY(cod_articol) REFERENCES ARTICOL(cod_articol)
);
;

CREATE TABLE DETALII_TRANZACTIE (
    cod_detalii_tranzactie INT CONSTRAINT cod_detalii_tranzactie_pk PRIMARY KEY,
    cod_beneficiar INT,
    data_tranzactie DATE DEFAULT SYSDATE CONSTRAINT data_tranzactie_nn NOT NULL,
    cost_total NUMBER(10, 2) CONSTRAINT cost_tranzactie_nn NOT NULL,
    puncte_fidelitate_castigate NUMBER(8) DEFAULT 0 CONSTRAINT transactie_puncte_fidelitate_castigate_nn NOT NULL,
    puncte_fidelitate_folosite NUMBER(8) DEFAULT 0 CONSTRAINT transactie_puncte_fidelitate_folosite_nn NOT NULL,
    suma_platita_in_puncte NUMBER(10, 2) DEFAULT 0 CONSTRAINT transactie_plata_puncte_nn NOT NULL,
    suma_platita_card NUMBER(10, 2) DEFAULT 0 CONSTRAINT transactie_plata_card_nn NOT NULL,
    suma_platita_numar NUMBER(10, 2) DEFAULT 0 CONSTRAINT transactie_plata_numar_nn NOT NULL,
    constraint transactie_beneficiar_fk FOREIGN KEY(cod_beneficiar) REFERENCES BENEFICIAR(cod_beneficiar)
);
;

CREATE TABLE VANZARE_FIZICA (
    cod_vanzare_fizica INT CONSTRAINT cod_vanzare_pk PRIMARY KEY,
);
;

Table ARTICOL_IN_SEDIU created.

Table DETALII_TRANZACTIE created.

Task completed in 0.169 seconds

```

Compiler - Log

SQL Worksheet History

Worksheet Query Builder

```

    );
create table VANZARE_FIZICA (
    cod_vanzare_fizica int constraint cod_vanzare_pk primary key,
    cod_sediu constraint vanzare_sediu_nn NOT NULL,
    cod_detaliu_tranzactie constraint vanzare_detaliu_tranzactie_nn NOT NULL,
    constraint vanzare_detaliu_fk foreign key(cod_detaliu_tranzactie) references DETALII_TRANZACTIE(cod_detaliu_tranzactie),
    constraint vanzare_sediu_fk foreign key(cod_sediu) references SEDIU(cod_sediu)
);
create table ARTICOL_IN_VANZARE (
    cod_vanzare_fizica int,
    cod_articol int,
    cantitate number(3) constraint vanzare_cantitate_nn NOT NULL,
    constraint articol_in_vanzare_pk primary key(cod_vanzare_fizica, cod_articol),
    constraint articol_in_vanzare_vanzare_fk foreign key(cod_vanzare_fizica) references VANZARE_FIZICA(cod_vanzare_fizica),
    constraint articol_in_vanzare_articol_fk foreign key(cod_articol) references ARTICOL(cod_articol)
);
create table COMANDA (
    cod_comanda int constraint cod_comanda_pk primary key,
    cod_detaliu_tranzactie constraint comanda_detaliu_tranzactie_nn NOT NULL,
    status_comanda varchar2(20) default 'se proceseaza',
    tip_comanda varchar2(30) default 'comanda generala' constraint tip_comanda_nn NOT NULL,
    plata_la_livrare number(1) constraint comanda_plata_la_livrare_nn NOT NULL,
    constraint comanda_detaliu_fk foreign key(cod_detaliu_tranzactie) references DETALII_TRANZACTIE(cod_detaliu_tranzactie)
);

```

Table VANZARE\_FIZICA created.

Table ARTICOL\_IN\_VANZARE created.

Table COMANDA created.

Compiler - Log

SQL Worksheet History

Worksheet Query Builder

```

    );
create table COLET (
    cod_colet int,
    cod_comanda int,
    cod_angajat int,
    cod_sediu int,
    greutate number(3) constraint greutate_nn NOT NULL,
    data_livrare date,
    status_colet varchar2(30) default 'se proceseaza' constraint status_colet_nn NOT NULL,
    constraint greutate_ck check(greutate > 0),
    constraint colet_comanda_fk foreign key(cod_comanda) references COMANDA(cod_comanda),
    constraint colet_angajat_fk foreign key(cod_angajat) references CURIER(cod_angajat),
    constraint colet_sediu_fk foreign key(cod_sediu) references SEDIU(cod_sediu),
    constraint colet_pk primary key(cod_colet, cod_comanda)
);
create table RECENTZIE (
    cod_recenzie int constraint cod_recenzie_pk primary key,
    cod_articol int constraint cod_recenzie_articol_nn NOT NULL,
    cod_beneficiar int constraint cod_recenzie_beneficiar_nn NOT NULL,
    data_recenzie date default sysdate,
    nota number(2) constraint nota_recenzie_nn NOT NULL,
    comentariu varchar2(250),
    constraint recenzie_articol_fk foreign key(cod_articol) references ARTICOL(cod_articol),
    constraint recenzie_beneficiar_fk foreign key(cod_beneficiar) references BENEFICIAR(cod_beneficiar),
    constraint nota_recenzie_ck check(nota <= 10)
);

```

Table COLET created.

Table RECENTZIE created.

Compiler - Log

```

...ge [1] create_inserare.sql x [2] exercitiu9.sql x [3] exercitiu13.sql x [4] exercitiu11.sql x [5] exercitiu100.sql x [6] exercitiu8.sql x [7] exercitiu7.sql...
SQL Worksheet | History
Worksheet | Query Builder
constraint recenzie_articol_fk foreign key(cod_articol) references ARTICOL(cod_articol),
constraint recenzie_beneficiar_fk foreign key(cod_beneficiar) references BENEFICIAR(cod_beneficiar),
constraint nota_recenzie_ck check(nota <= 10)
);

create table ARTICOL_in_COLET (
cod_articol int,
cod_colet int,
cod_comanda int,
cantitate number(3) default 1 constraint canitate_articol_colet_nn NOT NULL,
constraint canitate_articol_colet_ck check(cantitate >= 0),
constraint articol_in_colet_pk primary key(cod_articol, cod_colet, cod_comanda),
constraint articol_in_colet_articol_fk foreign key(cod_articol) references ARTICOL(cod_articol),
constraint articol_in_colet_colet_fk foreign key(cod_colet, cod_comanda) REFERENCES COLET(cod_colet, cod_comanda)
);

create table COS(
cod_articol int,
cod_beneficiar int,
cantitate number(3) default 1 constraint canitate_producus_nn NOT NULL,
data_adaugare date default sysdate,
bifat number(1) default 1,
constraint canitate_cos_producus_ck check(cantitate > 0),
constraint adauga_in_cos_pk primary key(cod_articol, cod_beneficiar),
constraint cos_articol_fk foreign key(cod_articol) references ARTICOL(cod_articol),
constraint cos_beneficiar_fk foreign key(cod_beneficiar) references BENEFICIAR(cod_beneficiar)
);

```

Table ARTICOL\_IN\_COLET created.

Table COS created.

## 5) Inserare în tabele

```

create sequence seq_furnizor start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_concurs start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_beneficiar start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_articol start with 100 increment by 1 maxvalue 1000 nocycle nocache;
create sequence seq_premiu start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_angajat start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_reclama start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_sediu start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_detalii_tranzactie start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_vanzare_fizica start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_comanda start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_colet start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_recenzie start with 100 increment by 1 maxvalue 1000 nocycle nocache;

```

SQL Worksheet | History

Worksheet | Query Builder

```
--CREARE SEQUENCE
create sequence seq_furnizor start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_concurs start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_beneficiar start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_articol start with 100 increment by 1 maxvalue 1000 nocycle nocache;
create sequence seq_premiu start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_angajat start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_reclama start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_sediu start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_detailii_tranzactie start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_vanzare_fizica start with 100 increment by 1 maxvalue 10000 nocycle nocache;
```

Sequence SEQ\_FURNIZOR created.

Sequence SEQ\_CONCURS created.

Sequence SEQ\_BENEFICIAR created.

Sequence SEQ\_ARTICOL created.

Sequence SEQ\_PREMIU created.

Sequence SEQ\_ANGAJAT created.

Sequence SEQ\_RECLAMA created.

Sequence SEQ\_SEDIU created.

SQL Worksheet | History

Worksheet | Query Builder

```
--CREARE SEQUENCE
create sequence seq_furnizor start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_concurs start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_beneficiar start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_articol start with 100 increment by 1 maxvalue 1000 nocycle nocache;
create sequence seq_premiu start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_angajat start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_reclama start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_sediu start with 100 increment by 10 maxvalue 1000 nocycle nocache;
create sequence seq_detailii_tranzactie start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_vanzare_fizica start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_comanda start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_colet start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_recenzie start with 100 increment by 1 maxvalue 1000 nocycle nocache;
```

--INSERARE TABELE:

--FURNIZOR

```
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electronice high quality', 'popescu');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electric party', 'ionescu.maria@yahoo.com');
```

Sequence SEQ\_DETALII\_TRANZACTIE created.

Sequence SEQ\_VANZARE\_FIZICA created.

Sequence SEQ\_COMANDA created.

Sequence SEQ\_COLET created.

Sequence SEQ\_RECENZIE created.

## --FURNIZOR

```
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electronice high quality', 'popescu.ioan@gmail.com');

insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Elecrtic party', 'ionescu.maria@yahoo.com');

insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electronics++', 'marinescu.mirela@gmail.com');

insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'IT Direct', 'paul.enescu45@gmail.com');

insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Ultra circuite', NULL);

insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Best Techno', 'maria.dobrescu@gmail.com');

insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Furnizor Electronice', 'andrei.matei@gmail.com');

select * from furnizor;
```

## --CONCURS

```
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('12-04-2023', 'dd-mm-yyyy'), 'Biologie', 10, 15, 20);

insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('18-04-2023', 'dd-mm-yyyy'), 'Matematica', 20, 20, 30);

insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('27-02-2023', 'dd-mm-yyyy'), 'Literatura', 10, 16, 10);

insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('14-11-2022', 'dd-mm-yyyy'), 'Diverse', 7, 8, 10);

insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('15-01-2023', 'dd-mm-yyyy'), 'Vulcani', 13, 15, 25);

select * from concurs;
```

## --BENEFICIAR

```
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Andronic',
```

```

'Marcel', '+400733456753', 5001210208170,  'marcel233', 1, to_date('12-05-2021', 'dd-mm-yyyy'),
100);

insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator,
cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Suditu',
'Mara', '+400745839485', 2990310209242,  'marra4', 0, to_date('25-04-2021', 'dd-mm-yyyy'),
500);

insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator,
cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Lupu',
'Eugen', '+400748201839', 5010410205807, 'dvd34_a', 1, to_date('30-05-2021', 'dd-mm-yyyy'),
800);

insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator,
cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Lupu',
'Eugen', '+400745678542', 1990915204669, 'eugen_123', 0,to_date('12-04-2021', 'dd-mm-yyyy'),
0);

insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator,
cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Alexandru',
'Miruna', '+400706543678', 6001130203816, 'mira__12', 0, to_date('12-04-2021', 'dd-mm-yyyy'),
900);

insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator,
cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Munteanu',
'Ana', '+400733456784', 6010413100131, 'mnt_ana', 0, to_date('02-05-2022', 'dd-mm-yyyy'), 0);

insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator,
cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Maria', 'Ana',
'+400733456784', 6010413123013, 'm_ana', 0, to_date('02-05-2022', 'dd-mm-yyyy'), 0);

select * from beneficiar;

```

## --ARTICOL

```

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 100, 'Casti
wireless', 12);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 120, 'Bec
ultra', 24);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 300,
'Incarcator telefon', 13);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 400,
'Lanterna', 18);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 100,
'Frigider', 19);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 80,
'Aspirator robot', 25);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 10, 'Fier de
calcat', 0);

```

```

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 100,
'Uscator de par',28);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 120,
'Uscator de par Julie Pro', 15);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 90,
'Telefon', 10);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 80,
'Laptop', 0);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval,
70,'Tableta', 3);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 1000,
'Pachet Pro', 28);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 800,
'Pachet electro', 28);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 1000,
'Pachet super', 29);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 5000,
'Pachet techno', 30);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 8000,
'Pachet all', 30);

select * from articol;

```

#### --PRODUS

```

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(100, NULL, 12);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(101, 100, 18);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(102, 100, 0);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(103, 110, 18);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(104, 120, 24);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(105, 120, 18);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(106, 130, 24);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(107, 130, 18);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(108, 140, 18);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(109, 150, 15);

insert into produs (cod_articol, cod_furnizor, luni_garantie)values(110, 150, 6);

```

```
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(111, 150, 3);
select * from produs;
```

#### --PACHET

```
insert into pachet_promotional (cod_articol)values(112);
insert into pachet_promotional (cod_articol)values(113);
insert into pachet_promotional (cod_articol)values(114);
insert into pachet_promotional (cod_articol)values(115);
insert into pachet_promotional (cod_articol)values(116);
select* from pachet_promotional;
```

#### --PRODUS\_IN\_PACHET

```
insert into produs_in_pachet(cod_produs, cod_pachet)values(100, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(101, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(102, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(103, 113);
insert into produs_in_pachet(cod_produs, cod_pachet)values(104, 113);
insert into produs_in_pachet(cod_produs, cod_pachet)values(105, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(106, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(107, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(108, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(109, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(100, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(102, 115);
insert into produs_in_pachet(cod_produs, cod_pachet)values(105, 115);
insert into produs_in_pachet(cod_produs, cod_pachet)values(104, 116);
insert into produs_in_pachet(cod_produs, cod_pachet)values(108, 116);
select * from produs_in_pachet;
```

## --PREMIU

```
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 100, 103, 60, 20);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 100, 100, 70, 30);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 100, 108, 90, 40);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 110, NULL, 80, 30);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 110, 103, 90, 35);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 120, NULL, 70, 210);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 120, 102, 75, 20);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 130, 101, 80, 0);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 130, 105, 99, 30);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 140, 111, 100, 0);

insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim,
suma)values(seq_premiu.nextval, 140, 104, 50, 10);

select* from premiu;
```

## --PARTICIPA

```
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (100, 110, 50);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (100, 100, 66);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (100, 130, 74);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (100, 140, 95);

insert into participa (cod_concurs, cod_beneficiar, punctaj) values (110, 100, 92);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (110, 110, 94);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (110, 130, 88);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (110, 140, 50);
```

```
insert into participa (cod_concurs, codBeneficiar, punctaj) values (120, 100, 97);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (120, 110, 73);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (120, 140, 60);
```

```
insert into participa (cod_concurs, codBeneficiar, punctaj) values (130, 100, 99);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (130, 110, 90);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (130, 130, 79);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (130, 140, 73);
```

```
insert into participa (cod_concurs, codBeneficiar, punctaj) values (140, 100, 99);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (140, 110, 45);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (140, 130, 74);
insert into participa (cod_concurs, codBeneficiar, punctaj) values (140, 140, 100);
select * from participa;
```

#### --ANGAJAT

```
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Lopataru', 'Alexandra', 3500, '+400758465748', to_date('15-08-2021',
'dd-mm-yyyy'));
```

```
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Cujbescu', 'Marius', 3600, '+400745678954', to_date('12-10-2021',
'dd-mm-yyyy'));
```

```
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Serban', 'Alina', 3700, '+400767584957', to_date('12-09-2022',
'dd-mm-yyyy'));
```

```
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Mihai', 'Costin', 3650, '+400778965435', to_date('09-09-2021',
'dd-mm-yyyy'));
```

```
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Lazar', 'Simon', 3700, '+400789657453', to_date('05-09-2021',
'dd-mm-yyyy'));
```

```
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Moldovan', 'Andreea', 5000, '+400709876547', to_date('12-04-2021',
'dd-mm-yyyy'));
```

```

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Oprea', 'Pavel', 5500, '+400798765467', to_date('12-01-2022',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Dumitrescu', 'Bianca', 5600, '+400785768594', to_date('18-06-2022',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Dinu', 'Dobrin', 5400, '+400734567895', to_date('19-10-2021',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Cristea', 'Cristina', 5400, NULL, to_date('28-11-2021', 'dd-mm-yyyy'));


insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Todorescu', 'Simon', 6000, '+400734567845', to_date('12-09-2023',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Anontescu', 'Briana', 7000, '+400787657893', to_date('12-11-2023',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Cutas', 'Mirel', 5600, '+400797865467', to_date('12-11-2023',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Chiristigiu', 'Madalina', 5700, '+400785958594', to_date('12-02-2023',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Costetchi', 'Catalina', 5700, '+400734569595', to_date('28-05-2022',
'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(seq_angajat.nextval, 'Ciubeica', 'Cristina', 6500, NULL, to_date('29-04-2024', 'dd-mm-yyyy'));

```

select\* from angajat;

#### --SCENARIST

```

insert into scenarist (cod_angajat, zi_libera) values (100, 'miercuri');

insert into scenarist (cod_angajat, zi_libera) values (110, 'luni');

insert into scenarist (cod_angajat, zi_libera) values (120, 'luni');

insert into scenarist (cod_angajat, zi_libera) values (130, 'joi');

insert into scenarist (cod_angajat, zi_libera) values (140, 'vineri');

```

```
select * from scenarist;
```

#### --ADMINISTRATOR

```
insert into administrator (cod_angajat) values (150);
insert into administrator (cod_angajat) values (160);
insert into administrator (cod_angajat) values (170);
insert into administrator (cod_angajat) values (180);
insert into administrator (cod_angajat) values (190);
select* from administrator;
```

#### --CURIER

```
insert into curier (cod_angajat, categorie_permis) values (200, 'A2');
insert into curier (cod_angajat, categorie_permis) values (210, 'B');
insert into curier (cod_angajat, categorie_permis) values (220, 'C');
insert into curier (cod_angajat, categorie_permis) values (230, 'CE');
insert into curier (cod_angajat, categorie_permis) values (240, 'C');
insert into curier (cod_angajat, categorie_permis) values (250, 'A1');
select* from curier;
```

#### --RECLAMA

```
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 100, 140, 30, 2150);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 100, 130, 40, 2250);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, NULL, 130, 40, 1400);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 101, 110, 50, 1400);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 101, 130, 60, 2250);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 101, 140, 10, 3000);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 102, 110, 20, 3000);
```

```

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 102, 140, 30, 2250);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 102, 130, 35, 2250);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 103, 120, 35, 1220);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 103, 110, 35, 1400);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 105, 120, 30, 1220);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 105, 140, 45, 1400);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 106, 130, 24, 1220);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 106, 120, 45, 3000);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 107, 110, 30, 1220);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 109, 140, 30, 500);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 110, 130, 45, 4000);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 111, 110, 55, 5000);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 110, 120, 30, 4030);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, NULL, 120, 42, 2000);

insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values
(seq_reclama.nextval, 108, 130, 25, 600);

select * from reclama;

```

#### --SEDIU

```

insert into sediu (cod_sediu, cod_administrator, nume, adresa, suprafata) values
(seq_sediu.nextval, 150, 'Electric center', 'Bucuresti Sud', 10000);

insert into sediu (cod_sediu, cod_administrator, nume, adresa, suprafata) values
(seq_sediu.nextval, 160, 'Cable power', 'Centru Bucuresti City', 12000);

```

```
insert into sediu (cod_sedi, cod_administrator, nume, adresa, suprafata) values  
(seq_sedi.nextval, 170, 'Atacul tehnologiei', 'Ploiesti Periferie', 5000);  
  
insert into sediu (cod_sedi, cod_administrator, nume, adresa, suprafata) values  
(seq_sedi.nextval, 170, 'IT Party', 'Buzau Cartier Dorobanti', 6000);  
  
insert into sediu (cod_sedi, cod_administrator, nume, adresa, suprafata) values  
(seq_sedi.nextval, 190, 'ELECTRO LIGHT MAIN', 'Bucuresti Strada Pacii', 12000);  
  
select* from sediu;
```

#### --ARTICOL\_IN\_SEDIU

```
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 100, 60, 105.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 101, 70, 115.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 102, 80, 290.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 103, 50, 410.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 104, 90, 395.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 107, 60, 107.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 108, 100, 132.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 109, 70, 425.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 110, 55, 885.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 111, 45, 1390.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 112, 30, 980.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 113, 25, 800.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 114, 15, 1020.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 115, 10, 5100.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (100, 116, 5, 8100.00);  
  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 100, 65, 100.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 101, 75, 110.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 102, 85, 295.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 103, 55, 420.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 104, 85, 400.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 105, 1, 90.00);  
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 107, 65, 110.00);
```

```
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 108, 105, 130.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 109, 75, 420.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 110, 60, 880.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 111, 50, 1400.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 112, 28, 950.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 113, 18, 790.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 114, 8, 1050.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 115, 5, 5200.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (110, 116, 3, 8200.00);

insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 100, 55, 110.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 101, 65, 120.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 102, 75, 285.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 103, 50, 425.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 104, 85, 405.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 107, 70, 105.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 108, 95, 130.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 110, 65, 875.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 111, 55, 1420.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 112, 26, 970.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 113, 15, 810.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 114, 7, 1020.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 115, 4, 5000.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (120, 116, 2, 8050.00);

insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (130, 100, 70, 100.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (130, 102, 90, 295.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (130, 103, 60, 430.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (130, 104, 100, 390.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (130, 106, 150, 27.00);
insert into articol_in_sedi (cod_sedi, cod_articol, cantitate, pret) values (130, 107, 80, 108.00);
```

```

insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 108, 120, 135.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 109, 90, 430.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 110, 70, 880.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 113, 12, 805.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 114, 5, 1015.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 115, 3, 5050.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 116, 2, 8050.00);

insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 100, 55, 95.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 101, 75, 110.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 102, 85, 290.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 103, 65, 425.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 108, 110, 130.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 109, 75, 420.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 110, 60, 875.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 111, 50, 1420.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 112, 20, 950.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 113, 10, 810.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 114, 5, 1010.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 115, 4, 5100.00);
select * from articol_in_sediu;

```

#### --DETALII\_TRANZACTIE

```

insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 120,
to_date('01-03-2024', 'dd-mm-yyyy'), 550, 25, 0, 0, 550, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 110,
to_date('02-01-2024', 'dd-mm-yyyy'), 820, 22, 0, 0, 820, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 100,
to_date('03-08-2024', 'dd-mm-yyyy'), 940, 24, 55, 20, 0, 920);

```

```

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 100,
to_date('04-11-2024', 'dd-mm-yyyy'), 570, 27, 0, 0, 570, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 110,
to_date('06-12-2024', 'dd-mm-yyyy'), 1060, 26, 30, 10, 0, 1050);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 140,
to_date('05-01-2023', 'dd-mm-yyyy'), 800, 20, 25, 10, 0, 790);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 110,
to_date('07-02-2023', 'dd-mm-yyyy'), 930, 23, 0, 0, 930, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 110,
to_date('08-03-2023', 'dd-mm-yyyy'), 250, 25, 0, 0, 250, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 100,
to_date('09-04-2023', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 130,
to_date('10-05-2023', 'dd-mm-yyyy'), 1020, 22, 0, 0, 1020, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 100,
to_date('11-06-2023', 'dd-mm-yyyy'), 240, 24, 0, 0, 100, 140);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 130,
to_date('10-07-2022', 'dd-mm-yyyy'), 1060, 26, 10, 10, 0, 1050);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 100,
to_date('13-08-2023', 'dd-mm-yyyy'), 220, 22, 0, 0, 220, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
```

```
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 130,
to_date('14-10-2023', 'dd-mm-yyyy'), 950, 25, 0, 0, 950, 0);

insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte,
suma_platita_card, suma_platita_numerar) values (seq_detalii_tranzactie.nextval, 140,
to_date('15-11-2023', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);
```

```
select * from detalii_tranzactie;
```

#### --COMANDA

```
insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda) values (seq_comanda.nextval, 100, 'helivrata', 0, 'generala');

insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda) values (seq_comanda.nextval, 101, 'helivrata', 0, 'generala');

insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda) values (seq_comanda.nextval, 103, 'helivrata', 0, 'generala');

insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda) values (seq_comanda.nextval, 102, 'livrata', 1, 'generala');

insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda) values (seq_comanda.nextval, 104, 'livrata', 1, 'generala');

select * from comanda;
```

#### --COLET

```
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet,
data_livrare) values (seq_colet.nextval, 100, 200, 100, 200, 'nelivrat', NULL);

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet,
data_livrare) values (seq_colet.nextval, 100, 200, 110, 100, 'nelivrat', NULL);

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet,
data_livrare) values (seq_colet.nextval, 100, 210, 120, 300, 'nelivrat', NULL);

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet,
data_livrare) values (seq_colet.nextval, 101, NULL, NULL, 350, 'nelivrat', NULL);

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet,
data_livrare) values (seq_colet.nextval, 101, 220, 110, 400, 'nelivrat', NULL);

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet,
data_livrare) values (seq_colet.nextval, 102, NULL, 110, 210, 'nelivrat', NULL);

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet,
data_livrare) values (seq_colet.nextval, 102, 230, 120, 310, 'nelivrat', NULL);
```

```
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 102, NULL, 130, 280, 'nelivrat', NULL);

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 103, 230, 130, 210, 'livrat', to_date('01-01-2024', 'dd-mm-yyyy'));

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 103, 240, 100, 200, 'livrat', to_date('01-03-2024', 'dd-mm-yyyy'));

insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 104, 240, 120, 500, 'livrat', to_date('01-08-2023', 'dd-mm-yyyy'));

select * from colet;
```

#### --VANZARE FIZICA

```
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 100, 105);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 100, 106);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 110, 107);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 110, 108);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 120, 109);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 110);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 111);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 112);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 113);

insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 114);

select * from vanzare_fizica;
```

#### --COS

```

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(100, 100, 1,
to_date('01-01-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(101, 100, 2,
to_date('02-01-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(102, 100, 3,
to_date('03-01-2024', 'dd-mm-yyyy'), 0);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(104, 110, 1,
to_date('04-02-2024', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(105, 110, 1,
to_date('05-03-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(106, 110, 1,
to_date('06-01-2023', 'dd-mm-yyyy'), 0);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(113, 120, 1,
to_date('07-04-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(114, 120, 2,
to_date('08-05-2023', 'dd-mm-yyyy'), 0);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(115, 120, 3,
to_date('09-01-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(102, 130, 4,
to_date('10-06-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(101, 130, 1,
to_date('11-01-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(104, 130, 1,
to_date('12-07-2024', 'dd-mm-yyyy'), 0);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(106, 140, 1,
to_date('13-08-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(109, 140, 1,
to_date('14-09-2024', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(105, 140, 2,
to_date('15-01-2023', 'dd-mm-yyyy'), 1);

insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(107, 140, 1,
to_date('16-01-2024', 'dd-mm-yyyy'), 0);

select * from cos;

```

## --RECENZIE

```

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, dataRecenzie, nota, comentariu)
values (seq_recenzie.nextval, 100, 100, to_date('07-08-2023', 'dd-mm-yyyy'), 8, 'recomand
tuturor');

```

```

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 100, 100, to_date('09-08-2023', 'dd-mm-yyyy'), 10, 'mai eficient
decat ma asteptam');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 101, 110, to_date('10-08-2023', 'dd-mm-yyyy'), 10, 'recomand
tuturor');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 113, 110, to_date('06-08-2023', 'dd-mm-yyyy'), 7, 'recomand
tuturor');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 103, 120, to_date('07-05-2023', 'dd-mm-yyyy'), 4, 'nu recomand');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 104, 120, to_date('09-08-2023', 'dd-mm-yyyy'), 6, 'poate fi
imbunatatit');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 105, 140, to_date('07-03-2023', 'dd-mm-yyyy'), 8, 'recomand cu
caldura');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 114, 140, to_date('03-08-2023', 'dd-mm-yyyy'), 9, 'de inalta calitate');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 109, 140, to_date('07-02-2023', 'dd-mm-yyyy'), 7, 'destul de bun');

insert into recenzie (cod_recenzie, cod_articol, codBeneficiar, data_recenzie, nota, comentariu)
values (seq_recenzie.nextval, 110, 140, to_date('04-08-2023', 'dd-mm-yyyy'), 3, 'nu recomand
deloc');

select * from recenzie;

```

#### --ARTICOL\_IN\_COLET

```

insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (100, 100, 100,
3);

insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (102, 100, 100,
2);

insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (104, 100, 100,
1);

insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (115, 101, 100,
1);

insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (103, 101, 100,
1);

insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (106, 102, 100,
2);

```

```

insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (107, 102, 100, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (104, 103, 101, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (109, 103, 101, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (113, 104, 101, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (101, 104, 101, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (112, 104, 101, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (105, 105, 102, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (114, 105, 102, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (115, 105, 102, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (108, 106, 102, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (110, 107, 102, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (107, 108, 103, 3);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (103, 109, 103, 4);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (114, 109, 103, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (100, 110, 104, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (112, 110, 104, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (116, 110, 104, 2);
select * from articol_in_colet;

```

#### --ARTICOL\_IN\_VANZARE

```
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (100, 100, 1);
```

```

insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (100, 101, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (100, 102, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (101, 101, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (101, 104, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (101,108, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (102, 109, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (102, 111, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (102, 112, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (103,113, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (103,116, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (103,112, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (104, 114, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (104, 115, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 116, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 112, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 114, 2);
--insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 102, 2);
--insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 116, 2);
--insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 109, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (106, 114, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (106, 115, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (106, 116, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (107, 112, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (107, 113, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (107, 107, 1 );
--insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (107, 104, 1 );
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (108, 107, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (108, 104, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (108, 108, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (109, 102, 1);
select * from articol_in_vanzare;

```

lab5.sql Welcome Page create\_inserare.sql exercitul9.sql exercitul13.sql exercitul11.sql exercitul100.sql exercitul8.sql exercitul7.sql exercitul6.sql

SQL Worksheet History

Worksheet Query Builder

```
create sequence seq_detalii_tranzactie start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_vanzare_fizica start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_comanda start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_colet start with 100 increment by 1 maxvalue 10000 nocycle nocache;
create sequence seq_recenzie start with 100 increment by 1 maxvalue 1000 nocycle nocache;
```

--INSERARE TABELE:

--FURNIZOR

```
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electronice high quality', 'popescu.ioan@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Elecrtic party', 'ionescu.maria@yahoo.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electronics++', 'marinescu.mirela@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'IT Direct', 'paul.enescu45@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Ultra circuite', NULL);
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Best Techno', 'maria.dobrescu@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Furnizor Electronice', 'andrei.matei@gmail.com');
select * from furnizor;
```

--CONCURS

```
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('12-04-2023', 'dd-mm-yyyy'), 'Biologie', 10, 15, 20);
```

All Rows Fetched: 7 in 0.068 seconds

COD_FURNIZOR	NUME	ADRESA_MAIL_REPR
1	100 Electronice high quality	popescu.ioan@gmail.com
2	110 Elecrtic party	ionescu.maria@yahoo.com
3	120 Electronics++	marinescu.mirela@gmail.com
4	130 IT Direct	paul.enescu45@gmail.com
5	140 Ultra circuite	(null)
6	150 Best Techno	maria.dobrescu@gmail.com
7	160 Furnizor Electronice	andrei.matei@gmail.com

Compiler - Log

Messages Statements Compiler Logging Page

lab5.sql Welcome Page create\_inserare.sql exercitul9.sql exercitul13.sql exercitul11.sql exercitul100.sql exercitul8.sql exercitul7.sql exercitul12.sql

SQL Worksheet History

Worksheet Query Builder

```
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electronice high quality', 'popescu.ioan@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Elecrtic party', 'ionescu.maria@yahoo.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Electronics++', 'marinescu.mirela@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'IT Direct', 'paul.enescu45@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Ultra circuite', NULL);
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Best Techno', 'maria.dobrescu@gmail.com');
insert into furnizor (cod_furnizor, nume, adresa_mail_repr)values(seq_furnizor.nextval, 'Furnizor Electronice', 'andrei.matei@gmail.com');
select * from furnizor;
```

--CONCURS

```
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('12-04-2023', 'dd-mm-yyyy'), 'Biologie', 10, 15, 20);
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('18-04-2023', 'dd-mm-yyyy'), 'Matematica', 20, 20, 30);
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('27-02-2023', 'dd-mm-yyyy'), 'Literatura', 10, 16, 10);
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('14-11-2022', 'dd-mm-yyyy'), 'Diverse', 7, 8, 10);
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscriere)values(seq_concurs.nextval, to_date('15-01-2023', 'dd-mm-yyyy'), 'Vulcani', 13, 15, 25);
select * from concurs;
```

--BENEFICIAR

```
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Andronic', 'Marcel');
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data autentificare, puncte fidelitate)values(seq_beneficiar.nextval, 'Suditu', 'Mara');
```

All Rows Fetched: 5 in 0.029 seconds

COD_CONCURS	DATA_CONCURS	SUBIECT	DURATA	NR_INTREBARI	TAXA_INSCRIERE
1	100 12-APR-23	Biologie	10	15	20
2	110 18-APR-23	Matematica	20	20	30
3	120 27-FEB-23	Literatura	10	16	10
4	130 14-NOV-22	Diverse	7	8	10
5	140 15-JAN-23	Vulcani	13	15	25

SQL Worksheet | History

Worksheet | Query Builder

```

insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscrisere)values(seq_concurs.nextval, to_date('18-04-2023', 'dd-mm-yyyy'), 'Matematica', 20, 20, 30);
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscrisere)values(seq_concurs.nextval, to_date('27-02-2023', 'dd-mm-yyyy'), 'Literatura', 10, 16, 10);
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscrisere)values(seq_concurs.nextval, to_date('14-11-2022', 'dd-mm-yyyy'), 'Univers', 7, 9, 10);
insert into concurs (cod_concurs, data_concurs, subiect, durata, nr_intrebari, taxa_inscrisere)values(seq_concurs.nextval, to_date('15-01-2023', 'dd-mm-yyyy'), 'Vizitanti', 13, 15, 25);

