

Proiect SGBD
Lant de Magazine de Instrumente Muzicale
Radoi Dragos Cosmin
Grupa 244

1)Prezența pe scurt baza de date (utilitatea ei).

Proiectul meu va prezenta modelul unei baze de date care reprezintă un lanț de magazine de instrumente muzicale. Magazinele vor avea mai multe instrumente muzicale(furnizate de companiile respective) și un Id care reprezintă locația magazinului. Ca angajați, magazinele vor avea fiecare unul sau mai mulți vânzători, un tehnician care se ocupă cu reparatul instrumentelor aduse de clienți și un manager. Angajații vor avea fiecare un cod și un email de muncă. Clienții, odată ce își fac un cont pe site-ul magazinului, pot crea comenzi pentru orice produs (dacă nu este în stoc, livrarea o să dureze mai mult). Livrarea va prezenta instrumentul, marca, durata de livrare și locația din care vine.

Locație – Magazin(One to many): Prezintă mai mulți magazine într-o locație.

Magazin – Angajați(One to many): Prezintă mai mulți angajați într-un magazin.

Magazin – Clienți(Many to many): Prezintă mai mulți clienți la mai multe magazine.

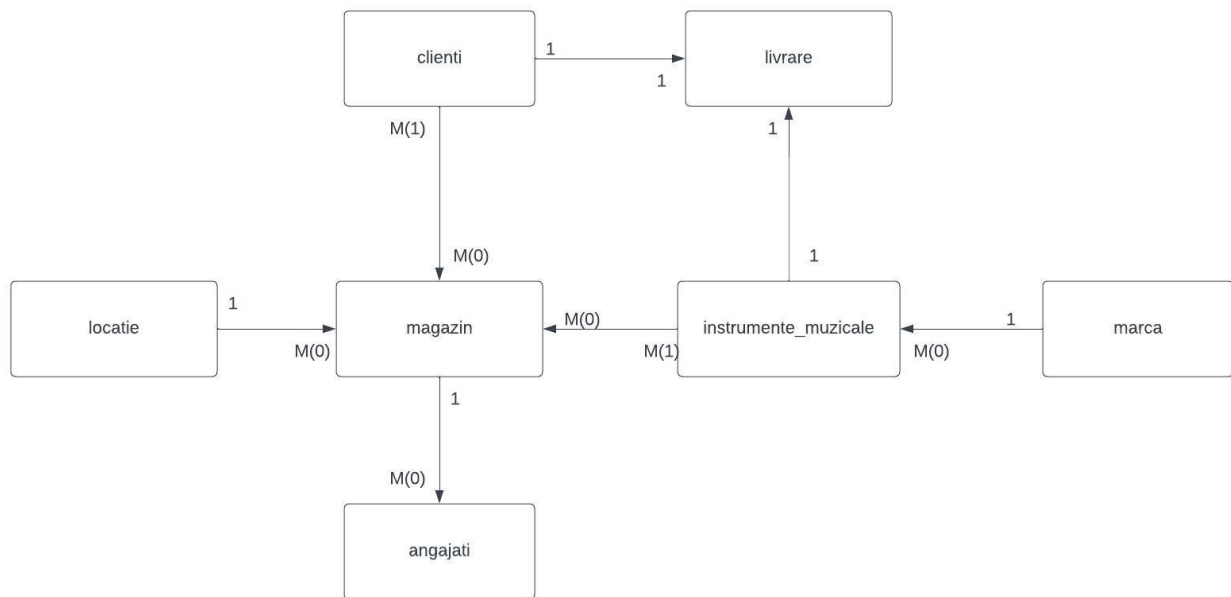
Magazin – Instrumente(Many to many): Prezintă mai mulți instrumente în mai multe magazine.

Instrumente – Marca(Many to one): Fiecare instrument poate avea o marcă diferită față de alt instrument, dar mai multe instrumente pot avea aceeași marcă.

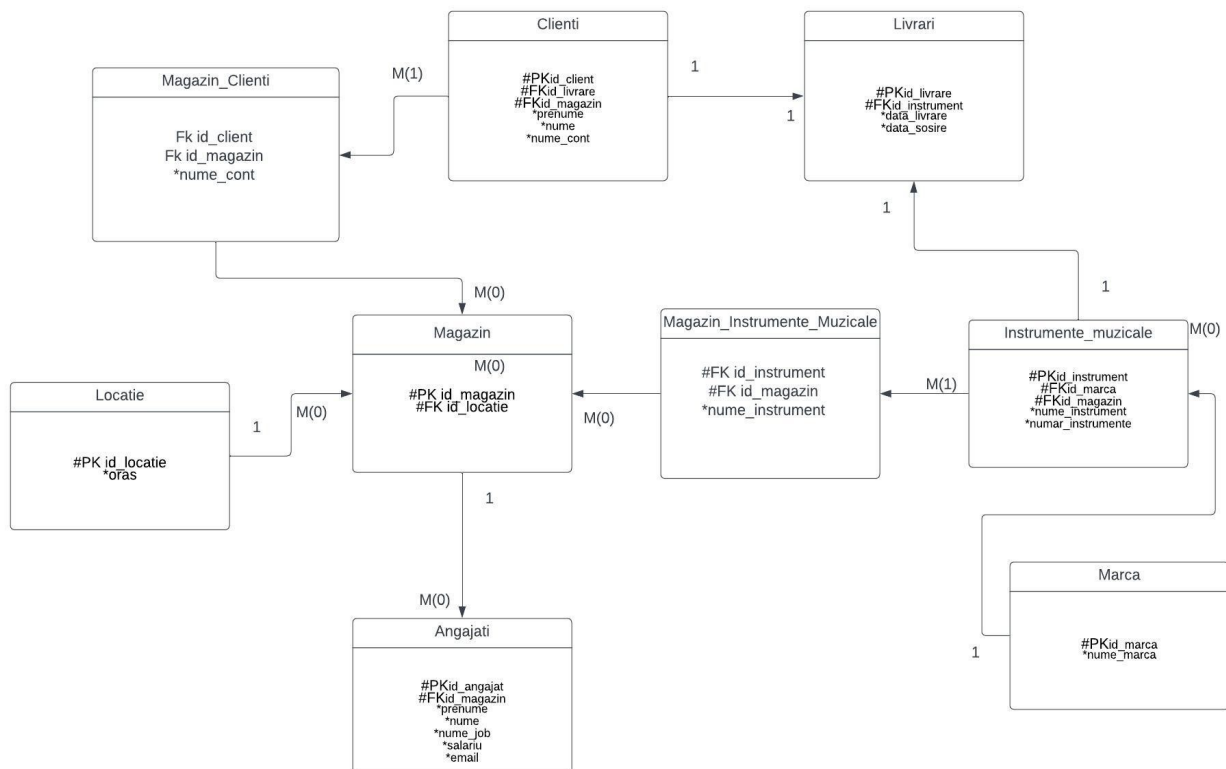
Instrumente – Livrare(One to one): Un instrument se poate încadra într-o singură livrare.

Clienți – Livrare(One to one): Un client poate avea o singură livrare.

