

Proiect Sisteme de Gestionare a Bazelor de Date

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

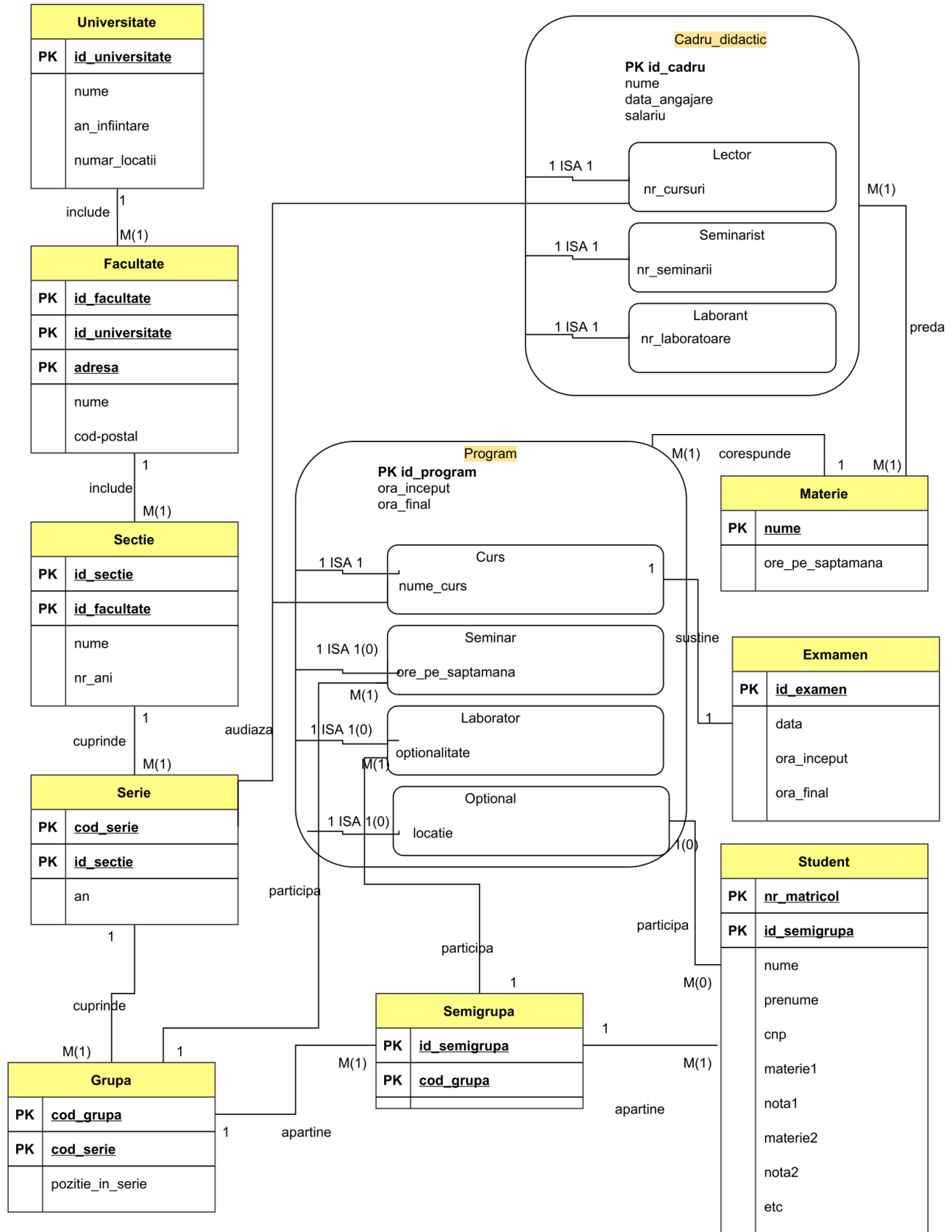
Modelul de date gestionează informații legate de organizarea și buna funcționare a universităților. Fiecare universitate este împărțită pe facultăți, serii, grupe, semigrupe, din care fac parte studenții. Ei participă la diverse programe (cursuri, seminarii, laboratoare, optionale), dar numai ale anumitor profesori. De asemenea, cursurile se soldează cu examene, iar studenții obțin mai multe note pe parcursul anului. Cadrele didactice sunt împărțite, de asemenea, pe categorii (lector, seminarist, laborant), predau una sau mai multe materii și le revine un salariu corespunzător.

Universitățile pot fi din diverse orașe ale țării și pot avea facultăți în mai multe locuri ale orașului. Astfel, adresa este specifică fiecărei facultăți în parte. Fiecare facultate are mai multe secții, ce cuprind serii formate dintr-un număr diferit de grupe, împărțite fiecare în două semigrupe. Împărțirea în acest mod este necesară, deoarece cursurile sunt audiate pe serii, o grupă întreagă participă la un seminar, laboratoarele sunt organizate pe semigrupe, iar fiecare student în parte își poate alege unul sau mai multe optionale. De asemenea, notele sunt specifice studenților, deoarece studenți diferiți pot primi număr diferit de note și la discipline diferite, în funcție de opțiuni.

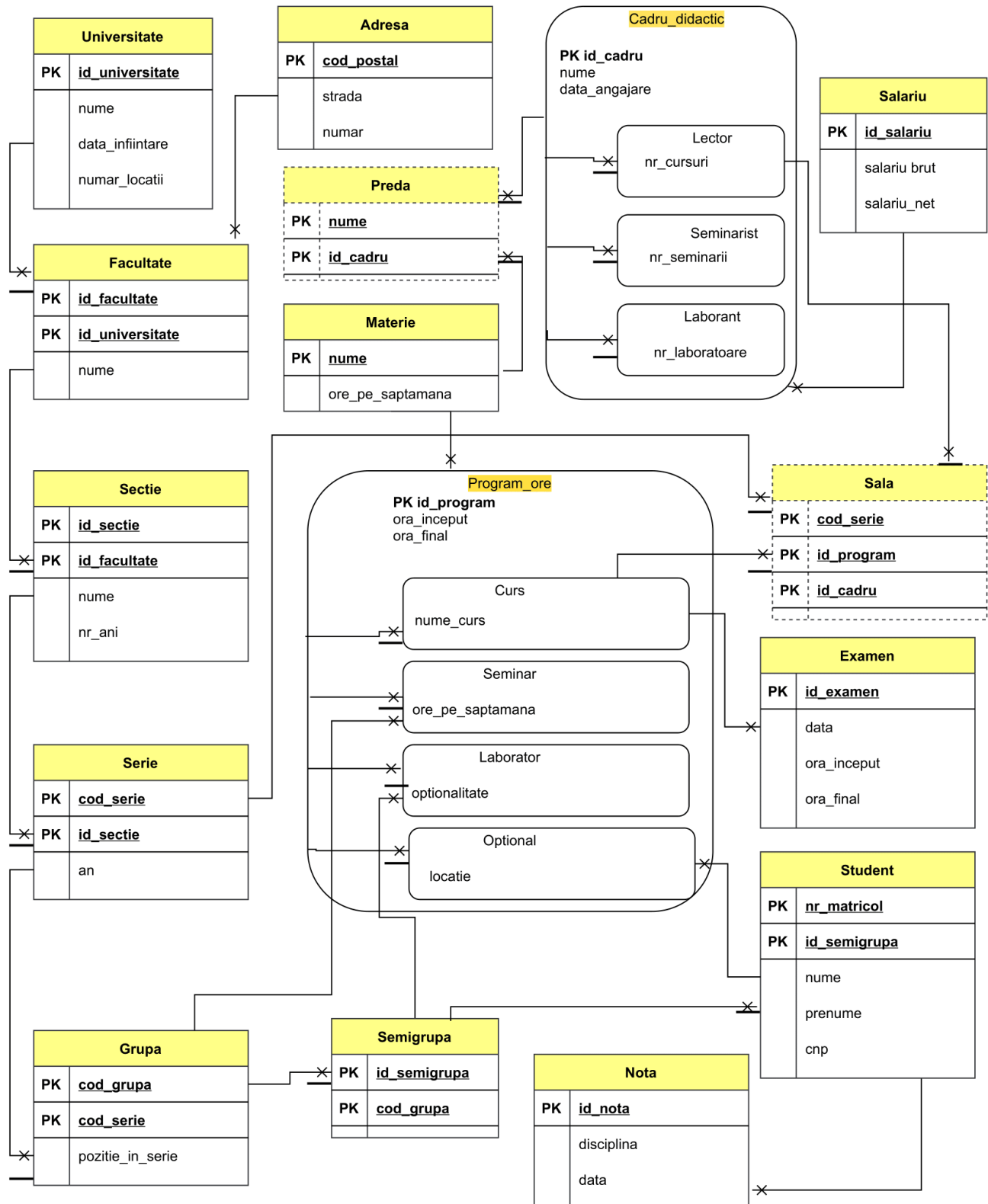
Cadrele didactice ale universității sunt lectori, seminariști și laboranți, dintre care orice tip de cadru poate susține optionale, și, de asemenea, oricine poate ține mai multe tipuri de programe la materii diferite.