--Beneficiar
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Andronic', 'Marcel', '+400733456753', '50012102009170', 'marcel123', 1, to_date('13-05-2021', 'dd-mm-yyyy'));
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Suditu', 'Mara', '+400745539465', '2980310309242', 'marsi1', 0, to_date('24-04-2021', 'dd-mm-yyyy'));
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Iuga', 'Eugen', '+400748201839', '5010410205807', 'dvd34_a', 1, to_date('30-05-2021', 'dd-mm-yyyy'));
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Lugoj', 'Eugen', '+400745678542', '13909515204669', 'eugen_123', 0, to_date('12-04-2021', 'dd-mm-yyyy'));
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Munteanu', 'Ana', '+400733456784', '6010413101111', 'ana_m', 0, to_date('16-05-2021', 'dd-mm-yyyy'));
insert into beneficiar (cod_beneficiar, nume, prenume, telefon, CNP, nume_utilizator, cont_premium, data_autentificare, puncte_fidelitate)values(seq_beneficiar.nextval, 'Maria', 'Ana', '+400733456784', '6010413123015', 'm_ana', 0, to_date('02-05-2022', 'dd-mm-yyyy'));

--ARTICOL
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 100, 'Casti wireless', 12);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 120, 'Bec ultra', 24);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 300, 'Incaricator telefon', 13);

```

All Rows Fetched: 7 in 0.127 seconds

COD_BENEFICIAR	NUME	PRENUME	TELEFON	CNP	NUME_UTILIZATOR	CONT_PREMIUM	DATA_AUTENTIFICARE	PUNCTE_FIDELITATE
1	100	Andronic	Marcel	+400733456753	50012102009170	marcel123	11-MAJ-21	100
2	110	Suditu	Mara	+400745539465	2980310309242	marsi1	02-APR-21	500
3	120	Iuga	Eugen	+400748201839	5010410205807	dvd34_a	13-MAJ-21	800
4	130	Lugoj	Eugen	+400745678542	13909515204669	eugen_123	08-APR-21	0
5	140	Alexandru	Mircea	+400704543478	6011041303816	mircea_12	01-APR-21	900
6	150	Munteanu	Ana	+400733456784	6010413101111	ana_m	02-MAR-22	0
7	160	Maria	Ana	+400733456784	6010413123015	m_ana	02-APR-22	0

Compiler - Log

Messages | Statements | Compiler | Logging Page |

lab5.sql | Welcome Page | creare\_inserare.sql | exercitiu19.sql | exercitiu13.sql | exercitiu11.sql | exercitiu100.sql | exercitiu8.sql | exercitiu111.sql

SQL Worksheet | History

Worksheet | Query Builder

```

--ARTICOL
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 100, 'Casti wireless', 12);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 120, 'Bec ultra', 24);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 300, 'Incaricator telefon', 13);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 400, 'Lanterna', 18);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 100, 'Frigider', 19);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 80, 'Aspirator robot', 25);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 10, 'Fier de calcat', 0);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 100, 'Uscator de par', 28);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 120, 'Uscator de par Julie Pro', 15);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 90, 'Telefon', 10);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 80, 'Laptop', 0);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 70, 'Tableta', 3);

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 1000, 'Pachet Pro', 28);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 800, 'Pachet electro', 28);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 1000, 'Pachet super', 29);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 5000, 'Pachet techno', 30);
insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 8000, 'Pachet all', 30);
select * from articol;

```

All Rows Fetched: 17 in 0.023 seconds

COD_ARTICOL	PRET	NUME	PUNCTE_FIDELITATE
103	400	Lanterna	18
5	104	Frigider	19
6	105	Aspirator robot	25
7	106	10 Fier de calcat	0
8	107	100 Uscator de par	28
9	108	120 Uscator de par Julie Pro	15
10	109	90 Telefon	10
11	110	80 Laptop	0
12	111	70 Tableta	3
13	112	1000 Pachet Pro	28
14	113	800 Pachet electro	28
15	114	1000 Pachet super	29
16	115	5000 Pachet techno	30
17	116	8000 Pachet all	30

SQL Worksheet | History

Worksheet | Query Builder

```

insert into articol (cod_articol, pret, nume, puncte_fidelitate)values(seq_articol.nextval, 8000, 'Pachet all', 30);
select * from articol;

--PRODUS
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(100, NULL, 12);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(101, 100, 18);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(102, 100, 0);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(103, 110, 18);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(104, 120, 24);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(105, 120, 18);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(106, 130, 24);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(107, 130, 18);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(108, 140, 18);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(109, 150, 15);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(110, 150, 6);
insert into produs (cod_articol, cod_furnizor, luni_garantie)values(111, 150, 3);
select * from produs;

--PACHET
insert into pachet_promotional (cod_articol)values(112);
insert into pachet_promotional (cod_articol)values(113);

```

All Rows Fetched: 12 in 0.025 seconds

COD_ARTICOL	COD_FURNIZOR	LUNI_GARANTIE
1	100	(null)
2	101	100
3	102	100
4	103	110
5	104	120
6	105	120
7	106	130
8	107	130
9	108	140
10	109	150
11	110	150
12	111	150

Compiler - Log | Messages | Statements | Compiler | Logging Page

Dbsm Output

```

--PACHET
insert into pachet_promotional (cod_articol)values(112);
insert into pachet_promotional (cod_articol)values(113);
insert into pachet_promotional (cod_articol)values(114);
insert into pachet_promotional (cod_articol)values(115);
insert into pachet_promotional (cod_articol)values(116);
select* from pachet_promotional;

--PRODUS_IN_PACHET
insert into produs_in_pachet(cod_produs, cod_pachet)values(100, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(101, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(102, 112);

```

All Rows Fetched: 5 in 0.018 seconds

COD_ARTICOL	
1	112
2	113
3	114
4	115
5	116

File Edit View Navigate Run Source Team Tools Window Help

SQL Worksheet History

Worksheet Query Builder

```

select * from packet_promotional;

--PRODUS_IN_PACHET
insert into produs_in_pachet(cod_produs, cod_pachet)values(100, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(101, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(102, 112);
insert into produs_in_pachet(cod_produs, cod_pachet)values(103, 113);
insert into produs_in_pachet(cod_produs, cod_pachet)values(104, 113);
insert into produs_in_pachet(cod_produs, cod_pachet)values(105, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(106, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(107, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(108, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(109, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(100, 114);
insert into produs_in_pachet(cod_produs, cod_pachet)values(102, 115);
insert into produs_in_pachet(cod_produs, cod_pachet)values(105, 115);
insert into produs_in_pachet(cod_produs, cod_pachet)values(104, 116);
insert into produs_in_pachet(cod_produs, cod_pachet)values(108, 116);
select * from produs_in_pachet;

```

All Rows Fetched: 15 in 0.022 seconds

COD_PRODUS	COD_PACHET
1	100
2	100
3	101
4	102
5	102
6	103
7	104
8	104
9	105
10	105
11	106
12	107
13	108
14	108
15	109

```

--PREMIU
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 100, 103, 60, 20);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 100, 100, 70, 30);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 100, 108, 90, 40);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 110, NULL, 80, 30);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 110, 103, 90, 35);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 120, NULL, 70, 210);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 120, 102, 75, 20);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 130, 101, 80, 0);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 130, 105, 99, 30);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 140, 111, 100, 0);
insert into premiu (cod_premiu, cod_concurs, cod_articol, punctaj_minim, suma)values(seq_premiu.nextval, 140, 104, 50, 10);
select * from premiu;

--PARTICIPA
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (100, 110, 50);

```

All Rows Fetched: 11 in 0.036 seconds

COD_PREMIU	COD_CONCURS	COD_ARTICOL	PUNCTAJ_MINIM	SUMA
1	100	100	103	60 20
2	110	100	100	70 30
3	120	100	108	90 40
4	130	110	(null)	80 30
5	140	110	103	90 35
6	150	120	(null)	70 210
7	160	120	102	75 20
8	170	130	101	80 0
9	180	130	105	99 30
10	190	140	111	100 0
11	200	140	104	50 10

SQL Worksheet | history

Worksheet | Query Builder

```

insert into participa (cod_concurs, cod_beneficiar, punctaj) values (110, 110, 94);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (110, 130, 88);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (110, 140, 50);
;
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (120, 100, 97);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (120, 110, 73);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (120, 140, 60);
;
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (130, 100, 99);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (130, 110, 90);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (130, 130, 79);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (130, 140, 73);
;
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (140, 100, 99);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (140, 110, 45);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (140, 130, 74);
insert into participa (cod_concurs, cod_beneficiar, punctaj) values (140, 140, 100);
select * from participa;

```

All Rows Fetched: 19 in 0.028 seconds

COD_CONCURS	COD_BENEFICIAR	PUNCTAJ
1	100	110
2	100	100
3	100	130
4	100	140
5	110	100
6	110	110
7	110	130
8	110	140
9	120	100
10	120	110
11	120	140
12	130	100
13	130	110
14	130	130
15	130	140

Compiler - Log

Messages | Statements | Compiler | Logging Page

Worksheet | Query Builder

```

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Serban', 'Alina', 3700, '+400767584957', to_date('12-09-2022', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Mihai', 'Costin', 3650, '+400778965435', to_date('09-09-2021', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Lazar', 'Simon', 3700, '+400785657453', to_date('05-09-2021', 'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Moldovan', 'Andreea', 5000, '+400709876547', to_date('12-04-2021', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Oprea', 'Pavel', 5500, '+400798765467', to_date('12-01-2022', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Dumitrescu', 'Bianca', 5600, '+400785768594', to_date('18-06-2022', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Dinu', 'Dobrin', 5400, '+400734567895', to_date('19-10-2021', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Cristea', 'Cristina', 5400, NULL, to_date('28-11-2021', 'dd-mm-yyyy'));

insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Todorescu', 'Simon', 6000, '+400734567845', to_date('12-09-2023', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Anonescu', 'Briana', 7000, '+400787657893', to_date('12-11-2023', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Cutas', 'Mirel', 5600, '+400797865467', to_date('12-11-2023', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Christigiu', 'Madelina', 5700, '+400785955954', to_date('12-02-2023', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Costetchi', 'Catalina', 5700, '+400734569595', to_date('28-05-2022', 'dd-mm-yyyy'));
insert into angajat (cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (seq_angajat.nextval, 'Clubelica', 'Cristina', 6500, NULL, to_date('29-04-2024', 'dd-mm-yyyy'));

select * from angajat;

```

All Rows Fetched: 16 in 0.029 seconds

COD_ANGAJAT	NUME	PRENUME	SALARIU	TELEFON	DATA_ANGAJARE
1	100	Lopataru	Alexandra	3500	+400758465748
2	110	Cujbescu	Marius	3600	+400745678954
3	120	Serban	Alina	3700	+400767584957
4	130	Mihai	Costin	3650	+400778965435
5	140	Lazar	Simon	3700	+400785657453
6	150	Moldovan	Andreea	5000	+400709876547
7	160	Oprea	Pavel	5500	+400798765467
8	170	Dumitrescu	Bianca	5600	+400785768594
9	180	Dinu	Dobrin	5400	+400734567895
10	190	Cristea	Cristina	5400	(null)
11	200	Todorescu	Simon	6000	+400734567845
12	210	Anonescu	Briana	7000	+400787657893
13	220	Cutas	Mirel	5600	+400797865467
14	230	Christigiu	Madelina	5700	+400785955954
15	240	Costetchi	Catalina	5700	+400734569595

```
SELECT * FROM angajat;

--SCENARIST
insert into scenarist (cod_angajat, zi_libera) values (100, 'miercuri');
insert into scenarist (cod_angajat, zi_libera) values (110, 'luni');
insert into scenarist (cod_angajat, zi_libera) values (120, 'luni');
insert into scenarist (cod_angajat, zi_libera) values (130, 'joi');
insert into scenarist (cod_angajat, zi_libera) values (140, 'vineri');
select * from scenarist;

--ADMINISTRATOR
insert into administrator (cod_angajat) values (150);
insert into administrator (cod_angajat) values (160);
insert into administrator (cod_angajat) values (170);

SQL | All Rows Fetched: 5 in 0.018 seconds
```

	COD_ANGAJAT	ZI_LIBERA
1	100	miercuri
2	110	luni
3	120	luni
4	130	joi
5	140	vineri

```
--ADMINISTRATOR
insert into administrator (cod_angajat) values (150);
insert into administrator (cod_angajat) values (160);
insert into administrator (cod_angajat) values (170);
insert into administrator (cod_angajat) values (180);
insert into administrator (cod_angajat) values (190);
select * from administrator;

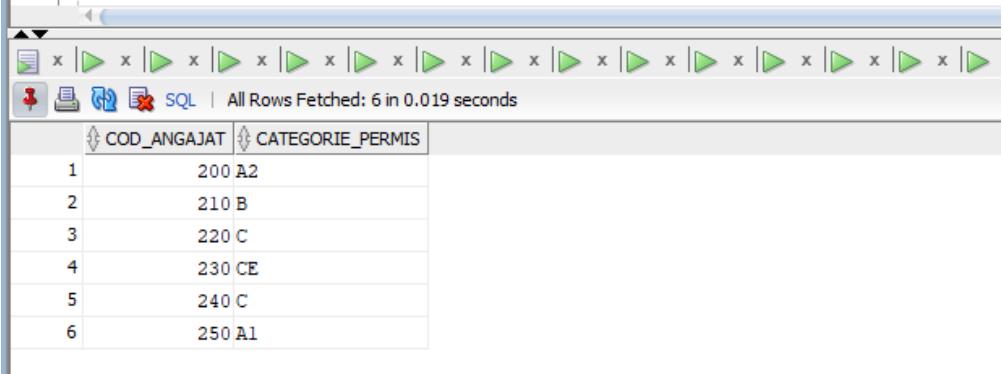
--CURIER
insert into curier (cod_angajat, categorie_permis) values (200, 'A2');

SQL | All Rows Fetched: 5 in 0.016 seconds
```

	COD_ANGAJAT
1	150
2	160
3	170
4	180
5	190

```
--CURIER
insert into curier (cod_angajat, categorie_permis) values (200, 'A2');
insert into curier (cod_angajat, categorie_permis) values (210, 'B');
insert into curier (cod_angajat, categorie_permis) values (220, 'C');
insert into curier (cod_angajat, categorie_permis) values (230, 'CE');
insert into curier (cod_angajat, categorie_permis) values (240, 'C');
insert into curier (cod_angajat, categorie_permis) values (250, 'A1');
select * from curier;

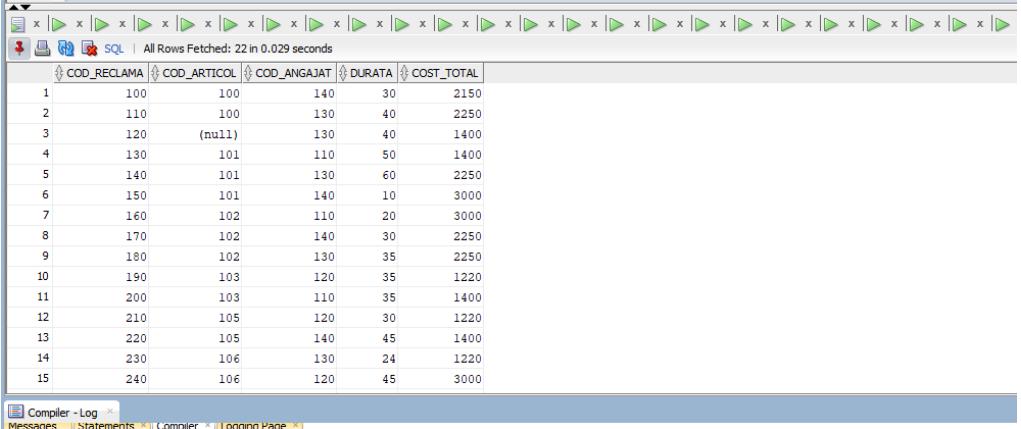
--RECLAMA
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) v
```



SQL Worksheet | History

Worksheet | Query Builder

```
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 102, 110, 20, 3000);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 102, 140, 30, 2250);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 102, 130, 35, 2250);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 103, 120, 35, 1200);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 103, 110, 35, 1400);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 105, 120, 30, 1220);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 105, 140, 45, 1400);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 106, 130, 24, 1220);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 106, 120, 45, 3000);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 107, 110, 30, 1220);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 109, 140, 30, 500);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 110, 130, 45, 4000);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 111, 110, 55, 5000);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 110, 120, 30, 4030);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, NULL, 120, 42, 2000);
insert into reclama (cod_reclama, cod_articol, cod_angajat, durata, cost_total) values (seq_reclama.nextval, 108, 130, 25, 600);
select * from reclama;
```



```
--SEDIU
insert into sediu (cod_sediu, cod_administrator, nume, adresa, suprafata) values (seq_sediu.nextval, 150, 'Electric center', 'Bucuresti Sud', 10000);
insert into sediu (cod_sediu, cod_administrator, nume, adresa, suprafata) values (seq_sediu.nextval, 160, 'Cable power', 'Centru Bucuresti City', 12000);
insert into sediu (cod_sediu, cod_administrator, nume, adresa, suprafata) values (seq_sediu.nextval, 170, 'Atacul tehnologiei', 'Ploiesti Periferie', 5000);
insert into sediu (cod_sediu, cod_administrator, nume, adresa, suprafata) values (seq_sediu.nextval, 170, 'IT Party', 'Buzau Cartier Dorobanti', 6000);
insert into sediu (cod_sediu, cod_administrator, nume, adresa, suprafata) values (seq_sediu.nextval, 190, 'ELECTRO LIGHT MAIN', 'Bucuresti Strada Pacii', 12000);
select * from sediu;

--ARTICOL_IN_SEDIU
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (100, 100, 60, 105.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (100, 101, 70, 115.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (100, 102, 80, 290.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (100, 103, 50, 410.00);

(

```

All Rows Fetched: 5 in 0.043 seconds

COD_SEDIU	COD_ADMINISTRATOR	NUME	ADRESA	SUPRAFATA
1	100	150 Electric center	Bucuresti Sud	10000
2	110	160 Cable power	Centru Bucuresti City	12000
3	120	170 Atacul tehnologiei	Ploiesti Periferie	5000
4	130	170 IT Party	Buzau Cartier Dorobanti	6000
5	140	190 ELECTRO LIGHT MAIN	Bucuresti Strada Pacii	12000

Worksheet | Query Builder

```
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 110, 70, 880.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 113, 12, 805.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 114, 5, 1015.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 115, 3, 5050.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (130, 116, 2, 8050.00);

insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 100, 55, 95.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 101, 75, 110.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 102, 85, 290.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 103, 65, 425.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 108, 110, 130.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 109, 75, 420.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 110, 60, 875.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 111, 50, 1420.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 112, 20, 950.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 113, 10, 810.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 114, 5, 1010.00);
insert into articol_in_sediu (cod_sediu, cod_articol, cantitate, pret) values (140, 115, 4, 5100.00);
select * from articol_in_sediu;
```

Fetched 50 rows in 0.021 seconds

COD_SEDIU	COD_ARTICOL	CANTITATE	PRET
1	100	100	60 105
2	100	101	70 115
3	100	102	80 290
4	100	103	50 410
5	100	104	90 395
6	100	107	60 107
7	100	108	100 132
8	100	109	70 425
9	100	110	55 885
10	100	111	45 1390
11	100	112	30 980
12	100	113	25 800
13	100	114	15 1020
14	100	115	10 5100
15	100	116	5 8100

Compiler - Log

```
--DETALII TRANZACTIE
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 100, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 101, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 102, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 103, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 104, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 105, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 106, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 107, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 108, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 109, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 110, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 111, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 112, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 113, 'nelivrata', 0, 0, 0, 0, 0, 0)
insert into detalii_tranzactie (cod_detaliu_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_puncte, suma_platita_card, suma_platita_numar) values (seq_detaliu_tranzactie.nextval, 114, 'nelivrata', 0, 0, 0, 0, 0, 0)

select * from detalii_tranzactie;
```

All Rows Fetched: 15 0.027 seconds

COD_DETALII_TRANZACTIE	COD_BENEFICIAR	DATA_TRANZACTIE	COST_TOTAL	PUNCTE_FIDELITATE_CASTIGATE	PUNCTE_FIDELITATE_FOLOSITE	SUMA_PLATITA_IN_PUNCTE	SUMA_PLATITA_CARD	SUMA_PLATITA_NUMAR
1	100	120-01-MAR-24	550	25	0	0	550	0
2	101	110-02-JAN-24	820	22	0	0	820	0
3	102	100-03-AUD-24	940	24	55	20	0	920
4	103	100-04-NOV-24	570	27	0	0	570	0
5	104	110-06-DEC-24	1060	26	30	10	0	1050
6	105	110-07-FEB-24	850	20	25	10	0	750
7	106	110-07-FEB-23	930	23	0	0	930	0
8	107	110-08-MAR-23	250	25	0	0	250	0
9	108	100-09-APR-23	280	28	0	0	280	0
10	109	130-10-MAY-23	1020	22	0	0	1020	0
11	110	100-11-JUN-23	240	24	0	0	100	140
12	111	130-10-JUN-22	1060	26	10	10	0	1050
13	112	100-13-AUG-23	220	22	0	0	220	0
14	113	130-14-OCT-23	950	25	0	0	950	0
15	114	140-15-NOV-23	280	28	0	0	280	0

Compiler - Log

```
--COMANDA
insert into comanda (cod_comanda, cod_detaliu_tranzactie, status_comanda, plata_la_livrare, tip_comanda) values (seq_comanda.nextval, 100, 'nelivrata', 0, 'generală');
insert into comanda (cod_comanda, cod_detaliu_tranzactie, status_comanda, plata_la_livrare, tip_comanda) values (seq_comanda.nextval, 101, 'nelivrata', 0, 'generală');
insert into comanda (cod_comanda, cod_detaliu_tranzactie, status_comanda, plata_la_livrare, tip_comanda) values (seq_comanda.nextval, 103, 'nelivrata', 0, 'generală');
insert into comanda (cod_comanda, cod_detaliu_tranzactie, status_comanda, plata_la_livrare, tip_comanda) values (seq_comanda.nextval, 102, 'livrata', 1, 'generală');
insert into comanda (cod_comanda, cod_detaliu_tranzactie, status_comanda, plata_la_livrare, tip_comanda) values (seq_comanda.nextval, 104, 'livrata', 1, 'generală');

select * from comanda;
```

```
--COLET
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 100, 200, 100, 200, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 100, 200, 100, 200, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 100, 210, 120, 300, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 100, 210, 120, 300, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 101, 220, 110, 400, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 101, 220, 110, 400, 'nelivrat', NULL);

select * from colet;
```

All Rows Fetched: 5 0.026 seconds

COD_COMANDA	COD_DETALII_TRANZACTIE	STATUS_COMANDA	TIP_COMANDA	PLATA_LA_LIVRARE
1	100	100nelivrata	generală	0
2	101	101nelivrata	generală	0
3	102	103 nelivrata	generală	0
4	103	102 livrata	generală	1
5	104	104 livrata	generală	1

```
--COLET
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 100, 200, 100, 200, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 100, 200, 110, 100, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 100, 210, 120, 300, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq_colet.nextval, 101, NULL, NULL, 350, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq.colet.nextval, 101, 220, 110, 400, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq.colet.nextval, 102, NULL, 110, 210, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq.colet.nextval, 102, 230, 120, 310, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq.colet.nextval, 102, NULL, 130, 280, 'nelivrat', NULL);
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq.colet.nextval, 103, 230, 130, 210, 'nelivrat', to_date('01-01-2024', 'dd-mm-yyyy'));
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq.colet.nextval, 103, 240, 100, 200, 'nelivrat', to_date('01-03-2024', 'dd-mm-yyyy'));
insert into colet (cod_colet, cod_comanda, cod_angajat, cod_sediu, greutate, status_colet, data_livrare) values (seq.colet.nextval, 104, 240, 120, 500, 'nelivrat', to_date('01-08-2023', 'dd-mm-yyyy'));

select * from colet;
```

```
--VANZARE FIZICA
4
```

All Rows Fetched: 11 0.04 seconds

COD_COLET	COD_COMANDA	COD_ANGAJAT	COD_SEDIU	GREUTATE	DATA_LIVRARE	STATUS_COLET
1	100	100	200	100	200 (null)	nelivrat
2	101	100	200	110	100 (null)	nelivrat
3	102	100	210	120	300 (null)	nelivrat
4	103	101 (null)	(null)	350 (null)	nelivrat	nelivrat
5	104	101	220	110	400 (null)	nelivrat
6	105	102 (null)	110	210 (null)	nelivrat	nelivrat
7	106	102	230	120	310 (null)	nelivrat
8	107	102 (null)	130	280 (null)	nelivrat	nelivrat
9	108	103	230	130	210 01-JAN-24	livrat
10	109	103	240	100	200 01-MAR-24	livrat
11	110	104	240	120	500 01-AUG-23	livrat

```
--VANZARE FIZICA
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 100, 105);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 100, 106);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 110, 107);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 110, 108);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 120, 109);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 110);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 111);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 112);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 113);
insert into vanzare_fizica(cod_vanzare_fizica, cod_sediu, cod_detalii_tranzactie) values (seq_vanzare_fizica.nextval, 130, 114);

select * from vanzare_fizica;

--COS
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(100, 100, 1, to_date('01-01-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(101, 100, 2, to_date('02-01-2023', 'dd-mm-yyyy'), 1)

```

All Rows Fetched: 10 in 0.025 seconds

COD_VANZARE_FIZICA	COD_SEDIU	COD_DETALII_TRANZACTIE
1	100	100
2	101	100
3	102	110
4	103	110
5	104	120
6	105	130
7	106	130
8	107	130
9	108	130
10	109	130
		114

```
--COS
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(100, 100, 1, to_date('01-01-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(101, 100, 2, to_date('02-01-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(102, 100, 3, to_date('03-01-2024', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(104, 110, 1, to_date('04-02-2024', 'dd-mm-yyyy'), 0);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(105, 110, 1, to_date('05-03-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(106, 110, 1, to_date('06-01-2023', 'dd-mm-yyyy'), 0);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(113, 120, 1, to_date('07-04-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(114, 120, 2, to_date('08-05-2023', 'dd-mm-yyyy'), 0);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(115, 120, 3, to_date('09-01-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(102, 100, 4, to_date('10-06-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(101, 100, 1, to_date('11-01-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(104, 130, 1, to_date('12-07-2024', 'dd-mm-yyyy'), 0);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(106, 140, 1, to_date('13-08-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(109, 140, 1, to_date('14-09-2024', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(105, 140, 2, to_date('15-01-2023', 'dd-mm-yyyy'), 1);
insert into cos(cod_articol, codBeneficiar, cantitate, dataAdaugare, bifat) values(107, 140, 1, to_date('16-01-2024', 'dd-mm-yyyy'), 0);

select * from cos;
```

All Rows Fetched: 16 in 0.029 seconds

COD_ARTICOL	COD_BENEFICIAR	CANTITATE	DATA_ADAUGARE	BIAT
1	100	100	01-JAN-23	1
2	101	100	02-JAN-23	1
3	102	100	03-JAN-24	0
4	104	110	04-FEB-24	1
5	105	110	05-MAR-23	1
6	106	110	06-JAN-23	0
7	113	120	07-APR-23	1
8	114	120	09-MAY-23	0
9	115	120	09-JAN-23	1
10	102	130	10-JUN-23	1
11	101	130	11-JAN-23	1
12	104	130	12-JUL-24	0
13	106	140	13-AUG-23	1
14	109	140	14-SEP-24	1
15	105	140	21-NOV-23	1

```
--RECENTIE
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 100, 100, to_date('07-08-2023', 'dd-mm-yyyy'), 8, 'recomand tuturor');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 100, 100, to_date('09-08-2023', 'dd-mm-yyyy'), 10, 'mai efficient decat ma asteptam');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 101, 110, to_date('10-08-2023', 'dd-mm-yyyy'), 10, 'recomand tuturor');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 113, 110, to_date('06-08-2023', 'dd-mm-yyyy'), 7, 'recomand tuturor');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 105, 120, to_date('07-05-2023', 'dd-mm-yyyy'), 4, 'nu recomand');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 104, 120, to_date('09-08-2023', 'dd-mm-yyyy'), 6, 'poate fi imbunatatit');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 105, 140, to_date('10-03-2023', 'dd-mm-yyyy'), 8, 'recomand cu caldura');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 100, 140, to_date('10-03-2023', 'dd-mm-yyyy'), 9, 'de inalta calitate');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 110, 140, to_date('10-03-2023', 'dd-mm-yyyy'), 7, 'destul de bun');
insert into recenzie (cod_recenzie, cod_articol, cod_beneficiar, data_recenzie, nota, comentariu) values (seq_recenzie.nextval, 110, 140, to_date('04-08-2023', 'dd-mm-yyyy'), 3, 'nu recomand deloc');

--ARTICOL_IN_COLET
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (100, 100, 100, 3);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (102, 100, 100, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (104, 100, 100, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (115, 101, 100, 1);

All Rows Fetched: 10 in 0.056 seconds
```

COD_RECENZIE	COD_ARTICOL	COD_BENEFICIAR	DATA_RECENZIE	NOTA	COMENTARIU
1	100	100	100-07-AUG-23	8	recomand tuturor
2	101	100	100-09-AUG-23	10	mai efficient decat ma asteptam
3	102	101	110-10-AUG-23	10	recomand tuturor
4	103	113	110-06-AUG-23	7	recomand tuturor
5	104	103	120-07-MAY-23	4	nu recomand
6	105	104	120-09-AUG-23	6	poate fi imbunatatit
7	106	105	140-07-MAR-23	8	recomand cu caldura
8	107	114	140-03-AUG-23	9	de inalta calitate
9	108	109	140-07-FEB-23	7	destul de bun
10	109	110	140-04-AUG-23	3	nu recomand deloc

```
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (107, 102, 100, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (104, 103, 101, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (109, 103, 101, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (113, 104, 101, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (101, 104, 101, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (112, 104, 101, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (105, 105, 102, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (114, 105, 102, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (115, 105, 102, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (108, 106, 102, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (110, 107, 102, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (107, 108, 103, 3);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (103, 109, 103, 4);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (114, 109, 103, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (100, 110, 104, 1);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (112, 110, 104, 2);
insert into articol_in_colet (cod_articol, cod_colet, cod_comanda, cantitate) values (116, 110, 104, 2);
select * from articol_in_colet;
```

COD_RECENZIE	COD_ARTICOL	COD_COLET	COD_COMANDA	CANTITATE
1	100	100	100	3
2	102	100	100	2
3	104	100	100	1
4	115	101	100	1
5	103	101	100	1
6	106	102	100	2
7	107	102	100	2
8	104	103	101	1
9	109	103	101	1
10	113	104	101	2
11	101	104	101	2
12	112	104	101	2
13	105	105	102	2
14	114	105	102	2
15	115	105	102	2

SQl Worksheet History

Worksheet | Query Builder

```

insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (103,116, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (103,112, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (104, 114, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (104, 115, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 116, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 112, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (105, 114, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (106, 114, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (106, 115, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (106, 116, 2);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (107, 112, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (107, 113, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (107, 107, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (108, 107, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (108, 104, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (108, 108, 1);
insert into articol_in_vanzare(cod_vanzare_fizica, cod_articol, cantitate) values (109, 102, 1);
select * from articol_in_vanzare;

```

SQL | All Rows Fetched: 27 in 0.022 seconds

	COD_VANZARE_FIZICA	COD_ARTICOL	CANTITATE
1	100	100	1
2	100	101	2
3	100	102	1
4	101	101	1
5	101	104	1
6	101	108	2
7	102	109	1
8	102	111	1
9	102	112	2
10	103	113	1
11	103	116	1
12	103	112	1
13	104	114	1
14	104	115	1
15	105	116	2

Compiler - Log ×  
Messages Statements Compiler Logging Page

**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.**

**Cerința în limbaj natural:**

Să se creeze o procedură care are ca parametru de intrare codul unui sediu.

În acest sediu a ajuns un transport cu produsele cu codurile 101, 102, 105 în cantitățile 100, 120, 200.

Creșteți stocul. Dacă nu există o înregistrare cu un articol puneti prețul universal, din tabela articol.

După aceea, pentru o imagine de ansamblu, afișați pentru fiecare furnizor numele, produsele sale și cantitatea totală din fiecare produs, plus cantitatea în care se găsește în fiecare sediu (inclusiv 0).