2) Realizați diagrama entitate-relație (ERD): entitățile, relațiile și atributele trebuie definite în limba română (vezi curs SGBD / model de diagrama ERD; nu se va accepta alt format).



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.



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

--locatie

```
CREATE TABLE locatie(id_locatie NUMBER(4), oras VARCHAR(20));
```

ALTER TABLE locatie

```
ADD CONSTRAINT PK_locatie PRIMARY KEY(id_locatie);
```

--magazin

```
CREATE TABLE magazin(id_magazin NUMBER(4), id_locatie NUMBER(4));
```

ALTER TABLE magazin

```
ADD CONSTRAINT PK_magazin PRIMARY KEY(id_magazin)
```

```
ADD CONSTRAINT FK_locatie2 FOREIGN KEY (id_locatie) REFERENCES locatie(id_locatie);
```

--angajati

```
CREATE TABLE angajati(id_angajat NUMBER(4), id_magazin NUMBER(4), prenume  
VARCHAR2(20),
```

```
nume VARCHAR2(20) NOT NULL, nume_job VARCHAR2(15), salariu NUMBER(8, 2) NOT  
NULL, email CHAR(15));
```

ALTER TABLE angajati

```
ADD CONSTRAINT PK_angajati PRIMARY KEY(id_angajat)
```

```
ADD CONSTRAINT FK_magazin2 FOREIGN KEY (id_magazin) REFERENCES  
magazin(id_magazin);
```

--marca

```
CREATE TABLE marca(id_marca NUMBER(4), nume_marca VARCHAR2(15));
```

ALTER TABLE marca

```
ADD CONSTRAINT PK_marca PRIMARY KEY(id_marca);
```

--instrumente muzicale

```
CREATE TABLE instrumente_muzicale(id_instrument NUMBER(4), id_marca NUMBER(4),  
id_magazin NUMBER(4), nume_instrument VARCHAR2(15), numar_instrumente NUMBER(4));
```

ALTER TABLE instrumente_muzicale

```
ADD CONSTRAINT PK_instrument PRIMARY KEY(id_instrument)
```

```
ADD CONSTRAINT FK_magazin3 FOREIGN KEY (id_magazin) REFERENCES  
magazin(id_magazin)
```

```
ADD CONSTRAINT FK_marca2 FOREIGN KEY (id_marca) REFERENCES marca(id_marca);
```

--livrare

```
CREATE TABLE livrare(id_livrare NUMBER(4), id_instrument NUMBER(4), data_livrare DATE,  
data_sosire DATE);
```

ALTER TABLE livrare

```
ADD CONSTRAINT PK_livrare PRIMARY KEY(id_livrare)
```

```
ADD CONSTRAINT FK_instrument2 FOREIGN KEY(id_instrument) REFERENCES  
instrumente_muzicale(id_instrument);
```

--clienti

```
CREATE TABLE clienti(id_client NUMBER(4), id_livrare NUMBER(4), id_magazin  
NUMBER(4),
```

```
prenume VARCHAR2(20), nume VARCHAR2(20), nume_cont CHAR(15));
```

```
ALTER TABLE clienti  
ADD CONSTRAINT PK_client PRIMARY KEY(id_client)  
ADD CONSTRAINT FK_livrare2 FOREIGN KEY(id_livrare) REFERENCES livrare(id_livrare)  
ADD CONSTRAINT FK_magazin4 FOREIGN KEY(id_magazin) REFERENCES  
magazin(id_magazin);
```

```
--magazin-clienti  
CREATE TABLE magazin_clienti(id_client NUMBER(4), id_magazin NUMBER(4), nume_cont  
CHAR(15));
```

```
ALTER TABLE magazin_clienti  
ADD CONSTRAINT FK_ID_Magazin_Clienti FOREIGN KEY(id_client) REFERENCES  
clienti(id_client)  
ADD CONSTRAINT FK_ID_Clienti_Magazin FOREIGN KEY(id_magazin) REFERENCES  
magazin(id_magazin);
```

```
--magazin-instrumente-muzicale  
CREATE TABLE magazin_instrumente_muzicale(id_instrument NUMBER(4), id_magazin  
NUMBER(4), nume_instrument VARCHAR(15));
```

```
ALTER TABLE magazin_instrumente_muzicale  
ADD CONSTRAINT FK_ID_Magazin_Instrumente FOREIGN KEY(id_magazin) REFERENCES  
magazin(id_magazin)  
ADD CONSTRAINT FK_ID_Instrumente_Magazin FOREIGN KEY(id_instrument)  
REFERENCES instrumente_muzicale(id_instrument);
```

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

```
CREATE TABLE locatie(id_locatie NUMBER(4), oras VARCHAR(20));
```

```
ALTER TABLE locatie  
ADD CONSTRAINT PK_locatie PRIMARY KEY(id_locatie);
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(1, 'New York');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(2, 'London');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(3, 'Paris');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(4, 'Tokyo');
```

```
INSERT INTO locatie (id_locatie, oras)
```

```
VALUES(5, 'Sydney');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(6, 'Rome');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(7, 'Barcelona');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(8, 'Berlin');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(9, 'Mumbai');
```

```
INSERT INTO locatie (id_locatie, oras)  
VALUES(10, 'Cairo');
```

```
-----  
CREATE TABLE magazin(id_magazin NUMBER(4), id_locatie NUMBER(4));
```

```
ALTER TABLE magazin  
ADD CONSTRAINT PK_magazin PRIMARY KEY(id_magazin)  
ADD CONSTRAINT FK_locatie2 FOREIGN KEY (id_locatie) REFERENCES locatie(id_locatie);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(101, 1);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(102, 2);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(103, 3);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(104, 4);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(105, 5);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(106, 6);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(107, 7);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(108, 8);
```

```
INSERT INTO magazin (id_magazin, id_locatie)  
VALUES(109, 9);
```

```
INSERT INTO magazin (id_magazin, id_locatie)
```