O astfel de bază de date este necesară pentru a accesa cu ușurință informații despre studenți, profesori, cât și despre cursurile care se desfășoară în facultăți.

2. Realizați diagrama entitate-relație (ERD).



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



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

--CREARE TABELE

```
CREATE TABLE Universitate (  
    id_universitate NUMBER(10) PRIMARY KEY,  
    nume VARCHAR2(60) NOT NULL UNIQUE,  
    an_infiintare DATE NOT NULL,  
    ranking NUMBER(10),  
    numar_locatii NUMBER(10)  
    CONSTRAINT chk_facult CHECK (numar_locatii > 0)  
);
```

```
CREATE TABLE Adresa (  
    cod_postal NUMBER(10) PRIMARY KEY,  
    strada VARCHAR2(60) NOT NULL,  
    numar NUMBER(10),  
    CONSTRAINT chk_numar CHECK (numar>0)  
);
```

```
CREATE TABLE Facultate (  
    id_facultate NUMBER(10) UNIQUE NOT NULL,  
    id_universitate NUMBER(10) NOT NULL,  
    CONSTRAINT pk_fac PRIMARY KEY(id_facultate, id_universitate),  
    CONSTRAINT fkey_fac_univ FOREIGN KEY (id_universitate) REFERENCES Universitate(id_universitate) ON  
DELETE SET NULL,  
    nume VARCHAR2(60) NOT NULL,  
    nr_secretari NUMBER(10),  
    CONSTRAINT chk_secretari CHECK (nr_secretari >=0),  
    cod_postal NUMBER(10)  
    CONSTRAINT fkey_adresa REFERENCES Adresa(cod_postal)  
);
```

```
CREATE TABLE Sectie (  
    id_sectie NUMBER(10) UNIQUE NOT NULL,  
    id_facultate NUMBER(10) NOT NULL,  
    CONSTRAINT pk_sectie PRIMARY KEY (id_sectie, id_facultate),  
    CONSTRAINT fkey_sectie_fac FOREIGN KEY (id_facultate) REFERENCES Facultate(id_facultate) ON DELETE  
SET NULL,  
    denumire VARCHAR2(60) NOT NULL,  
    numar_ani NUMBER(10),  
    nr_programe_master NUMBER(10) DEFAULT(NULL)  
);
```

```
CREATE TABLE Serie (  
    cod_serie NUMBER(10) UNIQUE NOT NULL,  
    id_sectie NUMBER(10) NOT NULL,  
    an NUMBER(10),  
    coordonator_serie VARCHAR2(40),  
    CONSTRAINT pk_serie PRIMARY KEY (cod_serie, id_sectie),  
    CONSTRAINT fkey_serie_sectie FOREIGN KEY (id_sectie) REFERENCES Sectie(id_sectie) ON DELETE SET  
NULL,  
    CONSTRAINT chk_an CHECK (0 < an AND an < 5)  
);
```

```
CREATE TABLE Grupa (  
    cod_grupa NUMBER(10) UNIQUE NOT NULL,  
    cod_serie NUMBER(10) NOT NULL,
```

```

    pozitie_in_serie NUMBER(10),
    CONSTRAINT pk_grupa PRIMARY KEY (cod_grupa, cod_serie),
    CONSTRAINT fkey_grupa_serie FOREIGN KEY (cod_serie) REFERENCES Serie(cod_serie) ON DELETE SET
    NULL
);

```

```

CREATE TABLE Semigrupa (
    id_semigrupa NUMBER(10) UNIQUE NOT NULL,
    cod_grupa NUMBER(10) NOT NULL,
    saptamana VARCHAR2(10),
    CONSTRAINT pk_semigrupa PRIMARY KEY (id_semigrupa, cod_grupa),
    CONSTRAINT fkey_semi_grupa FOREIGN KEY (id_semigrupa) REFERENCES Semigrupa (id_semigrupa) ON
    DELETE SET NULL,
    CONSTRAINT chk_sapt CHECK (saptamana = 'par' OR saptamana = 'impar')
);

```

```

CREATE TABLE Student (
    nr_matricol NUMBER(10) UNIQUE NOT NULL,
    id_semigrupa NUMBER(10)
    CONSTRAINT fkey_semi REFERENCES Semigrupa (id_semigrupa),
    nume VARCHAR2(30) NOT NULL,
    prenume VARCHAR2 (30) NOT NULL,
    cnp NUMBER(10) NOT NULL UNIQUE,
    data_nastere DATE NOT NULL,
    CONSTRAINT pk_student PRIMARY KEY (nr_matricol, id_semigrupa)
);

```

```

CREATE TABLE Nota (
    id_nota NUMBER(10) PRIMARY KEY,
    disciplina VARCHAR2(30) NOT NULL,
    data_nota DATE DEFAULT (sysdate)
);

```

```

CREATE TABLE Examen (
    id_examen NUMBER(10) PRIMARY KEY,
    data_examen DATE DEFAULT(sysdate),
    ora_inceput NUMBER(10),
    ora_final NUMBER(10)
);

```

```

CREATE TABLE Materie (
    nume VARCHAR2(30) PRIMARY KEY,
    ore_pe_saptamana NUMBER(10),
    CONSTRAINT chk_ore_sapt CHECK (ore_pe_saptamana > 1)
);

```

```

CREATE TABLE Cadru_didactic(
    id_cadru NUMBER(10) PRIMARY KEY,
    nume VARCHAR2(20),
    prenume VARCHAR2(20),
    data_angajare DATE NOT NULL,
    data_nastere DATE NOT NULL,
    tip_cadru VARCHAR2(20) NOT NULL
    CONSTRAINT chk_tip CHECK (tip_cadru IN ('lector', 'seminarist', 'laborant')),
    nr_cursuri NUMBER(10) DEFAULT(NULL),
    nr_seminarii NUMBER(10) DEFAULT(NULL),
    nr_laboratoare NUMBER(10) DEFAULT(NULL)
);

```

```

CREATE TABLE Salariu (
    id_cadru NUMBER(10)
    CONSTRAINT fkey_cadru REFERENCES Cadru_didactic (id_cadru) ON DELETE SET NULL,
    id_salariu NUMBER (10),
    CONSTRAINT pk_salariu PRIMARY KEY (id_cadru, id_salariu),
    salariu_brut NUMBER (10),
    salariu_net NUMBER (10)
);

```

```

CREATE TABLE Preda (
    nume VARCHAR2(30),
    id_cadru NUMBER(10),
    CONSTRAINT pkey_preda PRIMARY KEY (nume, id_cadru),
    CONSTRAINT fkey_preda_materie FOREIGN KEY (nume) REFERENCES Materie(nume) ON DELETE SET NULL,
    CONSTRAINT fkey_preda_cadru FOREIGN KEY (id_cadru) REFERENCES Cadru_didactic(id_cadru) ON DELETE
SET NULL
);

```

```

CREATE TABLE Program_ore (
    id_program NUMBER (10) PRIMARY KEY,
    ora_inceput NUMBER(10),
    ora_final NUMBER(10),
    zi DATE DEFAULT(sysdate),
    tip_program VARCHAR2(20) NOT NULL
    CONSTRAINT chk_tip_ore CHECK (tip_program IN ('curs', 'seminar', 'laborator', 'optional')),
    nume_curs VARCHAR2(10) DEFAULT(NULL),
    ore_pe_saptamana NUMBER(10) DEFAULT(NULL),
    optionalitate VARCHAR2(5)
    CONSTRAINT chk_optional CHECK (optionalitate IN (NULL, 'da', 'nu')),
    locatie VARCHAR2(20),
    nume VARCHAR2(30)
    CONSTRAINT fkey_ore_materie REFERENCES Materie(nume),
    id_examen NUMBER(10)
    CONSTRAINT fkey_ore_examen REFERENCES Examen(id_examen),
    cod_grupa NUMBER(10)
    CONSTRAINT fkey_ore_grupa REFERENCES Grupa(cod_grupa),
    id_semigrupa NUMBER(10)
    CONSTRAINT fkey_ore_semigrupa REFERENCES Semigrupa( id_semigrupa),
    nr_matricol NUMBER(10)
    CONSTRAINT fkey_ore_student REFERENCES Student(nr_matricol)
);