```
CREATE OR REPLACE PROCEDURE exercitiul6 (v_cod_sediu IN sediu.cod_sediu%TYPE)
```

```
IS
```

```
    TYPE transport_record IS RECORD (cod_articol articol.cod_articol%TYPE, cantitate  
    articol_in_sediu.cantitate%TYPE);
```

```
    TYPE tablou_indexat_transporturi IS TABLE OF transport_record INDEX BY  
    PLS_INTEGER;
```

```
    t_transporturi tablou_indexat_transporturi;
```

```
    v_pret articol_in_sediu.cantitate%TYPE;
```

```
    c_maxim_furnizori CONSTANT NUMBER := 1000;
```

```
    TYPE vector_furnizori IS VARRAY(c_maxim_furnizori) OF furnizor%ROWTYPE;
```

```
    t_furnizori vector_furnizori := vector_furnizori();
```

```
    TYPE produs_record IS RECORD (cod_articol articol.cod_articol%TYPE, nume  
    articol.nume%TYPE);
```

```
    TYPE tablou_imbricat_produse IS TABLE OF produs_record;
```

```

t_produse tablou_imbricat_produse := tablou_imbricat_produse();

TYPE sediu_record is RECORD (
    cod_sediu sediu.cod_sediu%TYPE,
    nume sediu.nume%TYPE,
    cantitate articol_in_sediu.cantitate%TYPE);

TYPE tablou_indexat_sedii IS TABLE OF sediu_record INDEX BY BINARY_INTEGER;
t_sedii tablou_indexat_sedii;
v_cantitate_totala int;

BEGIN

```

```

    t_transporturi(1) := transport_record(101, 100);
    t_transporturi(2) := transport_record(102, 120);
    t_transporturi(3) := transport_record(105, 200);

```

FOR i IN t\_transporturi.FIRST..t\_transporturi.LAST LOOP --iteram transporturile si le inseram in crestem stocurile

```

        UPDATE articol_in_sediu
        SET cantitate = cantitate + t_transporturi(i).cantitate
        WHERE cod_sediu = v_cod_sediu AND cod_articol = t_transporturi(i).cod_articol;

```

IF SQL%ROWCOUNT = 0 THEN --daca nu a updatat nicio linie inseamna ca produsul nu exista deloc in sediu si trebuie adaugata o inserare

```

        SELECT pret INTO v_pret FROM articol WHERE cod_articol =
        t_transporturi(i).cod_articol; --pretul il luam din tabela articol, adica pretul universal
        INSERT INTO articol_in_sediu (cod_sediu, cod_articol, cantitate, pret)
        VALUES (v_cod_sediu, t_transporturi(i).cod_articol, t_transporturi(i).cantitate,
        v_pret);
    END IF;

```

```
END LOOP;
```

```
FOR i in (SELECT * FROM furnizor) LOOP --retinem in vector toate datele din tabela  
furnizor (nume si cod, mai e si o adresa de mail dar nu ne trebuie)
```

```
    t_furnizori.EXTEND;
```

```
    t_furnizori(t_furnizori.LAST) := i;
```

```
END LOOP;
```

```
FOR i IN t_furnizori.FIRST..t_furnizori.LAST LOOP --luam produsele furnizate
```

```
    SELECT p.cod_articol, nume
```

```
    BULK COLLECT INTO t_produse
```

```
    FROM produs p, articol a
```

```
    WHERE cod_furnizor = t_furnizori(i).cod_furnizor AND p.cod_articol = a.cod_articol;
```

```
IF t_produse.COUNT = 0 THEN DBMS_OUTPUT.PUT_LINE('Furnizorul ' ||  
t_furnizori(i).nume || ' nu furnizeaza produse');
```

```
ELSE
```

```
    DBMS_OUTPUT.PUT_LINE('Furnizorul ' || t_furnizori(i).nume || ' ne vinde ' ||  
t_produse.COUNT || ' produs(e):');
```

```
FOR j IN t_produse.FIRST..t_produse.LAST LOOP
```

```
    v_cantitate_totala := 0;
```

```
    DBMS_OUTPUT.PUT_LINE('      ' || t_produse(j).nume || ', cantitati: ');
```

```
    SELECT s.cod_sediu, s.nume, NVL(ais.cantitate, 0) AS cantitate
```

```
    BULK COLLECT INTO t_sedii --pt un produs aflam cantitatea din fiecare  
    sediu
```

```
    FROM sediu s, articol_in_sediu ais
```

```

        WHERE cod_articol(+) = t_produse(j).cod_articol AND s.cod_sediu =
ais.cod_sediu(+)

        ORDER BY s.nume;

FOR k IN t_sedii.FIRST..t_sedii.LAST LOOP

        DBMS_OUTPUT.PUT_LINE('          sediu' || t_sedii(k).nume || ':' |
|| t_sedii(k).cantitate || ' bucati');

        v_cantitate_totala := v_cantitate_totala + t_sedii(k).cantitate;

        END LOOP;

--t_sedii.DELETE; --nu trebuie neaparat sters

        DBMS_OUTPUT.PUT_LINE('          Cantitatea totala: ' || |
v_cantitate_totala || ' bucati');

        END LOOP;

DBMS_OUTPUT.PUT_LINE('=====');
);

--t_produse.DELETE;

END IF;

END LOOP;

END exercitiul6;

/

--articolele din sediu 100 inainte de executare:

select * from articol_in_sediu where cod_sediu = 100;

BEGIN

    exercitiul6(100);

```

```
END;
```

```
/
```

--articolele din sediul 100 dupa executare:

```
select * from articol_in_sediu where cod_sediu = 100;
```

```
rollback;
```

```
--Dupa aceea, pentru o imagine de ansamblu, afisati pentru fiecare furnzior numele, produsele sale si cantitatea totala din f
CREATE OR REPLACE PROCEDURE exercitiul6 (v_cod_sediu IN sediu.cod_sediu%TYPE)
IS
    TYPE transport_record IS RECORD (cod_articol articol.cod_articol%TYPE, cantitate articol_in_sediu.cantitate%TYPE);
    TYPE tablou_indexat_transporturi IS TABLE OF transport_record INDEX BY PLS_INTEGER;
    t_transporturi tablou_indexat_transporturi;
    v_pret articol_in_sediu.cantitate%TYPE;
    c_maxim_furnizori CONSTANT NUMBER := 1000;
    TYPE vector_furnizori IS VARRAY(c_maxim_furnizori) OF furnizor%ROWTYPE;
    t_furnizori vector_furnizori := vector_furnizori();
    TYPE produs_record IS RECORD (cod_articol articol.cod_articol%TYPE, nume articol.nume%TYPE);
    TYPE tablou_imbricat_produse IS TABLE OF produs_record;
    t_produse tablou_imbricat_produse := tablou_imbricat_produse();
    TYPE sediu_record IS RECORD (cod_sediu sediu.cod_sediu%TYPE, nume sediu.nume%TYPE, cantitate articol_in_sediu.cantitate%TYPE);
    TYPE tablou_indexat_sediu IS TABLE OF sediu_record INDEX BY BINARY_INTEGER;
    t_sediu tablou_indexat_sediu;
    v_cantitate_totala int;
BEGIN
    t_transporturi(1) := transport_record(101, 100);
    t_transporturi(2) := transport_record(102, 120);
    t_transporturi(3) := transport_record(105, 200);

    FOR i IN t_transporturi.FIRST..t_transporturi.LAST LOOP --iteram transporturile si le inseram in restul stocurilor
        UPDATE articol_in_sediu
        SET cantitate = cantitate + t_transporturi(i).cantitate
        WHERE cod_sediu = v_cod_sediu AND cod_articol = t_transporturi(i).cod_articol;

        IF SQL%ROWCOUNT = 0 THEN --daca nu a updateat nicio linie inseamna ca produsul nu exista deloc in sediu si trebuie adaugat
            SELECT pret INTO v_pret FROM articol WHERE cod_articol = t_transporturi(i).cod_articol; --pretul il luam din articol
            INSERT INTO articol_in_sediu (cod_sediu, cod_articol, cantitate) VALUES (v_cod_sediu, t_transporturi(i).cod_articol, v_pret);
        END IF;
    END LOOP;
    COMMIT;
END;
```

Procedure EXERCITIUL6 compiled

Compiler - Log

## Articolele din sediul 100 înainte de executare:

The screenshot shows the Oracle SQL Developer interface. In the top tab bar, several files are listed: 'create\_inserare.sql', 'exercitiul100.sql', 'exercitiu11.sql', 'exercitiul8.sql', and 'exercitiul6.sql'. The main window has two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab contains the following SQL code:

```
--articolele din sediul 100 înainte de executare:  
select * from articol_in_sediul where cod_sediul = 100;  
  
BEGIN  
    exercitiul6(100);  
END;
```

The 'Query Result' tab shows the output of the query, which is a table with 15 rows. The columns are labeled 'COD\_SEDIU', 'COD\_ARTICOL', 'CANTITATE', and 'PRET'. The data is as follows:

	COD_SEDIU	COD_ARTICOL	CANTITATE	PRET
1	100	100	60	105
2	100	101	70	115
3	100	102	80	290
4	100	103	50	410
5	100	104	90	395
6	100	107	60	107
7	100	108	100	132
8	100	109	70	425
9	100	110	55	885
10	100	111	45	1390
11	100	112	30	980
12	100	113	25	800
13	100	114	15	1020
14	100	115	10	5100
15	100	116	5	8100

```

SQL Worksheet | History
Worksheet | Query Builder | conexiune_noua
-- exercitiul6;
/
--articolele din sediul 100 inainte de executare:
select * from articol_in_sediul where cod_sediul = 100;

BEGIN
    exercitiul6(100);
END;
/

```

Script Output | Query Result | Query Result 1 | Task completed in 0.229 seconds  
PL/SQL procedure successfully completed.

Compiler - Log | Messages | Logging Page | Statements | Compiler | Dbsms Output | Buffer Size: 20000 | conexiune\_noua

```

Furnizorul Electronice high quality ne vinde 2 produs(e):
Bec ultra, cantitati:
    sediulAtacul tehnologiei: 65 bucati
    sediulCable power: 75 bucati
    sediulELECTRO LIGHT MAIN: 75 bucati
    sediulElectric center: 170 bucati
    sediulIT Party: 0 bucati
    Cantitatea totala: 385 bucati
Incarcator telefon, cantitati:
    sediulAtacul tehnologiei: 75 bucati
    sediulCable power: 85 bucati
    sediulELECTRO LIGHT MAIN: 85 bucati
    sediulElectric center: 200 bucati
    sediulIT Party: 90 bucati
    Cantitatea totala: 535 bucati
=====
Furnizorul Electric party ne vinde 1 produs(e):
Lanterna, cantitati:
    sediulAtacul tehnologiei: 50 bucati
    sediulCable power: 55 bucati
    sediulELECTRO LIGHT MAIN: 65 bucati

```

**Întreaga afișare:**

Furnizorul Electronice high quality ne vinde 2 produs(e):

Bec ultra, cantitati:

sediulAtacul tehnologiei: 65 bucati  
 sediulCable power: 75 bucati  
 sediulELECTRO LIGHT MAIN: 75 bucati  
 sediulElectric center: 170 bucati

sediulIT Party: 0 bucati

Cantitatea totala: 385 bucati

Incarcator telefon, cantitati:

sediulAtacul tehnologiei: 75 bucati

sediulCable power: 85 bucati

sediulELECTRO LIGHT MAIN: 85 bucati

sediulElectric center: 200 bucati

sediulIT Party: 90 bucati

Cantitatea totala: 535 bucati

=====

Furnizorul Electric party ne vinde 1 produs(e):

Lanterna, cantitati:

sediulAtacul tehnologiei: 50 bucati

sediulCable power: 55 bucati

sediulELECTRO LIGHT MAIN: 65 bucati

sediulElectric center: 50 bucati

sediulIT Party: 60 bucati

Cantitatea totala: 280 bucati

=====

Furnizorul Electronics++ ne vinde 2 produs(e):

Frigider, cantitati:

sediulAtacul tehnologiei: 85 bucati

sediulCable power: 85 bucati

sediulELECTRO LIGHT MAIN: 0 bucati

sediulElectric center: 90 bucati

sediulIT Party: 100 bucati

Cantitatea totala: 360 bucati

Aspirator robot, cantitati:

sediulAtacul tehnologiei: 0 bucati

sediuCable power: 1 bucati  
sediuELECTRO LIGHT MAIN: 0 bucati  
sediuElectric center: 200 bucati  
sediuIT Party: 0 bucati

Cantitatea totala: 201 bucati

---

Furnizorul IT Direct ne vinde 2 produs(e):

Fier de calcat, cantitati:

sediuAtacul tehnologiei: 0 bucati  
sediuCable power: 0 bucati  
sediuELECTRO LIGHT MAIN: 0 bucati  
sediuElectric center: 0 bucati  
sediuIT Party: 150 bucati

Cantitatea totala: 150 bucati

Uscator de par, cantitati:

sediuAtacul tehnologiei: 70 bucati  
sediuCable power: 65 bucati  
sediuELECTRO LIGHT MAIN: 0 bucati  
sediuElectric center: 60 bucati  
sediuIT Party: 80 bucati

Cantitatea totala: 275 bucati

---

Furnizorul Ultra circuite ne vinde 1 produs(e):

Uscator de par Julie Pro, cantitati:

sediuAtacul tehnologiei: 95 bucati  
sediuCable power: 105 bucati  
sediuELECTRO LIGHT MAIN: 110 bucati  
sediuElectric center: 100 bucati  
sediuIT Party: 120 bucati

Cantitatea totala: 530 bucati

---

Furnizorul Best Techno ne vinde 3 produs(e):

Telefon, cantitati:

sediulAtacul tehnologiei: 0 bucati

sediulCable power: 75 bucati

sediulELECTRO LIGHT MAIN: 75 bucati

sediulElectric center: 70 bucati

sediulIT Party: 90 bucati

Cantitatea totala: 310 bucati

Laptop, cantitati:

sediulAtacul tehnologiei: 65 bucati

sediulCable power: 60 bucati

sediulELECTRO LIGHT MAIN: 60 bucati

sediulElectric center: 55 bucati

sediulIT Party: 70 bucati

Cantitatea totala: 310 bucati

Tableta, cantitati:

sediulAtacul tehnologiei: 55 bucati

sediulCable power: 50 bucati

sediulELECTRO LIGHT MAIN: 50 bucati

sediulElectric center: 45 bucati

sediulIT Party: 0 bucati

Cantitatea totala: 200 bucati

---

Furnizorul Furnizor Electronice nu furnizeaza produse

**Articolele din sediul 100 după executare: a apărut la final înregistrarea pt articolul 105 și a crescut cantitatea pentru articolele 101 și 102.**

The screenshot shows the Oracle SQL Developer interface. At the top, there is a tab bar with several open files: 'creare\_inserare.sql', 'exercitiul100.sql', 'exercitiu11.sql', 'exercitiul8.sql', 'exercitiul13.sql', and 'exercitiul6.sql'. Below the tab bar is the 'Worksheet' tab, which contains a PL/SQL block:

```
select * from articol_in_sediu where cod_sediu = 100;  
--  
BEGIN  
    exercitiul6(100);  
END;  
/  
  
--articolele din sediul 100 dupa executare:  
select * from articol_in_sediu where cod_sediu = 100;
```

Below the worksheet is the 'Script Output' tab, which displays the results of the executed query. The output shows 16 rows of data with columns: COD\_SEDIU, COD\_ARTICOL, CANTITATE, and PRET. The data is as follows:

COD_SEDIU	COD_ARTICOL	CANTITATE	PRET	
1	100	100	60	105
2	100	101	170	115
3	100	102	200	290
4	100	103	50	410
5	100	104	90	395
6	100	107	60	107
7	100	108	100	132
8	100	109	70	425
9	100	110	55	885
10	100	111	45	1390
11	100	112	30	980
12	100	113	25	800
13	100	114	15	1020
14	100	115	10	5100
15	100	116	5	8100
16	100	105	200	80

**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.**

**Cerința în limbaj natural:**

Afișați pt fiecare colet nelivrăt și care trebuie să plece din sediul 'Cable Power' pachetele promotionale și produsele pe care le conțin aceste pachete, pentru a ajuta angajații să le împacheteze. Să se afișeze și curierul care îl va duce, dacă este cunoscut.

```
CREATE OR REPLACE PROCEDURE exercitiul7 (v_numere_sediul IN sediu.nume%TYPE  
DEFAULT 'CABLE POWER') IS
```

```
    TYPE refcursor IS REF CURSOR;
```

```
    v_cursor refcursor;
```

CURSOR c\_principal IS --ciclul cursor care selecteaza pentru un colet codul si curierul (daca exista); ia doar coletele care respecta cerinta

```
    SELECT c.cod_colet cod_colet, c.cod_comanda cod_comanda,  
    NVL(a.cod_angajat, -1) AS cod_curier,
```

```
        NVL(CONCAT(a.nume, ' ' || a.prenume), 'Curierul nu a fost inca asignat') AS  
        nume_curier
```

```
    FROM sediu s, colet c, angajat a
```

```
    WHERE c.cod_sediul = s.cod_sediul AND v_numere_sediul = UPPER(s.nume) AND  
    UPPER(c.status_colet) = 'NELIVRAT'
```

```
    AND a.cod_angajat = c.cod_angajat(+); --left join pentru cazul in care curierul are  
    cod_angajat NULL
```

CURSOR c\_secundar (v\_cod\_colet colet.cod\_colet%TYPE, v\_cod\_comanda  
colet.cod\_comanda%TYPE) IS --cursor clasic parametrizat cu un ref cursor in el;pentru o  
comanda afla cate produse din pachete promotionale are

```
    SELECT a.cod_articol as cod, a.nume as nume, aic.cantitate as cantitate,
```

```
    CURSOR(SELECT pip.cod_produs cod_produs, a2.nume nume2 -- expresie  
    cursor care pt un pachet afla ce produse are
```

```

        FROM produs_in_pachet pip, articol a2
        WHERE pip.cod_pachet = a.cod_articol AND a2.cod_articol =
pip.cod_produs)
        FROM articol_in_colet aic, pachet_promotional p, articol a
        WHERE aic.cod_colet = v_cod_colet and aic.cod_comanda = v_cod_comanda
and aic.cod_articol = p.cod_articol
        AND a.cod_articol = p.cod_articol;

v_detalii_curier varchar2(100);
v_cod_articol articol.cod_articol%TYPE;
v_nume_articol articol.nume%TYPE;
v_nume_produs articol.nume%TYPE;
v_cantitate articol_in_colet.cantitate%TYPE;
v_cod_produs produs.cod_articol%TYPE;

BEGIN
    FOR i IN c_principal LOOP
        EXIT WHEN c_principal%NOTFOUND;
        v_detalii_curier := ' (';
        IF i.cod_curier = -1
        THEN v_detalii_curier := v_detalii_curier || 'al carui curier nu a fost inca
asignat)';
        ELSE v_detalii_curier := v_detalii_curier || 'care va fi livrat de curierul ' ||
i.nume_curier || ' cod ' || i.cod_curier || ')';
        END IF;
        DBMS_OUTPUT.PUT_LINE('Coletul' || v_detalii_curier || ' cu codul ' ||
i.cod_colet || ' din comanda ' || i.cod_comanda || ',are pachetele:');
    END LOOP;
END;

```

```

OPEN c_secundar(i.cod_colet, i.cod_comanda);

    LOOP

        FETCH c_secundar INTO v_cod_articol, v_nume_articol, v_cantitate,
v_cursor;
        EXIT WHEN c_secundar%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('cod: ' || v_cod_articol || ',
nume: ' || v_nume_articol || ', cantitate: ' || v_cantitate || ', produse componente');

    LOOP
        FETCH v_cursor INTO v_cod_produs, v_nume_produs;
        EXIT WHEN v_cursor%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('' ||
v_nume_produs || ', cod:' || v_cod_produs);

    END LOOP;

END LOOP;

CLOSE c_secundar;

END LOOP;

END exercitiul7;
/

```

BEGIN

```

exercitiul7();
END;
/

```

--pentru a ajuta angajatii sa le impacheteze  
--sa se afiseze si curierul care il va duce, daca este cunoscut

```

CREATE OR REPLACE PROCEDURE exercitiul7 (v_numere_sediul IN sediu.numere%TYPE DEFAULT 'CABLE POWER') IS
    TYPE refcursor IS REF CURSOR;
    v_cursor refcursor;

    CURSOR c_principal IS --ciclu cursor care selecteaza pentru un colet codul si curierul (daca exista); ia doar coletele care respecta cerinta
        SELECT c.cod_colet cod_colet, c.cod_comanda cod_comanda, NVL(a.cod_angajat, -1) AS cod_curier,
        NVL(CONCAT(a.nume, ' ' || a.prenume), 'Curierul nu a fost inca asignat') AS nume_curier
        FROM sediu s, colet c, angajat a
        WHERE c.cod_sediul = s.cod_sediul AND v_numere_sediul = UPPER(s.nume) AND UPPER(c.status_colet) = 'NELIVRAT'
        AND a.cod_angajat = c.cod_angajat(+); --left join pentru cazul in care curierul are cod_angajat NULL

    CURSOR c_secundar (v_cod_colet colet.cod_colet%TYPE, v_cod_comanda colet.cod_comanda%TYPE) IS --cursor clasic parametrizat cu un ref cursor in el
        SELECT a.cod_articol as cod, a.nume as nume, aic.cantitate as cantitate,
        CURSOR(SELECT pip.cod_produs cod_produs, a2.nume nume2 -- expresie cursor care pt un pachet afla ce produse are
            FROM produs_in_pachet pip, articol a2
            WHERE pip.cod_pachet = a.cod_articol AND a2.cod_articol = pip.cod_produs)
        FROM articol_in_colet aic, pachet_promotional p, articol a
        WHERE aic.cod_colet = v_cod_colet and aic.cod_comanda = v_cod_comanda and aic.cod_articol = p.cod_articol
        AND a.cod_articol = p.cod_articol;

    v_detalii_curier varchar2(100);
    v_cod_articol articol.cod_articol%TYPE;
    v_nume_articol articol.nume%TYPE;
    v_nume_produs produs.nume%TYPE;
    v_cantitate_articol_in_colet.cantitate%TYPE;
    v_cod_produs produs.cod_articol%TYPE;

    BEGIN
        FOR i IN c_principal LOOP
            EXIT WHEN c_principal%NOTFOUND;
            v_detalii_curier := '(';

```

Script Output x | Task completed in 0.236 seconds

Procedure EXERCITIUL7 compiled

```

LOOP
    FETCH v_cursor INTO v_cod_produs, v_nume_produs;
    EXIT WHEN v_cursor%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE(' ' || v_nume_produs || ', cod:' || v_cod_produs);

    END LOOP;

    END LOOP;
    CLOSE c_secundar;
END LOOP;

END exercitiul7;
/

DECLARE
BEGIN
    exercitiul7();
END;
/

```

Script Output x | Task completed in 0.09 seconds  
PL/SQL procedure successfully completed.

Dbms Output x | Buffer Size:20000 | Baza x

Coletul (care va fi livrat de curierul Todorescu Simon cod 200) cu codul 101 din comanda 100,are pachetele:  
cod: 115, nume: Pachet techno, cantitate: 1, produse componente  
    Incarcator telefon, cod:102  
    Aspirator robot, cod:105  
Coletul (care va fi livrat de curierul Cutas Mirel cod 220) cu codul 104 din comanda 101,are pachetele:  
cod: 113, nume: Pachet electro, cantitate: 2, produse componente  
    Lanterna, cod:103  
    Frigider, cod:104  
cod: 112, nume: Pachet Pro, cantitate: 2, produse componente  
    Casti wireless, cod:100  
    Bec ultra, cod:101  
    Incarcator telefon, cod:102

**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 exceptii proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.**

**Cerință în limbaj natural:**

Se dă ca parametru pentru funcție codul unui articol.

Să se implementeze 3 operații:

1 - afișăm furnizorul și câte produse mai furnizează acesta

2 - afișăm media numărului de participanți din concursurile din 2023 care au ca premiu produsul. se afișează și câte astfel de concursuri sunt, și profitul total obținut din plata taxei de înscriere pentru participanți.

3 - aflăm dacă articolul are costul din reclame mai mare decât media

Excepții:

Dacă se introduce un cod de articol necunoscut se va arunca excepția 'COD\_ARTICOL\_INVALID'

Dacă operația aleasă nu e disponibilă pentru unul din cele 2 tipuri de articol, se va arunca excepția 'OPERATIE\_INVALIDA\_PT\_TIPUL\_ARTICOLULUI'. Operațiile 1 și 2 nu sunt disponibile pentru pachete.

Dacă utilizatorul introduce o comandă greșită se va arunca excepția 'COMANDA\_INVALIDA'

Dacă articolul e de tip produs dar totuși nu are furnizor, se va arunca excepția 'PRODUS\_NU\_ARE\_FURNIZOR'

Funcția va returna un sir de caractere explicând rezultatul obținut.

Pentru operația 2, dacă nu se găsesc concursuri se va arunca 'NU\_EXISTA\_CONCURSURI'.

```
CREATE OR REPLACE FUNCTION exercitiul8 (v_cod_articol IN  
articol.cod_articol%TYPE DEFAULT 100, v_operatie IN number)
```

```
RETURN varchar2
```

```
IS
```

```
    COD_ARTICOL_INVALID exception;  
    COMANDA_INVALIDA exception;  
    OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI exception;  
    PRODUS_NU_ARE_FURNIZOR exception;  
    NU_EXISTA_CONCURSURI exception;
```

```
    v_concluzie varchar2(100);  
    v_temporar_cod_articol articol.cod_articol%TYPE;  
    v_tip_articol varchar2(15);  
    v_nume_furnizor furnizor.nume%TYPE;
```

```
v_cod_furnizor furnizor.cod_furnizor%TYPE;  
v_nr_produse_furnizate number(6);  
v_nr_concursuri number(5);  
v_nr_participanti number(6, 2);  
v_profit number(8);  
v_cost_articol number(8);  
v_medie_reclame number(8);
```

```
BEGIN
```

```
v_concluzie := 'Nu a fost efectuata operatia';  
BEGIN --bloc pentru a verifica daca exista codul introdus  
    SELECT cod_articol INTO v_temporar_cod_articol FROM articol WHERE  
cod_articol = v_cod_articol;
```

```
EXCEPTION
```

```
WHEN NO_DATA_FOUND THEN  
    RAISE COD_ARTICOL_INVALID;  
END;
```

```
--aflam tipul articolului: produs sau pachet promotional
```

```
v_tip_articol := 'produs';  
BEGIN  
    SELECT cod_articol INTO v_temporar_cod_articol FROM produs WHERE  
cod_articol = v_cod_articol;
```

```
EXCEPTION
```

```
WHEN NO_DATA_FOUND THEN  
    v_tip_articol := 'pachet';  
END;
```

```

IF v_operatie = 1 THEN
    --verificam daca articolul este produs, altfel nu are furnizor.
    IF v_tip_articol = 'produs' THEN

        SELECT cod_furnizor INTO v_cod_furnizor
        FROM produs WHERE cod_articol = v_cod_articol;

        IF v_cod_furnizor IS NULL
        THEN RAISE PRODUS_NU_ARE_FURNIZOR;
        ELSE
            SELECT f.nume, (COUNT(*) - 1)
            INTO v_nume_furnizor, v_nr_produse_furnizate --COUNT - 1 pt ca un
            produs furnizat e parametrul functiei
            FROM furnizor f, produs p
            WHERE f.cod_furnizor = v_cod_furnizor AND p.cod_furnizor =
            f.cod_furnizor
            GROUP BY f.nume;

            RETURN 'Produsul e furnizat de ' || v_nume_furnizor || ' care are codul ' ||

v_cod_furnizor || ' si mai furnizeaza alte ' || v_nr_produse_furnizate || ' produse';

        END IF;
        RETURN v_concluzie;
    ELSE
        RAISE OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI;
    END IF;

ELSIF v_operatie = 2 THEN
    IF v_tip_articol = 'produs' THEN

        SELECT COUNT(*) nr_concursuri, AVG(nr_participari), SUM(profit_concurs)
        INTO v_nr_concursuri, v_nr_participanti, v_profit

```

FROM( --luam detalii despre toate concursurile care au printre premii produsul: cod, nr participanti, profit

```

    SELECT c.cod_concurs cod_concurs, COUNT(DISTINCT
    pp.cod_beneficiar) nr_participari,
    (SELECT COUNT(*) * c.taxa_inscriere --doar beneficiarii fara
    cont_premium platesc taxa, deci numaram cati sunt
    FROM participa pp2, beneficiar b
    WHERE pp2.cod_concurs = c.cod_concurs AND b.cod_beneficiar =
    pp2.cod_beneficiar
    AND b.cont_premium = 0) AS profit_concurs
    FROM premiu p, concurs c, participa pp
    WHERE p.cod_articol = v_cod_articol AND p.cod_concurs =
    c.cod_concurs AND pp.cod_concurs = c.cod_concurs
    AND TO_CHAR(c.data_concurs, 'YYYY') = 2023
    GROUP BY c.cod_concurs, c.taxa_inscriere
  );
  
```

IF v\_nr\_concursuri = 0 THEN RAISE NU\_EXISTA\_CONCURSURI;

ELSE RETURN 'Articolul de tip ' || v\_tip\_articol || ' apare ca premiu in ' ||
 v\_nr\_concursuri || ' concursuri din 2023, la care au participant in medie ' ||
 v\_nr\_participanti || ' persoane.

Profitul total al concursurilor este ' || v\_profit || ' RON.';

END IF;

ELSE
 RAISE OPERATIE\_INVALIDA\_PT\_TIPUL\_ARTICOLULUI;
 END IF;

ELSIF v\_operatie = 3 THEN
 --aflam daca articolul are costul din reclame mai mare decat media
 SELECT AVG(cost\_per\_reclama)
 INTO v\_medie\_reclame FROM(

```
        SELECT a.cod_articol, SUM(NVL(r.cost_total, 0)) cost_per_reclama --costul  
reclamelor per articol
```

```
            FROM articol a, reclama r  
            WHERE a.cod_articol = r.cod_articol(+)  
            GROUP BY a.cod_articol);
```

```
        DBMS_OUTPUT.PUT_LINE('Media costului reclamelor per articol este ' ||  
v_medie_reclame);
```

```
--calculam costul din reclame pt articol:  
SELECT NVL(SUM(r.cost_total), 0)  
INTO v_cost_articol  
FROM articol a, reclama r  
WHERE a.cod_articol = r.cod_articol;
```

```
        DBMS_OUTPUT.PUT_LINE('Costul reclamelor pentru articolul ' || v_cod_articol ||  
' este ' || v_cost_articol);
```

```
        IF v_cost_articol > v_medie_reclame  
        THEN RETURN 'Articolul are costul din reclame mai mare decat media';  
        ELSE RETURN 'Articolul NU are costul din reclame mai mare decat media';  
        END IF;
```

```
    ELSE  
        RAISE COMANDA_INVALIDA;  
    END IF;  
    RETURN v_concluzie;
```

```
EXCEPTION
```

```
    WHEN COD_ARTICOL_INVALID THEN  
        RETURN 'Exceptie: Codul introdus nu se alfa in baza de date';  
    WHEN COMANDA_INVALIDA THEN  
        RETURN 'Exceptie: Comanda aleasa nu este valida';
```

```

WHEN OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI THEN
    RETURN 'Exceptie: Operatie invalida pentru tipul articolului';
WHEN PRODUS_NU_ARE_FURNIZOR THEN
    RETURN 'Exceptie: Produsl nu are furnizor';
WHEN NU_EXISTA_CONCURSURI THEN
    RETURN 'Exceptie: Nu s au gasit concursuri';
WHEN OTHERS THEN
    RETURN 'Exceptie necunoscuta';

END exercitiul8;
/

BEGIN
--EXCEPTII:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 8));
--comanda_invalida

    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(500, 3));
--cod_articol_invalid

    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(114, 1)); --operatie
invalida pt tipul articolului (un pachet nu are furnizor)

    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 1)); --produs_nu
are_furnizor

    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(106, 2));
--nu_exista_concursuri

END;
/

--produsul 100 nu are furnizor:
select * from produs where cod_articol = 100;
--codul 500 nu exista pt articole:
select cod_articol from articol;
--produsul 106 nu apare printre premiile vreunui concurs:

```

```
select * from concurs c, premiu p, produs pp  
where c.cod_concurs = p.cod_concurs and p.cod_articol = pp.cod_articol and  
pp.cod_articol = 106;  
  
BEGIN  
    --RULARE CORECTA PT CAZUL 1:  
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(110, 1));  
END;  
/  
  
BEGIN  
    --RULARE CORECTA PT CAZUL 2:  
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 2));  
END;  
/  
BEGIN  
    --RULARE CORECTA PT CAZUL 3:  
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(114, 3));  
END;  
/
```

SQL Worksheet | History | 0.076 seconds | Baza

```

CREATE OR REPLACE FUNCTION exercitiul8 (v_cod_articol IN articol.cod_articol%TYPE DEFAULT 100, v_operatie IN number)
RETURN varchar2
IS
    COD_ARTICOL_INVALID exception;
    COMANDA_INVALIDA exception;
    OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI exception;
    PRODUS_NU_ARE_FURNIZOR exception;
    NU_EXISTA_CONCURSURI exception;
    v_concluzie varchar2(100);
    v_temporar_cod_articol articol.cod_articol%TYPE;
    v_tip_articol varchar2(15);
    v_nume_furnizor furnizor.nume%TYPE;
    v_cod_furnizor furnizor.cod_furnizor%TYPE;
    v_nr_producuse_furnizate number(6);
    v_nr_concursuri number(5);
    v_nr_participanti number(6, 2);
    v_profit number(8);
    v_cost_articol number(8);
    v_mediareclame number(8);
BEGIN
    v_concluzie := 'Nu a fost efectuata operatia';
    BEGIN --Bloc pentru a verifica daca exista codul introdus
        SELECT cod_articol INTO v_temporar_cod_articol FROM articol WHERE cod_articol = v_cod_articol;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            RAISE COD_ARTICOL_INVALID;
    END;
    --aflam tipul articolului: produs sau pachet promotional

```

Script Output | Query Result | Task completed in 0.076 seconds

Function EXERCITIUL8 compiled

SQL Worksheet | History | 0.063 seconds | Baza

```

    RETURN 'Exceptie: Codul introdus nu se alfa in baza de date';
WHEN COMANDA_INVALIDA THEN
    RETURN 'Exceptie: Comanda aleasa nu este valida';
WHEN OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI THEN
    RETURN 'Exceptie: Operatie invalida pentru tipul articolului';
WHEN PRODUS_NU_ARE_FURNIZOR THEN
    RETURN 'Exceptie: Produsul nu are furnizor';
WHEN NU_EXISTA_CONCURSURI THEN
    RETURN 'Exceptie: Nu s au gasit concursuri';
WHEN OTHERS THEN
    RETURN 'Exceptie necunoscuta';
END exercitiul8;
/

BEGIN
--EXCEPTII:
- DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 8)); --comanda incalida
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(500, 3)); --cod articol invalid
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(114, 1)); --operatie invalida pt tipul articolului (un pachet)
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 1)); --produs_nu are furnizor
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(106, 2)); --nu_exista_concursuri
END;
/

```

PL/SQL procedure successfully completed.

Compiler - Log | Messages | Statements | Compiler | Logging Page | Obns Output | Buffer Size: 20000 | Baza

Rezultatul functiei: Exceptie: Comanda aleasa nu este valida  
Rezultatul functiei: Exceptie: Codul introdus nu se alfa in baza de date  
Rezultatul functiei: Exceptie: Operatie invalida pentru tipul articolului  
Rezultatul functiei: Exceptie: Produsul nu are furnizor  
Rezultatul functiei: Exceptie: Nu s au gasit concursuri

```

--produsul 100 nu are furnizor:
select * from produs where cod_articol = 100;

```

```

BEGIN
    --RULARE CORECTA PT CAZUL 1:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(110,
END;
/

```

```

BEGIN
    --RULARE CORECTA PT CAZUL 1:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(110, 1));
END;

```

COD_ARTICOL	COD_FURNIZOR	LUNI_GARANTIE
1	100	(null)
		12

```

select cod_articol from articol;
--produsul 106 nu apare printre premiile vreunui concurs:
select * from concurs c, premiu p, produs pp
where c.cod_concurs = p.cod_concurs and p.cod_articol = pp.cod_articol and pp.cod_articol = 106;

```

```

BEGIN
    --RULARE CORECTA PT CAZUL 1:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(110, 1));
END;

```

COD_CO...	DATA_CO...	SUBIECT	DURATA	NR_INTR...	TAXA_INS...	COD_PRE...	COD_CO...	COD_ART...	PUNCTAJ...	SUMA	COD_ART...