VALUES(110, 10);

```
CREATE TABLE angajati(id_angajat NUMBER(4), id_magazin NUMBER(4), prenume
VARCHAR2(20),
nume VARCHAR2(20) NOT NULL, nume_job VARCHAR2(15), salariu NUMBER(8, 2) NOT
NULL, email CHAR(15));
```

```
ALTER TABLE angajati
ADD CONSTRAINT PK_angajati PRIMARY KEY(id_angajat)
ADD CONSTRAINT FK_magazin2 FOREIGN KEY (id_magazin) REFERENCES
magazin(id_magazin);
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1001, 101, 'John', 'Smith', 'manager', 50000, 'john@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1002, 102, 'Emily', 'Johnson', 'tehnician', 40000, 'emi@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1003, 103, 'Sarah', 'Brown', 'manager', 50000, 'sarah@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1004, 104, 'Michael', 'Williams', 'vanzator', 30000, 'mike@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1005, 105, 'David', 'Davis', 'tehnician', 40000, 'david@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1006, 106, 'Jessica', 'Miller', 'vanzator', 30000, 'mill@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1007, 107, 'Matthew', 'Wilson', 'manager', 50000, 'matt@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1008, 108, 'Olivia', 'Anderson', 'tehnician', 40000, 'olie@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1009, 109, 'Ethan', 'Taylor', 'vanzator', 30000, 'ethan@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1010, 110, 'Ava', 'Martinez', 'manager', 50000, 'ava@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1011, 101, 'Horia', 'Mihai', 'tehnician', 40000, 'horia@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
VALUES(1012, 101, 'Nick', 'Hincu', 'vanzator', 30000, 'Nick@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)
```

```
VALUES(1013, 101, 'Cristi', 'Andrei', 'tehnician', 40000, 'Cris@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)  
VALUES(1014, 107, 'Andrei', 'Cherciu', 'manager', 50000, 'Andii@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)  
VALUES(1015, 107, 'Vlad', 'Tudose', 'vanzator', 30000, 'Vlad@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)  
VALUES(1016, 107, 'Alin', 'Iosif', 'tehnician', 40000, 'Alin@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)  
VALUES(1017, 109, 'Alex', 'Iosif', 'magager', 50000, 'Alex@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)  
VALUES(1018, 110, 'David', 'Ranaciu', 'vanzator', 30000, 'David@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)  
VALUES(1019, 103, 'Robert', 'Trifu', 'tehnician', 40000, 'Rob@gmail.com');
```

```
INSERT INTO angajati (id_angajat, id_magazin, prenume, nume, nume_job, salariu, email)  
VALUES(1020, 103, 'Filip', 'Dima', 'vanzator', 30000, 'Filip@gmail.com');
```


```
CREATE TABLE marca(id_marca NUMBER(4), nume_marca VARCHAR2(15));
```

```
ALTER TABLE marca  
ADD CONSTRAINT PK_marca PRIMARY KEY(id_marca);
```

```
INSERT INTO marca(id_marca, nume_marca)  
VALUES(201, 'Gibson');
```

```
INSERT INTO marca(id_marca, nume_marca)  
VALUES(202, 'Fender');
```

```
INSERT INTO marca(id_marca, nume_marca)  
VALUES(203, 'Pearl');
```

```
INSERT INTO marca(id_marca, nume_marca)  
VALUES(204, 'Yamaha');
```

```
INSERT INTO marca(id_marca, nume_marca)  
VALUES(205, 'Stradivarius');
```

```
INSERT INTO marca(id_marca, nume_marca)  
VALUES(206, 'Selmer');
```

```
INSERT INTO marca(id_marca, nume_marca)  
VALUES(207, 'Bach');
```

```
INSERT INTO marca(id_marca, nume_marca)
```

VALUES(208, 'Muramatsu');

INSERT INTO marca(id_marca, nume_marca)
VALUES(209, 'Ibanez');

INSERT INTO marca(id_marca, nume_marca)
VALUES(210, 'Roland');

CREATE TABLE instrumente_muzicale(id_instrument NUMBER(4), id_marca NUMBER(4),
id_magazin NUMBER(4), nume_instrument VARCHAR2(15), numar_instrumente NUMBER(4));

ALTER TABLE instrumente_muzicale
ADD CONSTRAINT PK_instrument PRIMARY KEY(id_instrument)
ADD CONSTRAINT FK_magazin3 FOREIGN KEY (id_magazin) REFERENCES
magazin(id_magazin)
ADD CONSTRAINT FK_marca2 FOREIGN KEY (id_marca) REFERENCES marca(id_marca);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(301, 201, 101, 'Chit. Acustica', 23);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(302, 202, 102, 'Chit. Electrica', 57);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(303, 203, 103, 'Tobe', 14);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(304, 204, 104, 'Pian', 11);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(305, 205, 105, 'Vioara', 54);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(306, 206, 106, 'Saxofon', 67);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(307, 207, 107, 'Trompeta', 45);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)
VALUES(308, 208, 108, 'Flaut', 67);

INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,
numar_instrumente)


```
VALUES(309, 209, 109, 'Chitara Bas', 34);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(310, 210, 110, 'Clape', 55);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(311, 201, 103, 'Cello', 66);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(312, 202, 110, 'Banjo', 13);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(313, 203, 107, 'Harpa', 9);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(314, 204, 106, 'Mandolina', 24);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(315, 205, 110, 'Acordeon', 9);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(316, 206, 107, 'Trombon', 14);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(317, 207, 101, 'Clarinet', 39);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(318, 208, 102, 'Oboi', 15);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(319, 209, 109, 'Cimpoi', 7);
```

```
INSERT INTO instrumente_muzicale (id_instrument, id_marca, id_magazin, nume_instrument,  
numar_instrumente)  
VALUES(320, 210, 107, 'Sitar', 33);
```

```
commit;
```

```
CREATE TABLE livrare(id_livrare NUMBER(4), id_instrument NUMBER(4), data_livrare DATE, data_sosire DATE);
```

```
ALTER TABLE livrare  
ADD CONSTRAINT PK_livrare PRIMARY KEY(id_livrare)  
ADD CONSTRAINT FK_instrument2 FOREIGN KEY(id_instrument) REFERENCES  
instrumente_muzicale(id_instrument);
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(1234, 301, TO_DATE('2023-05-10', 'YYYY-MM-DD'), TO_DATE('2023-05-15',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(5678, 302, TO_DATE('2023-05-11', 'YYYY-MM-DD'), TO_DATE('2023-05-16',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(9876, 303, TO_DATE('2023-05-12', 'YYYY-MM-DD'), TO_DATE('2023-05-17',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(5432, 304, TO_DATE('2023-05-13', 'YYYY-MM-DD'), TO_DATE(' 2023-05-18',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(2468, 305, TO_DATE('2023-05-14', 'YYYY-MM-DD'), TO_DATE(' 2023-05-19',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(1357, 306, TO_DATE('2023-05-15', 'YYYY-MM-DD'), TO_DATE(' 2023-05-20',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(8642, 307, TO_DATE('2023-05-16', 'YYYY-MM-DD'), TO_DATE(' 2023-05-21',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(9753, 308, TO_DATE('2023-05-17', 'YYYY-MM-DD'), TO_DATE(' 2023-05-22',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(7531, 309, TO_DATE('2023-05-18', 'YYYY-MM-DD'), TO_DATE(' 2023-05-23',  
'YYYY-MM-DD'));
```

```
INSERT INTO livrare (id_livrare, id_instrument, data_livrare, data_sosire)  
VALUES(1029, 310, TO_DATE('2023-05-19', 'YYYY-MM-DD'), TO_DATE(' 2023-05-24',  
'YYYY-MM-DD'));
```

```
-----  
-----  
CREATE TABLE clienti(id_client NUMBER(4), id_livrare NUMBER(4), id_magazin
```

```
NUMBER(4),  
prenume VARCHAR2(20), nume VARCHAR2(20), nume_cont CHAR(15));
```

```
ALTER TABLE clienti  
ADD CONSTRAINT PK_client PRIMARY KEY(id_client)  
ADD CONSTRAINT FK_livrare2 FOREIGN KEY(id_livrare) REFERENCES livrare(id_livrare)  
ADD CONSTRAINT FK_magazin4 FOREIGN KEY(id_magazin) REFERENCES  
magazin(id_magazin);
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3011, 1234, 110, 'Ethan', 'Lewis', 'ethanlewis234');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3012, 5678, 101, 'Emma', 'Lee', 'emmalee567');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3013, 9876, 102, 'Daniel', 'Walker', 'danielwalker890');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3014, 5432, 103, 'Mia', 'Green', 'miagreen123');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3015, 9876, 104, 'Alexander', 'Hall', 'alexanderhall');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3016, 5432, 105, 'Sophia', 'Turner', 'sophiaturner789');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3017, 2468, 106, 'William', 'White', 'williamwhite321');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3018, 8642, 107, 'Charlotte', 'Harris', 'charlotteh654');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3019, 7531, 108, 'James', 'King', 'jamesking987');
```

```
INSERT INTO clienti(id_client, id_livrare, id_magazin, prenume, nume, nume_cont)  
VALUES(3020, 1029, 109, 'Harper', 'Scott', 'harperscott012');
```

```
-----  
CREATE TABLE magazin_clienti(id_client NUMBER(4), id_magazin NUMBER(4), nume_cont  
CHAR(15));
```

```
ALTER TABLE magazin_clienti  
ADD CONSTRAINT FK_ID_Magazin_Clienti FOREIGN KEY(id_client) REFERENCES  
clienti(id_client)  
ADD CONSTRAINT FK_ID_Clienti_Magazin FOREIGN KEY(id_magazin) REFERENCES  
magazin(id_magazin);
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)  
VALUES(3011, 110, 'ethanlewis234');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3012, 101, 'emmalee567');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3013, 102, 'danielwalker890');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3014, 103, 'miagreen123');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3015, 104, 'alexanderhall');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3016, 105, 'sophiaturner789');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3017, 106, 'williamwhite321');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3018, 107, 'charlotteh654');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3019, 108, 'jamesking987');
```

```
INSERT INTO magazin_clienti(id_client, id_magazin, nume_cont)
VALUES(3020, 109, 'harperscott012');
```

```
-----
CREATE TABLE magazin_instrumente_muzicale(id_instrument NUMBER(4), id_magazin
NUMBER(4), nume_instrument VARCHAR(15), numar_instrumente NUMBER(4));
```

```
ALTER TABLE magazin_instrumente_muzicale
ADD CONSTRAINT FK_ID_Magazin_Instrumente FOREIGN KEY(id_magazin) REFERENCES
magazin(id_magazin)
ADD CONSTRAINT FK_ID_Instrumente_Magazin FOREIGN KEY(id_instrument)
REFERENCES instrumente_muzicale(id_instrument);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(301, 101, 'Chit. Acustica', 23);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(302, 102, 'Chit. Electrica', 57);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(303, 103, 'Tobe', 14);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(304, 104, 'Pian', 11);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(305, 105, 'Vioara', 54);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(306, 106, 'Saxofon', 67);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(307, 107, 'Trompeta', 45);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(308, 108, 'Flaut', 67);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(309, 109, 'Chitara Bas', 34);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(310, 110, 'Clape', 55);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(311, 103, 'Cello', 66);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(312,110, 'Banjo', 13);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(313, 107, 'Harpa', 9);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(314, 106, 'Mandolina', 24);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(315,110, 'Acordeon', 9);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(316,107, 'Trombon', 14);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(317, 101, 'Clarinet', 39);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(318, 102, 'Oboi', 15);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(319, 109, 'Cimpoi', 7);
```

```
INSERT INTO magazin_instrumente_muzicale(id_instrument, id_magazin, nume_instrument,
numar_instrumente)
VALUES(320, 107, 'Sitar', 33);
```

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.

Scrieti un program care, sa introduca intr-un tablou indexat informatii despre instrumentele muzicale si numarul acestora in stoc si afisati informatiile acestea, sa genereze un tablou imbricat care sa afiseze lista celor 5 cei mai platiti angajati si ordonatii crescatori in functie de salariu si afisati pentru fiecare angajat din lista numele si id-ul magazinului unde lucreaza si creati un vector care sa contina o lista ordonata numeric crescator a id-urilor magazinelor in care lucreaza toti angajatii.

```
CREATE OR REPLACE PROCEDURE Informatii_Angajati
IS
```

```
    TYPE t_instrumente_radoi IS record (
        NumeInstrument instrumente_muzicale.nume_instrument%TYPE,
        NumarStoc instrumente_muzicale.numar_instrumente%TYPE
    );
```

```
    TYPE tablou_indexat_radoi IS TABLE OF t_instrumente_radoi INDEX BY PLS_INTEGER;
    t_indexat tablou_indexat_radoi;
```

```
    TYPE tablou_imbricat_radoi IS TABLE OF angajati%ROWTYPE;
    t_imbricat tablou_imbricat_radoi := tablou_imbricat_radoi();
```

```
    TYPE tablou_vector_radoi IS VARRAY(100) OF angajati.id_magazin%TYPE;
```

```
    t_vector_magazin tablou_vector_radoi := tablou_vector_radoi();
```

```
BEGIN
```

```
    for informatii in (select i.nume_instrument, i.numar_instrumente from instrumente_muzicale i)
loop
    t_indexat(t_indexat.count + 1).NumeInstrument := informatii.nume_instrument;
    t_indexat(t_indexat.count).NumarStoc := informatii.numar_instrumente;
end loop;
```

```

    for i in t_indexat.first..t_indexat.last loop
        DBMS_OUTPUT.PUT_LINE(t_indexat(i).NumeInstrument || ' are in stoc un numar de ' ||
t_indexat(i).NumarStoc || ' exemplare' );
    end loop;
    DBMS_OUTPUT.NEW_LINE;

    for informatii in (select * from angajati order by salariu desc) loop
        t_imbricat.extend;
        t_imbricat(t_imbricat.count) := informatii;
        exit when t_imbricat.count = 5;
    end loop;

    for i in t_imbricat.first..t_imbricat.last loop
        DBMS_OUTPUT.PUT_LINE(t_imbricat(i).nume || ' lucreaza in magazinul ' ||
t_imbricat(i).id_magazin);
    end loop;
    DBMS_OUTPUT.NEW_LINE;

    FOR informatii IN (SELECT DISTINCT id_magazin FROM angajati ORDER BY id_magazin)
LOOP
    IF t_vector_magazin.COUNT < t_vector_magazin.LIMIT THEN
        t_vector_magazin.EXTEND;
        t_vector_magazin(t_vector_magazin.COUNT) := informatii.id_magazin;
    END IF;
END LOOP;

FOR i IN 1..t_vector_magazin.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE('Magazinul: ' || t_vector_magazin(i));
END LOOP;

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('A aparut o eroare: ' || SQLERRM);
END Informatii_Angajati;
/

BEGIN
    Informatii_Angajati;
END;
/

```

Tabloul indexat:

Worksheet

Query Builder

```
BEGIN
for informatii in (select i.numr_instrument, i.numr_instrumente from instrumente_muzicale i) loop
    t_indexat(t_indexat.count + 1).NumeInstrument := informatii.numr_instrument;
    t_indexat(t_indexat.count).NumarStoc := informatii.numr_instrumente;
end loop;

for i in t_indexat.first..t_indexat.last loop
    DBMS_OUTPUT.PUT_LINE(t_indexat(i).NumeInstrument || ' are in stoc un numar de ' || t_indexat(i).NumarStoc || ' exemplare' );
end loop;
DBMS_OUTPUT.NEW_LINE;

for informatii in (select * from angajati order by salariu desc) loop
    t_imbricat.extend;
    t_imbricat(t_imbricat.count) := informatii;
```

Script Output x

Task completed in 0.101 seconds

Procedure INFORMATII_ANGAJATI compiled

PL/SQL procedure successfully completed.