```

```

CREATE TABLE Sala (
    id_program NUMBER(10) NOT NULL
    CONSTRAINT fkey_sala_program REFERENCES Program_ore(id_program),
    id_cadru NUMBER (10) NOT NULL
    CONSTRAINT fkey_sala_cadru REFERENCES Cadru_didactic(id_cadru),
    cod_serie NUMBER (10) NOT NULL
    CONSTRAINT fkey_sala_serie REFERENCES Serie (cod_serie),
    CONSTRAINT pk_sala PRIMARY KEY (id_program, id_cadru, cod_serie)
);

```

```

ALTER TABLE Adresa
ADD Oras VARCHAR2(20);

```

```

ALTER TABLE FACULTATE
ADD COD_POSTAL NUMBER(10);

```

```
ALTER TABLE fACULTATE
ADD CONSTRAINT fkey_adresa FOREIGN KEY (cod_postal) REFERENCES Adresa (cod_postal);
```

```
ALTER TABLE Nota
ADD nr_matricol NUMBER(10) CONSTRAINT fkey_student REFERENCES Student(nr_matricol);
```

```
ALTER TABLE Nota
ADD nota NUMBER(10);
```

```
ALTER TABLE Examen
ADD id_program NUMBER(10) CONSTRAINT fkey_curs REFERENCES Program_ore(id_program);
```

```
ALTER TABLE Program_ore
DROP COLUMN id_examen;
```

```
ALTER TABLE Cadru_didactic
DROP COLUMN nr_cursuri;
```

```
ALTER TABLE Cadru_didactic
ADD nr_cursuri_sapt NUMBER(10);
select * from universitate;
```

```
ALTER TABLE PROGRAM_ORE
MODIFY zi VARCHAR2(15) CONSTRAINT chk_zi CHECK (zi IN ('luni','marti','miercuri','joi','vineri'));
```

```
ALTER TABLE PROGRAM_ORE
DROP CONSTRAINT chk_zi;
```

```
alter table program_ore
rename column nume_curs to prescurtare_curs;
```

```
alter table universitate
rename column an_infiintare to data_infiintare;
```

```
ALTER TABLE salariu
DROP column id_cadru;
```

```
ALTER TABLE CADRU_DIDACTIC
ADD id_salariu CONSTRAINT fkey_salariu REFERENCES Salariu(id_salariu);
```

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

```
INSERT INTO Universitate (id_universitate, nume, an_infiintare, ranking, numar_locatii)
VALUES (100, 'Universitatea din Bucuresti', to_date('04/07/1864','DD/MM/YYYY'), 235, 5);
INSERT INTO Universitate (id_universitate, nume, an_infiintare, ranking, numar_locatii)
VALUES (101, 'Universitatea Babes-Bolyai din Cluj', to_date('12/05/1919','DD/MM/YYYY'), 267, 2);
INSERT INTO Universitate (id_universitate, nume, an_infiintare, ranking, numar_locatii)
VALUES (102, 'Universitatea Politehnica din Bucuresti', to_date('10/06/1920','DD/MM/YYYY'), 301, 2);
INSERT INTO Universitate (id_universitate, nume, an_infiintare, ranking, numar_locatii)
VALUES (103, 'Universitatea de Studii Economice Bucuresti', to_date('06/04/1913','DD/MM/YYYY'), 572, 1);
INSERT INTO Universitate (id_universitate, nume, an_infiintare, ranking, numar_locatii)
VALUES (104, 'Universitatea de Vest din Timisoara', to_date('15/02/1944','DD/MM/YYYY'), 450, 3);
INSERT INTO Universitate VALUES (105, 'Universitatea de medicina si farmacie', to_date('09/09/1903',
'DD/MM/YYYY'), 105, 1);
```

```
INSERT INTO ADRESA (cod_postal, strada, numar, oras)
```

```

VALUES (010014, 'Strada Academiei', 14, 'Bucuresti');
INSERT INTO ADRESA (cod_postal, strada, numar, oras)
VALUES (050663, 'Soseaua Panduri', 90, 'Bucuresti');
INSERT INTO ADRESA (cod_postal, strada, numar, oras)
VALUES (060024, 'Splaiul Independentei', 204, 'Bucuresti');
INSERT INTO ADRESA (cod_postal, strada, numar, oras)
VALUES (010014, 'Bulevardul Mihail Kogalniceanu', 36, 'Bucuresti');
INSERT INTO ADRESA (cod_postal, strada, numar, oras)
VALUES (060042, 'Splaiul Independentei', 313, 'Bucuresti');
INSERT INTO ADRESA (cod_postal, strada, numar, oras)
VALUES (015014, 'Strada Mihail Kogalniceanu', 1, 'Cluj-Napoca');
INSERT INTO Adresa (cod_postal, strada, numar, oras)
VALUES (034579, 'Strada Caderea Bastiliei', 3, 'Bucuresti');
INSERT INTO Adresa (cod_postal, strada, numar, oras)
VALUES (300223, 'Bulevardul Vasile Parvan', 4, 'Timisoara');
SELECT * FROM ADRESA;

```

```

INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (1, 100, 'Facultatea de Matematica si Informatica', 10, 010014);
INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (2, 100, 'Facultatea de Drept', 7, 010014);
INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (3, 100, 'Facultatea de Filosofie', 3, 060024);
INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (4, 102, 'Facultatea de Automatica si Calculatoare', 15, 060042);
INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (5, 102, 'Facultatea de Stiinte Aplicate', 12, 060042);
INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (6, 104, 'Facultatea de Limbi Moderne Aplicate', 9, 300223);
INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (7, 101, 'Cibernetica', 5, 034579);
INSERT INTO Facultate (id_facultate, id_universitate, nume, nr_secretari, cod_postal)
VALUES (8, 103, 'Facultatea de Fizica', 9, 015010);
SELECT * FROM FACULTATE;

```

```

INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (1, 1, 'Informatica', 3, 2);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (2, 1, 'Matematica', 3, 1);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (3, 1, 'CTI', 4, 1);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (4, 2, 'Drept Civil', 3, 3);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (5, 2, 'Avocatura', 4, 1);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (6, 3, 'Filosofie generala', 3, 2);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (7, 3, 'Istoria filosofiei', 3, 1);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (8, 4, 'CTI', 4, 3);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (9, 4, 'Ingineria Sistemelor', 4, 3);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (10, 6, 'Engleza', 3, 1);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)
VALUES (11, 6, 'Germana', 3, 1);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_ani, nr_programe_master)

```



```
VALUES (12, 5, 'Chimie', 3, 2);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_an, nr_programe_master)
VALUES (13, 5, 'Biologie', 3, 2);
INSERT INTO Sectie (id_sectie, id_facultate, denumire, numar_an, nr_programe_master)
VALUES (14, 7, 'Informatica economica', 3, 1);
select * from sectie;
```

```
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (11, 2, 1, 'Vasile Dumitru');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (12, 2, 1, 'Vasile Dumitru');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (13, 1, 1, 'Popescu Maria');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (14, 1, 1, 'Rosu Macela');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (15, 1, 1, 'Dan Gabriel');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (21, 2, 2, 'Mirela Gabriela');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (22, 1, 2, 'Negru Ioan');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (23, 1, 2, 'Marin Irina');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (31, 2, 3, 'Alex Mihai');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (32, 1, 3, 'Ionut Teodorescu');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (16, 3, 1, 'Marin Irina');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (26, 3, 2, 'Alex Mihai');
INSERT INTO Serie (cod_serie, id_sectie, an, coordonator_serie)
VALUES (36, 3, 3, 'Ionut Teodorescu');
SELECT * FROM SERIE;
```

```
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (141, 14, 1);
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (142, 14, 2);
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (143, 14, 3);
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (151, 15, 1);
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (152, 15, 2);
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (211, 21, 1);
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (212, 21, 2);
INSERT INTO Grupa (cod_grupa, cod_serie, pozitie_in_serie)
VALUES (213, 21, 3);
INSERT INTO Grupa VALUES (131, 13, 1);
INSERT INTO Grupa VALUES (133, 13, 3);
SELECT * FROM GRUPA;
```

```
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1411, 141, 'impar');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
```

```

VALUES (1412, 141, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1421, 142, 'impar');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1422, 142, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1431, 143, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1432, 143, 'impar');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1511, 151, 'impar');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1512, 151, 'impar');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1521, 152, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1452, 152, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (2111, 211, 'impar');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (2112, 211, 'impar');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (2121, 212, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (1222, 212, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (2131, 213, 'par');
INSERT INTO Semigrupa (id_semigrupa, cod_grupa, saptamana)
VALUES (2132, 213, 'impar');
insert into semigrupa values (20, 211, 'par');
INSERT INTO Semigrupa VALUES(1311, 131, 'par');
INSERT INTO Semigrupa VALUES (1312, 131, 'impar');
INSERT INTO Semigrupa VALUES (1331, 133, 'impar');