```

select * from produs where cod_articol = 100;
--codul 500 nu exista pt articole:
select cod_articol from articol;

```

```

BEGIN
    --RULARE CORECTA PT CAZUL 1:

```

COD_ARTICOL	
1	100
2	101
3	102
4	103
5	104
6	105
7	106
8	107
9	108
10	109
11	110
12	111
13	112
14	113
15	114
16	115
17	116

SQL Worksheet History

Worksheet Query Builder

```

    BEGIN
    --EXCEPTII:
        DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 8)); --comanda invalida
        DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(500, 3)); --cod_articol_invalid
        DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(114, 1)); --operatie invalida pt tipul articolului (un pacchet nu are furnizor)
        DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 1)); --produs_nu_are_furnizor
        DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(106, 2)); --nu_exista_concursuri
    END;
    /
BEGIN
    --RULARE CORECTA PT CAZUL 1:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(110, 1));
END;
/
BEGIN
    --RULARE CORECTA PT CAZUL 2:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 2));
END;

```

Script Output x | Task completed in 0.534 seconds

PL/SQL procedure successfully completed.

Dbms Output | Buffer Size: 20000

Baza x

Rezultatul functiei: Produsul e furnizat de Best Techno care are codul 150 si mai furnizeaza alte 2 produse

```

    /
BEGIN
    --RULARE CORECTA PT CAZUL 2:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(100, 2));
END;
/
BEGIN
    --RULARE CORECTA PT CAZUL 3:
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul8(114, 3));
END;
/

```

Script Output x | Task completed in 0.188 seconds

PL/SQL procedure successfully completed.

Dbms Output | Buffer Size: 20000

Baza x

Rezultatul functiei: Articoul de tip produs apare ca premiu in 1 concurs(uri) din 2023, la care au participant in medie 4 persoane.  
Profitul total al concursurilor este 60 RON.

SQL Worksheet | History

Worksheet | Query Builder

```

WHEN OTHERS THEN
    RETURN 'Exceptie necunoscuta';
END exercitiu8;
/

BEGIN
--EXCEPTII:
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(100, 8)); --comanda invalida
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(500, 3)); --cod_articol_invalid
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(114, 1)); --operatie invalida pt tipul articolului
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(100, 1)); --produs_nu_are_furnizor
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(106, 2)); --nu_exista_concursuri
END;
/
BEGIN
--RULARE CORECTA PT CAZUL 1:
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(110, 1));
END;
/
BEGIN
--RULARE CORECTA PT CAZUL 2:
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(100, 2));
END;
/
BEGIN
--RULARE CORECTA PT CAZUL 3:
DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiu8(114, 3));
END;
/

```

Script Output x

Task completed in 0.218 seconds

PL/SQL procedure successfully completed.

Dbms Output

Baza x

```

Media costului reclamelor per articol este 2551
Costul reclamelor pentru articolul 114 este 43360
Rezultatul functiei: Articolul are costul din reclame mai mare decat media

```

**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 exceptiile care pot apărea, inclusiv exceptiile NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.**

**Cerință în limbaj natural:**

Pentru numele și prenumele unui beneficiar să se afișeze toate produsele cumpărate, cantitatea, și să scrie la fiecare dacă se mai aplică garanția.

Afișați de asemenea separat pentru comenzi și vânzări prețul total plătit de beneficiar din comenzi/vânzări când a făcut tranzacțiile și cel pe care l-ar fi plătit acum (prețurile articolelor se mai pot schimba). Afișați și punctele de fidelitate pe care le ar fi avut acum dacă nu ar fi folosit din ele (aici se calculează împreună pt toate produsele cumpărate).

Dacă beneficiarul nu a cumpărat nimic fizic sau online să se arunce exceptia NU\_EXISTA\_TRANZACTII (folosesc NO\_DATA\_FOUND într-un bloc interior de unde arunc NU\_EXISTA\_TRANZACTII)

Dacă sunt 2 persoane cu același nume să se arunce exceptia PERSOANA CU ACELASI NUME (folosesc TOO MANY ROWS într-un bloc interior).

```
CREATE OR REPLACE PROCEDURE exercitiul9(v_nume beneficiar.nume%TYPE,  
v_prenume beneficiar.prenume%TYPE)
```

```
IS
```

```
    PERSOANA_CU_ACELASI_NUME exception;  
  
    NU_EXISTA_TRANZACTII exception;  
  
    v_cod_beneficiar beneficiar.cod_beneficiar%TYPE;  
  
    v_temporar_cod_beneficiar beneficiar.cod_beneficiar%TYPE;  
  
    TYPE date_articole_record is RECORD  
        (tip_tranzactie varchar2(30),  
         tip_articol varchar2(30),  
         puncte_fidelitate articol.puncte_fidelitate%TYPE,  
         cod_articol articol.cod_articol%TYPE,  
         nume articol.nume%TYPE,
```

```

        pret_la_cumparare articol.pret%TYPE,
        data_cumparare date,
        cantitate number(8));

TYPE tablou_imbricat_articole IS TABLE OF date_articole_record;
t_articole tablou_imbricat_articole := tablou_imbricat_articole();
t_articole_temporar tablou_imbricat_articole := tablou_imbricat_articole(); --vom
retine aici datele despre toate produsele cumparate (din vanzari si comenzi)

v_temporar_nr number(6);
v_suma_totala_comenzi number(11) := 0; -- suma de cost total al tranzactiilor
legate de comenzi
v_suma_totala_vanzari number(11) := 0; -- suma de cost total al tranzactiilor
legate de vanzari
v_suma_totala_comenzi_atunci number(11) := 0;
v_suma_totala_vanzari_atunci number(11) := 0;
v_detalii_garantie varchar2(150);
v_luni_garantie produs.luni_garantie%TYPE;
v_data_expirare_garantie date;
v_nr_total_puncte_fidelitate number(11) := 0; --adunam pt fiecare articol
cumparat punctele de fidelitate
v_cod_temporar int;

BEGIN
    --verificam daca mai exista alta persoana cu acelasi nume si prenume
    BEGIN
        SELECT codBeneficiar INTO v_codBeneficiar FROM beneficiar WHERE
prenume = v_prenume AND nume = v_nume;
    EXCEPTION
        WHEN TOO_MANY_ROWS
        THEN RAISE PERSOANA_CU_ACELASI_NUME;
    END;

```

```

--verificam daca persoana a facut cumparaturi
BEGIN
    SELECT codBeneficiar INTO v_temporar_nr FROM detalii_tranzactie
    WHERE v_codBeneficiar = codBeneficiar;
EXCEPTION
    WHEN NO_DATA_FOUND
        THEN RAISE NU_EXISTA_TRANZACTII;
    WHEN TOO_MANY_ROWS THEN NULL; --se va arunca too many rows
daca avem tranzactii. oprim exceptia.
END;

```

```

--calculam datele pentru comenzi:
SELECT 'comanda' tip_tranzactie,
CASE WHEN pp.cod_articol IS NULL THEN 'produs'
    ELSE 'pachet'
END AS tip_articol,
a.puncte_fidelitate puncte_fidelitate, a.cod_articol cod_articol, a.nume nume,
a.pret pret_la_cumparare, cc.data_livrare data_cumparare, aic.cantitate
cantitate
BULK COLLECT INTO t_articole
FROM detalii_tranzactie dt, comanda c, colet cc, articol_in_colet aic, articol a,
pachet_promotional pp
WHERE dt.codBeneficiar = v_codBeneficiar AND dt.cod_detalii_tranzactie =
c.cod_detalii_tranzactie
AND c.cod_comanda = cc.cod_comanda
AND aic.cod_colet = cc.cod_colet AND aic.cod_comanda = cc.cod_comanda
AND aic.cod_articol = a.cod_articol
AND a.cod_articol = pp.cod_articol(+);

```

--datele pentru vanzari fizice:

```
SELECT 'vanzare fizica' tip_tranzactie,
```

```

        CASE WHEN a.cod_articol IN (SELECT cod_articol FROM produs) THEN
'produs'
        ELSE 'pachet'
        END AS tip_articol,
        a.puncte_fidelitate puncte_fidelitate, a.cod_articol cod_articol, a.numere nume,
ais.pret pret_la_cumparare,
dt.data_tranzactie data_cumparare, aiv.cantitate cantitate
        BULK COLLECT INTO t_articole_temporar
        FROM detalii_tranzactie dt, vanzare_fizica vf, articol_in_vanzare aiv, articol a,
articol_in_sediu ais
        WHERE dt.codBeneficiar = v_codBeneficiar AND dt.cod_detalii_tranzactie =
vf.cod_detalii_tranzactie
        AND vf.cod_vanzare_fizica = aiv.cod_vanzare_fizica
        AND aiv.cod_articol = a.cod_articol AND vf.cod_sediu = ais.cod_sediu AND
ais.cod_articol = a.cod_articol;

```

t\_articole := t\_articole MULTISET UNION t\_articole\_temporar; --adaugam in t\_articole si datele vanzarilor fizice. nu puteam direct pentru ca bulk collect ar fi sters comenzile.

```

DBMS_OUTPUT.PUT_LINE('COMENZI:');
FOR i IN t_articole.FIRST..t_articole.LAST LOOP
    IF i > t_articole.FIRST AND t_articole(i - 1).tip_tranzactie = 'comanda' AND
t_articole(i).tip_tranzactie = 'vanzare fizica' --cand am gasit primul articol dintr-o vanzare fizica
        THEN DBMS_OUTPUT.PUT_LINE('VANZARI:');
        END IF;
    IF t_articole(i).tip_tranzactie = 'comanda'
        THEN      v_suma_totala_comenzi      :=      v_suma_totala_comenzi      +
(t_articole(i).pret_la_cumparare * t_articole(i).cantitate); --calculam cele 2 sume totale pe care le ar plati cu preturile de acum

```

```
    ELSE      v_suma_totala_vanzari      :=      v_suma_totala_vanzari      +
(t_articole(i).pret_la_cumparare * t_articole(i).cantitate);
END IF;
```

```
        v_nr_total_puncte_fidelitate      :=      v_nr_total_puncte_fidelitate      +
(t_articole(i).puncte_fidelitate * t_articole(i).cantitate);
```

```
IF t_articole(i).tip_articol = 'pachet' THEN --daca articolul curent e pachet
trebuie sa aflam toate produsele din el
```

```
    DBMS_OUTPUT.PUT_LINE('Din pachetul: ' || t_articole(i).cod_articol || ':');
');
```

```
    FOR j IN
        (SELECT p.luni_garantie, a.nume, a.cod_articol
        FROM produs p, produs_in_pachet pip, articol a
        WHERE t_articole(i).cod_articol = pip.cod_pachet AND pip.cod_produc =
p.cod_articol AND a.cod_articol = p.cod_articol)
    LOOP
```

```
    v_detalii_garantie := ";
```

```
IF t_articole(i).data_cumparare IS NULL --daca nu avem data
cumpararii inseamna ca e dintr o vanzare fizica unde clientul nu a platit prin contul
aplicatiei
```

```
    THEN v_detalii_garantie := 'produs nelivrat inca, garantia de ' ||
j.luni_garantie || ' luni inca nu se aplica ';
```

```
ELSE
```

```
    v_data_expirare_garantie          :=          :
ADD_MONTHS(t_articole(i).data_cumparare, j.luni_garantie); --calculam cand expira
garantia
```

```
IF v_data_expirare_garantie < sysdate
```

```
THEN v_detalii_garantie := 'garantie expirata';
```

```
ELSE  v_detalii_garantie  :=  'garantia  expira  pe  '  ||
TO_CHAR(v_data_expirare_garantie, 'dd-mm-yyyy');
```

```

        END IF;

        END IF;

        DBMS_OUTPUT.PUT_LINE('cod: ' || j.cod_articol || ' nume:
' || j.nume || ', cantitate ' || t_articole(i).cantitate || ', detalii garantie: ' || v_detalii_garantie);

    END LOOP;

    ELSE --daca e de tip produs il afisam direct
        SELECT luni_garantie INTO v_luni_garantie FROM produs WHERE
cod_articol = t_articole(i).cod_articol;
        v_detalii_garantie := "";

        IF t_articole(i).data_cumparare IS NULL
            THEN v_detalii_garantie := 'produs nelivrat inca, garantia de ' ||
v_luni_garantie || ' luni inca nu se aplica';
        ELSE
            v_data_expirare_garantie := ADD_MONTHS(t_articole(i).data_cumparare, v_luni_garantie);

            IF v_data_expirare_garantie < sysdate
                THEN v_detalii_garantie := 'garantie expirata';
                ELSE v_detalii_garantie := 'garantia expira pe ' ||
TO_CHAR(v_data_expirare_garantie, 'dd-mm-yyyy');
            END IF;

        END IF;

        DBMS_OUTPUT.PUT_LINE('cod: ' || t_articole(i).cod_articol || ', nume: ' ||
t_articole(i).nume || ', cantitate ' || t_articole(i).cantitate || ', detalii garantie: ' || v_detalii_garantie);

    END IF;

```

```
END LOOP;
```

```
SELECT dt.cod_beneficiar, SUM(dt.cost_total)  
INTO v_cod_temporar, v_suma_totala_comenzi_atunci --aflam cat a platit pe  
toate comenzile
```

```
FROM comanda c, detalii_tranzactie dt  
WHERE c.cod_detalii_tranzactie = dt.cod_detalii_tranzactie and  
dt.cod_beneficiar = v_cod_beneficiar  
GROUP BY dt.cod_beneficiar;
```

```
SELECT dt.cod_beneficiar, SUM(dt.cost_total) INTO v_cod_temporar,  
v_suma_totala_vanzari_atunci --aflam cat a platit pe toate vanzarile  
FROM vanzare_fizica vf, detalii_tranzactie dt  
WHERE vf.cod_detalii_tranzactie = dt.cod_detalii_tranzactie and  
dt.cod_beneficiar = v_cod_beneficiar  
GROUP BY dt.cod_beneficiar;
```

```
DBMS_OUTPUT.PUT_LINE('Beneficiarul a acumulat ' ||  
v_nr_total_puncte_fidelitate || ' puncte fidelitate');  
DBMS_OUTPUT.PUT_LINE('Cat a platit in total pt comenzi si vanzari:' ||  
v_suma_totala_comenzi_atunci || '' || v_suma_totala_vanzari_atunci);  
DBMS_OUTPUT.PUT_LINE('Cat ar fi platit in total pt comenzi si vanzari cu  
preturile actuale:' || v_suma_totala_comenzi || '' || v_suma_totala_vanzari);
```

```
EXCEPTION
```

```
WHEN PERSOANA CU ACELASI NUME THEN
```

```
DBMS_OUTPUT.PUT_LINE('Mai exista o persoana cu acelasi nume');
```

```
WHEN NU_EXISTA_TRANZACTII THEN
```

```
DBMS_OUTPUT.PUT_LINE('Persoana nu a efectuat nicio tranzactie');
```

```
END exercitiul9;
```

```
/
```

```
BEGIN  
    exercitiu9('Andronic', 'Marcel'); --va merge  
END;  
/
```

```
BEGIN  
    exercitiu9('Lupu', 'Eugen'); --too many rows  
    exercitiu9('Maria', 'Ana'); --no data found  
END;  
/
```

--exista 2 beneficiari cu numele Lupu Eugen:

```
select * from beneficiar where nume = 'Lupu' and prenume = 'Eugen';
```

--beneficiarul Maria Ana nu a cumparat nimic:

```
select * from beneficiar b, detalii_tranzactie dt  
where b.cod_beneficiar = dt.cod_beneficiar and nume = 'Maria' and prenume = 'Ana';
```

SQL Worksheet History

Worksheet Query Builder

```

CREATE OR REPLACE PROCEDURE exercitiul9(v_nume beneficiar.nume%TYPE, v_prenume beneficiar.prenume%TYPE)
IS
    PERSOANA_CU_ACELASI_NUME exception;
    NU_EXISTA_TRANZACTII exception;
    v_cod_beneficiar beneficiar.cod_beneficiar%TYPE;
    v_temporar_cod_beneficiar beneficiar.cod_beneficiar%TYPE;
    TYPE date_articole_record is RECORD (tip_tranzactie varchar2(30), tip_articol varchar2(30), puncte_fidelitate articol.cod_articol.articol.cod_articol%TYPE, nume_articol.nume%TYPE, pret_la_cumparare articol.pret%TYPE, data_cumparare DATE);
    TYPE tablou_imbricat_articole IS TABLE OF date_articole_record;
    t_articole tablou_imbricat_articole := tablou_imbricat_articole();
    t_articole_temporar tablou_imbricat_articole := tablou_imbricat_articole(); -- vom retine aici datele despre toate produsele
    v_temporar_nr number(6);
    v_suma_totala_comenzi number(11) := 0; -- suma de cost total al tranzactiilor legate de comenzi
    v_suma_totala_vanzari number(11) := 0; -- suma de cost total al tranzactiilor legate de vanzari
    v_suma_totala_comenzi_atunci number(11) := 0;
    v_suma_totala_vanzari_atunci number(11) := 0;
    v_detalii_garantie varchar2(150);
    v_luni_garantie produs.luni_garantie%TYPE;
    v_data_expirare_garantie date;
    v_nr_total_puncte_fidelitate number(11) := 0; -- adunam pt fiecare articol cumparat punctele de fidelitate
    v_cod_temporar int;
BEGIN
    -- verificam daca mai exista alta persoana cu acelasi nume si prenume
    BEGIN
        SELECT cod_beneficiar INTO v_cod_beneficiar FROM beneficiar WHERE prenume = v_prenume AND nume = v_nume;
    EXCEPTION
        WHEN TOO_MANY_ROWS THEN RAISE PERSOANA_CU_ACELASI_NUME;
    END;

    -- verificam daca persoana a facut cumparaturi
    BEGIN
        SELECT cod_beneficiar INTO v_temporar_nr FROM detalii_tranzactie WHERE v_cod_beneficiar = cod_beneficiar;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN RAISE NU_EXISTA_TRANZACTII;
    END;

```

Script Output

Procedure EXERCITIUL9 compiled

Compiler - Log

Messages Logging Page Statements Compiler

Dbms Output

```

grupa231* | lab3.sql | lab2.sql | Welcome Page | creare_inserare.sql | exercitiu100.sql | exercitiu11.sql | exercitiu13.sql | exercitiu6.sql | exercitiu8.sql | exercitiu9.sql >
SQL Worksheet History
Worksheet Query Builder
SELECT dt.cod_beneficiar, SUM(dt.cost_total) INTO v_cod_temporar, v_suma_totala_comenzi_atunci --aflam cat a platit pe toate comenziile
FROM comanda c, detalii_tranzactie dt
WHERE c.cod_detalii_tranzactie = dt.cod_detalii_tranzactie AND dt.cod_beneficiar = v_cod_beneficiar
GROUP BY dt.cod_beneficiar;

SELECT dt.cod_beneficiar, SUM(dt.cost_total) INTO v_cod_temporar, v_suma_totala_vanzari_atunci --aflam cat a platit pe toate vanzarile
FROM vanzare_fizica vf, detalii_tranzactie dt
WHERE vf.cod_detalii_tranzactie = dt.cod_detalii_tranzactie AND dt.cod_beneficiar = v_cod_beneficiar
GROUP BY dt.cod_beneficiar;

DBMS_OUTPUT.PUT_LINE('Beneficiarul a acumulat ' || v_nr_total_puncte_fidelitate || ' puncte fidelitate');
DBMS_OUTPUT.PUT_LINE('Cat a platit in total pt comenzi si vanzari:' || v_suma_totala_comenzi_atunci || ' || v_suma_totala_vanzari_atunci);
DBMS_OUTPUT.PUT_LINE('Cat ar fi platit in total pt comenzi si vanzari cu preturile actuale:' || v_suma_totala_comenzi || ' || v_suma_totala_vanzari);

EXCEPTION
  WHEN PERSONA_CU_ACELASI_NUME THEN
    DBMS_OUTPUT.PUT_LINE('Mai exista o persoana cu acelasi nume');
  WHEN NU_EXISTA_TRANZACTII THEN
    DBMS_OUTPUT.PUT_LINE('Persoana nu a efectuat nicio tranzactie');
END exercitiu9;
/

BEGIN
  exercitiu9('Andronic', 'Marcel'); --va merge
END;
/
BEGIN
  exercitiu9('Lupu', 'Eugen'); --too many rows
  exercitiu9('Maria', 'Ana'); --no data found
END;

```

Script Output x Task completed in 0.172 seconds  
PL/SQL procedure successfully completed.

Dbms Output + Buffer Size: 20000 |  
Baza x  
Mai exista o persoana cu acelasi nume  
Persoana nu a efectuat nicio tranzactie

```
--exista 2 beneficiari cu numele Lupu Eugen:
select * from beneficiar where nume = 'Lupu' and prenume = 'Eugen';

--beneficiarul Maria Ana nu a cumparat nimic:
select * from beneficiar b, detalii_tranzactie dt
where b.cod_beneficiar = dt.cod_beneficiar and nume = 'Maria' and prenume = 'Ana';

```

Script Output x Query Result x SQL | All Rows Fetched: 2 in 0.166 seconds

COD_BENEFICIAR	NUME	PRENUME	TELEFON	CNP	NUME_UTILIZATOR	CONT_PREMIUM	DATA_AUTENTIFICARE	PUNCTE_FIDELITATE
1	120	Lupu	Eugen	+400748201839	5010410205807	dvd34_a	1 30-MAY-21	800
2	130	Lupu	Eugen	+400745678542	1990915204669	eugen_123	0 12-APR-21	0

```

--exista 2 beneficiari cu numele Lupu Eugen:
select * from beneficiar where nume = 'Lupu' and prenume = 'Eugen';

--beneficiarul Maria Ana nu a cumparat nimic:
select * from beneficiar b, detalii_tranzactie dt
where b.cod_beneficiar = dt.cod_beneficiar and nume = 'Maria' and prenume = 'Ana';

```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 0 in 0.324 seconds

COD_BEN...	NUME	PRENUME	TELEFON	CNP	NUME_UT...	CONT_PR...	DATA_AU...	PUNCTE_...

```

PL/SQL procedure successfully completed.

Doms Output
+ - Buffer Size:20000 |
Baza x
COMENZI:
cod: 105, nume: Aspirator robot, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica
Din pachetul: 114:
cod: 100 nume: Casti wireless, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 12 luni inca nu se aplica
cod: 105 nume: Aspirator robot, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica
cod: 106 nume: Fier de calcat, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 24 luni inca nu se aplica
cod: 107 nume: Uscator de par, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica
cod: 108 nume: Uscator de par Julie Pro, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica
cod: 109 nume: Telefon, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 15 luni inca nu se aplica
Din pachetul: 115:
cod: 102 nume: Incarcator telefon, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 0 luni inca nu se aplica
cod: 105 nume: Aspirator robot, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica
cod: 108, nume: Uscator de par Julie Pro, cantitate 1, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica
cod: 110, nume: Laptop, cantitate 1, detalii garantie: produs nelivrat inca, garantia de 6 luni inca nu se aplica
cod: 107, nume: Uscator de par, cantitate 3, detalii garantie: garantia expira pe 01-07-2025
cod: 103, nume: Lanterna, cantitate 4, detalii garantie: garantia expira pe 01-09-2025
Din pachetul: 114:
cod: 100 nume: Casti wireless, cantitate 2, detalii garantie: garantia expira pe 01-03-2025
cod: 105 nume: Aspirator robot, cantitate 2, detalii garantie: garantia expira pe 01-06-2025

```

## Afișarea completă:

### COMENZI:

cod: 105, nume: Aspirator robot, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica

Din pachetul: 114:

cod: 100 nume: Casti wireless, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 12 luni inca nu se aplica

cod: 105 nume: Aspirator robot, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica

cod: 106 nume: Fier de calcat, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 24 luni inca nu se aplica

cod: 107 nume: Uscator de par, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica

cod: 108 nume: Uscator de par Julie Pro, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica

cod: 109 nume: Telefon, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 15 luni inca nu se aplica

Din pachetul: 115:

cod: 102 nume: Incarcator telefon, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 0 luni inca nu se aplica

cod: 105 nume: Aspirator robot, cantitate 2, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica

cod: 108, nume: Uscator de par Julie Pro, cantitate 1, detalii garantie: produs nelivrat inca, garantia de 18 luni inca nu se aplica

cod: 110, nume: Laptop, cantitate 1, detalii garantie: produs nelivrat inca, garantia de 6 luni inca nu se aplica

cod: 107, nume: Uscator de par, cantitate 3, detalii garantie: garantia expira pe 01-07-2025

cod: 103, nume: Lanterna, cantitate 4, detalii garantie: garantia expira pe 01-09-2025

Din pachetul: 114:

cod: 100 nume: Casti wireless, cantitate 2, detalii garantie: garantia expira pe 01-03-2025

cod: 105 nume: Aspirator robot, cantitate 2, detalii garantie: garantia expira pe 01-09-2025

cod: 106 nume: Fier de calcat, cantitate 2, detalii garantie: garantia expira pe 01-03-2026

cod: 107 nume: Uscator de par, cantitate 2, detalii garantie: garantia expira pe 01-09-2025

cod: 108 nume: Uscator de par Julie Pro, cantitate 2, detalii garantie: garantia expira pe 01-09-2025

cod: 109 nume: Telefon, cantitate 2, detalii garantie: garantia expira pe 01-06-2025

VANZARI:

Din pachetul: 112:

cod: 100 nume: Casti wireless, cantitate 1, detalii garantie: garantie expirata

cod: 101 nume: Bec ultra, cantitate 1, detalii garantie: garantia expira pe  
09-10-2024

cod: 102 nume: Incarcator telefon, cantitate 1, detalii garantie: garantie expirata

Din pachetul: 113:

cod: 103 nume: Lanterna, cantitate 1, detalii garantie: garantia expira pe  
09-10-2024

cod: 104 nume: Frigider, cantitate 1, detalii garantie: garantia expira pe  
09-04-2025

Din pachetul: 116:

cod: 104 nume: Frigider, cantitate 1, detalii garantie: garantia expira pe  
09-04-2025

cod: 108 nume: Uscator de par Julie Pro, cantitate 1, detalii garantie: garantia  
expira pe 09-10-2024

Din pachetul: 114:

cod: 100 nume: Casti wireless, cantitate 2, detalii garantie: garantie expirata

cod: 105 nume: Aspirator robot, cantitate 2, detalii garantie: garantia expira pe  
11-12-2024

cod: 106 nume: Fier de calcat, cantitate 2, detalii garantie: garantia expira pe  
11-06-2025

cod: 107 nume: Uscator de par, cantitate 2, detalii garantie: garantia expira pe  
11-12-2024

cod: 108 nume: Uscator de par Julie Pro, cantitate 2, detalii garantie: garantia  
expira pe 11-12-2024

cod: 109 nume: Telefon, cantitate 2, detalii garantie: garantia expira pe  
11-09-2024

Din pachetul: 116:

cod: 104 nume: Frigider, cantitate 2, detalii garantie: garantia expira pe  
11-06-2025

cod: 108 nume: Uscator de par Julie Pro, cantitate 2, detalii garantie: garantia  
expira pe 11-12-2024

cod: 107, nume: Uscator de par, cantitate 1, detalii garantie: garantia expira pe 13-02-2025

Din pachetul: 113:

cod: 103 nume: Lanterna, cantitate 1, detalii garantie: garantia expira pe 13-02-2025

cod: 104 nume: Frigider, cantitate 1, detalii garantie: garantia expira pe 13-08-2025

Beneficiarul a acumulat 657 puncte fidelitate

Cat a platit in total pt comenzi si vanzari: 1510 740

Cat ar fi platit in total pt comenzi si vanzari cu preturile actuale: 16260 28983

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

Definiți un trigger care permite adăugarea unui angajat nou doar dacă  $S < M$ ;

$S$  = suma salariilor angajaților care au activitate

$M$  = media încasărilor din vânzări fizice din tot anul 2023 a sediilor care respectă ambele condiții:

1. au avut în 2023 mai mult de 5 luni cu profit sub 300
2. au vândut mai puține produse vândute decât media. Produsele din pachete se iau separat.

De asemenea, nu se permite adăugarea de angajați în baza de date între orele 00 și 03.

CREATE OR REPLACE VIEW nr\_produse\_per\_sediu AS --view sa retin care produse a vandut un sediu adunat din toate vanzarile lui

(SELECT s.cod\_sediu as cod\_sediu,

COALESCE(SUM(nr\_produse\_per\_vanzare.nr\_articole\_tip\_produc\_per\_vanzare +  
nr\_produse\_per\_vanzare.nr\_produse\_din\_pachete\_per\_vanzare), 0)

AS nr\_produse\_vandute\_per\_sediu-- cate produse a vandut fiecare sediu

FROM

(SELECT vf.cod\_vanzare\_fizica as cod\_vanzare\_fizica, vf.cod\_sediu as cod\_sediu, -- cate produse are fiecare vanzare si sediul unde a avut loc vanzarea

```

        COUNT(CASE WHEN pp.cod_articol IS NULL THEN aiv.cod_articol ELSE
NULL END) nr_articole_tip_produs_per_vanzare,
        SUM(CASE WHEN pp.cod_articol IS NOT NULL
        THEN (SELECT COUNT(*) FROM produs_in_pachet pip WHERE
pp.cod_articol = pip.cod_pachet) --daca e pachet trb sa numaram toate produsele din el
        ELSE 0 END) nr_produse_din_pachete_per_vanzare
        FROM vanzare_fizica vf, articol_in_vanzare aiv, articol a, pachet_promotional pp
        WHERE aiv.cod_vanzare_fizica = vf.cod_vanzare_fizica AND aiv.cod_articol =
a.cod_articol
        AND a.cod_articol = pp.cod_articol(+) --daca codul din pp e nenul inseamna ca e
pachet
        GROUP BY vf.cod_vanzare_fizica, vf.cod_sediu) nr_produse_per_vanzare,
        sediu s
        WHERE s.cod_sediu = nr_produse_per_vanzare.cod_sediu(+)
        GROUP BY s.cod_sediu);

```

```
SELECT * FROM nr_produse_per_sediu;
```

```

CREATE OR REPLACE VIEW medie_produse_vandute_per_sediu AS (
    SELECT AVG(nr_produse_per_sediu.nr_produse_vandute_per_sediu) AS
medie_produse_vandute_per_sediu
    FROM nr_produse_per_sediu);

```

```

CREATE OR REPLACE TRIGGER exercitiul10
    BEFORE INSERT ON angajat
DECLARE
    v_ora_inceput number(2) := 0;
    v_ora_sfarsit number(2) := 3;
    v_suma_salarii number(10);

```

```

v_numar_inregistrari number(4);
v_medie_produse number(10);
v_medie_produse_vandute_per_sediu number(10);
v_nr_sedii_indeplinesc_conditia number(10) := 0;
v_suma_profit_sedii number(10) := 0;
v_medie_profit number(10) := 0;
v_produse_per_sediu number(10) := 0;
v_cod_sediu sediu.cod_sediu%TYPE;
v_profit number(10);

```

cursor c\_profit\_anual is

```

    WITH luna AS (
        SELECT ADD_MONTHS(TO_DATE('2023-01', 'YYYY-MM'), LEVEL - 1) AS
luna --cream o tabela cu toate lunile lui 2023

```

--pentru a include si lunile unui sediu cand nu are vanzari, adica profit 0

```

        FROM dual

```

```

        CONNECT BY LEVEL <= 12),

```

profit\_lunar\_per\_sediu AS

```

        (SELECT s.cod_sediu AS cod_sediu, l.luna AS luna_vanzare,
            SUM(NVL(dt.cost_total, 0)) AS profit_lunar

```

```

        FROM sediu s, luna l, vanzare_fizica vf, detalii_tranzactie dt --luam toate
        sediile cu toate lunile si apoi outer join cu vanzarile

```

```

        WHERE s.cod_sediu= vf.cod_sediu(+) AND vf.cod_detalii_tranzactie =
dt.cod_detalii_tranzactie(+)

```

```

        AND TO_CHAR(dt.data_tranzactie(+), 'YYYY-MM') =
TO_CHAR(l.luna, 'YYYY-MM')

```

```

        GROUP BY s.cod_sediu, l.luna),

```

300 sedii\_cu\_profit\_sub\_300 AS ( --sedii care au mai mult de 5 luni cu profit sub

```

        SELECT cod_sediu
        FROM profit_lunar_per_sediu
        WHERE profit_lunar < 300
        GROUP BY cod_sediu
        HAVING COUNT(*) > 5
    )

--calculam profitul din 2023 pt sediile care au mai mult de 5 luni cu profit sub
300:

        SELECT SUM(pl.profit_lunar) AS profit_anual, pl.cod_sediu
        FROM profit_lunar_per_sediu pl, sedii_cu_profit_sub_300 s
        WHERE pl.cod_sediu = s.cod_sediu
        GROUP BY pl.cod_sediu;

```

cursor c\_nr\_produse\_per\_sediu (cod\_sediu.cod\_sediu%TYPE) is --cursor care pentru un sediu dat arata cate produse a vandut