Dbms Output

Buffer Size: 20000

grupa244 x

Chit. Acustica are in stoc un numar de 23 exemplare
Chit. Electrica are in stoc un numar de 57 exemplare
Tobe are in stoc un numar de 14 exemplare
Pian are in stoc un numar de 11 exemplare
Vioara are in stoc un numar de 54 exemplare
Saxofon are in stoc un numar de 67 exemplare
Trompeta are in stoc un numar de 45 exemplare
Flaut are in stoc un numar de 67 exemplare
Chitara Bas are in stoc un numar de 34 exemplare
Clape are in stoc un numar de 55 exemplare
Cello are in stoc un numar de 66 exemplare
Banjo are in stoc un numar de 13 exemplare
Harpa are in stoc un numar de 9 exemplare
Mandolina are in stoc un numar de 24 exemplare
Acordeon are in stoc un numar de 9 exemplare
Trombon are in stoc un numar de 14 exemplare
Clarinet are in stoc un numar de 39 exemplare
Oboi are in stoc un numar de 15 exemplare
Cimpoi are in stoc un numar de 7 exemplare
Sitar are in stoc un numar de 33 exemplare
Violoncel are in stoc un numar de 2 exemplare

Tabloul imbricat si vectorul:

Worksheet

Query Builder

```
for informatii in (select * from angajati order by salariu desc) loop
    t_imbricat.extend;
    t_imbricat(t_imbricat.count) := informatii;
    exit when t_imbricat.count = 5;
end loop;

for i in t_imbricat.first..t_imbricat.last loop
    DBMS_OUTPUT.PUT_LINE(t_imbricat(i).nume || ' lucreaza in magazinul ' || t_imbricat(i).id_magazin);
end loop;
DBMS_OUTPUT.NEW_LINE;

FOR informatii IN (SELECT DISTINCT id_magazin FROM angajati ORDER BY id_magazin) LOOP
    IF t_vector_magazin.COUNT < t_vector_magazin.LIMIT THEN
        t_vector_magazin.EXTEND;
        t_vector_magazin(t_vector_magazin.COUNT) := informatii.id_magazin;
    END IF;
END LOOP;

FOR i IN 1..t_vector_magazin.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE('Magazinul: ' || t_vector_magazin(i));
END LOOP;
```

Script Output x

Task completed in 0.101 seconds

Procedure INFORMATII_ANGAJATI compiled

Dbms Output

Buffer Size: 20000

grupa244 x

Iosif lucreaza in magazinul 109
Wilson lucreaza in magazinul 107
Brown lucreaza in magazinul 103
Cherciu lucreaza in magazinul 107
Smith lucreaza in magazinul 101

Magazinul: 101
Magazinul: 102
Magazinul: 103
Magazinul: 104
Magazinul: 105
Magazinul: 106
Magazinul: 107
Magazinul: 108
Magazinul: 109
Magazinul: 110

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.

Scrieti un program care sa introduca intr-un cursor id-ul si numele marcilor si sa introduca intr-un cursor parametrizat numele si id-ul tuturor instrumentelor. Afisati toate marcile si sub fiecare marca sa afisati instrumentele care au acel tip de marca

```
create or replace procedure cursoare_radoi is
  cursor cursor_dragos is select id_marca, nume_marca
    from marca;
  cursor cursor_cosmin (param marca.id_marca%type) is
    select nume_instrument, id_instrument
    from instrumente_muzicale
    where id_marca = param;

  d_id_marca marca.id_marca%type;
  d_nume_marca marca.nume_marca%type;

begin
  open cursor_dragos;
  loop
    fetch cursor_dragos into d_id_marca, d_nume_marca;
    exit when cursor_dragos%notfound;

    dbms_output.put_line('Marca ' || d_nume_marca);

    for i in cursor_cosmin(d_id_marca) loop
      dbms_output.put_line(i.nume_instrument);
    end loop;
  end loop;
end cursoare_radoi;
/

begin
  cursoare_radoi;
end;
/
```

create or replace procedure cursoare_radoi is
 cursor cursor_dragos is select id_marca, nume_marca
 from marca;
 cursor cursor_cosmin (param marca.id_marca&type) is

Script Output x

| Task completed in 0.123 seconds

Procedure CURSOARE_RADOI compiled

PL/SQL procedure successfully completed.

Dbms Output

| Buffer Size: |

grupa244 x

Marca Gibson
Chit. Acustica
Cello
Marca Fender
Chit. Electrica
Banjo
Marca Pearl
Tobe
Harpa
Marca Yamaha
Pian
Marca Stradivarius
Vioara
Mandolina
Acordeon
Marca Selmer
Saxofon
Trombon
Marca Bach
Trompeta
Clarinet
Marca Muramatsu
Flaut
Oboi
Marca Ibanez
Chitara Bas
Cimpoi
Marca Roland
Clape
Sitar
Violoncel

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.

Scrieti un subprogram care afiseaza orasul in care se afla instrumentul mentionat, iar instrumentul sa aiba intre 20 si 50 de exemplare in magazin. Apelati subprogramul astfel incat sa fie tratate toate cazurile definite si tratate.

--Ex_8

create or replace function cautare_locatie_instrument

(v_ume_instrument instrumente_muzicale.ume_instrument%TYPE DEFAULT 'Tobe')

return VARCHAR is

ume_oras locatie.oras%type;

numar_instrumente instrumente_muzicale.numar_instrumente%type;

exceptie_1 EXCEPTION;

exceptie_2 EXCEPTION;

begin

select oras, i.numar_instrumente into ume_oras, numar_instrumente

from instrumente_muzicale i, magazin m, locatie l

where ume_instrument = v_ume_instrument

and i.id_magazin = m.id_magazin

and l.id_locatie = m.id_locatie;

if numar_instrumente < 20 then

raise exceptie_1;

elsif numar_instrumente > 50 then

raise exceptie_2;

else

return ume_oras;

end if;

EXCEPTION

WHEN exceptie_1 THEN

DBMS_OUTPUT.PUT_LINE('Magazinul nu are destule instrumente');

WHEN exceptie_2 THEN

DBMS_OUTPUT.PUT_LINE('Magazinul are prea multe instrumente');

WHEN NO_DATA_FOUND THEN

RAISE_APPLICATION_ERROR(-20000, 'Nu exista instrumente cu numele dat');

WHEN OTHERS THEN

RAISE_APPLICATION_ERROR(-20001, 'Alta eroare!');

end cautare_locatie_instrument;