```

```

INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (42, 1431, 'Gherghescu', 'Andreea', 5010506442, to_date('27/09/2001','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (45, 1431, 'Rusu', 'Rares', 6010506492, to_date('05/09/2001','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (87, 1432, 'Ionescu', 'Pavel', 5270506448, to_date('14/10/2001','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (12, 1432, 'Popescu', 'Ion', 5010516442, to_date('20/11/2001','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (65, 1511, 'Georgescu', 'Laura', 6015506436, to_date('06/02/2001','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (63, 1511, 'Marinescu', 'Irina', 5410506446, to_date('27/10/2001','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (32, 2131, 'Iliescu', 'Raluca', 5019907442, to_date('21/04/2000','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (99, 2121, 'Radulescu', 'Andrei', 6980506412, to_date('22/05/2000','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (74, 1521, 'Mihailescu', 'Eduard', 5010504140, to_date('01/09/2000','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (44, 2132, 'Petrescu', 'Alex', 5018307142, to_date('08/01/2001','DD/MM/YYYY'));
INSERT INTO Student (nr_matricol, id_semigrupa, nume, prenume, cnp, data_nastere)
VALUES (23, 2132, 'Teodorescu', 'Iulian', 6077524135, to_date('19/12/2000','DD/MM/YYYY'));
insert into student values (10, 20, 'Marin', 'George', 5623458962, to_date('10/10/2000','DD/MM/YYYY'));
insert into student values (11, 20, 'Stan', 'Alexandra', 5248965326, to_date('30/08/2001','DD/MM/YYYY'));

```

```
select * from student;
```

```
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (127, 'Baze de date', to_date('12/12/2020','DD/MM/YYYY'), 99, 10);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (547, 'Programarea algoritmilor', to_date('10/12/2020','DD/MM/YYYY'), 99, 8);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (836, 'Logica matematica', to_date('12/12/2020','DD/MM/YYYY'), 12, 7);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (385, 'POO', to_date('27/02/2021','DD/MM/YYYY'), 74, 5);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (184, 'Algebra', to_date('13/03/2021','DD/MM/YYYY'), 44, 3);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (563, 'Gandire critica', to_date('01/01/2021','DD/MM/YYYY'), 87, 6);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (978, 'Baze de date', to_date('01/02/2021','DD/MM/YYYY'), 12, 10);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (355, 'Tehnici WEB', to_date('20/04/2021','DD/MM/YYYY'), 65, 9);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (586, 'POO', to_date('29/05/2021','DD/MM/YYYY'), 65, 9);
INSERT INTO Nota (id_nota, disciplina, data_nota, nr_matricol, nota)
VALUES (552, 'Algebra', to_date('23/03/2021','DD/MM/YYYY'), 23, 6);
select * from nota;
```

```
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('Baze de date', 4);
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('Logica matematica', 2);
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('Algebra', 4);
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('POO', 5);
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('Programarea algoritmilor', 3);
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('Gandire critica', 2);
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('Analiza', 3);
INSERT INTO Materie (nume, ore_pe_saptamana)
VALUES ('Tehnici WEB', 2);
select * from materie;
```

```
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_cursuri_sapt,
id_salariu)
VALUES (21, 'Enache', 'Daniel', sysdate, to_date('23/08/1979','DD/MM/YYYY'), 'lector', 2, 4);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_cursuri_sapt,
id_salariu)
VALUES (11, 'Popa', 'Ionut', to_date('01/10/2011','DD/MM/YYYY'), to_date('16/11/1978','DD/MM/YYYY'), 'lector', 3, 4);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_cursuri_sapt,
id_salariu)
VALUES (50, 'Anghel', 'Irina', to_date('03/07/2018','DD/MM/YYYY'), to_date('24/06/1990','DD/MM/YYYY'), 'lector', 1,
5);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_cursuri_sapt,
id_salariu)
VALUES (37, 'Marinescu', 'Marin', to_date('01/09/2009','DD/MM/YYYY'), to_date('01/04/1982','DD/MM/YYYY'), 'lector',
1, 4);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_seminarii, id_salariu)
```

```

VALUES (25, 'Dobre', 'Adelin', to_date('07/05/2018','DD/MM/YYYY'), to_date('11/05/1987','DD/MM/YYYY'),
'seminarist', 2, 2);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_seminarii, id_salariu)
VALUES (14, 'Buica', 'Iustin', to_date('21/07/2014','DD/MM/YYYY'), to_date('20/01/1990','DD/MM/YYYY'), 'seminarist',
3, 2);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_seminarii, id_salariu)
VALUES (71, 'Bruma', 'Adrian', to_date('15/01/2013','DD/MM/YYYY'), to_date('30/11/1993','DD/MM/YYYY'),
'seminarist', 4, 4);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_seminarii, id_salariu)
VALUES (39, 'Negru', 'Alexandra', to_date('17/09/2017','DD/MM/YYYY'), to_date('05/07/1986','DD/MM/YYYY'),
'seminarist', 6, 3);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_laboratoare,
id_salariu)
VALUES (5, 'Safta', 'Calin', sysdate, to_date('15/10/1987','DD/MM/YYYY'), 'laborant', 2, 3);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_laboratoare,
id_salariu)
VALUES (12, 'Lazar', 'Teodor', to_date('21/06/2020','DD/MM/YYYY'), to_date('04/04/1992','DD/MM/YYYY'), 'laborant',
1, 1);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_laboratoare,
id_salariu)
VALUES (54, 'Nae', 'Nicolae', to_date('03/04/2019','DD/MM/YYYY'), to_date('18/07/1995','DD/MM/YYYY'), 'laborant', 3,
2);
INSERT INTO Cadru_didactic (id_cadru, nume, prenume, data_angajare, data_nastere, tip_cadru, nr_laboratoare,
id_salariu)
VALUES (27, 'Nasture', 'Cosmin', to_date('03/01/2021','DD/MM/YYYY'), to_date('21/09/1999','DD/MM/YYYY'),
'laborant', 4, 1);
INSERT INTO Cadru_didactic VALUES (2, 'Popa', 'Mirela', sysdate, to_date('27/03/1980', 'DD/MM/YYYY'), 'lector',
NULL, NULL, 1, 4);
select * from cadru_didactic;
delete from cadru_didactic;

INSERT INTO Salariu (id_salariu, salariu_brut, salariu_net)
VALUES (1, 3194, 3098);
INSERT INTO Salariu (id_salariu, salariu_brut, salariu_net)
VALUES (2, 4000, 3500);
INSERT INTO Salariu (id_salariu, salariu_brut, salariu_net)
VALUES (3, 2900, 2500);
INSERT INTO Salariu (id_salariu, salariu_brut, salariu_net)
VALUES (4, 51020, 5000);
INSERT INTO Salariu (id_salariu, salariu_brut, salariu_net)
VALUES (5, 4070, 4000);

select * from materie;
select * from examen;
select * from cadru_didactic;

INSERT INTO Preda (nume, id_cadru)
VALUES ('Baze de date', 14);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Tehnici WEB', 5);
INSERT INTO Preda (nume, id_cadru)
VALUES ('POO', 27);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Logica matematica', 54);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Algebra', 50);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Analiza', 11);

```

```

INSERT INTO Preda (nume, id_cadru)
VALUES ('Baze de date', 39);
INSERT INTO Preda (nume, id_cadru)
VALUES ('POO', 12);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Tehnici WEB', 37);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Analiza', 25);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Programarea algoritmilor', 71);
INSERT INTO Preda (nume, id_cadru)
VALUES ('Programarea algoritmilor', 21);

```

```

select * from program_ore;

```

```

--cursuri

```

```

INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, nume_curs, nume)
VALUES (01, 10, 12, 'luni', 'curs', 'PA', 'Programarea algoritmilor');
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, nume_curs, nume)
VALUES (02, 8, 10, 'miercuri', 'curs', 'POO', 'POO');
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, nume_curs, nume)
VALUES (03, 12, 14, 'joi', 'curs', 'GAL', 'Algebra');
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, nume_curs, nume)
VALUES (04, 16, 18, 'luni', 'curs', 'BD', 'Baze de date');
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, nume_curs, nume)
VALUES (18, 12, 14, 'marti', 'curs', 'A', 'Analiza');
insert into program_ore values (29, 10, 13, 'luni', 'curs', 'POO', null, null, null, 'POO', null, null, null);
insert into program_ore values (30, 13, 17, 'marti', 'curs', 'LM', null, null, null, 'Logica matematica', null, null, null);
insert into program_ore values (31, 14, 17, 'luni', 'curs', 'TW', null, null, null, 'Tehnici WEB', null, null, null);
insert into program_ore values (32, 8, 12, 'joi', 'curs', 'GC', null, null, null, 'Gandire critica', null, null, null);
insert into program_ore values (33, 16, 20, 'miercuri', 'curs', 'GAL', null, null, null, 'Algebra', null, null, null);