```

        SELECT n.nr_produse_vandute_per_sediu nr
        FROM nr_produse_per_sediu n
        WHERE n.cod_sediu = cod;

```

```

BEGIN

--mai intai verificam ora

IF TO_CHAR(SYSDATE, 'HH24') BETWEEN v_ora_inceput AND v_ora_sfarsit

    THEN RAISE_APPLICATION_ERROR(-20004, 'Aplicatia este in mentenanta, nu
puteti insera intre orele ' || v_ora_inceput || ' si ' || v_ora_sfarsit);

END IF;

```

--suma salariilor angajatilor care au activitate:

```

        SELECT sum(salariu) INTO v_suma_salarii FROM angajat a
        WHERE EXISTS (SELECT 1 FROM reclama r WHERE r.cod_angajat =
a.cod_angajat)

```

```

        OR EXISTS (SELECT 1 FROM colet c WHERE c.cod_angajat = a.cod_angajat)
        OR EXISTS (SELECT 1 FROM sediu s WHERE s.cod_administrator =
a.cod_angajat);
        DBMS_OUTPUT.PUT_LINE('Suma salariilor angajatilor care au activitate este ' ||
v_suma_salarii);

```

```

--calculam media produselor vandute de toate sediile
SELECT *
INTO v_mediie_produse_vandute_per_sediu
FROM medie_produse_vandute_per_sediu;
DBMS_OUTPUT.PUT_LINE('Media produselor vandute de toate sediile este ' ||
v_mediie_produse_vandute_per_sediu);

--cursor cu care mergem prin profitul sediilor care au mai mult de 5 luni cu profit sub
300
--vom verificca pt fiecare daca a vandut mai putine produse decat media
DBMS_OUTPUT.PUT_LINE('Sediile cu mai mult de 5 luni cu profitul sub 300: ');
OPEN c_profit_anual;
LOOP
    FETCH c_profit_anual INTO v_profit, v_cod_sediu;
    EXIT WHEN c_profit_anual%NOTFOUND;

    OPEN c_nr_produse_per_sediu(v_cod_sediu);
        FETCH c_nr_produse_per_sediu INTO v_produse_per_sediu;
        -- aflam cate produse a vandut sediul
        CLOSE c_nr_produse_per_sediu;

        DBMS_OUTPUT.PUT('    Sediul ' || v_cod_sediu || ' are profit anual ' ||
v_profit || ' si a vandut ' || v_produse_per_sediu || ' produse ');

        IF v_produse_per_sediu < v_mediie_produse_vandute_per_sediu THEN
            v_nr_sediile_indeplinesc_conditia := v_nr_sediile_indeplinesc_conditia + 1;

```

```

        v_suma_profit_sedii := v_suma_profit_sedii + v_profit;
        DBMS_OUTPUT.PUT_LINE(' adica mai putine produse decat media.');
    ELSE DBMS_OUTPUT.PUT_LINE('');
    END IF;

    END LOOP;
CLOSE c_profit_anual;

--calculam media profiturilor:
IF v_nr_sedii_indeplinesc_condiita > 0 THEN
    v_medie_profit := v_suma_profit_sedii / v_nr_sedii_indeplinesc_condiita;
END IF;

DBMS_OUTPUT.PUT_LINE('Media profiturilor sediilor care respecta conditiile e: ' || v_medie_profit);
IF v_suma_salarii > v_medie_profit THEN
    RAISE_APPLICATION_ERROR(-20006, 'Suma salariilor este mai mare decat media profiturilor in sedii cu activitate redusa');
END IF;

END;
/

```

```

insert into angajat(cod_angajat, nume, prenume, salariu, telefon, data_angajare) values
(400, 'Todorescu', 'Simon', 6000, '+400734567845', to_date('12-09-2023', 'dd-mm-yyyy'));
delete from angajat where cod_angajat = 400;
rollback;

drop trigger exercitiul10;

```

```

CREATE OR REPLACE VIEW nr_produse_per_sedi AS --view sa retin care produse a vandut un sediu adunat din toate vanzarile lui
  (SELECT s.cod_sediu as cod_sediu,
         COALESCE(SUM(nr_produse_per_vanzare.nr_articole_tip_produs_per_vanzare + nr_produse_per_vanzare.nr_produse_din_pachet
                     AS nr_produse_vandute_per_sedi-- cate produse a vandut fiecare sediu
    FROM
      (SELECT vf.cod_vanzare_fizica as cod_vanzare_fizica, vf.cod_sediu as cod_sediu, -- cate produse are fiecare vanzare sunt
             COUNT(CASE WHEN pp.cod_articol IS NULL THEN aiv.cod_articol ELSE NULL END) nr_articole_tip_produs_per_vanzare,
             SUM(CASE WHEN pp.cod_articol IS NOT NULL
                      THEN (SELECT COUNT(*) FROM produs_in_pachet pip WHERE pp.cod_articol = pip.cod_pachet) --daca e pachet trb sa se aduna
                           ELSE 0 END) nr_produse_din_pachete_per_vanzare
    FROM vanzare_fizica vf, articol_in_vanzare aiv, articol a, pachet_promotional pp
    WHERE aiv.cod_vanzare_fizica = vf.cod_vanzare_fizica AND aiv.cod_articol = a.cod_articol
    AND a.cod_articol = pp.cod_articol(+) --daca codul din pp e nenul inseamna ca e pachet
    GROUP BY vf.cod_vanzare_fizica, vf.cod_sediu) nr_produse_per_vanzare,
    sediu s
   WHERE s.cod_sediu = nr_produse_per_vanzare.cod_sediu(+)
  GROUP BY s.cod_sediu);

CREATE OR REPLACE VIEW medie_produse_vandute_per_sedi AS (
  SELECT AVG(nr_produse_per_sedi.nr_produse_vandute_per_sedi) AS medie_produse_vandute_per_sedi
);

```

Script Output | Query Result | Task completed in 0.315 seconds

View NR\_PRODUSE\_PER\_SEDIU created.

SELECT \* FROM nr\_produse\_per\_sedi;

Script Output | Query Result | SQL | All Rows Fetched: 5 in 0.113 seconds

COD_SEDIU	NR_PRODUSE_VANDUTE_PER_SEDIU
1	100
2	110
3	120
4	130
5	140

```

        GROUP BY s.cod_sediu);

CREATE OR REPLACE VIEW medie_produse_vandute_per_sedi AS (
    SELECT AVG(nr_produse_per_sediu.nr_produse_vandute_per_sediu) AS medie_produse_vandute_per_sediu
    FROM nr_produse_per_sediu);

```

Script Output x | Task completed in 0.079 seconds

View MEDIE\_PRODUSE\_VANDUTE\_PER\_SEDIU created.

Dbms Output x | Buffer Size: 20000 | conexiune\_noua x

```

create_inserare.sql x exercitul00.sql x exercitul11.sql x exercitul13.sql x exercitul6.sql x exercitul8.sql x exercitul9.sql x
SQL Worksheet History
Worksheet Query Builder
SELECT AVG(nr_produse_per_sediu.nr_produse_vandute_per_sediu) AS medie_produse_vandute_per_sediu
FROM nr_produse_per_sediu;

--acum retinem sediile care au vandut mai putine produse decat media si pt fiecare media incasarilor pe luna din 2023
CREATE OR REPLACE TRIGGER exercitiul10
    BEFORE INSERT ON angajat
DECLARE
    v_ora_inceput number(2) := 0;
    v_ora_sfarsit number(2) := 3;
    v_suma_salarii number(10);
    v_numar_inregistrari number(4);
    v_medie_produse number(10);
    v_medie_produse_vandute_per_sediu number(10);
    v_nr_sediile_indeplinesc_conditia number(10) := 0;
    v_suma_profit_sedii number(10) := 0;
    v_medie_profit number(10) := 0;
    v_produse_per_sediu number(10) := 0;
    v_cod_sediu sediu.cod_sediu%TYPE;
    v_profit number(10);
    cursor c_profit_anual is
        WITH luna AS (
            SELECT ADD_MONTHS(TO_DATE('2023-01', 'YYYY-MM'), LEVEL - 1) AS luna --cream o tabela cu toate lunile lui 2023
            --pentru a include si lunile unui sediu cand nu are vanzari, adica profit 0
            FROM dual
            CONNECT BY LEVEL <= 12),
        profit_lunar_per_sediu AS
            (SELECT s.cod_sediu AS cod_sediu,
                l.luna AS luna_vanzare,
                SUM(NVL(dt.cost_total, 0)) AS profit_lunar
            FROM sediu s, luna l, vanzare_fizica vf, detalii_tranzactie dt --luam toate sediile cu toate lunile si apoi
            WHERE s.cod_sediu= vf.cod_sediu(+) AND vf.cod_detalii_tranzactie = dt.cod_detalii_tranzactie(+)
            AND TO_CHAR(dt.data_tranzactie(+), 'YYYY-MM') = TO_CHAR(l.luna, 'YYYY-MM'))

```

Script Output x | Task completed in 0.155 seconds

Trigger EXERCITIUL10 compiled

The screenshot shows the Oracle SQL Developer interface. The top window is a Worksheet titled "Query Builder" containing PL/SQL code for creating a trigger. The code includes a cursor loop to calculate average profit per branch, a condition to check if the sum of salaries exceeds the average profit, and a raise application error if true. It also includes an insert statement for an employee, a delete statement, a rollback command, and a drop trigger command.

```

v_mediu_profit := v_suma_profit_sedii / v_nr_sedii_lidelipiente_comunita;
END IF;

DBMS_OUTPUT.PUT_LINE('Media profiturilor sediilor care respecta conditiile e: ' || v_mediu_profit);
IF v_suma_salarii > v_mediu_profit THEN
    RAISE_APPLICATION_ERROR(-20006, 'Suma salariilor este mai mare decat media profiturilor in sedii cu activitate redusa');
END IF;
END;
/

insert into angajat(cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (400, 'Todorescu', 'Simon', 6000, '+400734567845');
delete from angajat where cod_angajat = 400;
rollback;

drop trigger exercitiul10;

```

The "Script Output" tab shows the execution results, including the error message:

```

Error starting at line : 142 in command -
insert into angajat(cod_angajat, nume, prenume, salariu, telefon, data_angajare) values (400, 'Todorescu', 'Simon', 6000, '+400734567845'
Error report -
ORA-20006: Suma salariilor este mai mare decat media profiturilor in sedii cu activitate redusa
ORA-06512: at "EU.EXERCITIUL10", line 99
ORA-04088: error during execution of trigger 'EU.EXERCITIUL10'

```

The bottom window is the DBMS Output tab, showing the output of the trigger's print statements:

```

Suma salariilor angajatilor care au activitate este 66150
Media produselor vandute de toate sediile este 11
Sediile cu mai mult de 5 luni cu profitul sub 300:
    Sediul 130 are profit anual 1690 si a vandut 31 produse
    Sediul 100 are profit anual 1730 si a vandut 6 produse , adica mai putine produse decat media.
    Sediul 110 are profit anual 530 si a vandut 12 produse
    Sediul 140 are profit anual 0 si a vandut 0 produse , adica mai putine produse decat media.
    Sediul 120 are profit anual 1020 si a vandut 8 produse , adica mai putine produse decat media.
Media profiturilor sediilor care respecta conditiile e: 917

```

Declanșatorul nu permite inserarea angajatului pentru că suma salariilor angajaților care au activitate e mai mare decât media profiturilor anuale ale sediilor care respectă cele 2 condiții. ( $72650 > 917$ ).

Pentru a testa faptul că triggerul nu permite inserarea între anumite ore, vom schimba intervalul nepermis din  $(0, 3)$  în  $(0, 16)$ , pentru a cuprinde ora curentă.

```

CREATE OR REPLACE TRIGGER exercitiu110
  BEFORE INSERT ON angajat
DECLARE
  v_ora_inceput number(2) := 0;
  v_ora_sfarsit number(2) := 16;

  DBMS_OUTPUT.PUT_LINE('Media profiturilor sediilor care respecta conditiile e: ' || v_medie_profit);
  IF v_suma_salarii > v_medie_profit THEN
    RAISE_APPLICATION_ERROR(-20006, 'Suma salariilor este mai mare decat media profiturilor in sedii cu activitate redusa');
  END IF;
END;
/

```

insert into angajat(cod\_angajat, nume, prenume, salariu, telefon, data\_angajare) values (400, 'Todorescu', 'Simon', 6000, '+40073456789');
delete from angajat where cod\_angajat = 400;

Script Output | Query Result | Task completed in 0.092 seconds

Error starting at line : 135 in command -  
 insert into angajat(cod\_angajat, nume, prenume, salariu, telefon, data\_angajare) values (400, 'Todorescu', 'Simon', 6000, '+40073456789'  
 Error report -  
 ORA-20004: Aplicatia este in mentenanta, nu puteti insera intre orele 0 si 16  
 ORA-06512: at "EU.EXERCITIU110", line 52  
 ORA-04088: error during execution of trigger 'EU.EXERCITIU110'

Compiler - Log | Messages | Logging Page | Statements | Compiler

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

Să se definească 2 triggere:

**Primul** se activează înaintea inserării unei noi linii în tabela COMANDA sau înaintea update-ului unei linii;

În caz de inserare, verifică dacă beneficiarul de care e legată comanda prin tabela detalii\_tranzactie are articole bifate.

Dacă nu are, nu avem ce articole să punem în comandă deci trigger-ul va opri inserarea.

Se presupune că înregistrarea corespunzătoare comenzii din detalii\_tranzactie este deja existență.

În caz de update, se verifică dacă noul status este livrată și cel vechi nelivrata.

Pentru ca o comandă să fie livrată trebuie ca toate coletele ei să fie livrate, deci vom verifica dacă există vreun colet nelivrat.

În caz afirmativ, triggerul nu permite update-ul.

**Al doilea** trigger creează după inserarea unei comenzi un colet cu statusul 'temporar' care să păstreze toate articolele comandate până la distribuirea lor în colete finale.

Articolele din comandă sunt cele care se află bifate în coșul beneficiarului.

După salvarea articolelor debifați acele înregistrări din coș

Trebuie să vedem dacă putem păstra un singur colet, adică dacă există vreun sediu în care se găsesc toate produsele lui componente

Să se aleagă sediul cu cât mai puține colete de livrat.

Dacă găsim sediul schimbăm statusul coletului din temporar în 'în procesare' și îi punem codul sediului găsit. Deci nu va mai fi colet temporar.

Dacă nu găsim acel sediu ordonăm sediile crescător după numărul de colete nelivrante din sediu și vedem în care sediu putem pune un articol din comandă.

Se va crea cate un colet care să conțină articolele repartizate în fiecare sediu.

La final să se specifică dacă au rămas articole nerepartizate și care, plus cantitatea.

### **Primul trigger:**

```
CREATE OR REPLACE TRIGGER exercitiul11_inainte
    BEFORE INSERT OR UPDATE ON comanda FOR EACH ROW
DECLARE
    v_numar_inregistrari number(4);

BEGIN
    IF INSERTING THEN
        SELECT COUNT(*)
        INTO v_numar_inregistrari --cate articole are bifate in cos
        FROM detalii_tranzactie dt, cos c
        WHERE :NEW.cod_detalii_tranzactie = dt.cod_detalii_tranzactie AND dt.cod_beneficiar
        = c.cod_beneficiar AND c.bifat = 1;

        IF v_numar_inregistrari = 0 THEN
            RAISE_APPLICATION_ERROR(-20004,'Beneficiarul nu are articole bifate in cos
            pentru a face comanda');
        END IF;
    END IF;
```

```

ELSIF UPDATING THEN
    IF :NEW.status_comanda = 'livrata' AND :OLD.status_comanda = 'nelivrata'
    THEN
        SELECT COUNT(*)
        INTO v_numar_inregistrari -- verificam ca toate coletele componente sa fie livrate,
        afand cate sunt nelivrante
        FROM colet c
        WHERE c.cod_comanda = :OLD.cod_comanda AND c.status_colet <> 'livrat';

        IF v_numar_inregistrari > 0
            THEN RAISE_APPLICATION_ERROR(-20005,'O comanda poate avea statusul
            livrata doar daca toate coletele sale au acest status');
        END IF;

        END IF;

    END IF;

--beneficiarul 150 nu are produse in cos
select * from cos where cod_beneficiar = 150;

insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite,
suma_platita_in_puncte, suma_platita_card, suma_platita_numerar)
values (80000, 150, to_date('15-08-2024', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);

insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda)
values (7000, 80000, 'neprocesata', 1, 'general');

rollback;

--comanda 100 e nelivrata si are si colete nelivrante

```

```

select * from comanda where cod_comanda = 100;
select * from colet where cod_comanda = 100;
UPDATE comanda SET status_comanda = 'livrata' WHERE cod_comanda = 100;

```

```

UPDATE colet SET status_colet = 'livrat' WHERE cod_comanda = 100;
select * from colet where cod_comanda = 100;
UPDATE comanda SET status_comanda = 'livrata' WHERE cod_comanda = 100;
select * from comanda where cod_comanda = 100;
rollback;

```

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, several tabs are open, including 'exercitul11.sql'. The main area is the 'Worksheet' tab, which contains the following PL/SQL code:

```

--delete from detalii_tranzactie where cod_detalii_tranzactie = 300;
CREATE OR REPLACE TRIGGER exercitiul11_inainte
    BEFORE INSERT OR UPDATE ON comanda FOR EACH ROW
DECLARE
    v_numar_inregistrari number(4);
BEGIN
    IF INSERTING THEN
        SELECT COUNT(*)
        INTO v_numar_inregistrari --cate articole are bifate in cos
        FROM detalii_tranzactie dt, cos c
        WHERE :NEW.cod_detalii_tranzactie = dt.cod_detalii_tranzactie AND dt.cod_beneficiar = c.cod_beneficiar AND c.bifat = 1;

        IF v_numar_inregistrari = 0 THEN
            RAISE_APPLICATION_ERROR(-20004,'Beneficiarul nu are articole bifate in cos pentru a face comanda');
        END IF;
    ELSIF UPDATING THEN
        IF :NEW.status_comanda = 'livrata' AND :OLD.status_comanda = 'nelivrata'
        THEN SELECT COUNT(*) INTO v_numar_inregistrari -- verificam ca toate coletele componente sa fie livrate, afland cate
            FROM colet c
            WHERE c.cod_comanda = :OLD.cod_comanda AND c.status_colet <> 'livrat';

            IF v_numar_inregistrari > 0
            THEN RAISE_APPLICATION_ERROR(-20005,'O comanda poate avea statusul livrata doar daca toate coletele sale au acest status');
            END IF;
        END IF;
    END IF;
END;
/

```

Below the code, there is additional commented-out code related to beneficiary 150:

```

--beneficiarul 150 nu are produse in cos
select * from cos where cod_beneficiar = 150;
insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castig)
insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare, tip_comanda) values (7000, 80000, 'livrata');

```

In the bottom 'Script Output' tab, the message 'Trigger EXERCITIUL11\_INAINTE compiled' is displayed.

**Pentru inserare:** Beneficiarul 150 nu are produse în cos.

```
--beneficiarul 150 nu are produse in cos
select * from cos where cod_beneficiar = 150;
insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castig, suma_platita_in_puncte, suma_platita_card, suma_platita_numar)
values (80000, 150, to_date('15-08-2024', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);
insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare, tip_comanda)
values (7000, 80000, 'neprocesata', 1, 'general');
rollback;
```

Script Output | Query Result

All Rows Fetched: 0 in 0.009 seconds

COD_ARTICOL	COD_BENEFICIAR	CANTITATE	DATA_ADDED	BIFAT
-------------	----------------	-----------	------------	-------

```
--beneficiarul 150 nu are produse in cos
select * from cos where cod_beneficiar = 150;
insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castig, suma_platita_in_puncte, suma_platita_card, suma_platita_numar)
values (80000, 150, to_date('15-08-2024', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);

insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare, tip_comanda)
values (7000, 80000, 'neprocesata', 1, 'general');
rollback;
```

Script Output | Query Result

Task completed in 0.13 seconds

1 row inserted.

Error starting at line : 60 in command -  
insert into comanda (cod\_comanda, cod\_detalii\_tranzactie, status\_comanda, plata\_la\_livrare, tip\_comanda)  
values (7000, 80000, 'neprocesata', 1, 'general')  
Error report -  
ORA-20004: Beneficiarul nu are articole bifate in cos pentru a face comanda  
ORA-06512: at "EU.EXERCITIUL11\_INAINTE", line 11  
ORA-04088: error during execution of trigger 'EU.EXERCITIUL11\_INAINTE'

**Pentru update:** Comanda 100 e neliivrata și are colete neliivrate.

```
--comanda 100 e neliivrata si are si colete neliivrate
select * from comanda where cod_comanda = 100;
```

Script Output | Query Result | Query Result 1

All Rows Fetched: 1 in 0.034 seconds

COD_COMANDA	COD_DETALII_TRANZACTIE	STATUS_COMANDA	TIP_COMANDA	PLATA_LA_LIVRARE
1	100	100 neliivrata	general	0

```
--comanda 100 e nelivrata si are si colete nelivrate
select * from comanda where cod_comanda = 100;
select * from colet where cod_comanda = 100;
UPDATE comanda SET status_comanda = 'livrata' WHERE cod_comanda = 100;
```

Script Output x | Query Result x | Query Result 1 x

SQL | All Rows Fetched: 3 in 0.011 seconds

	COD_COLET	COD_COMANDA	COD_ANGAJAT	COD_SEDIU	GREUTATE	DATA_LIVRARE	STATUS_COLET
1	100	100	200	100	200 (null)		nelivrat
2	101	100	200	110	100 (null)		nelivrat
3	102	100	210	120	300 (null)		nelivrat

```
--comanda 100 e nelivrata si are si colete nelivrate
select * from comanda where cod_comanda = 100;
select * from colet where cod_comanda = 100;
UPDATE comanda SET status_comanda = 'livrata' WHERE cod_comanda = 100;

rollback;

CREATE OR REPLACE TRIGGER exercitiul11_dupa
AFTER INSERT ON comanda FOR EACH ROW
```

Script Output x | Query Result x | Query Result 1 x

Task completed in 0.087 seconds

Error starting at line : 66 in command -
UPDATE comanda SET status\_comanda = 'livrata' WHERE cod\_comanda = 100
Error report -
ORA-20005: O comanda poate avea statusul livrata doar daca toate coletele sale au acest status
ORA-06512: at "EU.EXERCITIUL11\_INAINTE", line 21
ORA-04088: error during execution of trigger 'EU.EXERCITIUL11\_INAINTE'

Îl setăm toate coletele ca fiind livrate și încercăm din nou update-ul:

```
UPDATE colet SET status_colet = 'livrat' WHERE cod_comanda = 100;
select * from colet where cod_comanda = 100;
UPDATE comanda SET status_comanda = 'livrata' WHERE cod_comanda = 100;

rollback;
```

COD_COLET	COD_COMANDA	COD_ANGAJAT	COD_SEDIU	GREUTATE	DATA_LIVRARE	STATUS_COLET
1	100	100	200	100	200 (null)	livrat
2	101	100	200	110	100 (null)	livrat
3	102	100	210	120	300 (null)	livrat

```
UPDATE colet SET status_colet = 'livrat' WHERE cod_comanda = 100;
select * from colet where cod_comanda = 100;
UPDATE comanda SET status_comanda = 'livrata' WHERE cod_comanda = 100;
select * from comanda where cod_comanda = 100;
rollback;
```

1 row updated.

Trigger-ul a permis update-ul.

```
select * from colet where cod_comanda = 100;
UPDATE comanda SET status_comanda = 'livrata' WHERE cod_comanda = 100;
select * from comanda where cod_comanda = 100;
rollback;
```

	COD_COMANDA	COD_DETALII_TRANZACTIE	STATUS_COMANDA	TIP_COMANDA	PLATA_LA_LIVRARE
1	100		100 livrata	generală	0

**Al doilea trigger:**

```
CREATE OR REPLACE TRIGGER exercitiul11_dupa
  AFTER INSERT ON comanda FOR EACH ROW
DECLARE
  v_minim_colete int := 99999999;
  v_colete int;
  v_cod_minim number(10) := -1; --pt a gasi sediul cu cat mai putine colete de livrat
  v_cod_beneficiar beneficiar.cod_beneficiar%TYPE;
  TYPE articol_record is RECORD (cod_articol articol.cod_articol%TYPE, cantitate
  articol_in_colet.cantitate%TYPE);
  TYPE tablou_articole IS TABLE OF articol_record INDEX BY PLS_INTEGER;
  t_articole tablou_articole;
  v_articol_gasit boolean;
  v_cantitate_articol_in_sediu.cantitate%TYPE;
  v_cantitate_luata int;
  v_cod_colet int;

  CURSOR c_cursor IS --retine articolele din cos
    SELECT cod_articol, cantitate
    FROM cos c
    WHERE c.cod_beneficiar = v_cod_beneficiar AND bifat = 1;
BEGIN
  SELECT cod_beneficiar --aflam cine a facut comanda
  INTO v_cod_beneficiar
```

```

FROM detalii_tranzactie dt
WHERE dt.cod_detalii_tranzactie = :NEW.cod_detalii_tranzactie;

DBMS_OUTPUT.PUT_LINE('Beneficiarul care a facut comanda are codul ' ||
v_cod_beneficiar || ' si a comandat articolele: ');

--cream coletul temporar care poate deveni colet normal daca nu va trebui sa impartim
articolele in mai multe colete

--ii punem codul 1

INSERT INTO colet(cod_colet, cod_comanda, greutate, status_colet)
VALUES (1, :NEW.cod_comanda, 1, 'colet_temporar');

FOR i IN c_cursor LOOP

DBMS_OUTPUT.PUT_LINE(' cod_articol: ' || i.cod_articol || ', cantitate: ' || i.cantitate);
-- adaugam fiecare produs in noul colet:

INSERT INTO articol_in_colet(cod_colet, cod_comanda, cod_articol, cantitate)
VALUES (1, :NEW.cod_comanda, i.cod_articol, i.cantitate);

END LOOP;

--gasim sediile care au toate produsele din comanda:
DBMS_OUTPUT.PUT_LINE('Sediile care contin toate articolele din comanda sunt: ');

FOR i IN

(SELECT s.cod_sediu as cod_sediu
FROM sediu s
WHERE NOT EXISTS ( -- nu exista articol din sediu care sa nu existe in comanda
SELECT 1 FROM articol_in_colet aic WHERE aic.cod_comanda =
:NEW.cod_comanda AND aic.cod_colet = 1
AND NOT EXISTS (
SELECT 1
FROM articol_in_sediu ais
WHERE ais.cod_sediu = s.cod_sediu AND ais.cod_articol = aic.cod_articol

```

```

        AND ais.cantitate >= aic.cantitate
    )))

LOOP

--calculam cate colete are deja de livrat sediul
SELECT COUNT(*)
INTO v_colete
FROM colet c
WHERE c.cod_sediul = i.cod_sediul;
DBMS_OUTPUT.PUT_LINE(i.cod_sediul || ', din care pleaca ' || v_colete || ' colete');

IF v_colete < v_minim_colete THEN --daca sediul curent are mai putine colete de
livrat updatam minimul
    v_minim_colete := v_colete;
    v_cod_minim := i.cod_sediul;
END IF;

END LOOP;

IF v_cod_minim = -1 THEN DBMS_OUTPUT.PUT_LINE('Nu am gasit un sediu care sa aiba
toate articolele cerute.');
--impartim articolele in cate mai putine colete; luam sediile pe rand si punem in coletul
catre sediul respectiv toate produsele care se gasesc acolo
OPEN c_cursor;
FETCH c_cursor BULK COLLECT INTO t_articole; -- in t_articole retinem articolul si
cantitatea care a ramas nedistribuita in sediu(sedii)
CLOSE c_cursor;

FOR i IN (SELECT s.cod_sediul as cod_sediul --toate sediile ordonate dupa numarul de
colete
        FROM colet c, sediu s

```

```

WHERE c.cod_sedi(+)=s.cod_sedi
GROUP BY s.cod_sedi
ORDER BY COUNT(c.cod_sedi)) LOOP

```

v\_articol\_gasit := FALSE; --momentan nu am gasit niciun articol din comanda care sa se afle in sediu

FOR j IN t\_articole.FIRST..t\_articole.LAST LOOP --vedem ce articole si in ce cantitati mai avem de distribuit

IF t\_articole(j).cantitate > 0 THEN --daca cantitatea ar fi fost 0 inseamna ca am distribuit tot articolul

```

BEGIN
    SELECT cantitate --vedem in ce cantitate il avem in sediul curent
    INTO v_cantitate
    FROM articol_in_sediu
    WHERE cod_sedi = i.cod_sedi AND cod_articol =
t_articole(j).cod_articol;

```

```

EXCEPTION
    WHEN NO_DATA_FOUND THEN --daca nu s-a gasit inregistrarea
    inseamna ca cantitatea e 0

```

```

        v_cantitate := 0;
    END;

```

```

IF v_cantitate > 0 THEN
    DBMS_OUTPUT.PUT('Am gasit articolul ' || t_articole(j).cod_articol ||
    ' in sediul ' || i.cod_sediu || ' in cantitatea ' || v_cantitate || '. Ne trebuie ' || t_articole(j).cantitate || '
    bucati.');
    v_cantitate_luata := 0; --cat vom lua din sediu

```

IF v\_cantitate <= t\_articole(j).cantitate --daca nu avem destul in sediu  
luam tot si restul va trebui distribuit in alt sediu

THEN t\_articole(j).cantitate := t\_articole(j).cantitate - v\_cantitate;

v\_cantitate\_luata := v\_cantitate;

v\_cantitate := 0; --in sediu nu a mai ramas nimic

ELSE v\_cantitate := v\_cantitate - t\_articole(j).cantitate;

v\_cantitate\_luata := t\_articole(j).cantitate;

t\_articole(j).cantitate := 0;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Luam ' || v\_cantitate\_luata || ' bucati.  
Au mai ramas in sediu ' || v\_cantitate || ' bucati.');

DBMS\_OUTPUT.PUT\_LINE('Mai avem ' || t\_articole(j).cantitate || '  
bucati din articol de distribuit in colete in alte sedii.');

IF v\_articol\_gasit = FALSE THEN --nu exista coletul din sediul  
current, il cream si punem in el articolul

v\_articol\_gasit := TRUE;

v\_cod\_colet := seq\_colet.nextval;

insert into colet (cod\_colet, cod\_comanda, cod\_angajat,  
cod\_sediu, greutate, status\_colet, data\_livrare)

values (v\_cod\_colet, :NEW.cod\_comanda, NULL, i.cod\_sediu, 1,  
'nelivrat', NULL);

END IF;

insert into articol\_in\_colet (cod\_articol, cod\_colet, cod\_comanda,  
cantitate)

values (t\_articole(j).cod\_articol, v\_cod\_colet, :NEW.cod\_comanda,  
v\_cantitate\_luata);

--actualizam cantitatea in sediu si in coletul initial lasam cantitatea  
care mai trebuie distribuita:

```

        UPDATE articol_in_sediul
        SET cantitate = v_cantitate
        WHERE cod_sediul = i.cod_sediul and cod_articol =
t_articole(j).cod_articol;

        UPDATE articol_in_colet
        SET cantitate = t_articole(j).cantitate
        WHERE cod_articol = t_articole(j).cod_articol AND cod_colet = 1
AND cod_comanda = :NEW.cod_comanda;
        END IF;

    END IF;

    END LOOP;

    END LOOP;

FOR i in t_articole.FIRST..t_articole.LAST LOOP -- afisam articolele ramase nedistribuite
    IF t_articole(i).cantitate > 0
        THEN DBMS_OUTPUT.PUT_LINE('Ne mai trebuie ' || t_articole(i).cantitate || ' bucati
din articolul ' || t_articole(i).cod_articol || ' care nu au fost repartizate in niciun colet. Asteptam noi
aprovisionari.');
    END IF;
    END LOOP;

    ELSE DBMS_OUTPUT.PUT_LINE('Unul dintre sediile cu activitate putina este ' ||
v_cod_minim || '. Aici vom distribui coletul');

        UPDATE colet SET cod_sediul = v_cod_minim, status_colet = 'nelivrat' WHERE
cod_colet = 1 and cod_comanda = :NEW.cod_comanda;
        END IF;

--debifam in cos:
```

```
UPDATE cos
SET bifat = 0
WHERE cod_beneficiar = v_cod_beneficiar;
END;
/
```

--va gasi toate articolele intr-un sediu:

```
insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_punche,
suma_platita_card, suma_platita_numerar)
values (80000, 130, to_date('15-08-2024', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);
insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda)
values (7000, 80000, 'neprocesata', 1, 'general');
```

--vedem daca starea coletului temporar s-a schimbat si daca e repartizat intr-un sediu:

```
select * from colet c where c.cod_colet = 1 and c.cod_comanda = 7000;
```

--vedem ce articole are in el:

```
select c.cod_colet, c.cod_comanda, aic.cod_articol from colet c, articol_in_colet aic
where c.cod_colet = 1 and c.cod_comanda = 7000 and aic.cod_colet = 1 and aic.cod_comanda =
c.cod_comanda;
```

--vedem daca mai sunt bifate articolele in cos:

```
select * from cos c where cod_beneficiar = 130;
```

--nu va gasi toate articolele intr-un sediu:

```
insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total,
puncte_fidelitate_castigate, puncte_fidelitate_folosite, suma_platita_in_punche,
suma_platita_card, suma_platita_numerar)
values (500, 140, to_date('15-08-2024', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);
insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare,
tip_comanda)
```

```
values (400, 500, 'neprocesata', 1, 'general');
```

--vedem ce a mai ramas in coletul initial:

```
select c.cod_colet, c.cod_comanda, aic.cod_articol, cantitate from colet c, articol_in_colet aic  
where c.cod_colet = 1 and c.cod_comanda = 400 and aic.cod_colet = 1 and aic.cod_comanda =  
c.cod_comanda;
```

--vedem restul coletelor in care s-a impartit primul:

```
select * from colet c where c.cod_comanda = 400;
```

--si ce articole are fiecare:

```
select c.cod_colet, aic.cod_articol, cantitate from colet c, articol_in_colet aic  
where c.cod_comanda = 400 and aic.cod_colet = c.cod_colet and aic.cod_comanda =  
c.cod_comanda;
```

```
rollback;
```

```
DROP TRIGGER exercitiul11_inainte;
```

```
DROP TRIGGER exercitiul11_dupa;
```

SQL Worksheet History

0.26300001 seconds

Worksheet Query Builder

```

| select * from comanda where cod_comanda = 100;
| rollback;

CREATE OR REPLACE TRIGGER exercitiul11_dupa
    AFTER INSERT ON comanda FOR EACH ROW
DECLARE
    v_minim_colete int := 99999999;
    v_colete int;
    v_cod_minim number(10) := -1; --pt a gasi sediul cu cat mai putine colete de livrat
    v_cod_beneficiar beneficiar.cod_beneficiar%TYPE;
    TYPE articol_record is RECORD (cod_articol articol.cod_articol%TYPE, cantitate articol_in_colet.cantitate%TYPE);
    TYPE tablou_articole IS TABLE OF articol_record INDEX BY PLS_INTEGER;
    t_articole tablou_articole;
    v_articol_gasit boolean;
    v_cantitate articol_in_sediu.cantitate%TYPE;
    v_cantitate_luata int;
    v_cod_colet int;

    CURSOR c_cursor IS --retine articolele din cos
        SELECT cod_articol, cantitate
        FROM cos c
        WHERE c.cod_beneficiar = v_cod_beneficiar AND bifat = 1;
BEGIN
    SELECT cod_beneficiar --aflam cine a facut comanda
    INTO v_cod_beneficiar
    FROM detalii_tranzactie dt
    WHERE dt.cod_detalii_tranzactie = :NEW.cod_detalii_tranzactie;

    DBMS_OUTPUT.PUT_LINE('Beneficiarul care a facut comanda are codul ' || v_cod_beneficiar || ' si a comandat articolele: ');

```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x

Task completed in 0.263 seconds

Trigger EXERCITIUL11\_DUPA compiled

## Cazul 1:

SQL Worksheet History | 0.131 seconds | conexiune\_noua

```

Worksheet | Query Builder
WHERE cod_beneficiar = v_cod_beneficiar;
END;
/
--va gasi toate articolele intr-un sediu:
insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_caducare)
values (80000, 130, to_date('15-08-2024', 'dd-mm-yyyy'), 280, 28, 0, 280, 0);
insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare, tip_comanda)
values (7000, 80000, 'neprocesata', 1, 'generală');

--vedem daca starea coletului temporar s-a schimbat si daca e repartizat intr-un sediu:
select * from colet c where c.cod_colet = 1 and c.cod_comanda = 70000;
--vedem ce articole are in el:
select c.cod_colet, c.cod_comanda, aic.cod_articol from colet c, articol_in_colet aic
where c.cod_colet = 1 and c.cod_comanda = 70000 and aic.cod_colet = 1 and aic.cod_comanda = c.cod_comanda;

```

Script Output | Task completed in 0.131 seconds | 1 row inserted.

1 row inserted.

Compiler - Log | Messages | Logging Page | Statements | Compiler

Dbs Output | Buffer Size: 20000 | conexiune\_noua

```

Beneficiarul care a facut comanda are codul 130 si a comandat articolele:
cod_articol: 102, cantitate: 4
cod_articol: 101, cantitate: 1
cod_articol: 103, cantitate: 4
cod_articol: 108, cantitate: 2
cod_articol: 113, cantitate: 4
Sediile care contin toate articolele din comanda sunt:
100, din care pleaca 2 colete
110, din care pleaca 3 colete
120, din care pleaca 3 colete
140, din care pleaca 0 colete
Unul dintre sediile cu activitate putina este 140. Aici vom distribui coletul

```

Click on an identifier with the Control key down to perform "Go to Declaration" | Line 226 Column 1 | Insert | Windows:

```

--vedem daca starea coletului temporar s-a schimbat si daca e repartizat intr-un sediu:
select * from colet c where c.cod_colet = 1 and c.cod_comanda = 70000;
--vedem ce articole are in el:
select c.cod_colet, c.cod_comanda, aic.cod_articol from colet c, articol_in_colet aic
where c.cod_colet = 1 and c.cod_comanda = 70000 and aic.cod_colet = 1 and aic.cod_comanda = c.cod_comanda;
--vedem daca mai sunt bifate articolele in cos:
select * from cos c where cod_beneficiar = 130;

```

Script Output | All Rows Fetched: 1 in 0.017 seconds

COD_COLET	COD_COMANDA	COD_ANGAJAT	COD_SEDIU	GREUTATE	DATA_LIVRARE	STATUS_COLET
1	1	7000	(null)	140	1 (null)	nelivrata

```
--vedem daca starea coletului temporar s-a schimbat si daca e repartizat intr-un sediu:
select * from colet c where c.cod_colet = 1 and c.cod_comanda = 7000;
--vedem ce articole are in el:
select c.cod_colet, c.cod_comanda, aic.cod_articol, aic.cantitate from colet c, articol_in_colet aic
where c.cod_colet = 1 and c.cod_comanda = 70000 and aic.cod_colet = 1 and aic.cod_comanda = c.cod_comanda;
--vedem daca mai sunt bifate articolele in cos:
select * from cos c where cod_beneficiar = 130;

--nu va gasi toate articolele intr-un sediu:
insert into detalii_tranzactie (cod_detalii_tranzactie, cod_beneficiar, data_tranzactie, cost_total, puncte_fidelitate)
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x

SQL | All Rows Fetched: 5 in 0.018 seconds

COD_COLET	COD_COMANDA	COD_ARTICOL	CANTITATE
1	1	70000	102
2	1	70000	101
3	1	70000	103
4	1	70000	108
5	1	70000	113

```
select * from colet c where c.cod_colet = 1 and c.cod_comanda = 7000;
--vedem ce articole are in el:
select c.cod_colet, c.cod_comanda, aic.cod_articol from colet c, articol_in_colet aic
where c.cod_colet = 1 and c.cod_comanda = 7000 and aic.cod_colet = 1 and aic.cod_comanda = c.cod_comanda;
--vedem daca mai sunt bifate articolele in cos:
select * from cos c where cod_beneficiar = 130;
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x

SQL | All Rows Fetched: 6 in 0.019 seconds

COD_ARTICOL	COD_BENEFICIAR	CANTITATE	DATA ADAUGARE	BIFAT
102	130	4	10-JUN-23	0
101	130	1	11-JAN-23	0
104	130	1	12-JUL-24	0
103	130	4	15-JAN-23	0
108	130	2	15-JAN-23	0
113	130	4	15-JAN-23	0

## Cazul 2:

--nu va gasi toate articolele intr-un sediu:  
`insert into detalii_tranzactie (cod_detalii_tranzactie, codBeneficiar, data_tranzactie, cost_total, puncte_fidelitate_castigate, puncte_fidelitate_consumate) values (500, 140, to_date('15-08-2024', 'dd-mm-yyyy'), 280, 28, 0, 0, 280, 0);  
insert into comanda (cod_comanda, cod_detalii_tranzactie, status_comanda, plata_la_livrare, tip_comanda)  
values (400, 500, 'neprocesata', 1, 'general');`

--vedem ce a mai ramas in coletul initial:  
`select c.cod_colet, c.cod_comanda, aic.cod_articol, cantitate from colet c, articol_in_colet aic  
where c.cod_colet = 1 and c.cod_comanda = 400 and aic.cod_colet = 1 and aic.cod_comanda = c.cod_comanda;`

**Script Output** | All Rows Fetched: 12 in 0.026 seconds

COD_COLET	COD_ARTICOL	CANTITATE
1	106	1
1	109	1
1	105	2
1	103	4
1	108	2
1	113	4

**Dbsm Output** | **Compiler - Log**

conexiune\_noua

Unui client se stie ca activitatea putina este 140. Aici vom distribui coletele.

Beneficiarul care a facut comanda are codul 140 si a comandat articolele:

```
cod_articol: 106, cantitate: 1
cod_articol: 109, cantitate: 1
cod_articol: 105, cantitate: 2
cod_articol: 103, cantitate: 4
cod_articol: 108, cantitate: 2
cod_articol: 113, cantitate: 4
```

Sedile care contin toate articolele din comanda sunt:

Nu am gasit un sediu care sa aiba toate articolele cerute.  
Am gasit articolul 109 in sediul 140 in cantitatea 75.Ne trebuie 1 bucati.Luam 1 bucati. Au mai ramas in sediu 74 bucati.  
Mai avem 0 bucati din articol de distribuit in colete in alte sedii.  
Am gasit articolul 103 in sediul 140 in cantitatea 65.Ne trebuie 4 bucati.Luam 4 bucati. Au mai ramas in sediu 61 bucati.  
Mai avem 0 bucati din articol de distribuit in colete in alte sedii.  
Am gasit articolul 108 in sediul 140 in cantitatea 65.Ne trebuie 2 bucati.Luam 2 bucati. Au mai ramas in sediu 108 bucati.  
Mai avem 0 bucati din articol de distribuit in colete in alte sedii.  
Am gasit articolul 113 in sediul 140 in cantitatea 10.Ne trebuie 4 bucati.Luam 4 bucati. Au mai ramas in sediu 6 bucati.  
Mai avem 0 bucati din articol de distribuit in colete in alte sedii.  
Am gasit articolul 106 in sediul 130 in cantitatea 150.Ne trebuie 1 bucati.Luam 1 bucati. Au mai ramas in sediu 149 bucati.  
Mai avem 0 bucati din articol de distribuit in colete in alte sedii.  
Am gasit articolul 105 in sediul 110 in cantitatea 1.Ne trebuie 2 bucati.Luam 1 bucati. Au mai ramas in sediu 0 bucati.  
Mai avem 1 bucati din articol de distribuit in colete in alte sedii.  
Ne mai trebuie 1 bucati din articolul 105 care nu au fost repartizate in niciun colet. Asteptam noi aprovizionari.

```

--vedem ce a mai ramas in coletul initial:
select c.cod_colet, c.cod_comanda, aic.cod_articol, cantitate from colet c, articol_in_colet aic
where c.cod_colet = 1 and c.cod_comanda = 400 and aic.cod_colet = 1 and aic.cod_comanda = c.cod_comanda;

--vedem restul coletelor in care s-a impartit primul:
select * from colet c where c.cod_comanda = 400;

--si ce articole sunt fiecare:
select c.cod_colet, aic.cod_articol, cantitate from colet c, articol_in_colet aic
where c.cod_comanda = 400 and aic.cod_colet = c.cod_colet and aic.cod_comanda = c.cod_comanda;
rollback;

DROP TRIGGER exercitiul11_inainte;
DROP TRIGGER exercitiul11_dupa;

```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x | Query Result 5 x | .

SQL | All Rows Fetched: 6 in 0.097 seconds

	COD_COLET	COD_COMANDA	COD_ARTICOL	CANTITATE
1	1	400	106	0
2	1	400	109	0
3	1	400	105	1
4	1	400	103	0
5	1	400	108	0
6	1	400	113	0

```
where c.cod_colet = 1 and c.cod_comanda = 400 and aic.cod_colet = 1 and aic.cod_comanda = c.cod_comanda;

--vedem restul coletelor in care s-a impartit primul:
select * from colet c where c.cod_comanda = 400;

--si ce articole sunt fiecare:
select c.cod_colet, aic.cod_articol, cantitate from colet c, articol_in_colet aic
where c.cod_comanda = 400 and aic.cod_colet = c.cod_colet and aic.cod_comanda = c.cod_comanda;
rollback;

DROP TRIGGER exercitiu111_ainte;
DROP TRIGGER exercitiu111_dupa;
```

	COD_COLET	COD_COMANDA	COD_ANGAJAT	COD_SEDIU	GREUTATE	DATA_LIVRARE	STATUS_COLET	
1	1	400	(null)	(null)	1	(null)	colet_temporar	
2	111	400	(null)	140	1	(null)	nelivrat	
3	112	400	(null)	130	1	(null)	nelivrat	
4	113	400	(null)	110	1	(null)	nelivrat	

```
--vedem restul coletelor in care s-a impartit primul:  
select * from colet c where c.cod_comanda = 400;  
  
--si ce articole are fiecare:  
select c.cod_colet, aic.cod_articol, cantitate from colet c, articol_in_colet aic  
where c.cod_comanda = 400 and aic.cod_colet = c.cod_colet and aic.cod_comanda = c.cod_comanda;  
rollback;
```

All Rows Fetched: 12 in 0.007 seconds

	COD_COLET	COD_ARTICOL	CANTITATE
1		106	0
2		109	0
3		105	1
4		103	0
5		108	0
6		113	0
7	111	109	1
8	111	103	4
9	111	108	2
10	111	113	4
11	112	106	1
12	113	105	1

## 12) Definiți un trigger de tip LDD. Declanșați trigger-ul.

Creați un tabel în care să stocați operațiile aplicate asupra structurii bazei de date.

Afişați în dbms output când se creează, sterge sau modifică un obiect.

The screenshot shows the Oracle SQL Developer interface with the following code in the SQL Worksheet:

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

--Creeati un tabel in care sa stocati operatiile aplicate asupra structurii bazei de date.
--Afisati in dbms output cand se creeaza, sterge sau modifica un obiect.

create sequence seq_operatie start with 100 increment by 1 maxvalue 100000 nocycle nocache;

create table operatie(
    cod_operatie int constraint cod_operatie_pk primary key,
    operatie varchar2(40) NOT NULL,
    baza_de_date varchar2(40),
    nume_utilizator varchar2(40),
    data_actiune date default sysdate,
    obiect_modificat varchar2(40) NOT NULL
);

CREATE OR REPLACE TRIGGER exercitiul12
    AFTER CREATE OR ALTER OR DROP ON SCHEMA
DECLARE
    v_object_modificat varchar2(100);
    v_operatie varchar2(100);
BEGIN
    v_object_modificat := sys.dictionary_obj_name;
    v_operatie := sys.event;
    INSERT INTO operatie(cod_operatie, operatie, baza_de_date, nume_utilizator, obiect_modificat)
    VALUES (seq_operatie.nextval, v_operatie, sys.database_name, sys.login_user, v_object_modificat);

    IF v_operatie = 'CREATE' THEN
        DBMS_OUTPUT.PUT_LINE('A fost creat obiectul ' || v_object_modificat);
    ELSIF v_operatie = 'ALTER' THEN
        DBMS_OUTPUT.PUT_LINE('A fost modificat obiectul ' || v_object_modificat);
    ELSIF v_operatie = 'DROP' THEN
        DBMS_OUTPUT.PUT_LINE('A fost sters obiectul ' || v_object_modificat);
    END IF;

```

The Script Output window shows the results of the execution:

```
Sequence SEQ_OPERATIE created.

Table OPERATIE created.
```

...sql | exercitiu8.sql | exercitiu13.sql | exercitiu9.sql | exercitiu6.sql | exercitiu7.sql | exercitiu10.sql | exercitiu12.sql |

SQL Worksheet History

Worksheet Query Builder

```

create table operatie(
    cod_operatie int constraint cod_operatie_pk primary key,
    operatie varchar2(40) NOT NULL,
    baza_de_date varchar2(40),
    nume_utilizator varchar2(40),
    data_actiune date default sysdate,
    obiect_modificat varchar2(40) NOT NULL
)

CREATE OR REPLACE TRIGGER exercitiu12
    AFTER CREATE OR ALTER OR DROP ON SCHEMA
DECLARE
    v_object_modificat varchar2(100);
    v_operatie varchar2(100);
BEGIN
    v_object_modificat := sys.dictionary_obj_name;
    v_operatie := sys.sysevent;
    INSERT INTO operatie(cod_operatie, operatie, baza_de_date, nume_utilizator, obiect_modificat)
    VALUES (seq_operatie.nextval, v_operatie, sys.database_name, sys.login_user, v_object_modificat);

    IF v_operatie = 'CREATE' THEN
        DBMS_OUTPUT.PUT_LINE('A fost creat obiectul ' || v_object_modificat);
    ELSIF v_operatie = 'ALTER' THEN
        DBMS_OUTPUT.PUT_LINE('A fost modificat obiectul ' || v_object_modificat);
    ELSIF v_operatie = 'DROP' THEN
        DBMS_OUTPUT.PUT_LINE('A fost sters obiectul ' || v_object_modificat);
    END IF;
END;

create table tabel (
    cod_tabel int
)
insert into tabel values(2);

```

Script Output | Task completed in 0.461 seconds

Table OPERATIE created.

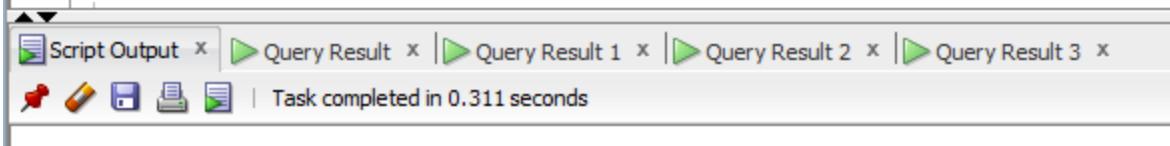
Trigger EXERCITIUL12 compiled

Dbsm Output | Compiler - Log

```
create table tabel (
    cod_tabel int
);

insert into tabel values(2);
delete from tabel where cod_tabel = 2;
truncate table tabel;
drop table tabel;

select * from operatie;
rollback;
DROP TRIGGER exercitiu112;
```

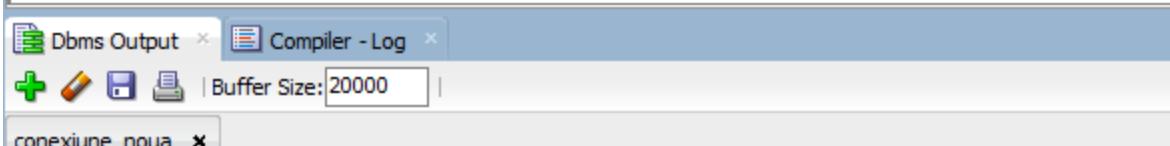


1 row inserted.

1 row deleted.

Table TABEL truncated.

Table TABEL dropped.



A fost creat obiectul TABEL

A fost sters obiectul TABEL

```

truncate table tabel;
drop table tabel;

select * from operatie;
rollback;
DROP TRIGGER exercitiull2;

```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x

SQL | All Rows Fetched: 7 in 0.005 seconds

COD_OPERATIE	OPERATIE	BAZA_DE_DATE	NUME_UTILIZATOR	DATA_ACTIUNE	OBIECT_MODIFICAT
1	100 DROP	XE	EU	25-AUG-24	TABEL
2	101 CREATE	XE	EU	25-AUG-24	TABEL
3	102 DROP	XE	EU	25-AUG-24	TABEL
4	103 CREATE	XE	EU	25-AUG-24	TABEL
5	104 DROP	XE	EU	25-AUG-24	TABEL
6	105 CREATE	XE	EU	25-AUG-24	TABEL
7	106 DROP	XE	EU	25-AUG-24	TABEL

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

```

CREATE OR REPLACE PACKAGE exercitiul13 IS

    PROCEDURE exercitiul6 (v_cod_sediu IN sediu.cod_sediu%TYPE);

    PROCEDURE exercitiul7 (v_nume_sediu IN sediu.nume%TYPE DEFAULT 'CABLE
POWER');

    FUNCTION exercitiul8 (v_cod_articol IN articol.cod_articol%TYPE DEFAULT 100,
v_operatie IN number) RETURN varchar2;

    PROCEDURE exercitiul9(v_nume_beneficiar.nume%TYPE, v_prenume
beneficiar.prenume%TYPE);

END exercitiul13;
/

```

```
CREATE OR REPLACE PACKAGE BODY exercitiul13 IS
```

```
PROCEDURE exercitiul6 (v_cod_sediu IN sediu.cod_sediu%TYPE)
```

IS

    TYPE transport\_record is RECORD (cod\_articol articol.cod\_articol%TYPE, cantitate articol\_in\_sediu.cantitate%TYPE);

    TYPE tablou\_indexat\_transporturi IS TABLE OF transport\_record INDEX BY PLS\_INTEGER;

        t\_transporturi tablou\_indexat\_transporturi;

        v\_pret articol\_in\_sediu.cantitate%TYPE;

        c\_maxim\_furnizori CONSTANT NUMBER := 1000;

        TYPE vector\_furnizori IS VARRAY(c\_maxim\_furnizori) OF furnizor%ROWTYPE;

        t\_furnizori vector\_furnizori := vector\_furnizori();

    TYPE produs\_record is RECORD (cod\_articol articol.cod\_articol%TYPE, nume articol.nume%TYPE);

    TYPE tablou\_imbricat\_produse IS TABLE OF produs\_record;

        t\_produse tablou\_imbricat\_produse := tablou\_imbricat\_produse();

    TYPE sediu\_record is RECORD (cod\_sediu sediu.cod\_sediu%TYPE, nume sediu.nume%TYPE, cantitate articol\_in\_sediu.cantitate%TYPE);

    TYPE tablou\_indexat\_sedii IS TABLE OF sediu\_record INDEX BY BINARY\_INTEGER;

        t\_sedii tablou\_indexat\_sedii;

        v\_cantitate\_totala int;

BEGIN

        t\_transporturi(1) := transport\_record(101, 100);

        t\_transporturi(2) := transport\_record(102, 120);

        t\_transporturi(3) := transport\_record(105, 200);

FOR i IN t\_transporturi.FIRST..t\_transporturi.LAST LOOP --iteram transporturile si le inseram in restem stocurile

```

UPDATE articol_in_sediul
SET cantitate = cantitate + t_transporturi(i).cantitate
WHERE cod_sediul = v_cod_sediul AND cod_articol =
t_transporturi(i).cod_articol;

```

IF SQL%ROWCOUNT = 0 THEN --daca nu a updatat nicio linie inseamna ca produsul nu exista deloc in sediu si trebuie adaugata o inserare

```

SELECT pret INTO v_pret FROM articol WHERE cod_articol =
t_transporturi(i).cod_articol; --pretul il luam din tabela articol, adica pretul universal

```

```

INSERT INTO articol_in_sediul (cod_sediul, cod_articol, cantitate, pret)
VALUES (v_cod_sediul, t_transporturi(i).cod_articol, t_transporturi(i).cantitate, v_pret);

```

```
END IF;
```

```
END LOOP;
```

FOR i in (SELECT \* FROM furnizor) LOOP --retinem in vector toate datele din tabela furnizor (nume si cod, mai e si o adresa de mail dar nu ne trebuie)

```

t_furnizori.EXTEND;
t_furnizori(t_furnizori.LAST) := i;
END LOOP;

```

FOR i IN t\_furnizori.FIRST..t\_furnizori.LAST LOOP --luam produsele furnizate

```

SELECT p.cod_articol, nume BULK COLLECT INTO t_produse
FROM produs p, articol a
WHERE cod_furnizor = t_furnizori(i).cod_furnizor AND p.cod_articol =
a.cod_articol;

```

IF t\_produse.COUNT = 0 THEN DBMS\_OUTPUT.PUT\_LINE('Furnizorul ' ||  
t\_furnizori(i).nume || ' nu furnizeaza produse');

ELSE

```

DBMS_OUTPUT.PUT_LINE('Furnizorul ' || t_furnizori(i).nume || ' ne vinde ' ||
t_produse.COUNT || ' produs(e):');

```

```

FOR j IN t_produse.FIRST..t_produse.LAST LOOP

    v_cantitate_totala := 0;
    DBMS_OUTPUT.PUT_LINE('      ' || t_produse(j).nume || ', cantitati: ');

    SELECT s.cod_sediu, s.nume, NVL(ais.cantitate, 0) AS cantitate BULK
    COLLECT INTO t_sedii --pt un produs aflam cantitatea din fiecare sediu

        FROM sediu s, articol_in_sediu ais
        WHERE cod_articol(+) = t_produse(j).cod_articol AND s.cod_sediu =
    ais.cod_sediu(+)
        ORDER BY s.nume;

    FOR k IN t_sedii.FIRST..t_sedii.LAST LOOP

        DBMS_OUTPUT.PUT_LINE('      sediul' || t_sedii(k).nume
    || ': ' || t_sedii(k).cantitate || ' bucati');

        v_cantitate_totala := v_cantitate_totala + t_sedii(k).cantitate;
    END LOOP;
    --t_sedii.DELETE; --nu trebuie neaparat sters

    DBMS_OUTPUT.PUT_LINE('      Cantitatea totala: ' ||
v_cantitate_totala || ' bucati');

    END LOOP;

DBMS_OUTPUT.PUT_LINE('=====');
--t_produse.DELETE;

END IF;

END LOOP;

END exercitiul6;

```

--EXERCITIUL 7

---

```
PROCEDURE exercitiul7 (v_nume_sediu IN sediu.nume%TYPE DEFAULT 'CABLE
POWER') IS
```

```
    TYPE refcursor IS REF CURSOR;
```

```
    v_cursor refcursor;
```

CURSOR c\_principal IS --ciclu cursor care selecteaza pentru un colet codul si curierul (daca exista); ia doar coletele care respecta cerinta

```
    SELECT c.cod_colet cod_colet, c.cod_comanda cod_comanda, NVL(a.cod_angajat,
-1) AS cod_curier,
```

```
        NVL(CONCAT(a.nume, ' ' || a.prenume), 'Curierul nu a fost inca asignat') AS
nume_curier
```

```
    FROM sediu s, colet c, angajat a
```

```
    WHERE c.cod_sediu = s.cod_sediu AND v_nume_sediu = UPPER(s.nume) AND
UPPER(c.status_colet) = 'NELIVRAT'
```

```
    AND a.cod_angajat = c.cod_angajat(+); --left join pentru cazul in care curierul are
cod_angajat NULL
```

CURSOR c\_secundar (v\_cod\_colet colet.cod\_colet%TYPE, v\_cod\_comanda
colet.cod\_comanda%TYPE) IS --cursor clasic parametrizat cu un ref cursor in el;pentru o
comanda afla cate produse din pachete promotionale are

```
    SELECT a.cod_articol as cod, a.nume as nume, aic.cantitate as cantitate,
```

```
        CURSOR(SELECT pip.cod_produs cod_produs, a2.nume nume2 -- expresie
cursor care pt un pachet afla ce produse are
```

```
        FROM produs_in_pachet pip, articol a2
```

```
        WHERE pip.cod_pachet = a.cod_articol AND a2.cod_articol =
pip.cod_produs)
```

```
        FROM articol_in_colet aic, pachet_promotional p, articol a
```

```
        WHERE aic.cod_colet = v_cod_colet and aic.cod_comanda = v_cod_comanda and
aic.cod_articol = p.cod_articol
```

```
        AND a.cod_articol = p.cod_articol;
```

```

v_detalii_curier varchar2(100);
v_cod_articol articol.cod_articol%TYPE;
v_nume_articol articol.nume%TYPE;
v_nume_produs articol.nume%TYPE;
v_cantitate articol_in_colet.cantitate%TYPE;
v_cod_produs produs.cod_articol%TYPE;

BEGIN
    FOR i IN c_principal LOOP

        EXIT WHEN c_principal%NOTFOUND;
        v_detalii_curier := ' (';

        IF i.cod_curier = -1
            THEN v_detalii_curier := v_detalii_curier || 'al carui curier nu a fost inca asignat)';
        ELSE v_detalii_curier := v_detalii_curier || 'care va fi livrat de curierul ' ||
i.nume_curier || ' cod ' || i.cod_curier || ')';
        END IF;

        DBMS_OUTPUT.PUT_LINE('Coletul' || v_detalii_curier || ' cu codul ' || i.cod_colet
|| ' din comanda ' || i.cod_comanda || ', are pachetele:');

        OPEN c_secundar(i.cod_colet, i.cod_comanda);
        LOOP
            FETCH c_secundar INTO v_cod_articol, v_nume_articol, v_cantitate,
v_cursor;
            EXIT WHEN c_secundar%NOTFOUND;
            DBMS_OUTPUT.PUT_LINE('cod: ' || v_cod_articol || ', nume: ' || v_nume_articol || ', cantitate: ' || v_cantitate || ', produse componente');
        END LOOP;
    END LOOP;
END;

```

```

        LOOP
            FETCH v_cursor INTO v_cod_produs, v_nume_produs;
            EXIT WHEN v_cursor%NOTFOUND;
            DBMS_OUTPUT.PUT_LINE('' || v_nume_produs
|| ', cod:' || v_cod_produs);
        END LOOP;

        END LOOP;
        CLOSE c_secundar;
    END LOOP;

END exercitiul7;

```

--EXERCITIUL 8

---

```

FUNCTION exercitiul8 (v_cod_articol IN articol.cod_articol%TYPE DEFAULT 100,
v_operatie IN number)

```

```

    RETURN varchar2

```

```

    IS

```

```

        COD_ARTICOL_INVALID exception;

```

```

        COMANDA_INVALIDA exception;

```

```

        OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI exception;

```

```

        PRODUS_NU_ARE_FURNIZOR exception;

```

```

        NU_EXISTA_CONCURSURI exception;

```

```

        v_concluzie varchar2(100);

```

```

        v_temporar_cod_articol articol.cod_articol%TYPE;

```

```

        v_tip_articol varchar2(15);

```

```

v_nume_furnizor furnizor.nume%TYPE;
v_cod_furnizor furnizor.cod_furnizor%TYPE;
v_nr_produse_furnizate number(6);
v_nr_concursuri number(5);
v_nr_participanti number(6, 2);
v_profit number(8);
v_cost_articol number(8);
v_medie_reclame number(8);

BEGIN

    v_concluzie := 'Nu a fost efectuata operatia';

    BEGIN --bloc pentru a verifica daca exista codul introdus
        SELECT cod_articol INTO v_temporar_cod_articol FROM articol WHERE
cod_articol = v_cod_articol;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            RAISE COD_ARTICOL_INVALID;
    END;

    --aflam tipul articolului: produs sau pachet promotional
    v_tip_articol := 'produs';
    BEGIN
        SELECT cod_articol INTO v_temporar_cod_articol FROM produs WHERE cod_articol =
v_cod_articol;

    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            v_tip_articol := 'pachet';
    END;

```

```

IF v_operatie = 1 THEN
    --verificam daca articolul este produs, altfel nu are furnizor.
    IF v_tip_articol = 'produs' THEN
        SELECT cod_furnizor INTO v_cod_furnizor
        FROM produs WHERE cod_articol = v_cod_articol;

        IF v_cod_furnizor IS NULL
        THEN RAISE PRODUS_NU_ARE_FURNIZOR;
        ELSE
            SELECT f.nume, (COUNT(*) - 1) INTO v_nume_furnizor,
            v_nr_produse_furnizate --COUNT - 1 pt ca un produs furnizat e parametrul functiei
            FROM furnizor f, produs p WHERE f.cod_furnizor = v_cod_furnizor AND
            p.cod_furnizor = f.cod_furnizor
            GROUP BY f.nume;

            RETURN 'Produsul e furnizat de ' || v_nume_furnizor || ' care are codul ' ||
            v_cod_furnizor || ' si mai furnizeaza alte ' || v_nr_produse_furnizate || ' produse';
        END IF;
        RETURN v_concluzie;
    ELSE
        RAISE OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI;
    END IF;

ELSIF v_operatie = 2 THEN
    IF v_tip_articol = 'produs' THEN
        SELECT COUNT(*) nr_concursuri, AVG(nr_participari), SUM(profit_concurs)
        INTO v_nr_concursuri, v_nr_participanti, v_profit
        FROM( --luam detalii despre toate concursurile care au printre premii produsul
        SELECT c.cod_concurs cod_concurs, COUNT(pp.codBeneficiar)
        nr_participari,

```

```

        (SELECT COUNT(*) * c.taxa_inscriere --doar beneficiarii fara
cont_premium platesc taxa
        FROM participa pp2, beneficiar b
        WHERE pp2.cod_concurs = c.cod_concurs AND b.cod_beneficiar =
pp2.cod_beneficiar
        AND b.cont_premium = 0) AS profit_concurs
        FROM premiu p, concurs c, participa pp
        WHERE p.cod_articol = v_cod_articol AND p.cod_concurs = c.cod_concurs
AND pp.cod_concurs = c.cod_concurs
        AND TO_CHAR(c.data_concurs, 'YYYY') = 2023
        GROUP BY c.cod_concurs, c.taxa_inscriere
);

```

```

IF v_nr_concursuri = 0 THEN RAISE NU_EXISTA_CONCURSURI;
ELSE RETURN 'Articolul de tip ' || v_tip_articol || ' apare ca premiu in ' ||
v_nr_concursuri || ' concursuri din 2023, la care au participant in medie ' || v_nr_participanti || '
persoane.
Profitul total al concursurilor este ' || v_profit || ' RON.';

END IF;

```

```

ELSE
    RAISE OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI;
END IF;

```

```

ELSIF v_operatie = 3 THEN
    --aflam daca articolul are costul din reclame mai mare decat media
    SELECT AVG(cost_per_reclama)
    INTO v_medie_reclame FROM(
        SELECT a.cod_articol, SUM(NVL(r.cost_total, 0)) cost_per_reclama --costul
reclamelor per articol
        FROM articol a, reclama r
        WHERE a.cod_articol = r.cod_articol(+)

```

```

        GROUP BY a.cod_articol);

        DBMS_OUTPUT.PUT_LINE('Media costului reclamelor per articol este ' || v_medie_reclame);

        --calculam costul din reclame pt articol:
        SELECT NVL(SUM(r.cost_total), 0)
        INTO v_cost_articol
        FROM articol a, reclama r
        WHERE a.cod_articol = r.cod_articol;

        DBMS_OUTPUT.PUT_LINE('Costul reclamelor pentru articolul ' || v_cod_articol || ' este ' || v_cost_articol);

        IF v_cost_articol > v_medie_reclame
        THEN RETURN 'Articolul are costul din reclame mai mare decat media';
        ELSE RETURN 'Articolul NU are costul din reclame mai mare decat media';
        END IF;

        ELSE
        RAISE COMANDA_INVALIDA;
        END IF;
        RETURN v_concluzie;

EXCEPTION
    WHEN COD_ARTICOL_INVALID THEN
        RETURN 'Exceptie: Codul introdus nu se alfa in baza de date';
    WHEN COMANDA_INVALIDA THEN
        RETURN 'Exceptie: Comanda aleasa nu este valida';
    WHEN OPERATIE_INVALIDA_PT_TIPUL_ARTICOLULUI THEN
        RETURN 'Exceptie: Operatie invalida pentru tipul articoului';
    WHEN PRODUS_NU_ARE_FURNIZOR THEN