/

BEGIN

DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul '||
cautare_locatie_instrument('Saxofon'));

END;

/

Cazul in care exista instrumentul:

The screenshot displays the Oracle SQL Developer interface. The top pane, titled 'Query Builder', contains the following PL/SQL code for a function named `CAUTARE_LOCATIE_INSTRUMENT`:

```
19 and l.id_locatie = m.id_locatie;
20
21 if numar_instrumente < 20 then
22     raise exceptie_1;
23 elsif numar_instrumente > 50 then
24     raise exceptie_2;
25 else
26     return nume_oras;
27 end if;
28
29 EXCEPTION
30     WHEN exceptie_1 THEN
31         DBMS_OUTPUT.PUT_LINE('Magazinul nu are destule instrumente');
32     WHEN exceptie_2 THEN
33         DBMS_OUTPUT.PUT_LINE('Magazinul are prea multe instrumente');
34     WHEN NO_DATA_FOUND THEN
35         RAISE_APPLICATION_ERROR(-20000, 'Nu exista instrumente cu numele dat');
36     WHEN OTHERS THEN
37         RAISE_APPLICATION_ERROR(-20001, 'Alta eroare!');
38
39 end cautare_locatie_instrument;
40 /
41
42 BEGIN
43     DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul '|| cautare_locatie_instrument('Clarinet'));
44 END;
45 /
46
```

The bottom pane shows the execution results. The 'Script Output' tab indicates that the function was compiled successfully and the PL/SQL procedure was completed. The 'Query Result' tab shows the output of the function call, which is 'New York'.

Function CAUTARE_LOCATIE_INSTRUMENT compiled

PL/SQL procedure successfully completed.

Dbms Output

Buffer Size: 20000

Instrumentul se afla in orasul New York

Cazul in care nu exista instrumentul:

Worksheet

Query Builder

25

else

26

return nume_oras;

27

end if;

28

29

EXCEPTION

30

WHEN exceptie_1 THEN

31

DBMS_OUTPUT.PUT_LINE('Magazinul nu are destule instrumente');

32

WHEN exceptie_2 THEN

33

DBMS_OUTPUT.PUT_LINE('Magazinul are prea multe instrumente');

34

WHEN NO_DATA_FOUND THEN

35

RAISE_APPLICATION_ERROR(-20000, 'Nu exista instrumente cu numele dat');

36

WHEN OTHERS THEN

37

RAISE_APPLICATION_ERROR(-20001, 'Alta eroare!');

38

39

end cautare_locatie_instrument;

40

/

41

42

BEGIN

43

DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul '|| cautare_locatie_instrument('Cartof'));

44

END;

45

/

46

Script Output x



Task completed in 0.155 seconds

Error starting at line : 42 in command -

BEGIN

DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul '|| cautare_locatie_instrument('Cartof'));

END;

Error report -

ORA-20000: Nu exista instrumente cu numele dat

ORA-06512: at "GRUPA244.CAUTARE_LOCATIE_INSTRUMENT", line 31

ORA-06512: at line 2

20000. 00000 - "%s"

*Cause: The stored procedure 'raise_application_error' was called which causes this error to be generated.

*Action: Correct the problem as described in the error message or contact the application administrator or DBA for more information.

Cazul in care exista, dar are sub 20 de exemplare in magazin

Worksheet

Query Builder

25

else

26

return nume_oras;

27

end if;

28

29

EXCEPTION

30

WHEN exceptie_1 THEN

31

DBMS_OUTPUT.PUT_LINE('Magazinul nu are destule instrumente');

32

WHEN exceptie_2 THEN

33

DBMS_OUTPUT.PUT_LINE('Magazinul are prea multe instrumente');

34

WHEN NO_DATA_FOUND THEN

35

RAISE_APPLICATION_ERROR(-20000, 'Nu exista instrumente cu numele dat');

36

WHEN OTHERS THEN

37

RAISE_APPLICATION_ERROR(-20001, 'Alta eroare!');

38

39

end cautare_locatie_instrument;

40

/

41

42

BEGIN

43

DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul '|| cautare_locatie_instrument('Pian'));

44

END;

45

/

46

Script Output x

Task completed in 0.074 seconds

Error starting at line : 42 in command -

BEGIN

DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul '|| cautare_locatie_instrument('Pian'));

END;

Error report -

ORA-06503: PL/SQL: Function returned without value

ORA-06512: at "GRUPA244.CAUTARE_LOCATIE_INSTRUMENT", line 35

ORA-06512: at line 2

06503. 00000 - "PL/SQL: Function returned without value"

*Cause: A call to PL/SQL function completed, but no RETURN statement was executed.

*Action: Rewrite PL/SQL function, making sure that it always returns a value of a proper type.

Dbms Output

Buffer Size: 20000

grupa244 x

Magazinul nu are destule instrumente

Cazul in care exista, dar are peste 50 de instrumente

The screenshot displays the Oracle SQL Developer environment. The top pane, titled 'Query Builder', contains a PL/SQL script. The script defines a function `cautare_locatie_instrument` that searches for an instrument in a city. It includes several exception handlers: `exceptie_1` for no instruments, `exceptie_2` for too many instruments, `NO_DATA_FOUND` for no results, and `OTHERS` for general errors. The script is executed from line 42, where it calls `cautare_locatie_instrument('Saxofon')`.

The bottom pane shows the 'Script Output' and 'Query Result' tabs. The 'Script Output' tab displays the execution error, which is an ORA-06503 error: 'PL/SQL: Function returned without value'. The error report indicates that the function completed but did not return a value, which is a programming error. The 'Query Result' tab shows the output of the script, which is 'Magazinul are prea multe instrumente' (The store has too many instruments).

```
25 else
26     return nume_oras;
27 end if;
28
29 EXCEPTION
30     WHEN exceptie_1 THEN
31         DBMS_OUTPUT.PUT_LINE('Magazinul nu are destule instrumente');
32     WHEN exceptie_2 THEN
33         DBMS_OUTPUT.PUT_LINE('Magazinul are prea multe instrumente');
34     WHEN NO_DATA_FOUND THEN
35         RAISE_APPLICATION_ERROR(-20000, 'Nu exista instrumente cu numele dat');
36     WHEN OTHERS THEN
37         RAISE_APPLICATION_ERROR(-20001, 'Alta eroare!');
38
39 end cautare_locatie_instrument;
40 /
41
42 BEGIN
43     DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul ' || cautare_locatie_instrument('Saxofon'));
44 END;
45 /
46
```

Script Output x Query Result x

Task completed in 0.05 seconds

Error starting at line : 42 in command -
BEGIN
 DBMS_OUTPUT.PUT_LINE('Instrumentul se afla in orasul ' || cautare_locatie_instrument('Saxofon'));
END;
Error report -
ORA-06503: PL/SQL: Function returned without value
ORA-06512: at "GRUPA244.CAUTARE_LOCATIE_INSTRUMENT", line 35
ORA-06512: at line 2
06503. 00000 - "PL/SQL: Function returned without value"
*Cause: A call to PL/SQL function completed, but no RETURN statement was
 executed.
*Action: Rewrite PL/SQL function, making sure that it always returns
 a value of a proper type.