```

```

--seminarii

```

```

INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, ore_pe_saptamana, nume, cod_grupa)
VALUES (05, 12, 14, 'marti', 'seminar', 2, 'Algebra', 143);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, ore_pe_saptamana, nume, cod_grupa)
VALUES (06, 14, 16, 'marti', 'seminar', 1, 'POO', 142);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, ore_pe_saptamana, nume, cod_grupa)
VALUES (07, 16, 18, 'vineri', 'seminar', 2, 'Logica matematica', 213);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, ore_pe_saptamana, nume, cod_grupa)
VALUES (08, 10, 12, 'luni', 'seminar', 3, 'Baze de date', 212);
insert into program_ore values (22, 11, 14, 'miercuri', 'seminar', null, 3, null, null, 'Algebra', 213, null, null);
insert into program_ore values (23, 12, 16, 'vineri', 'seminar', null, 2, null, null, 'POO', 143, null, null);
insert into program_ore values (24, 8, 11, 'luni', 'seminar', null, 4, null, null, 'Logica matematica', 143, null, null);
insert into program_ore values (25, 14, 16, 'luni', 'seminar', null, 2, null, null, 'POO', 213, null, null);
insert into program_ore values (26, 8, 11, 'marti', 'seminar', null, 2, null, null, 'POO', 152, null, null);
insert into program_ore values (27, 11, 14, 'marti', 'seminar', null, 3, null, null, 'Algebra', 152, null, null);
insert into program_ore values (28, 14, 17, 'marti', 'seminar', null, 2, null, null, 'Logica matematica', 152, null, null);

```

```

--laburi

```

```

INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, optionalitate, nume, id_semigrupa)
VALUES (09, 8, 10, 'joi', 'laborator', 'da', 'Analiza', 2121);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, optionalitate, nume, id_semigrupa)
VALUES (10, 16, 18, 'luni', 'laborator', 'nu', 'Tehnici WEB', 2111);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, optionalitate, nume, id_semigrupa)
VALUES (11, 18, 20, 'miercuri', 'laborator', 'nu', 'POO', 1411);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, optionalitate, nume, id_semigrupa)
VALUES (12, 10, 12, 'vineri', 'laborator', 'da', 'Baze de date', 1521);
insert into program_ore values (21, 12, 16, 'joi', 'laborator', null, null, 'nu', null, 'POO', null, 20, null);

```

--optionale

```
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, locatie, nume, nr_matricol)
VALUES (13, 18, 20, 'marti', 'optional', 'amfiteatru', 'Baze de date', 42);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, locatie, nume, nr_matricol)
VALUES (14, 18, 20, 'marti', 'optional', 'amfiteatru', 'Baze de date', 45);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, locatie, nume, nr_matricol)
VALUES (15, 18, 20, 'marti', 'optional', 'amfiteatru', 'Baze de date', 87);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, locatie, nume, nr_matricol)
VALUES (16, 18, 20, 'marti', 'optional', 'amfiteatru', 'Baze de date', 12);
INSERT INTO Program_ore (id_program, ora_inceput, ora_final, zi, tip_program, locatie, nume, nr_matricol)
VALUES (17, 18, 20, 'marti', 'optional', 'amfiteatru', 'Baze de date', 32);
insert into program_ore values (20, 16, 20, 'marti', 'optional', null, null, null, 'amfiteatru', 'POO', null, null, 11);
insert into program_ore values (19, 10, 13, 'luni', 'optional', null, null, null, 'amfiteatru', 'Baze de date', null, null, 10);
```

```
select * from student;
select * from program_ore;
```

```
INSERT INTO Examen (id_examen, data_examen, ora_inceput, ora_final, id_program)
VALUES (1, to_date('03/06/2021','DD/MM/YYYY'), 10, 12, 3);
INSERT INTO Examen (id_examen, data_examen, ora_inceput, ora_final, id_program)
VALUES (2, to_date('08/06/2021','DD/MM/YYYY'), 12, 15, 2);
INSERT INTO Examen (id_examen, data_examen, ora_inceput, ora_final, id_program)
VALUES (3, to_date('14/06/2021','DD/MM/YYYY'), 9, 12, 4);
INSERT INTO Examen (id_examen, data_examen, ora_inceput, ora_final, id_program)
VALUES (4, to_date('27/06/2021','DD/MM/YYYY'), 8, 10, 1);
INSERT INTO Examen (id_examen, data_examen, ora_inceput, ora_final, id_program)
VALUES (5, to_date('21/06/2021','DD/MM/YYYY'), 11, 14, 18);
```

```
select * from examen;
select * from program_ore;
select * from cadru_didactic;
select * from serie;
```

```
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (1, 11, 11);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (2, 50, 12);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (4, 37, 13);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (5, 25, 14);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (6, 14, 15);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (7, 71, 21);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (11, 39, 22);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (13, 5, 23);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (16, 12, 31);
INSERT INTO Sala (id_program, id_cadru, cod_serie)
VALUES (18, 54, 16);
insert into Sala (id_program, id_cadru, cod_serie)
values(3, 11, 12);
INSERT INTO Sala VALUES (20, 50 , 13);
select * from sala;
```

6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat care să utilizeze două tipuri de colecție studiate. Apelați subprogramul.

Creați o procedură care afișează toate numele facultăților pentru universitatea de pe poziția i (i dat ca parametru).

- id-urile universităților sunt stocate într-o colecție de tip vector, apoi facultățile universității de pe poziția i sunt stocate într-o colecție de tip tabel imbricat.

```
CREATE OR REPLACE PROCEDURE ex_6 (uni_nr universitate.id_universitate%TYPE)
IS
```

```
    TYPE ids IS VARRAY(6) OF universitate.id_universitate%TYPE;
    universitati ids;
```

```
    TYPE fac IS TABLE OF facultate%ROWTYPE;
    facultati fac := fac();
```

```
    cnt_uni NUMBER(2);
```

```
    NEGATIVE_NUMBER EXCEPTION;
    NO_DATA_FOUND EXCEPTION;
```

```
BEGIN
```

```
    IF uni_nr <= 0 THEN
        RAISE NEGATIVE_NUMBER;
    END IF;
```

```
    SELECT id_universitate
    BULK COLLECT INTO universitati
    FROM universitate;
```

```
    SELECT COUNT(*) INTO cnt_uni
    FROM universitate;
```

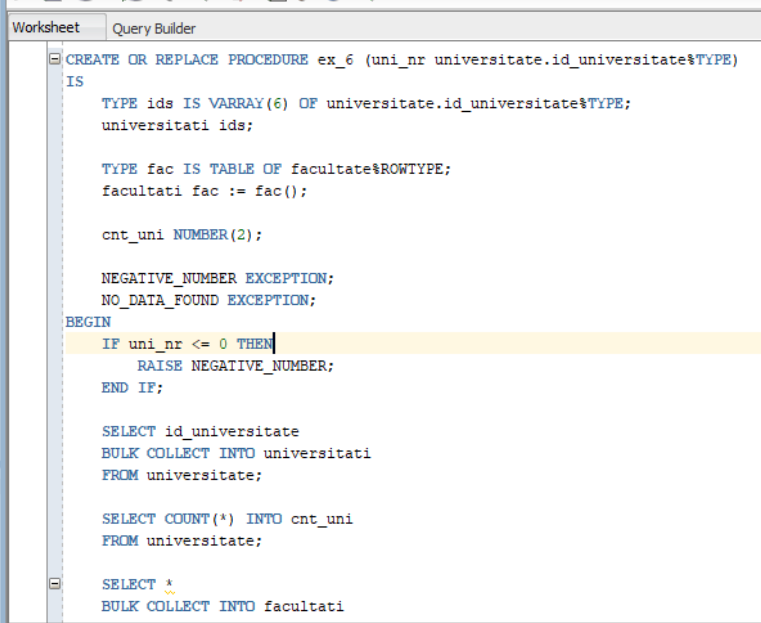
```
    SELECT *
    BULK COLLECT INTO facultati
    FROM facultate
    WHERE id_universitate = universitati(uni_nr);
```

```
    IF SQL%NOTFOUND THEN
        RAISE NO_DATA_FOUND;
    END IF;
```

```
    FOR i IN facultati.FIRST..facultati.LAST LOOP
        DBMS_OUTPUT.PUT_LINE(facultati(i).nume);
    END LOOP;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista facultati in universitatea de pe pozitia ' || uni_nr);
    WHEN NEGATIVE_NUMBER THEN
```



```
Worksheet  Query Builder
CREATE OR REPLACE PROCEDURE ex_6 (uni_nr universitate.id_universitate%TYPE)
IS
    TYPE ids IS VARRAY(6) OF universitate.id_universitate%TYPE;
    universitati ids;

    TYPE fac IS TABLE OF facultate%ROWTYPE;
    facultati fac := fac();

    cnt_uni NUMBER(2);

    NEGATIVE_NUMBER EXCEPTION;
    NO_DATA_FOUND EXCEPTION;
BEGIN
    IF uni_nr <= 0 THEN
        RAISE NEGATIVE_NUMBER;
    END IF;

    SELECT id_universitate
    BULK COLLECT INTO universitati
    FROM universitate;

    SELECT COUNT(*) INTO cnt_uni
    FROM universitate;

    SELECT *
    BULK COLLECT INTO facultati
```