```

```

        RETURN 'Exceptie: Produsl nu are furnizor';

WHEN NU_EXISTA_CONCURSURI THEN

        RETURN 'Exceptie: Nu s au gasit concursuri';

WHEN OTHERS THEN

        RETURN 'Exceptie necunoscuta';

END exercitiul8;

```

#### --EXERCITIUL 9

---

```

PROCEDURE exercitiul9(v_nume beneficiar.nume%TYPE, v_prenume
beneficiar.prenume%TYPE)
IS
    PERSOANA CU ACELASI NUME exception;
    NU_EXISTA_TRANZACTII exception;
    v_cod_beneficiar beneficiar.cod_beneficiar%TYPE;
    v_temporar_cod_beneficiar beneficiar.cod_beneficiar%TYPE;
    TYPE date_articole_record is RECORD (tip_tranzactie varchar2(30), tip_articol
varchar2(30),
                                         puncte_fidelitate articol.puncte_fidelitate%TYPE,
                                         cod_articol articol.cod_articol%TYPE, nume articol.nume%TYPE,
                                         pret_la_cumparare articol.pret%TYPE,
                                         data_cumparare date, cantitate number(8));
    TYPE tablou_imbricat_articole IS TABLE OF date_articole_record;
    t_articole tablou_imbricat_articole := tablou_imbricat_articole();
    t_articole_temporar tablou_imbricat_articole := tablou_imbricat_articole(); --vom
retine aici datele despre toate produsele cumparate (din vanzari si comenzi)

    v_temporar_nr number(6);
    v_suma_totala_comenzi number(11) := 0; -- suma de cost total al tranzactiilor legate
de comenzi

```

```
v_suma_totala_vanzari number(11) := 0; -- suma de cost total al tranzactiilor legate  
de vanzari
```

```
v_suma_totala_comenzi_atunci number(11) := 0;
```

```
v_suma_totala_vanzari_atunci number(11) := 0;
```

```
v_detalii_garantie varchar2(150);
```

```
v_luni_garantie produs.luni_garantie%TYPE;
```

```
v_data_expirare_garantie date;
```

```
v_nr_total_puncte_fidelitate number(11) := 0; --adunam pt fiecare articol cumparat  
punctele de fidelitate
```

```
v_cod_temporar int;
```

```
BEGIN
```

```
--verificam daca mai exista alta persoana cu acelasi nume si prenume
```

```
BEGIN
```

```
    SELECT codBeneficiar INTO v_codBeneficiar FROM beneficiar WHERE  
prenume = v_prenume AND nume = v_nume;
```

```
EXCEPTION
```

```
    WHEN TOO_MANY_ROWS THEN RAISE PERSOANA CU ACELASI NUME;
```

```
END;
```

```
--verificam daca persoana a facut cumparaturi
```

```
BEGIN
```

```
    SELECT codBeneficiar INTO v_temporar_nr FROM detalii_tranzactie WHERE  
v_codBeneficiar = codBeneficiar;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN RAISE NU_EXISTA_TRANZACTII;
```

```
    WHEN TOO_MANY_ROWS THEN NULL; --se va arunca too many rows daca  
avem tranzactii. oprim exceptia.
```

```
END;
```

```
--calculam datele pentru comenzi:
```

```
SELECT 'comanda' tip_tranzactie,
```

```

CASE WHEN pp.cod_articol IS NULL THEN 'produs'
      ELSE 'pachet'
END AS tip_articol,
a.puncte_fidelitate puncte_fidelitate, a.cod_articol cod_articol, a.nume nume,
a.pret pret_la_cumparare, cc.data_livrare data_cumparare, aic.cantitate
cantitate
BULK COLLECT INTO t_articole
FROM detalii_tranzactie dt, comanda c, colet cc, articol_in_colet aic, articol a,
pachet_promotional pp
WHERE dt.codBeneficiar = v_codBeneficiar AND dt.cod_detalii_tranzactie =
c.cod_detalii_tranzactie
AND c.cod_comanda = cc.cod_comanda
AND aic.cod_colet = cc.cod_colet AND aic.cod_comanda = cc.cod_comanda
AND aic.cod_articol = a.cod_articol
AND a.cod_articol = pp.cod_articol(+);

```

--datele pentru vanzari fizice:

```

SELECT 'vanzare fizica' tip_tranzactie,
CASE WHEN a.cod_articol IN (SELECT cod_articol FROM produs) THEN
'produs'
ELSE 'pachet'
END AS tip_articol,
a.puncte_fidelitate puncte_fidelitate, a.cod_articol cod_articol, a.nume nume,
ais.pret pret_la_cumparare,
dt.data_tranzactie data_cumparare, aiv.cantitate cantitate
BULK COLLECT INTO t_articole_temporar
FROM detalii_tranzactie dt, vanzare_fizica vf, articol_in_vanzare aiv, articol a,
articol_in_sediu ais
WHERE dt.codBeneficiar = v_codBeneficiar AND dt.cod_detalii_tranzactie =
vf.cod_detalii_tranzactie
AND vf.cod_vanzare_fizica = aiv.cod_vanzare_fizica
AND aiv.cod_articol = a.cod_articol AND vf.cod_sediu = ais.cod_sediu AND
ais.cod_articol = a.cod_articol;

```

t\_articole := t\_articole MULTISET UNION t\_articole\_temporar; --adaugam in t\_articole si datele vanzarilor fizice. nu puteam direct pentru ca bulk collect ar fi sters comenzile.

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

FOR i IN t\_articole.FIRST..t\_articole.LAST LOOP

IF i > t\_articole.FIRST AND t\_articole(i - 1).tip\_tranzactie = 'comanda' AND t\_articole(i).tip\_tranzactie = 'vanzare fizica' --cand am gasit primul articol dintr-o vanzare fizica

THEN DBMS\_OUTPUT.PUT\_LINE('VANZARI:');

END IF;

IF t\_articole(i).tip\_tranzactie = 'comanda'

THEN v\_suma\_totala\_comenzi := v\_suma\_totala\_comenzi +

(t\_articole(i).pret\_la\_cumparare \* t\_articole(i).cantitate); --calculam cele 2 sume totale pe care le ar plati cu preturile de acum

ELSE v\_suma\_totala\_vanzari := v\_suma\_totala\_vanzari +  
(t\_articole(i).pret\_la\_cumparare \* t\_articole(i).cantitate);

END IF;

v\_nr\_total\_puncte\_fidelitate := v\_nr\_total\_puncte\_fidelitate +  
(t\_articole(i).puncte\_fidelitate \* t\_articole(i).cantitate);

IF t\_articole(i).tip\_articol = 'pachet' THEN --daca articolul curent e pachet trebuie sa aflam toate produsele din el

DBMS\_OUTPUT.PUT\_LINE('Din pachetul: ' || t\_articole(i).cod\_articol || ':');

FOR j IN

(SELECT p.luni\_garantie, a.nume, a.cod\_articol

FROM produs p, produs\_in\_pachet pip, articol a

WHERE t\_articole(i).cod\_articol = pip.cod\_pachet AND pip.cod\_produs = p.cod\_articol AND a.cod\_articol = p.cod\_articol) LOOP

```

        v_detalii_garantie := "";

        IF t_articole(i).data_cumparare IS NULL --daca nu avem data cumpararii
inseamna ca e dintr o vanzare fizica unde clientul nu a platit prin contul aplicatiei

            THEN v_detalii_garantie := 'produs nelivrat inca, garantia de ' ||
j.luni_garantie || ' luni inca nu se aplica';

        ELSE

            v_data_expirare_garantie :=
ADD_MONTHS(t_articole(i).data_cumparare, j.luni_garantie); --calculam cand expira garantia

            IF v_data_expirare_garantie < sysdate
                THEN v_detalii_garantie := 'garantie expirata';
                ELSE v_detalii_garantie := 'garantia expira pe ' ||
TO_CHAR(v_data_expirare_garantie, 'dd-mm-yyyy');
                END IF;
            END IF;

            DBMS_OUTPUT.PUT_LINE('      cod: ' || j.cod_articol || ' nume: ' ||
j.nume || ', cantitate ' || t_articole(i).cantitate || ', detalii garantie: ' || v_detalii_garantie);

        END LOOP;

        ELSE --daca e de tip produs il afisam direct
            SELECT luni_garantie INTO v_luni_garantie FROM produs WHERE
cod_articol = t_articole(i).cod_articol;

            v_detalii_garantie := "";

            IF t_articole(i).data_cumparare IS NULL
                THEN v_detalii_garantie := 'produs nelivrat inca, garantia de ' ||
v_luni_garantie || ' luni inca nu se aplica';
                ELSE
                    v_data_expirare_garantie :=
ADD_MONTHS(t_articole(i).data_cumparare, v_luni_garantie);

```

```

        IF v_data_expirare_garantie < sysdate
            THEN v_detalii_garantie := 'garantie expirata';
        ELSE v_detalii_garantie := 'garantia expira pe ' ||
TO_CHAR(v_data_expirare_garantie, 'dd-mm-yyyy');
        END IF;
    END IF;

    DBMS_OUTPUT.PUT_LINE('cod: ' || t_articole(i).cod_articol || ', nume: ' ||
t_articole(i).nume || ', cantitate ' || t_articole(i).cantitate || ', detalii garantie: ' || v_detalii_garantie);
END IF;

END LOOP;

```

```

SELECT dt.cod_beneficiar, SUM(dt.cost_total) INTO v_cod_temporar,
v_suma_totala_comenzi_atunci --aflam cat a platit pe toate comenzile
FROM comanda c, detalii_tranzactie dt
WHERE c.cod_detalii_tranzactie = dt.cod_detalii_tranzactie and dt.cod_beneficiar =
v_cod_beneficiar
GROUP BY dt.cod_beneficiar;

```

```

SELECT dt.cod_beneficiar, SUM(dt.cost_total) INTO v_cod_temporar,
v_suma_totala_vanzari_atunci --aflam cat a platit pe toate vanzarile
FROM vanzare_fizica vf, detalii_tranzactie dt
WHERE vf.cod_detalii_tranzactie = dt.cod_detalii_tranzactie and dt.cod_beneficiar =
v_cod_beneficiar
GROUP BY dt.cod_beneficiar;

```

```

DBMS_OUTPUT.PUT_LINE('Beneficiarul a acumulat ' || v_nr_total_puncte_fidelitate
|| ' puncte fidelitate ');
DBMS_OUTPUT.PUT_LINE('Cat a platit in total pt comenzi si vanzari:' ||
v_suma_totala_comenzi_atunci || '' || v_suma_totala_vanzari_atunci);
DBMS_OUTPUT.PUT_LINE('Cat ar fi platit in total pt comenzi si vanzari cu preturile
actuale:' || v_suma_totala_comenzi || '' || v_suma_totala_vanzari);

```

```
EXCEPTION
    WHEN PERSOANA_CU_ACELASI_NUME THEN
        DBMS_OUTPUT.PUT_LINE('Mai exista o persoana cu acelasi nume');
    WHEN NU_EXISTA_TRANZACTII THEN
        DBMS_OUTPUT.PUT_LINE('Persoana nu a efectuat nicio tranzactie');
END exercitiul9;
```

```
END exercitiul13;
```

```
BEGIN
    exercitiul13.exercitiul6(100);
END;
/
```

```
BEGIN
    exercitiul13.exercitiul7();
END;
/
```

```
BEGIN
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul13.exercitiul8(100, 2));
END;
/
```

```
BEGIN
    exercitiul13.exercitiul9('Andronic', 'Marcel');
```

END;

/

The screenshot shows the Oracle SQL Developer interface with the 'Worksheet' tab selected. The code area contains the creation of a package named 'exercitiul13'. The package includes several procedures and functions, such as 'exercitiul6', 'exercitiul7', 'exercitiul8', 'exercitiul9', and 'exercitiul10'. The code uses various PL/SQL data types like RECORDs and TABLEs. The 'Script Output' pane at the bottom shows the message 'PL/SQL procedure successfully completed.' indicating the successful compilation of the package.

```
--13. Defini?i un pachet care s? con?in? toate obiectele definite in cadrul proiectului.
CREATE OR REPLACE PACKAGE exercitiul13 IS
    PROCEDURE exercitiul6 (v_cod_sediu IN sediu.cod_sediu%TYPE);
    PROCEDURE exercitiul7 (v_nume_sediu IN sediu.nume%TYPE DEFAULT 'CABLE POWER');
    FUNCTION exercitiul8 (v_cod_articol IN articol.cod_articol%TYPE DEFAULT 100, v_operatie IN number) RETURN varchar2;
    PROCEDURE exercitiul9(v_nume beneficiar.nume%TYPE, v_prenume beneficiar.prenume%TYPE);
END exercitiul13;
/
CREATE OR REPLACE PACKAGE BODY exercitiul13 IS
    PROCEDURE exercitiul6 (v_cod_sediu IN sediu.cod_sediu%TYPE)
    IS
        TYPE transport_record is RECORD (cod_articol articol.cod_articol%TYPE, cantitate articol_in_sediu.cantitate%TYPE);
        TYPE tablou_indexat_transporturi IS TABLE OF transport_record INDEX BY PLS_INTEGER;
        t_transporturi tablou_indexat_transporturi;
        v_pret articol_in_sediu.cantitate%TYPE;

        c_maxim_furnizori CONSTANT NUMBER := 1000;
        TYPE vector_furnizori IS VARRAY(c_maxim_furnizori) OF furnizor%ROWTYPE;
        t_furnizori vector_furnizori := vector_furnizori();

        TYPE produs_record is RECORD (cod_articol articol.cod_articol%TYPE, nume articol.nume%TYPE);
        TYPE tablou_imbricat_produse IS TABLE OF produs_record;
        t_produse tablou_imbricat_produse := tablou_imbricat_produse();

        TYPE sediu_record is RECORD (cod_sediu sediu.cod_sediu%TYPE, nume sediu.nume%TYPE, cantitate articol_in_sediu.cantitate%TYPE);
        TYPE tablou_indexat_sediil IS TABLE OF sediu_record INDEX BY BINARY_INTEGER;
        t_sediil tablou_indexat_sediil;
        v_cantitate_totala int;
    BEGIN
        t_transporturi(1) := transport_record(101, 100);
        t_transporturi(2) := transport_record(102, 120);
    END;

```

...sql | exercitiu100.sql | exercitiu11.sql | exercitiu8.sql | exercitiu13.sql | exercitiu9.sql | exercitiu6.sql | exercitiu7.sql | exercitiu10....

SQL Worksheet | History

Worksheet | Query Builder

```

END exercitiu13;
/
CREATE OR REPLACE PACKAGE BODY exercitiu13 IS
    PROCEDURE exercitiu6 (v_cod_sediu IN sediu.cod_sediu%TYPE)
    IS
        TYPE transport_record IS RECORD (cod_articol articol.cod_articol%TYPE, cantitate articol_in_sediu.cantitate%TYPE);
        TYPE tablou_indexat_transporturi IS TABLE OF transport_record INDEX BY PLS_INTEGER;
        t_transporturi tablou_indexat_transporturi;
        v_pret articol_in_sediu.cantitate%TYPE;

        c_maxim_furnizori CONSTANT NUMBER := 1000;
        TYPE vector_furnizori IS VARRAY(c_maxim_furnizori) OF furnizor%ROWTYPE;
        t_furnizori vector_furnizori := vector_furnizori();

        TYPE produs_record IS RECORD (cod_articol articol.cod_articol%TYPE, nume articol.nume%TYPE);
        TYPE tablou_imbricat_produse IS TABLE OF produs_record;
        t_produse tablou_imbricat_produse := tablou_imbricat_produse();

        TYPE sediu_record IS RECORD (cod_sediu sediu.cod_sediu%TYPE, nume sediu.nume%TYPE, cantitate articol_in_sediu.cantitate%TYPE);
        TYPE tablou_indexat_sedii IS TABLE OF sediu_record INDEX BY BINARY_INTEGER;
        t_sedii tablou_indexat_sedii;
        v_cantitate_totala int;
    BEGIN
        t_transporturi(1) := transport_record(101, 100);
        t_transporturi(2) := transport_record(102, 120);
        t_transporturi(3) := transport_record(105, 200);

        FOR i IN t_transporturi.FIRST..t_transporturi.LAST LOOP --iteram transporturile si le inseram in stocurile
            UPDATE articol_in_sediu
            SET cantitate = cantitate + t_transporturi(i).cantitate
        END LOOP;
    END;
END;
/

```

Script Output | Task completed in 0.074 seconds

Package BODY EXERCITIU13 compiled

...sql | exercitiu8.sql | exercitiu13.sql | exercitiu9.sql | exercitiu6.sql | exercitiu7.sql | exercitiu10.sql |

SQL Worksheet | History

Worksheet | Query Builder

```

END exercitiu13;
/
BEGIN
    exercitiu13.exercitiu16(100);
END;
/

```

Script Output x

| Task completed in 0.101 seconds

Package EXERCITIUL13 compiled

Package Body EXERCITIUL13 compiled

PL/SQL procedure successfully completed.

Dbms Output x Compiler - Log

+ | Buffer Size: 20000 |

conexiune\_noua x

```

Furnizorul Electronice high quality ne vinde 2 produs(e):
    Bec ultra, cantitati:
        sediulAtacul tehnologiei: 65 bucati
        sediulCable power: 75 bucati
        sediulELECTRO LIGHT MAIN: 75 bucati
        sediulElectric center: 270 bucati
        sediulIT Party: 0 bucati
    Cantitatea totala: 485 bucati
    Incarcator telefon, cantitati:
        sediulAtacul tehnologiei: 75 bucati
        sediulCable power: 85 bucati
        sediulELECTRO LIGHT MAIN: 85 bucati
        sediulElectric center: 320 bucati
        sediulIT Party: 90 bucati
    Cantitatea totala: 655 bucati
=====
Furnizorul Elecrtic party ne vinde 1 produs(e):
    Lanterna, cantitati:
        sediulAtacul tehnologiei: 50 bucati
        sediulCable power: 55 bucati
        sediulELECTRO LIGHT MAIN: 61 bucati

```

Saved: D:\facultate\II\SGBD\proiect SGBD\exercitiu8.sql | Lin

SQL Worksheet | History

Worksheet    Query Builder

```
BEGIN
    exercitiu13.exercitiu1();
END;
/
```

Script Output x

Task completed in 0.082 seconds

Package Body EXERCITIUL13 compiled

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x Compiler - Log

conexiune\_noua x

```
sediulELECTRO LIGHT MAIN: 50 bucăti
sediulElectric center: 45 bucăti
sediulIT Party: 0 bucăti
Cantitatea totală: 200 bucăti
=====
Furnizorul Furnizor Electronice nu furnizează produse

Coletul (care va fi livrat de curierul Todorescu Simon cod 200) cu codul 101 din comanda 100, are pachetele:
cod: 115, nume: Pachet techno, cantitate: 1, produse componente
    Incarcator telefon, cod:102
    Aspirator robot, cod:105
Coletul (care va fi livrat de curierul Cutas Mirel cod 220) cu codul 104 din comanda 101, are pachetele:
cod: 113, nume: Pachet electro, cantitate: 2, produse componente
    Lanterna, cod:103
    Frigider, cod:104
cod: 112, nume: Pachet Pro, cantitate: 2, produse componente
    Casti wireless, cod:100
    Bec ultra, cod:101
    Incarcator telefon, cod:102
```

The screenshot shows the Oracle SQL Developer interface with the following components:

- Worksheet Tab:** Displays a PL/SQL block. The code includes several BEGIN blocks, some containing function calls like `exercitiu13.exercitiu6(100)` or `exercitiu13.exercitiu7()`. It also contains a DBMS\_OUTPUT.PUT\_LINE statement and a call to `exercitiu13.exercitiu9`.
- Script Output Tab:** Shows the execution results:
  - PL/SQL procedure successfully completed.
  - PL/SQL procedure successfully completed.
- Dbsms Output Tab:** Shows the final output:

Rezultatul functiei: Articolul de tip produs apare ca premiu in 1 concurs(uri) din 2023, la care au participant in medie 4 persoane.  
Profitul total al concursurilor este 60 RON.

SQL Worksheet History

Worksheet Query Builder

```

END;
/
BEGIN
    DBMS_OUTPUT.PUT_LINE('Rezultatul functiei: ' || exercitiul3.exercitiul8(100, 2));
END;
/
BEGIN
    exercitiul3.exercitiul9('Andronic', 'Marcel');
END;
/

```

Script Output | Task completed in 0.117 seconds

PL/SQL procedure successfully completed.

Dbms Output Compiler - Log

Buffer Size: 20000

conexiune\_noua

```

Coletul (care va fi livrat de curierul Todorescu Simon cod 200) cu codul 101 din comanda 100, are pachetele:
    cod: 115, nume: Pachet techno, cantitate: 1, produse componente
        Incarcator telefon, cod:102
        Aspirator robot, cod:105
Coletul (care va fi livrat de curierul Cutas Mirel cod 220) cu codul 104 din comanda 101, are pachetele:
    cod: 113, nume: Pachet electro, cantitate: 2, produse componente
        Lanterna, cod:103
        Frigidex, cod:104
    cod: 112, nume: Pachet Pro, cantitate: 2, produse componente
        Casti wireless, cod:100
        Bec ultra, cod:101
        Incarcator telefon, cod:102

Rezultatul functiei: Articolul de tip produs apare ca premiu in 1 concurs(uri) din 2023, la care au participant in medie 4 persoane.
    Profitul total al concursurilor este 60 RON.

COMENZI:
cod: 103, nume: Lanterna, cantitate 4, detalii garantie: garantia expira pe 01-09-2025

```

Saved: D:\facultate\II\SGBD\project SGBD\exercitiul8.sql | Line 452 Column 2 | Insert | Windows: C

**14) Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).**

--Pentru prima functie:

--aflam mai intai ce a premiu a castigat fiecare si ce contine, plus punctajul necesar pt a-l primi si

--ce punctaj a obtinut concurrentul la concursul care ii ofera premiul

--si al cateleau premiu din concursul corespunzator e

CREATE OR REPLACE VIEW castiguri AS

(SELECT p.cod\_premiu as cod\_premiu, p.cod\_concurs as cod\_concurs, b.codBeneficiar as codBeneficiar,

nvl(p.suma, 0) as suma\_bani, produs.cod\_articol as cod\_articol, nvl(a.pret, 0) as pret\_produs\_castigat,

punctaj\_minim as punctaj\_cerut\_de\_premiu, punctaj as punctaj\_obtinut\_concurs,

(SELECT COUNT(\*) + 1 --numaram cate premii din concursul curent au punctaj necesar mai mare decat premiul castigat

FROM premiu p2, concurs c2

WHERE p2.cod\_concurs = c2.cod\_concurs AND c2.cod\_concurs = c.cod\_concurs

AND p2.punctaj\_minim > p.punctaj\_minim) as rang\_premiu\_in\_concurs

FROM premiu p, concurs c, participa pp, beneficiar b, produs, articol a

WHERE p.cod\_concurs = c.cod\_concurs AND c.cod\_concurs = pp.cod\_concurs

AND pp.codBeneficiar = b.codBeneficiar AND p.codArticol = produs.codArticol(+) --outer join pt ca e posibil ca premiul sa nu ofere produs

AND produs.codArticol = a.codArticol(+) --daca produsul e null si superentitatea lui va fi

AND pp.punctaj >= p.punctaj\_minim --pentru a vedea ce premiu a luat

AND p.punctaj\_minim = (SELECT(MAX(punctaj\_minim)) --punctajul maxim al unui premiu din concursul curent <= cat a obtinut concurrentul

FROM premiu p2

WHERE p2.cod\_concurs = p.cod\_concurs AND pp.punctaj >= p2.punctaj\_minim);

```
select * from castiguri;
```

```
CREATE OR REPLACE PACKAGE exercitiul14 IS
```

```
--prima functie:
```

```
    TYPE tablou_indexat_coduri IS TABLE OF number(10) INDEX BY PLS_INTEGER;
```

```
    TYPE castig_record IS RECORD --informatii despre fiecare premiu castigat de fiecare concurrent
```

```
        (cod_premiu int,  
         cod_concurs int,  
         cod_beneficiar int,  
         suma_bani int,  
         cod_articol int,  
         pret_produs_castigat int,  
         punctaj_cerut_de_premiu int,  
         punctaj_obtinut_concurs int,  
         rang_premiu_in_concurs int);
```

```
    TYPE tablou_indexat_castiguri IS TABLE OF castig_record INDEX BY PLS_INTEGER;
```

```
    FUNCTION premii_14(v_suma_maxima IN OUT number, t_coduri IN OUT  
        tablou_indexat_coduri) RETURN number;
```

```
--A doua functie:
```

```
    FUNCTION comparatie_pachet_produs_14 (v_cod_sediu IN sediu.cod_sediu%TYPE,  
        v_cod_pachet OUT pachet_promotional.cod_articol%TYPE)
```

```
    RETURN number;
```

```
--Prima procedura:
```

```
    TYPE articol_record IS RECORD (cod_articol int, procent number);
```

```
    TYPE tablou_imbricat_articole IS TABLE OF articol_record; --aici retinem fiecare articol si cat trebuie sa ii crestem pretul
```

```
    PROCEDURE recenzii_14(v_procent1 IN number, v_procent2 IN number);
```

```
--A doua procedura:  
PROCEDURE articole_comandate_14(v_nr_persoane IN number);  
END exercitiul14;  
/  
CREATE OR REPLACE PACKAGE BODY exercitiul14 IS
```

--Prima functie:

- Aflati pentru fiecare concurs castigatorii si ce premii au castigat.
- Se va afisa si al catelea premiu din concurs e (in functie de punctajul cerut pentru a primi premiul).
- Functia va returna numarul de premii castigate in total. Nu conteaza daca un beneficiar a castigat 2 premii.
- Pentru a castiga un premiu, beneficiarul trebuie sa aiba punctajul mai mare sau egal cu punctajul cerut de premiu.
- Se va lua cel mai mare premiu care indeplineste conditia.
- Sa se afle cate premii a castigat fiecare beneficiar
- Daca a participat si nu a castigat nimic se va specifica asta, la fel si daca nu a participat deloc.
- Functia va avea parametrii in out un tablou care retine beneficiarii care au castigat un premiu cu valoarea maxima
- si celalat parametru este acea valoare maxima
- cel mai scump premiu (suma de pret produs si bani dintr-un premiu) si codul celui care l-a castigat..
- Se presupune ca premiile din cadrul aceluiasi concurs au punctaje necesare distincte.

```
FUNCTION premii_14 (v_suma_maxima IN OUT number, t_coduri IN OUT  
tablou_indexat_coduri)  
RETURN number  
IS  
    v_nr_premii_castigate number(8):= 0;  
    t_castiguri tablou_indexat_castiguri;  
  
    --aflam pt fiecare beneficiar daca a participat si cate concursuri a castigat
```

```

CURSOR c_situatieBeneficiari IS
    SELECT b.codBeneficiar AS codBeneficiar,
        (CASE
            WHEN COUNT(c.codBeneficiar) > 0 -- cate concursuri a castigat
                THEN ('a castigat ' || COUNT(DISTINCT c.codConcurs) || '
concursuri') -- Punem distinct pt ca daca a participat la mai multe concursuri se va multiplica
                informatia
            WHEN COUNT(p.codConcurs) > 0
                THEN ('nu a castigat nimic dar a participat la ' || COUNT(p.codConcurs) || ' concursuri')
            ELSE 'nu a participat la concursuri'
        END) AS situatieBeneficiar
    FROM beneficiar b, participa p, castiguri c
    WHERE b.codBeneficiar = p.codBeneficiar(+) AND p.codBeneficiar =
        c.codBeneficiar(+)
    GROUP BY b.codBeneficiar;

```

```

BEGIN
    -- luam datele din view si le punem in tablou:
    SELECT *
    BULK COLLECT INTO t_castiguri
    FROM castiguri;

    FOR i IN t_castiguri.FIRST..t_castiguri.LAST LOOP
        DBMS_OUTPUT.PUT_LINE('Beneficiarul ' || t_castiguri(i).codBeneficiar || ' a
        castigat premiul ' || t_castiguri(i).codPremiu
        || ' din cadrul concursului ' || t_castiguri(i).codConcurs || ', obtinand ' ||
        t_castiguri(i).punctajObtinutConcurs || ' puncte');

        DBMS_OUTPUT.PUT_LINE('Premiul are punctajul minim necesar ' ||
        t_castiguri(i).punctajCerutDePremiu ||
        ' si are rangul ' || t_castiguri(i).rangPremiuInConcurs);

    END LOOP;

```

--Aflam cate premii s-au castigat in total folosindu-ne de view

```
SELECT COUNT(*)
```

```
INTO v_nr_premii_castigate
```

```
FROM castiguri;
```

--aflam cel mai scump premiu

```
SELECT MAX(suma_bani + pret_produs_castigat)
```

```
INTO v_suma_maxima
```

```
FROM castiguri;
```

--aflam beneficiarii care l-au castigat:

```
SELECT codBeneficiar
```

```
BULK COLLECT INTO t_coduri
```

```
FROM castiguri
```

```
WHERE suma_bani + pret_produs_castigat = v_suma_maxima;
```

FOR i IN c\_situatieBeneficiari LOOP --afisam datele din cursor

```
EXIT WHEN c_situatieBeneficiari%NOTFOUND;
```

```
DBMS_OUTPUT.PUT_LINE('Beneficiarul ' || i.codBeneficiar ||  
i.situatieBeneficiari || '');
```

```
END LOOP;
```

```
RETURN v_nr_premii_castigate;
```

```
END premii_14;
```

--Prima

procedura=====

--Sa se creasca cu un anume procent pretul articolelor care au cel putin o recenzie printre primele 2 cele mai mari note distincte

--Si cu alt procent daca e pe locurile 3-5.

--Procentul va fi dat ca parametru

```
PROCEDURE recenzi_14(v_procent1 IN number, v_procent2 IN number)
```

```
IS
```

CURSOR c\_recenzi\_bune IS --articolele carora trebuie sa li se creasca pretul si cu cat(in functie de recenzi)

```
WITH note_ordonate AS --luam primele 5 note distincte
```

```
(SELECT note.nota, rownum indice
```

```
FROM
```

```
(SELECT DISTINCT nota
```

```
FROM recenzie r
```

```
ORDER BY nota DESC) note
```

```
WHERE rownum <= 5)
```

SELECT a.cod\_articol cod\_articol, -- aflam care articole trebuie sa aiba pretul crescut si cu cat

```
(CASE
```

WHEN EXISTS (SELECT 1 --daca exista o recenzie a produsului care sa aiba nota in primele 2 inregistrari

```
FROM recenzie r, note_ordonate n
```

```
WHERE r.cod_articol = a.cod_articol and r.nota = n.nota and  
n.indice <= 2)
```

```
THEN v_procent1
```

```
WHEN EXISTS (SELECT 1
```

```
FROM recenzie r, note_ordonate n
```

```
WHERE r.cod_articol = a.cod_articol and r.nota = n.nota and  
n.indice BETWEEN 3 AND 5)
```

```
THEN v_procent2
```

```
END) as procent_crestere
```

```

FROM articol a
WHERE EXISTS (SELECT 1 --ca sa nu afiseze si articole fara recenzii bune
              FROM recenzie r, note_ordonate n
              WHERE r.cod_articol = a.cod_articol and r.nota = n.nota);

v_pret_initial number(8, 2);
v_pret_final number(8, 2);
v_cod_recenzie int;
v_nota number(2);

BEGIN

FOR i IN c_recenzii_bune LOOP
    EXIT WHEN c_recenzii_bune%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Articolul ' || i.cod_articol || ' va avea pretul mai mare cu '
    || i.procent_crestere || '%');

    SELECT pret
    INTO v_pret_initial
    FROM articol
    WHERE cod_articol = i.cod_articol;

    v_pret_final := v_pret_initial * (1 + i.procent_crestere / 100);
    DBMS_OUTPUT.PUT_LINE('    Pret initial: ' || v_pret_initial || ', pret final: ' ||
    v_pret_final);

    UPDATE articol
    SET pret = v_pret_final
    WHERE cod_articol = i.cod_articol;

--afisam pentru articolul curent una din recenziile cu nota cea mai mare
SELECT cod_recenzie, nota

```

```

INTO v_cod_recenzie, v_nota
FROM recenzie
WHERE cod_articol = i.cod_articol
AND nota = (SELECT MAX(nota) -- vedem daca nota e egala cu nota maxima a
recenziilor articoului
FROM recenzie
WHERE cod_articol = i.cod_articol)
AND ROWNUM = 1; ---luam doar prima inregistrare generata

DBMS_OUTPUT.PUT_LINE(' Una dintre recenziile cu nota cea mai mare pentru
articol este ' || v_cod_recenzie ||
' cu nota ' || v_nota);
END LOOP;
END recenzi_14;

```

--A doua  
procedura=====

--Sa se afle articolele comandate de cel putin n persoane diferite. N este dat ca parametru  
--Pentru aceste produse sa se afle exact cate persoane le au comandat, cate bucati au fost in  
total (o persoana poate cumpara mai multe)  
--si sedile diferite din care a plecat/va pleca cel putin un colet care contin acel articol  
--de asemenea, si curierii care au livrat cel putin un astfel de colet.