Dbms Output

Buffer Size: 20000

grupa244 x

Magazinul are prea multe instrumente

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.

Scrieti un subprogram stocat independent de tip procedura care sa afiseze numele orasului unde sa afla instrumentele cu o anumita marca.

```
create or replace procedure oras_marca
(v_marca marca.nume_marca%TYPE)
is
city locatie.oras%TYPE;
begin
select l.oras into city
from locatie l, magazin m, magazin_instrumente_muzicale mi, instrumente_muzicale i, marca
ma
where v_marca = nume_marca
and ma.id_marca = i.id_marca
and i.nume_instrument = mi.nume_instrument
and i.id_instrument = mi.id_instrument
and mi.id_magazin = m.id_magazin
and l.id_locatie = m.id_locatie;
DBMS_OUTPUT.PUT_LINE('Orasul in care se afla marca mentionata este '|| city);
EXCEPTION
WHEN NO_DATA_FOUND THEN
RAISE_APPLICATION_ERROR(-20000, 'Nu exista orase in care se afla instrumente cu
marca data');
WHEN TOO_MANY_ROWS THEN
RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe orase in care se afla
instrumente cu marca data');
WHEN OTHERS THEN
RAISE_APPLICATION_ERROR(-20002,'Alta eroare!');
end oras_marca;
/

begin
oras_marca('Yamaha');
end;
/
```

Cazul in care instrumentul exista:

Worksheet

Query Builder

```
is
    city locatie.oras%TYPE;
begin
    select l.oras into city
    from locatie l, magazin m, magazin_instrumente_muzicale mi, instrumente_muzicale i, marca ma
    where v_marca = nume_marca
    and ma.id_marca = i.id_marca
    and i.nume_instrument = mi.nume_instrument
    and i.id_instrument = mi.id_instrument
    and mi.id_magazin = m.id_magazin
    and l.id_locatie = m.id_locatie;
    DBMS_OUTPUT.PUT_LINE('Orasul in care se afla marca mentionata este '|| city);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista orase in care se afla instrumente cu marca data');
    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe orase in care se afla instrumente cu marca data');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
end oras_marca;
/

begin
    oras_marca('Yamaha');
end;
```

Script Output x

Task completed in 0.131 seconds

PL/SQL procedure successfully completed.

Dbms Output

Buffer Size: 20000

grupa244 x

Orasul in care se afla marca mentionata este Tokyo

Cazul in care instrumentul nu exista:

```
is
    city locatie.oras%TYPE;
begin
    select l.oras into city
    from locatie l, magazin m, magazin_instrumente_muzicale mi, instrumente_muzicale i, marca ma
    where v_marca = nume_marca
    and ma.id_marca = i.id_marca
    and i.nume_instrument = mi.nume_instrument
    and i.id_instrument = mi.id_instrument
    and mi.id_magazin = m.id_magazin
    and l.id_locatie = m.id_locatie;
    DBMS_OUTPUT.PUT_LINE('Orasul in care se afla marca mentionata este '|| city);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista orase in care se afla instrumente cu marca data');
    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe orase in care se afla instrumente cu marca data');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
end oras_marca;
/

begin
    oras_marca('Instrument');
end;
/
```

Script Output x

Task completed in 0.059 seconds

Error report -

ORA-20000: Nu exista orase in care se afla instrumente cu marca data

ORA-06512: at "GRUPA244.ORAS_MARCA", line 17

ORA-06512: at line 2

20000. 00000 - "%s"

*Cause: The stored procedure 'raise_application_error' was called which causes this error to be generated.

Dbms Output

Cazul TOO_MANY_ROWS:

```
create or replace procedure oras_marca
(v_marca marca.nume_marca%TYPE)
is
    city locatie.oras%TYPE;
begin
    select l.oras into city
    from locatie l, magazin m, magazin_instrumente_muzicale mi, instrumente_muzicale i, marca ma
    where v_marca = nume_marca
    and ma.id_marca = i.id_marca
    and i.nume_instrument = mi.nume_instrument
    and i.id_instrument = mi.id_instrument
    and mi.id_magazin = m.id_magazin
    and l.id_locatie = m.id_locatie;
    DBMS_OUTPUT.PUT_LINE('Orasul in care se afla marca mentionata este '|| city);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista orase in care se afla instrumente cu marca data');
    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multe orase in care se afla instrumente cu marca data');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
end oras_marca;
/

begin
    oras_marca('Gibson');
end;
/
```

Script Output x

Task completed in 0.053 seconds

20000. 00000 - "%s"

*Cause: The stored procedure 'raise_application_error' was called which causes this error to be generated.

*Action: Correct the problem as described in the error message or contact the application administrator or DBA for more information.

Error starting at line : 25 in command -

```
begin
    oras_marca('Gibson');
end;
```