Script Output x Query Result x

Task completed in 0.101 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Procedure EX_6 compiled

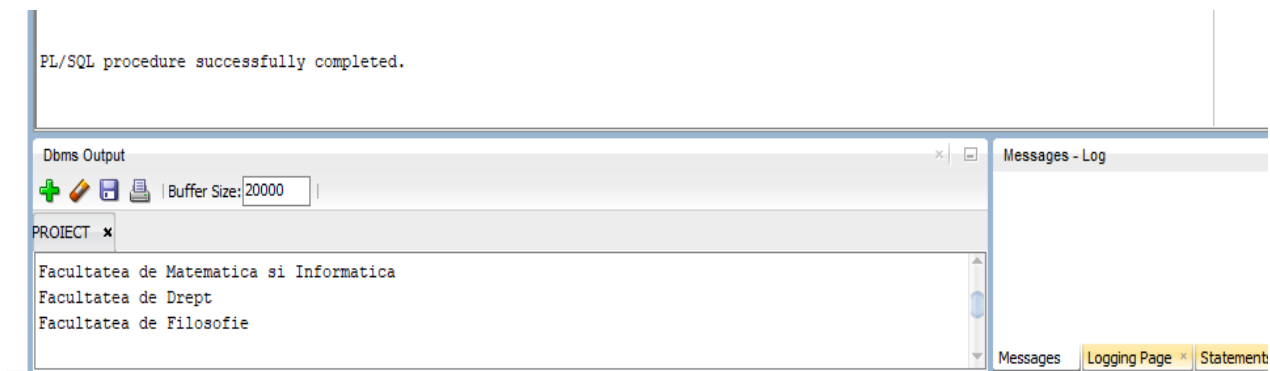
```

    DBMS_OUTPUT.PUT_LINE('Nu sunt permise valori negative');
WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE ('Codul erorii: ' || SQLCODE);
    DBMS_OUTPUT.PUT_LINE ('Mesajul erorii: ' || SQLERRM);
END ex_6;
/

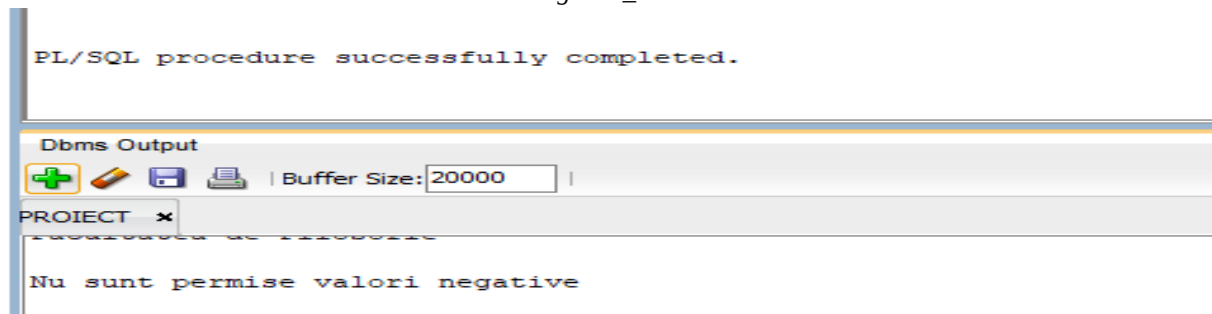
BEGIN
    ex_6 (1);
    --ex_6(-2); --NEGATIVE_NUMBER
    --ex_6(10); --OTHERS
END;
/

```

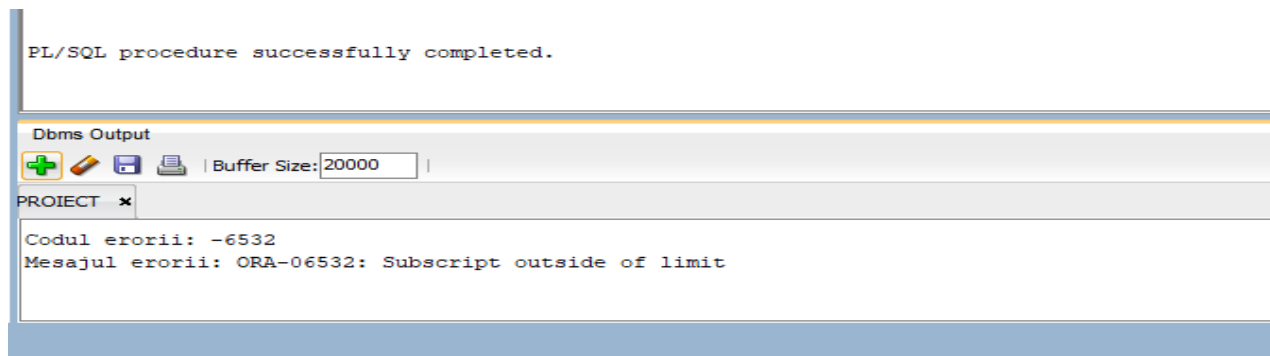
corect



negative_number



others



7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat care să utilizeze un tip de cursor studiat. Apelați subprogramul.

În funcție de o opțiune introdusă de la tastatură (1, 2 sau 3) să se deschidă un cursor astfel încât să se regăsească:

- informațiile tuturor studenților (opțiunea 1)
- informațiile studenților cu numele între 'A' și 'T' (opțiunea 2)
- informațiile studenților al căror prenume începe cu 'R' (opțiunea 3)

--cursor dinamic

--procedură stocată

CREATE OR REPLACE PROCEDURE ex_7 (opt NUMBER)

IS

TYPE student_tip IS REF CURSOR RETURN student%ROWTYPE;

std student_tip;

stud student%ROWTYPE;

BEGIN

IF opt = 1 THEN

OPEN std FOR SELECT *

FROM student;

ELSIF opt = 2 THEN

OPEN std FOR SELECT *

FROM student

WHERE lower(ume) >= 'a' AND lower(ume) < 'j';

ELSIF opt = 3 THEN

OPEN std FOR SELECT *

FROM student

WHERE lower(prenume) like 'r%';

ELSE

DBMS_OUTPUT.PUT_LINE('Opțiune incorectă');

END IF;

LOOP

FETCH std INTO stud;

EXIT WHEN std%NOTFOUND;

DBMS_OUTPUT.PUT_LINE(stud.ume || ' ' || stud.prenume);

END LOOP;

DBMS_OUTPUT.PUT_LINE ('Au fost procesate ' || std%ROWCOUNT || ' linii');

CLOSE std;

END;

/

BEGIN

--ex_7(1);

--ex_7(2);

ex_7(3);

END;

/

Worksheet Query Builder

```
CREATE OR REPLACE PROCEDURE ex_7 (opt NUMBER)
IS
    TYPE student_tip IS REF CURSOR RETURN student%ROWTYPE;
    std student_tip;
    stud student%ROWTYPE;
BEGIN
    IF opt = 1 THEN
        OPEN std FOR SELECT *
            FROM student;
    ELSIF opt = 2 THEN
        OPEN std FOR SELECT *
            FROM student
            WHERE lower(nume) >= 'a' AND lower(nume) < 'j';
    ELSIF opt = 3 THEN
        OPEN std FOR SELECT *
            FROM student
            WHERE lower(prenume) like 'r%';
    ELSE
        DBMS_OUTPUT.PUT_LINE('Optiune incorecta');
    END IF;

    LOOP
        FETCH std INTO stud;
        EXIT WHEN std%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE(stud.nume || ' ' || stud.prenume);
    END LOOP;
END;
```

Script Output x

Task completed in 0.064 seconds

PL/SQL procedure successfully completed.