PROCEDURE articole\_comandate\_14(v\_nr\_persoane IN number)

IS

v\_gasit boolean;

CURSOR c\_articole IS --retinem pt fiecare articol cumparat de minim n persoane cate  
persoane l au cumparat, si cate bucati au fost in total

WITH beneficiar\_articol AS (

```

    SELECT b.cod_beneficiar cod_beneficiar, aic.cod_articol cod_articol, --aflam
mai intai cat din fiecare articol a comandat cineva in total(din mai multe comenzi sau pachete)
        SUM(aic.cantitate) as cantitate
        FROM beneficiar b, detalii_tranzactie dt, comanda c, colet cc, articol_in_colet aic
        WHERE b.cod_beneficiar = dt.cod_beneficiar AND dt.cod_detalii_tranzactie =
c.cod_detalii_tranzactie
            AND c.cod_comanda = cc.cod_comanda AND cc.cod_colet = aic.cod_colet
            AND cc.cod_comanda = aic.cod_comanda
        GROUP BY b.cod_beneficiar, aic.cod_articol)

```

```

    SELECT ba.cod_articol as cod_articol, COUNT(ba.cod_beneficiar) as
nr_persoane_distincte,
        SUM(cantitate) as cantitate_totala
        FROM beneficiar_articol ba
        GROUP BY ba.cod_articol
        HAVING COUNT(ba.cod_beneficiar) >= v_nr_persoane; --iau linia doar daca articolul
a fost cumparat de minim n pers

```

```

BEGIN
    FOR i IN c_articole LOOP
        --EXIT WHEN c_articole%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Articolul ' || i.cod_articol || ' a fost comandat de ' ||
i.nr_persoane_distincte || ' persoane diferite in cantitatea totala de '
|| i.cantitate_totala || ' bucati. Sediile din care pleaca cel putin un colet care contine
articolul sunt: ');
        DBMS_OUTPUT.PUT('');
        --aflam sediile din care pleaca cel putin un colet cu articolul curent:
        FOR j IN (SELECT s.cod_sediu
        FROM sediu s
        WHERE EXISTS (SELECT 1 --sediile unde exista cel putin un colet care
contine articolul

```

```

        FROM colet c, articol_in_colet aic
        WHERE c.cod_colet = aic.cod_colet and c.cod_comanda =
aic.cod_comanda
                and aic.cod_articol = i.cod_articol and c.cod_sediu =
s.cod_sediu)
)
LOOP

        DBMS_OUTPUT.PUT(j.cod_sediu || ' ');

END LOOP;

        DBMS_OUTPUT.PUT_LINE("");
        DBMS_OUTPUT.PUT('Curierii care livreaza cel putin un astfel de colet sunt: ');

--gasim curierii care au livrat/vor livra cel putin un astfel de colet (calculam separat de
sedii pt ca dintr un sediu pot pleca mai multe colete

v_gasit := FALSE; -- pentru a afisa dupa un mesaj specific daca nu gasim curier(e
posibil sa fie colete care nu au fost asignate unui curier)

FOR j IN (SELECT cr.cod_angajat as cod_curier
        FROM curier cr
        WHERE EXISTS (SELECT 1 --curierii care livreaza cel putin un colet care
contine articolul
                FROM colet c, articol_in_colet aic
                WHERE c.cod_colet = aic.cod_colet and c.cod_comanda =
aic.cod_comanda
                        and aic.cod_articol = i.cod_articol and c.cod_angajat =
cr.cod_angajat)
)
LOOP

        DBMS_OUTPUT.PUT(j.cod_curier || ' ');

```

```

    v_gasit := TRUE; --am gasit cel putin un curier

    END LOOP;

    IF v_gasit = FALSE
        THEN DBMS_OUTPUT.PUT('Niciun colet nu a fost repartizat inca la un curier');
    END IF;

    DBMS_OUTPUT.PUT_LINE(");

END LOOP;

END articole_comandate_14;

```

--A doua functie

---



---

--Pentru un cod\_sediu dat, sa se compare pt toate pachetele pretul lor cu suma preturilor produselor componente.

--Daca vreun articol nu se gaseste in sediu se va lua pretul universal, din tabela articol.

--Sa se afle de asemenea pentru fiecare pachet cate produse comune are cu fiecare pachet.

--Sa se afle pachetul cu cea mai mare diferență intre pretul lui și suma preturilor produselor componente.

--Functia va avea parametru out cu codul acestui pachet, și va returna diferența maxima

```

FUNCTION comparatie_pachet_produc_14 (v_cod_sediu IN sediu.cod_sediu%TYPE,
v_cod_pachet OUT pachet_promotional.cod_articol%TYPE)
IS
    RETURN number

```

```
v_diferenta_maxima number(10):= 0;  
v_diferenta_curenta number(10); --pentru a calcula diferența maxima
```

CURSOR c\_pachet\_produse IS--cursor care retine pt fiecare pachet pretul lui si pretul adunat al produselor componente cu preturi din sediul parametru

```
WITH preturi_pachete AS --mai intai retinem doar pachetele si pretul lor  
(SELECT pp.cod_articol cod_pachet,  
       (SELECT NVL(ais.pret, a2.pret) -- daca nu gasim stocul in sediu luam pretul  
        din tabela articol  
         FROM articol a2, articol_in_sediu ais  
        WHERE pp.cod_articol = a2.cod_articol and a2.cod_articol =  
              ais.cod_articol(+)) -- e posibil sa nu se gaseasca in sediu stocul  
        and ais.cod_sediu(+) = v_cod_sediu)  
        as pret_pachet  
        FROM pachet_promotional pp)
```

```
SELECT pp.cod_pachet as cod_pachet, pp.pret_pachet as pret_pachet,  
       SUM (NVL(ais.pret, a.pret)) as suma_preturi_produse  
        FROM preturi_pachete pp, produs_in_pachet pip, articol a, articol_in_sediu  
              ais--pentru fiecare produs al pachetului trb sa vedem daca e in stoc  
        WHERE pp.cod_pachet = pip.cod_pachet and pip.cod_produs = a.cod_articol and  
              a.cod_articol = ais.cod_articol(+)  
        and ais.cod_sediu(+) = v_cod_sediu --trb sa fie din sediul parametru  
        GROUP BY pp.cod_pachet, pp.pret_pachet;
```

CURSOR c\_produse\_comune IS --aflam pt fiecare pachet cate produse comune are cu celelalte

--facem produs cartezian intre 2 pachete si produsele lor si luam inregistrarile unde corespund cele 2 produse componente

```
SELECT pp1.cod_articol cod_pachet1, pp2.cod_articol cod_pachet2,  
       COUNT(pip1.cod_produs) nr_produse_comune
```

```
FROM pachet_promotional pp1, produs_in_pachet pip1, pachet_promotional pp2,
produs_in_pachet pip2

WHERE pp1.cod_articol = pip1.cod_pachet and pp2.cod_articol = pip2.cod_pachet
and pp1.cod_articol < pp2.cod_articol and pip1.cod_produs = pip2.cod_produs --
semnul < pt a afisa doar o data perechea

GROUP BY pp1.cod_articol, pp2.cod_articol;
```

BEGIN

--vom parcurge ciclul cursorul, afisam rezultatele si calculam diferența maxima

FOR i IN c\_pachet\_produse LOOP

EXIT WHEN c\_pachet\_produse%NOTFOUND;

v\_diferenta\_curenta := ABS(i.pret\_pachet - i.suma\_preturi\_produse);

```
DBMS_OUTPUT.PUT_LINE('Pachetul promotional ' || i.cod_pachet || ' are pretul ' ||
i.pret_pachet ||
```

```
' si suma preturilor produselor ' || i.suma_preturi_produse || ', diferența fiind ' ||
v_diferenta_curenta);
```

IF v\_diferenta\_curenta > v\_diferenta\_maxima

THEN v\_diferenta\_maxima := v\_diferenta\_curenta;

v\_cod\_pachet := i.cod\_pachet;

END IF;

END LOOP;

--afisam produsele comune:

FOR i IN c\_produse\_comune LOOP

```
DBMS_OUTPUT.PUT_LINE('Pachetul ' || i.cod_pachet1 || ' are ' ||
i.nr_produse_comune || ' produs(e) comun(e) cu pachetul ' || i.cod_pachet2);
```

```
    END LOOP;

    RETURN v_diferenta_maxima;
END comparatie_pachet_produs_14;

END exercitiul14;
/
```

--A doua functie:

----PACHET\_PRODUS

```
truncate table articol_in_sediu; --stergem ce e in sediu pentru a demonstra ca se va lua pretul universal in acest caz
```

DECLARE

```
    v_diferenta_maxima number(10);
```

```
    v_cod_pachet int;
```

BEGIN

```
    v_diferenta_maxima := exercitiul14.comparatie_pachet_produs_14(100, v_cod_pachet);
```

```
    DBMS_OUTPUT.PUT_LINE('Diferenta maxima pret pachet <=> suma preturi produse este ' || v_diferenta_maxima || ' RON.');
```

```
    DBMS_OUTPUT.PUT_LINE('Pachetul promotional cu aceasta diferență maxima este ' || v_cod_pachet);
```

END;

/

--A doua procedura:

--ARTICOLE COMANDATE:

BEGIN

```
    exercitiul14.articole_comandate_14(2);
```

END;

/

rollback;

```

--Prima procedura:
--RECENZII
DECLARE
    v_procent1 number(3);
    v_procent2 number(3);
BEGIN
    exercitiul14.recenzi_14(20, 10);
END;
/
--Prima functie:
--PREMII
DECLARE
    v_suma_maxima number(10);
    t_coduri exercitiul14.tablou_indexat_coduri;
    v_numar_castiguri number(8);
BEGIN
    --inseram o inregistrare pentru a avea un caz in care un beneficiar a participat dar nu a castigat nimic.
    --insert into participa (cod_concurs, cod_beneficiar, punctaj) values (100, 150, 20);

    v_numar_castiguri := exercitiul14.premii_14(v_suma_maxima, t_coduri);
    DBMS_OUTPUT.PUT_LINE('In total au fost ' || v_numar_castiguri || ' premii castigate ');

    DBMS_OUTPUT.PUT_LINE('Cel mai scump premiu are valoarea de ' || v_suma_maxima || ' si a fost castigat de beneficiarii cu codurile: ');
    FOR i IN t_coduri.FIRST..t_coduri.LAST LOOP
        DBMS_OUTPUT.PUT_LINE('      ' || t_coduri(i));
    END LOOP;
    delete from participa where cod_concurs = 100 and cod_beneficiar = 150;

```

END;

/

### View pentru functia premii\_14 :

The screenshot shows the Oracle SQL Developer interface. In the top-left corner, there's a toolbar with various icons. Below it, the main window has tabs for 'Worksheet' and 'Query Builder'. The 'Worksheet' tab is active, displaying the following PL/SQL code:

```
--aflam mai intai ce a castigat fiecare si ce contine, plus punctajul necesar pt a-l primi si
--ce punctaj a obtinut concurrent la concursul care ii ofera premiul
--si al cateleau premiu din concursul corespunzator e
CREATE OR REPLACE VIEW castiguri AS
  (SELECT p.cod_premiu as cod_premiu, p.cod_concurs as cod_concurs, b.cod_beneficiar as cod_beneficiar,
    nvl(p.suma, 0) as suma_bani, produs.cod_articol as cod_articol, nvl(a.pret, 0) as pret_produs_castigat,
    punctaj_minim as punctaj_cerut_de_premiu, punctaj as punctaj_obtinut_concurs,
    (SELECT COUNT(*) + 1 --numaram cate premii din concursul curent au punctaj necesar mai mare decat premiul castigat
     FROM premiu p2, concurs c2
    WHERE p2.cod_concurs = c2.cod_concurs AND c2.cod_concurs = p.cod_concurs
      AND p2.punctaj_minim > p.punctaj_minim) as rang_premiu_in_concurs
   FROM premiu p, concurs c, participa pp, beneficiar b, produs, articol a
  WHERE p.cod_concurs = c.cod_concurs AND c.cod_concurs = pp.cod_concurs
    AND pp.cod_beneficiar = b.cod_beneficiar AND p.cod_articol = produs.cod_articol(+) -- outer join pt ca e posibil ca produs sa nu exista
    AND produs.cod_articol = a.cod_articol(+) --daca produsul e null si superentitatea lui va fi
    AND pp.punctaj >= p.punctaj_minim --pentru a vedea ce premiu a luat
    AND p.punctaj_minim = (SELECT MAX(punctaj_minim)
                           FROM premiu p2
                          WHERE p2.cod_concurs = p.cod_concurs AND pp.punctaj >= p2.punctaj_minim));
select * from castiguri;
```

Below the code editor, there's a status bar showing 'Task completed in 0.156 seconds'. At the bottom, there's a 'Compiler - Log' tab.

Oracle SQL Developer : D:\facultate\II\SGBD\project SGBD\exercitiul14.sql

```

File Edit View Navigate Run Source Team Tools Window Help
SQL Worksheet History
Worksheet Query Builder
    AND pp.punctaj_minim --pentru a vedea ce premiu a luat
    AND pp.punctaj_minim = (SELECT MAX(punctaj_minim) --punctajul maxim al unui premiu din concursul curent <= cat a obtinut concurrentul
                           FROM premiu p2
                           WHERE p2.cod_concurs = p.cod_concurs AND pp.punctaj >= p2.punctaj_minim);

select * from castiguri;

CREATE OR REPLACE PACKAGE exercitiul14 IS
    --prima functie:
    TYPE tablou_indexat_coduri IS TABLE OF number(10) INDEX BY PLS_INTEGER;
    TYPE castig_record IS RECORD --informatii despre fiecare premiu castigat de fiecare concurrent
        (cod_premiu int,

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | All Rows Fetched: 13 in 0.179 seconds

	COD_PREMIU	COD_CONCURS	COD_BENEFICIAR	SUMA_BANI	COD_ARTICOL	PRET_PRODUS_CASTIGAT	PUNCTAJ_CERUT_DE_PREMIU	PUNCTAJ_OBTINUT_CONCURS	RANG_PREMIU_IN_CONCURS
1	110	100	130	30	100	100	70	74	2
2	170	130	110	0	101	120	80	90	2
3	160	120	100	20	102	300	75	97	1
4	100	100	100	20	103	400	60	66	3
5	140	110	100	35	103	400	90	92	1
6	140	110	110	35	103	400	90	94	1
7	200	140	130	10	104	100	50	74	2
8	200	140	100	10	104	100	50	99	2
9	180	130	100	30	105	80	99	99	1
10	120	100	140	40	108	120	90	95	1
11	190	140	140	0	111	70	100	100	1
12	130	110	130	30	(null)	0	80	88	2
13	150	120	110	210	(null)	0	70	73	2

File Edit View Navigate Run Source Team Tools Window Help

SQL Worksheet History

Worksheet Query Builder

```

        WHERE p2.cod_concurs = p.cod_concurs AND pp.punctaj >= p2.punctaj_minim));

select * from castiguri;

CREATE OR REPLACE PACKAGE exercitiul14 IS
    --prima functie:
    TYPE tablou_indexat_coduri IS TABLE OF number(10) INDEX BY PLS_INTEGER;
    TYPE castig_record IS RECORD --informatii despre fiecare premiu castigat de fiecare concurrent
        (cod_premiu int,
        cod_concurs int,
        cod_beneficiar int,
        suma_bani int,
        cod_articol int,
        pret_produs_castigat int,
        punctaj_cerut_de_premiu int,
        punctaj_obtinut_concurs int,
        rang_premiu_in_concurs int);
    TYPE tablou_indexat_castiguri IS TABLE OF castig_record INDEX BY PLS_INTEGER;

    FUNCTION premii_14(v_suma_maxima IN OUT number, t_coduri IN OUT tablou_indexat_coduri) RETURN number;

    --A doua functie:
    FUNCTION comparatie_pachet_produs_14 (v_cod_sediu IN sediu.cod_sediu%TYPE, v_cod_pachet OUT pachet_promotional.cod_articol%TYPE)
    RETURN number;
    --Prima procedura:
    TYPE articol_record IS RECORD (cod_articol int, procent number);
    TYPE tablou_imbricat_articole IS TABLE OF articol_record; --aici retinem fiecare articol si cu cat trebuie sa ii crestem pretul
    PROCEDURE recenzii_14(v_procentul IN number, v_procent2 IN number);

    --A doua procedura:
    PROCEDURE articole_comandate_14(v_nr_persoane IN number);
END exercitiul14;
/
```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | Task completed in 0.319 seconds

Package EXERCITIUL14 compiled

Compiler - Log

```

PROCEDURE articole_comandate_14(v_nr_persoane IN number);
END exercitiu14;
/
CREATE OR REPLACE PACKAGE BODY exercitiu14 IS
--Prima functie:
--Aflati pentru fiecare concurs castigatorii si ce premii au castigat.
--Se va afisa si al catelea premiu din concurs e (in functie de punctajul cerut pentru a primi premiul).
--Functia va returna numarul de premii castigate in total. Nu conteaza daca un beneficiar a castigat 2 premii.
--Pentru a castiga un premiu, beneficiarul trebuie sa aiba punctajul mai mare sau egal cu punctajul cerut de premiu.
--Se va lua cel mai mare premiu care indeplineste conditia.
--Sa se afle cate premii a castigat fiecare beneficiar
--Daca a participat si nu a castigat nimic se va specifica asta, la fel si daca nu a participat deloc.
--Functia va avea parametrii in out un tablou care retine beneficiarii care au castigat un premiu cu valoarea maxima
--si celalalt parametru este acea valoare maxima
--cel mai scump premiu (suma de pret produs si bani dintr-un premiu) si codul celui care l-a castigat..
--Se presupune ca premiile din cadrul aceluiasi concurs au punctaje necesare distincte.
FUNCTION premii_14 (v_suma_maxima IN OUT number, t_coduri IN OUT tablou_indexat_coduri)
    RETURN number
IS
    v_nr_premii_castigate number(8):= 0;
    t_castiguri tablou_indexat_castiguri;

    --aflam pt fiecare beneficiar daca a participat si cate concursuri a castigat
    CURSOR c_situatie_beneficiari IS
        SELECT b.cod_beneficiar as cod_beneficiar,
        (CASE
            WHEN COUNT(c.cod_beneficiar) > 0 --cate concursuri a castigat
                THEN ('a castigat ' || COUNT(DISTINCT c.cod_concurs) || ' concursuri') --Punem distinct pt ca daca a participat in doua concursuri, sa nu se numere de doua ori
            WHEN COUNT(p.cod_concurs) > 0
                THEN ('nu a castigat nimic dar a participat la ' || COUNT(p.cod_concurs) || ' concursuri')
            ELSE 'nu a participat la concursuri'
        END) AS situatie_beneficiar

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | Task completed in 0.268 seconds

Package Body EXERCITIUL14 compiled

## Functia premii\_14 :

SQL Worksheet | History

Worksheet | Query Builder

```
--Prima functie:
--PREMII
DECLARE
    v_suma_maxima number(10);
    t_coduri exercitiul4.tablou_indexat_coduri;
    v_numar_castiguri number(8);
BEGIN
    --inseram o inregistrare pentru a avea un caz in care un beneficiar a participat dar nu a castigat nimic.
    --insert into participa (cod_concurs, cod_beneficiar, punctaj) values (100, 150, 20);

    v_numar_castiguri := exercitiul4.premii_14(v_suma_maxima, t_coduri);
    DBMS_OUTPUT.PUT_LINE('In total au fost ' || v_numar_castiguri || ' premii castigate ');

    DBMS_OUTPUT.PUT_LINE('Cel mai scump premiu are valoarea de ' || v_suma_maxima || ' si a fost castigat de beneficiarii cu codurile: ');
    FOR i IN t_coduri.FIRST..t_coduri.LAST LOOP
        DBMS_OUTPUT.PUT_LINE('      ' || t_coduri(i));
    END LOOP;

    delete from participa where cod_concurs = 100 and cod_beneficiar = 150;
END;
/

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | Task completed in 0.746 seconds

PL/SQL procedure successfully completed.

Compiler - Log | Messages | Logging Page | Statements | Compiler

Dbms Output | Buffer Size: 20000 | conexiune\_noua

```
Beneficiarul 130 a castigat premiul 110 din cadrul concursului 100, obtinand 74 puncte
Premiul are punctajul minim necesar 70 si are rangul 2
Beneficiarul 110 a castigat premiul 170 din cadrul concursului 130, obtinand 90 puncte
Premiul are punctajul minim necesar 80 si are rangul 2
Beneficiarul 100 a castigat premiul 160 din cadrul concursului 120, obtinand 97 puncte
Premiul are punctajul minim necesar 75 si are rangul 1
Beneficiarul 100 a castigat premiul 100 din cadrul concursului 100, obtinand 66 puncte
Premiul are punctajul minim necesar 60 si are rangul 3
Beneficiarul 100 a castigat premiul 140 din cadrul concursului 110, obtinand 92 puncte
Premiul are punctajul minim necesar 90 si are rangul 1
```

### Intregul output :

Beneficiarul 130 a castigat premiul 110 din cadrul concursului 100, obtinand 74 puncte

Premiul are punctajul minim necesar 70 si are rangul 2

Beneficiarul 110 a castigat premiul 170 din cadrul concursului 130, obtinand 90 puncte

Premiul are punctajul minim necesar 80 si are rangul 2

Beneficiarul 100 a castigat premiul 160 din cadrul concursului 120, obtinand 97 puncte

Premiul are punctajul minim necesar 75 si are rangul 1

Beneficiarul 100 a castigat premiul 100 din cadrul concursului 100, obtinand 66 puncte

Premiul are punctajul minim necesar 60 si are rangul 3

Beneficiarul 100 a castigat premiul 140 din cadrul concursului 110, obtinand 92 puncte

Premiul are punctajul minim necesar 90 si are rangul 1

Beneficiarul 110 a castigat premiul 140 din cadrul concursului 110, obtinand 94 puncte

Premiul are punctajul minim necesar 90 si are rangul 1

Beneficiarul 130 a castigat premiul 200 din cadrul concursului 140, obtinand 74 puncte

Premiul are punctajul minim necesar 50 si are rangul 2

Beneficiarul 100 a castigat premiul 200 din cadrul concursului 140, obtinand 99 puncte

Premiul are punctajul minim necesar 50 si are rangul 2

Beneficiarul 100 a castigat premiul 180 din cadrul concursului 130, obtinand 99 puncte

Premiul are punctajul minim necesar 99 si are rangul 1

Beneficiarul 140 a castigat premiul 120 din cadrul concursului 100, obtinand 95 puncte

Premiul are punctajul minim necesar 90 si are rangul 1

Beneficiarul 140 a castigat premiul 190 din cadrul concursului 140, obtinand 100 puncte

Premiul are punctajul minim necesar 100 si are rangul 1

Beneficiarul 130 a castigat premiul 130 din cadrul concursului 110, obtinand 88 puncte

Premiul are punctajul minim necesar 80 si are rangul 2

Beneficiarul 110 a castigat premiul 150 din cadrul concursului 120, obtinand 73 puncte

Premiul are punctajul minim necesar 70 si are rangul 2

Beneficiarul 100a castigat 5 concursuri

Beneficiarul 130a castigat 3 concursuri

Beneficiarul 140a castigat 2 concursuri

Beneficiarul 110a castigat 3 concursuri

Beneficiarul 120nu a participat la concursuri

Beneficiarul 150nu a participat la concursuri

Beneficiarul 160nu a participat la concursuri

In total au fost 13 premii castigate

Cel mai scump premiu are valoarea de 435 si a fost castigat de beneficiarii cu codurile:

100

110

## Procedura recenzi\_14 :

```

    .TOOLBOX;
    --Prima procedura:
    --RECENZII
    --preturile inainte:
    select * from articol;
    DECLARE
        v_procent1 number(3);
        v_procent2 number(3);

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | All Rows Fetched: 17 in 0.09 seconds

COD_ARTICOL	PRET	NUME	PUNCTE_FIDELITATE
1	100	100 Casti wireless	12
2	101	120 Bec ultra	24
3	102	300 Incarcator telefon	13
4	103	400 Lanterna	18
5	104	100 Frigider	19
6	105	80 Aspirator robot	25
7	106	10 Fier de calcat	0
8	107	100 Uscator de par	28
9	108	120 Uscator de par Julie Pro	15
10	109	90 Telefon	10
11	110	80 Laptop	0
12	111	70 Tableta	3
13	112	1000 Pachet Pro	28
14	113	800 Pachet electro	28
15	114	1000 Pachet super	29
16	115	5000 Pachet techno	30
17	116	8000 Pachet all	30

SQL Worksheet History

Worksheet | Query Builder

```

    --RECENZII
    --preturile inainte:
    select * from articol;
    DECLARE
        v_procent1 number(3);
        v_procent2 number(3);
    BEGIN
        exercitiul14.recenzi_14(20, 10);
    END;
    /
--Prima functie:
--PREMII

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | Task completed in 0.245 seconds

PL/SQL procedure successfully completed.

Obms Output | Buffer Size: 20000

conexiune\_noua

```

Articolul 100 va avea pretul mai mare cu 20%
    Pret initial: 100, pret final: 120
    Una dintre recenziile cu nota cea mai mare pentru articol este 101 cu nota 10
Articolul 101 va avea pretul mai mare cu 20%
    Pret initial: 120, pret final: 144
    Una dintre recenziile cu nota cea mai mare pentru articol este 102 cu nota 10
Articolul 113 va avea pretul mai mare cu 10%
    Pret initial: 800, pret final: 880
    Una dintre recenziile cu nota cea mai mare pentru articol este 103 cu nota 7
Articolul 104 va avea pretul mai mare cu 10%
    Pret initial: 100, pret final: 110
    Una dintre recenziile cu nota cea mai mare pentru articol este 105 cu nota 6
Articolul 105 va avea pretul mai mare cu 10%
    Pret initial: 80, pret final: 88
    Una dintre recenziile cu nota cea mai mare pentru articol este 106 cu nota 8
Articolul 114 va avea pretul mai mare cu 20%
    Pret initial: 1000, pret final: 1200
    Una dintre recenziile cu nota cea mai mare pentru articol este 107 cu nota 9
Articolul 109 va avea pretul mai mare cu 10%
    Pret initial: 90, pret final: 99
    Una dintre recenziile cu nota cea mai mare pentru articol este 108 cu nota 7

```

### Preturile după :

```
--RECENZII
--preturile inainte:
select * from articol;
DECLARE
    v_procent1 number(3);
    v_procent2 number(3);
BEGIN
    exercitiu14.recenzii_14(20, 10);
END;
```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result

SQL | All Rows Fetched: 17 in 0.011 seconds

COD_ARTICOL	PRET	NUME	PUNCTE_FIDELITATE
1	100	120 Casti wireless	12
2	101	144 Bec ultra	24
3	102	300 Incarcator telefon	13
4	103	400 Lanterna	18
5	104	110 Frigider	19
6	105	88 Aspirator robot	25
7	106	10 Fier de calcat	0
8	107	100 Uscator de par	28
9	108	120 Uscator de par Julie Pro	15
10	109	99 Telefon	10
11	110	80 Laptop	0
12	111	70 Tableta	3
13	112	1000 Pachet Pro	28
14	113	880 Pachet electro	28
15	114	1200 Pachet super	29
16	115	5000 Pachet techno	30
17	116	8000 Pachet all	30

## Functia comparatie\_pachet\_produs\_14

1) Cazul cand sunt articole in sediu si se ia pretul de acolo (sediul 100 dat ca parametru) :

```
--A doua functie:  
----PACHET_PRODUS  
truncate table articol_in_sediu; --stergem ce e in sediu pentru a demonstra ca se va lua pretul universal in acest caz  
DECLARE  
    v_diferenta_maxima number(10);  
    v_cod_pachet int;  
BEGIN  
    v_diferenta_maxima := exercitiul4.comparatie_pachet_produs_14(100, v_cod_pachet);  
    DBMS_OUTPUT.PUT_LINE('Diferenta maxima pret pachet <=> suma preturilor produse este ' || v_diferenta_maxima || ' RON.');//  
    DBMS_OUTPUT.PUT_LINE('Pachetul promotional cu aceasta diferență maxima este ' || v_cod_pachet);  
END;  
--A doua procedura:  
--ARTICOLE COMANDATE:
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x | Query Result 5 x | Query Result 6 x |  
Task completed in 0.079 seconds

PL/SQL procedure successfully completed.

Dbms Output x  
+ | Buffer Size: 20000 |  
conexiune\_noua x

```
Pachetul promotional 112 are pretul 980 si suma preturilor produselor 510, diferența fiind 470  
Pachetul promotional 114 are pretul 1020 si suma preturilor produselor 859, diferența fiind 161  
Pachetul promotional 115 are pretul 5100 si suma preturilor produselor 370, diferența fiind 4730  
Pachetul promotional 113 are pretul 800 si suma preturilor produselor 805, diferența fiind 5  
Pachetul promotional 116 are pretul 8100 si suma preturilor produselor 527, diferența fiind 7573  
Pachetul 112 are 1 produs(e) comun(e) cu pachetul 114  
Pachetul 112 are 1 produs(e) comun(e) cu pachetul 115  
Pachetul 113 are 1 produs(e) comun(e) cu pachetul 116  
Pachetul 114 are 1 produs(e) comun(e) cu pachetul 115  
Pachetul 114 are 1 produs(e) comun(e) cu pachetul 116  
Diferenta maxima pret pachet <=> suma preturilor produse este 7573 RON.  
Pachetul promotional cu aceasta diferență maxima este 116
```

2) Stergem stocurile si preturile vor fi cele din tabela articol, nu articol\_in\_sediu :

```
--A doua functie:
----PACHET_PRODUS
truncate table articol_in_sediu; --stergem ce e :
DECLARE
    v_diferenta_maxima number(10);
    v_cod_pachet int;
BEGIN
    v_diferenta_maxima := exercitiu14.comparatie_pachet_produc_14(100, v_cod_pachet);
    DBMS_OUTPUT.PUT_LINE('Diferenta maxima pret pachet <=> suma preturilor produse este ' || v_diferenta_maxima || ' RON.');
    DBMS_OUTPUT.PUT_LINE('Pachetul promotional cu aceasta diferență maxima este ' || v_cod_pachet);
END;
/
--A doua procedura:
--ARTICOLE COMANDATE:
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Task completed in 0.146 seconds

Table ARTICOL\_IN\_SEDIU truncated.

```
/
```

```
--A doua functie:
----PACHET_PRODUS
truncate table articol_in_sediu; --stergem ce e in sediu pentru a demonstra ca se va lua pretul universal in acest caz
DECLARE
    v_diferenta_maxima number(10);
    v_cod_pachet int;
BEGIN
    v_diferenta_maxima := exercitiu14.comparatie_pachet_produc_14(100, v_cod_pachet);
    DBMS_OUTPUT.PUT_LINE('Diferenta maxima pret pachet <=> suma preturilor produse este ' || v_diferenta_maxima || ' RON.');
    DBMS_OUTPUT.PUT_LINE('Pachetul promotional cu aceasta diferență maxima este ' || v_cod_pachet);
END;
/
--A doua procedura:
--ARTICOLE COMANDATE:
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x | Query Result 5 x | Query Result 6 x | Query Result 7 x | Task completed in 0.168 seconds

PL/SQL procedure successfully completed.

Dbms Output x  
+ | Buffer Size: 20000 | conexiune\_noua x

```
Pachetul promotional 114 are pretul 1000 si suma preturilor produselor 500, diferența fiind 500
Pachetul promotional 116 are pretul 8000 si suma preturilor produselor 220, diferența fiind 7780
Pachetul promotional 115 are pretul 5000 si suma preturilor produselor 380, diferența fiind 4620
Pachetul promotional 113 are pretul 800 si suma preturilor produselor 500, diferența fiind 300
Pachetul promotional 112 are pretul 1000 si suma preturilor produselor 520, diferența fiind 480
Pachetul 112 are 1 produs(e) comun(e) cu pachetul 114
Pachetul 112 are 1 produs(e) comun(e) cu pachetul 115
Pachetul 113 are 1 produs(e) comun(e) cu pachetul 116
Pachetul 114 are 1 produs(e) comun(e) cu pachetul 115
Pachetul 114 are 1 produs(e) comun(e) cu pachetul 116
Diferenta maxima pret pachet <=> suma preturilor produse este 7780 RON.
Pachetul promotional cu aceasta diferență maxima este 116
```

## Procedura articole\_comandate\_14 :

The screenshot shows the Oracle SQL Developer interface with two main panes. The top pane is a 'Worksheet' containing PL/SQL code. The bottom pane is a 'Dbms Output' window displaying the results of the procedure's execution.

```
v_diferenta_maxima := exercitiu14.comparatie_pachet_produs_14(100, v_cod_pachet);
DBMS_OUTPUT.PUT_LINE('Diferenta maxima pret pachet <=> suma preturi produse este ' || v_diferenta_maxima || ' RON.');
DBMS_OUTPUT.PUT_LINE('Pachetul promotional cu aceasta diferență maxima este ' || v_cod_pachet);

END;
/
--A doua procedura:
--ARTICOLE COMANDATE:
BEGIN
    exercitiu14.articole_comandate_14(2);
END;
/
rollback;
--Prima procedura:
```

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

```
conexiune_noua x
xxv

Articolul 100 a fost comandat de 2 persoane diferite in cantitatea totala de 4 bucati. Sedile din care pleaca cel putin un colet care contine articolul sunt:
100 120
Curierii care livreaza cel putin un astfel de colet sunt: 200 240
Articolul 104 a fost comandat de 2 persoane diferite in cantitatea totala de 2 bucati. Sedile din care pleaca cel putin un colet care contine articolul sunt:
100
Curierii care livreaza cel putin un astfel de colet sunt: 200
Articolul 115 a fost comandat de 2 persoane diferite in cantitatea totala de 3 bucati. Sedile din care pleaca cel putin un colet care contine articolul sunt:
110
Curierii care livreaza cel putin un astfel de colet sunt: 200
Articolul 103 a fost comandat de 2 persoane diferite in cantitatea totala de 5 bucati. Sedile din care pleaca cel putin un colet care contine articolul sunt:
110 100
Curierii care livreaza cel putin un astfel de colet sunt: 200 240
Articolul 107 a fost comandat de 2 persoane diferite in cantitatea totala de 5 bucati. Sedile din care pleaca cel putin un colet care contine articolul sunt:
120 130
Curierii care livreaza cel putin un astfel de colet sunt: 210 230
```