Error report -

ORA-20001: Exista mai multe orase in care se afla instrumente cu marca data

ORA-06512: at "GRUPA244.ORAS_MARCA", line 19

ORA-06512: at line 2

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

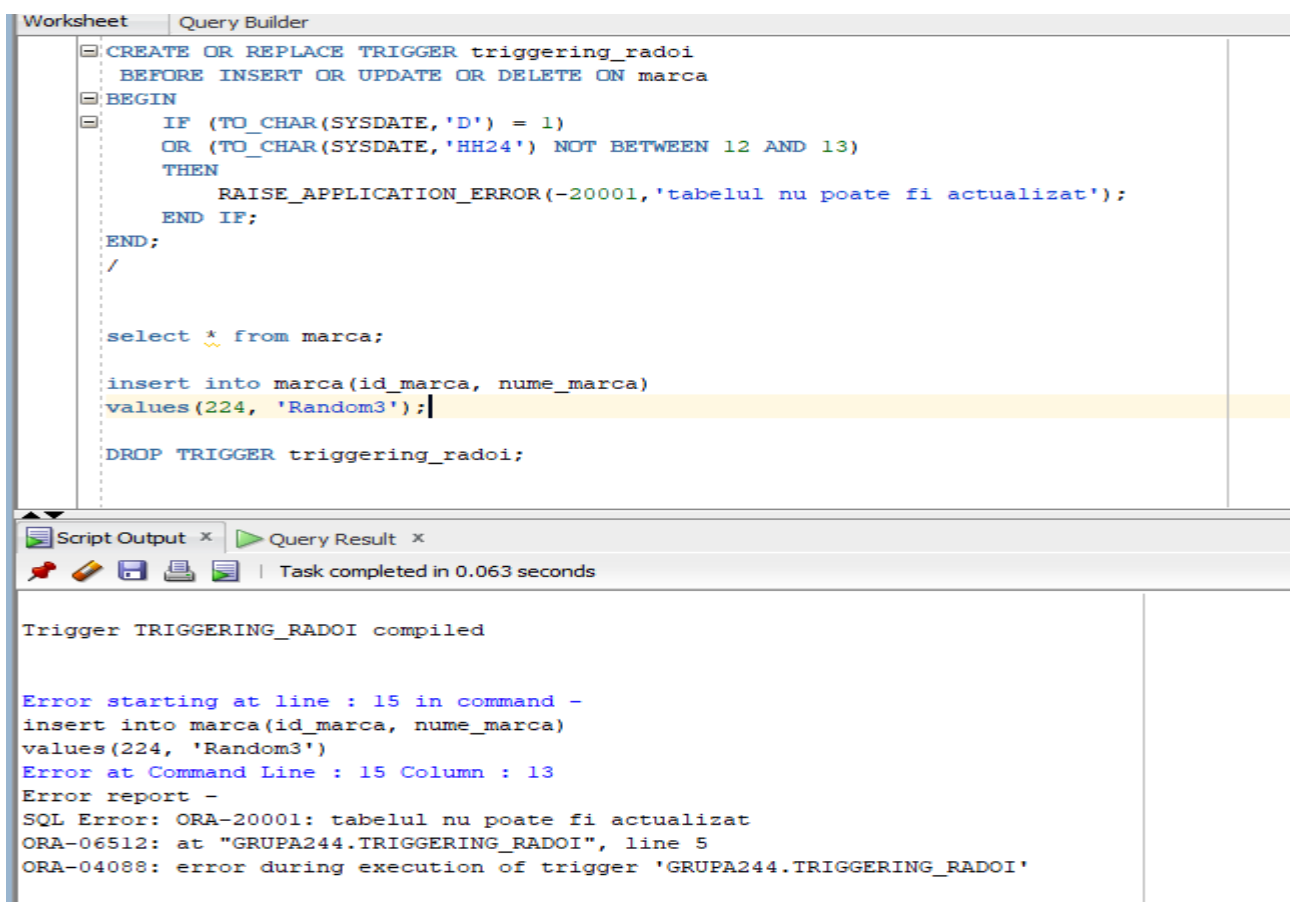
Definiți un declanșator care să permită lucrul asupra tabelului marca (INSERT, UPDATE, DELETE) doar în intervalul de ore 12:00 - 13:00, de luni până sâmbătă.

```
CREATE OR REPLACE TRIGGER triggering_radoi
BEFORE INSERT OR UPDATE OR DELETE ON marca
BEGIN
    IF (TO_CHAR(SYSDATE,'D') = 1)
    OR (TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 12 AND 13)
    THEN
        RAISE_APPLICATION_ERROR(-20001,'tabelul nu poate fi actualizat');
    END IF;
END;
/
```

```
select * from marca;
```

```
insert into marca(id_marca, nume_marca)
values(223, 'Random2');
```

```
DROP TRIGGER triggering_radoi;
```



```
Worksheet | Query Builder
CREATE OR REPLACE TRIGGER triggering_radoi
BEFORE INSERT OR UPDATE OR DELETE ON marca
BEGIN
    IF (TO_CHAR(SYSDATE,'D') = 1)
    OR (TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 12 AND 13)
    THEN
        RAISE_APPLICATION_ERROR(-20001,'tabelul nu poate fi actualizat');
    END IF;
END;
/

select * from marca;

insert into marca(id_marca, nume_marca)
values(224, 'Random3');

DROP TRIGGER triggering_radoi;
```

Script Output x | Query Result x

Task completed in 0.063 seconds

Trigger TRIGGERING_RADOI compiled

Error starting at line : 15 in command -
insert into marca(id_marca, nume_marca)
values(224, 'Random3')

Error at Command Line : 15 Column : 13

Error report -
SQL Error: ORA-20001: tabelul nu poate fi actualizat
ORA-06512: at "GRUPA244.TRIGGERING_RADOI", line 5
ORA-04088: error during execution of trigger 'GRUPA244.TRIGGERING_RADOI'

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

Definiți un declanșator prin care să nu se permită mărirea numărului de instrumente din tabelul instrumente_muzicale.

```
select * from instrumente_muzicale;
```

```
CREATE OR REPLACE TRIGGER triggering_radoi
BEFORE UPDATE OF numar_instrumente ON instrumente_muzicale
FOR EACH ROW
BEGIN
    IF (:NEW.numar_instrumente > :OLD.numar_instrumente) THEN
        RAISE_APPLICATION_ERROR(-20002,'numarul de instrumente nu poate fi marit');
    END IF;
END;
/
```

```
UPDATE instrumente_muzicale
SET numar_instrumente = numar_instrumente + 100;
```

```
DROP TRIGGER triggering_radoi;
```

The screenshot shows a database query editor with a 'Query Builder' tab. The SQL code is as follows:

```
select * from instrumente_muzicale;

CREATE OR REPLACE TRIGGER triggering_radoi
BEFORE UPDATE OF numar_instrumente ON instrumente_muzicale
FOR EACH ROW
BEGIN
    IF (:NEW.numar_instrumente > :OLD.numar_instrumente) THEN
        RAISE_APPLICATION_ERROR(-20002,'numarul de instrumente nu poate fi marit');
    END IF;
END;
/

UPDATE instrumente_muzicale
SET numar_instrumente = numar_instrumente + 100;

DROP TRIGGER triggering_radoi;
```

The 'Script Output' tab shows the following messages:

```
Trigger TRIGGERING_RADOI compiled

Error starting at line : 13 in command -
UPDATE instrumente_muzicale
SET numar_instrumente = numar_instrumente + 100
Error at Command Line : 13 Column : 8
Error report -
SQL Error: ORA-20002: numarul de instrumente nu poate fi marit
ORA-06512: at "GRUPA244.TRIGGERING_RADOI", line 3
ORA-04088: error during execution of trigger 'GRUPA244.TRIGGERING_RADOI'
```

12)Definiți un trigger de tip LDD. Declanșați trigger-ul.
Creați un trigger care confirma modificarea unui tabel.

```
CREATE OR REPLACE TRIGGER trigger_radoi_3
AFTER ALTER ON SCHEMA
BEGIN
    dbms_output.put_line('Un tabel a fost modificat: ' || SYS.DICTIONARY_OBJ_NAME);
END;
/
```

```
ALTER TABLE marca
ADD nume_marca2 varchar(15);
```

```
ALTER TABLE marca
DROP COLUMN nume_marca2;
```

```
select * from marca;
```

```
drop trigger trigger_radoi_3;
```

The screenshot displays the Oracle SQL Developer interface. The top pane, titled 'Query Builder', contains the following SQL script:

```
CREATE OR REPLACE TRIGGER trigger_radoi_3
AFTER ALTER ON SCHEMA
BEGIN
    dbms_output.put_line('Un tabel a fost modificat: ' || SYS.DICTIONARY_OBJ_NAME);
END;
/

ALTER TABLE marca
ADD nume_marca2 varchar(15);

ALTER TABLE marca
DROP COLUMN nume_marca2;

select * from marca;

drop trigger trigger_radoi_3;
```

The bottom pane, titled 'Script Output', shows the execution results:

```
Trigger TRIGGER_RADOI_3 compiled

Table MARCA altered.

Table MARCA altered.
```

Below the 'Script Output' pane is the 'Dbms Output' section, which shows the output of the trigger:

```
Un tabel a fost modificat: MARCA

Un tabel a fost modificat: MARCA
```