Procedure EX_7 compiled

Dbms Output x

procedura

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROIECT x

Gherghescu Andreea
Rusu Rares
Ionescu Pavel
Popescu Ion
Georgescu Laura
Marinescu Irina
Iliescu Raluca
Radulescu Andrei
Mihailescu Eduard
Petrescu Alex
Teodorescu Iulian
Marin George
Stan Alexandra
Au fost procesate 13 linii

opțiunea 1 (13 linii)

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROIECT x

Marinescu Irina
Iliescu Raluca
Radulescu Andrei
Mihailescu Eduard
Petrescu Alex
Teodorescu Iulian
Marin George
Stan Alexandra
Au fost procesate 13 linii

Gherghescu Andreea
Ionescu Pavel
Georgescu Laura
Iliescu Raluca
Au fost procesate 4 linii

opțiunea 2 (4 linii)

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROIECT x

Teodorescu Iulian
Marin George
Stan Alexandra
Au fost procesate 13 linii

Gherghescu Andreea
Ionescu Pavel
Georgescu Laura
Iliescu Raluca
Au fost procesate 4 linii

Rusu Rares
Iliescu Raluca
Au fost procesate 2 linii

opțiunea 3 (2 linii)

8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Tratați toate excepțiile care pot apărea. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Creați o funcție care returnează pentru un lector, dat ca parametru prin id, la câte serii predă, Excepții tratate:

- nu există cadru didactic cu id-ul dat
- cadrul didactic dat ca parametru nu este lector
- id-ul dat este negativ
- lectorul nu predă la nicio serie
- altele

```
CREATE OR REPLACE FUNCTION ex_8 (nrc NUMBER) RETURN NUMBER
IS
```

```
    nr_s NUMBER;
    TYPE tabel IS TABLE OF cadru_didactic%ROWTYPE INDEX BY PLS_INTEGER;
    aux tabel;
    tip_cadru cadru_didactic.tip_cadru%TYPE;
    NEGATIVE_NUMBER EXCEPTION;
    WRONG_DATA EXCEPTION;
    NO_DATA_FOUND1 EXCEPTION;
    NO_DATA_FOUND2 EXCEPTION;
BEGIN
    if nrc < 0 THEN
        RAISE NEGATIVE_NUMBER;
    END IF;
    SELECT * BULK COLLECT INTO aux FROM cadru_didactic WHERE id_cadru = nrc;
    IF SQL%NOTFOUND THEN
        RAISE NO_DATA_FOUND1;
    END IF;

    SELECT COUNT (ser.cod_serie) INTO nr_s
    FROM serie ser JOIN sala sal ON (ser.cod_serie = sal.cod_serie)
        JOIN cadru_didactic c ON (c.id_cadru = sal.id_cadru)
    WHERE c.id_cadru = nrc;
```

```
    SELECT tip_cadru INTO tip_cadru
    FROM cadru_didactic
    WHERE id_cadru = nrc;
```

```
    IF nr_s = 0 THEN
        RAISE NO_DATA_FOUND2;
    ELSIF tip_cadru != 'lector' THEN
        RAISE WRONG_DATA;
    ELSE
        RETURN nr_s;
    END IF;
```

```
EXCEPTION
```

```
Worksheet Query Builder
CREATE OR REPLACE FUNCTION ex_8 (nrc NUMBER) RETURN NUMBER
IS
    nr_s NUMBER;
    TYPE tabel IS TABLE OF cadru_didactic%ROWTYPE INDEX BY PLS_INTEGER;
    aux tabel;
    tip_cadru cadru_didactic.tip_cadru%TYPE;
    NEGATIVE_NUMBER EXCEPTION;
    WRONG_DATA EXCEPTION;
    NO_DATA_FOUND1 EXCEPTION;
    NO_DATA_FOUND2 EXCEPTION;
BEGIN
    if nrc < 0 THEN
        RAISE NEGATIVE_NUMBER;
    END IF;
    SELECT * BULK COLLECT INTO aux FROM cadru_didactic WHERE id_cadru = nrc;
    IF SQL%NOTFOUND THEN
        RAISE NO_DATA_FOUND1;
    END IF;

    SELECT COUNT (ser.cod_serie) INTO nr_s
    FROM serie ser JOIN sala sal ON (ser.cod_serie = sal.cod_serie)
        JOIN cadru_didactic c ON (c.id_cadru = sal.id_cadru)
    WHERE c.id_cadru = nrc;

    SELECT tip_cadru INTO tip_cadru
    FROM cadru_didactic
    WHERE id_cadru = nrc;

    IF nr_s = 0 THEN
        RAISE NO_DATA_FOUND2;
    ELSIF tip_cadru != 'lector' THEN
        RAISE WRONG_DATA;
    ELSE
        RETURN nr_s;
    END IF;
EXCEPTION
```

Script Output x

Task completed in 0.075 seconds

PL/SQL procedure successfully completed.

Function EX_8 compiled

Dbms Output

```

WHEN NO_DATA_FOUND1 THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista cadru didactic cu id-ul ' || nrc);
    RETURN -1;
WHEN NO_DATA_FOUND2 THEN
    DBMS_OUTPUT.PUT_LINE('Lectorul nu preda la nicio serie');
    RETURN -1;
WHEN NEGATIVE_NUMBER THEN
    DBMS_OUTPUT.PUT_LINE('Nu sunt permise valori negative');
    RETURN -1;
WHEN WRONG_DATA THEN
    DBMS_OUTPUT.PUT_LINE('Cadrul didactic cu id-ul ' || nrc || ' nu este lector');
    RETURN -1;
WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE ('Codul erorii: ' || SQLCODE);
    DBMS_OUTPUT.PUT_LINE ('Mesajul erorii: ' || SQLERRM);
END;
/

```

```

DECLARE
    aux NUMBER;
BEGIN
    --aux := ex_8(-1); --NEGATIVE_NUMBER
    --aux := ex_8(21); --NO_DATA_FOUND2
    --aux := ex_8(11);
    --aux := ex_8(5); --WRONG_DATA
    aux := ex_8(1); --NO_DATA_FOUND1
    -- -1, 21 are 0, 11 are 2, 5 nu e lector
    IF aux > -1 THEN
        dbms_output.put_line('Lectorul preda la ' || aux || ' serii');
    END IF;
END;
/

```

```

DECLARE
    aux NUMBER;
BEGIN
    --aux := ex_8(-1); --NEGATIVE_NUMBER
    --aux := ex_8(21); --NO_DATA_FOUND2
    --aux := ex_8(11);
    --aux := ex_8(5); --WRONG_DATA
    aux := ex_8(1); --NO_DATA_FOUND1
    -- -1, 21 are 0, 11 are 2, 5 nu e lector
    IF aux > -1 THEN
        dbms_output.put_line('Lectorul preda la ' || aux || ' serii');
    END IF;
END;
/

```

Script Output x Query Result x

Task completed in 0.069 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROJECT x

Nu sunt permise valori negative

Lectorul nu preda la nicio serie

```

DECLARE
    aux NUMBER;
BEGIN
    --aux := ex_8(-1); --NEGATIVE_NUMBER
    --aux := ex_8(21); --NO_DATA_FOUND2
    --aux := ex_8(11);
    --aux := ex_8(5); --WRONG_DATA
    --aux := ex_8(1); --NO_DATA_FOUND1
    -- -1, 21 are 0, 11 are 2, 5 nu e lector
    IF aux > -1 THEN
        dbms_output.put_line('Lectorul preda la ' || aux || ' serii');
    END IF;
END;
/

```

Script Output x Query Result x

Task completed in 0.064 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROJECT x

Lectorul nu preda la nicio serie

Nu sunt permise valori negative

Cadrul didactic cu id-ul 5 nu este lector

```

DECLARE
    aux NUMBER;
BEGIN
    --aux := ex_8(-1); --NEGATIVE_NUMBER
    --aux := ex_8(21); --NO_DATA_FOUND2
    --aux := ex_8(11);
    --aux := ex_8(5); --WRONG_DATA
    aux := ex_8(1); --NO_DATA_FOUND1
    -- -1, 21 are 0, 11 are 2, 5 nu e lector
    IF aux > -1 THEN
        dbms_output.put_line('Lectorul preda la ' || aux || ' serii');
    END IF;
END;
/

```

Script Output x Query Result x

Task completed in 0.036 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROJECT x

Lectorul preda la 2 serii

Cadrul didactic cu id-ul 5 nu este lector

Nu exista cadru didactic cu id-ul 1

Script Output x Query Result x

Task completed in 0.067 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROJECT x

Lectorul preda la 2 serii

Cadrul didactic cu id-ul 5 nu este lector

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROJECT x

Lectorul nu preda la nicio serie

Lectorul preda la 2 serii

9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat 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.

Creați o procedură care să afișeze ce semigrupe participă la orele predate de un anumit lector dat ca parametru prin nume.

```
CREATE OR REPLACE PROCEDURE ex_9 (nume_l VARCHAR2)
IS
    TYPE tabel_ind_s IS TABLE OF semigrupa.id_semigrupa%TYPE INDEX BY PLS_INTEGER;
    sgr tabel_ind_s;
    aux cadru_didactic.id_cadru%TYPE;
    NO_DATA_FOUND1 EXCEPTION;
    NO_DATA_FOUND2 EXCEPTION;
    WRONG_DATA EXCEPTION;
BEGIN
    SELECT id_cadru INTO aux
    FROM cadru_didactic
    WHERE cadru_didactic.nume = nume_l AND cadru_didactic.tip_cadru = 'lector';

    SELECT id_semigrupa BULK COLLECT INTO sgr
    FROM semigrupa sem JOIN grupa g ON (sem.cod_grupa = g.cod_grupa)
        JOIN serie ser ON (g.cod_serie = ser.cod_serie)
        JOIN sala sal ON (ser.cod_serie = sal.cod_serie)
        JOIN cadru_didactic c ON (c.id_cadru = sal.id_cadru)
    WHERE c.nume = nume_l;

    IF sgr.COUNT = 0 THEN
        RAISE NO_DATA_FOUND2;
    END IF;

    DBMS_OUTPUT.PUT_LINE('Semigrupele la care preda lectorul cautat: ');
    FOR i IN sgr.FIRST..sgr.LAST LOOP
        DBMS_OUTPUT.PUT_LINE(sgr(i));
    END LOOP;

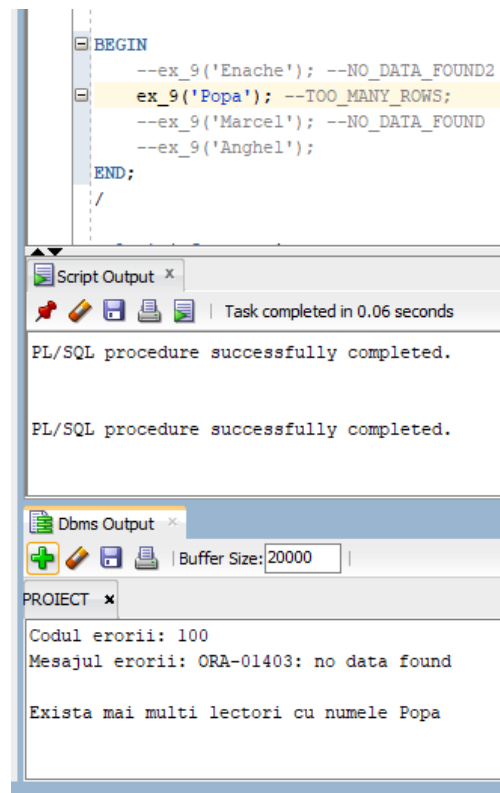
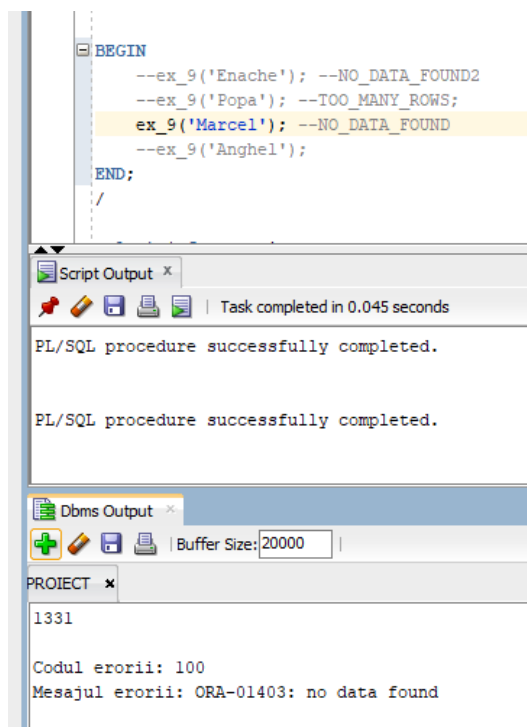
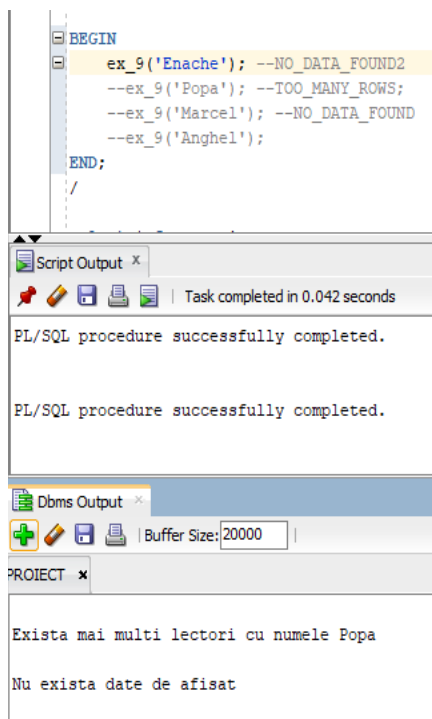
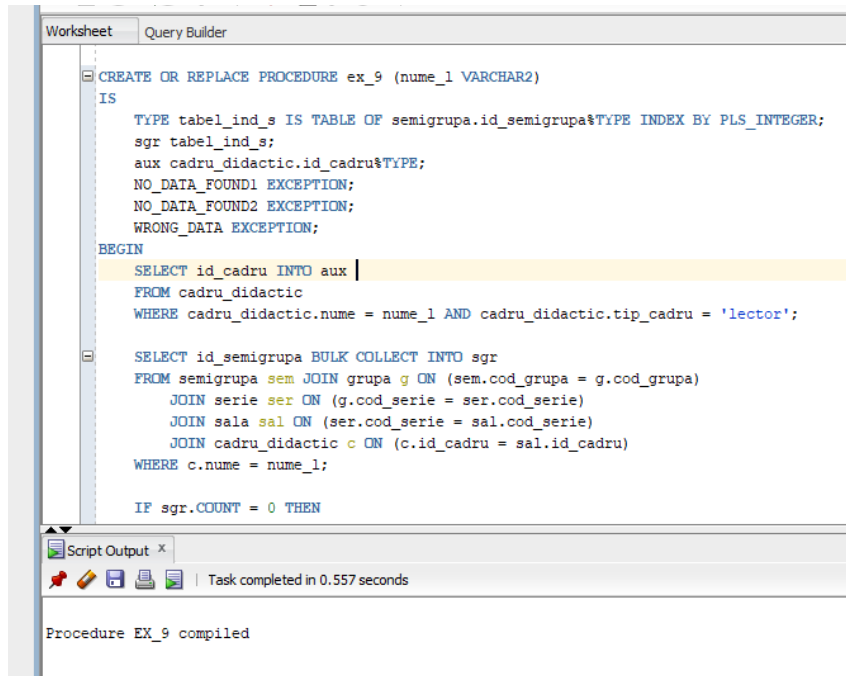
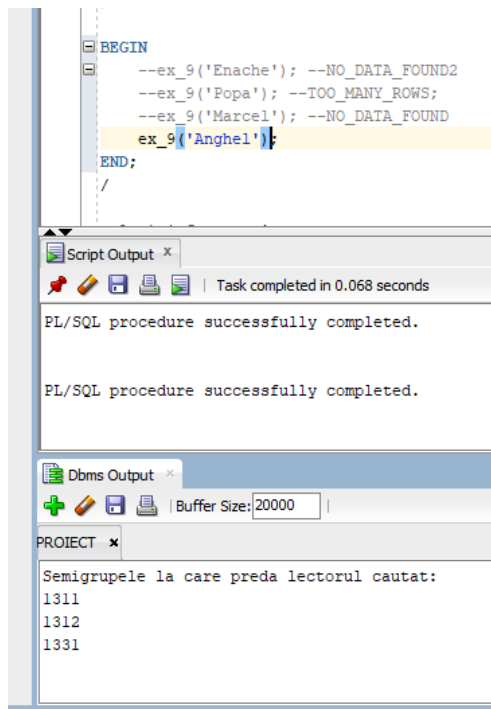
EXCEPTION
    WHEN NO_DATA_FOUND2 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista date de afisat');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multi lectori cu numele ' || nume_l);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE('Mesajul erorii: ' || SQLERRM);
END;
/

BEGIN
```

```

ex_9('Enache'); --NO_DATA_FOUND2
--ex_9('Popa'); --TOO_MANY_ROWS;
--ex_9('Marcel'); --NO_DATA_FOUND
--ex_9('Anghel');
END;

```



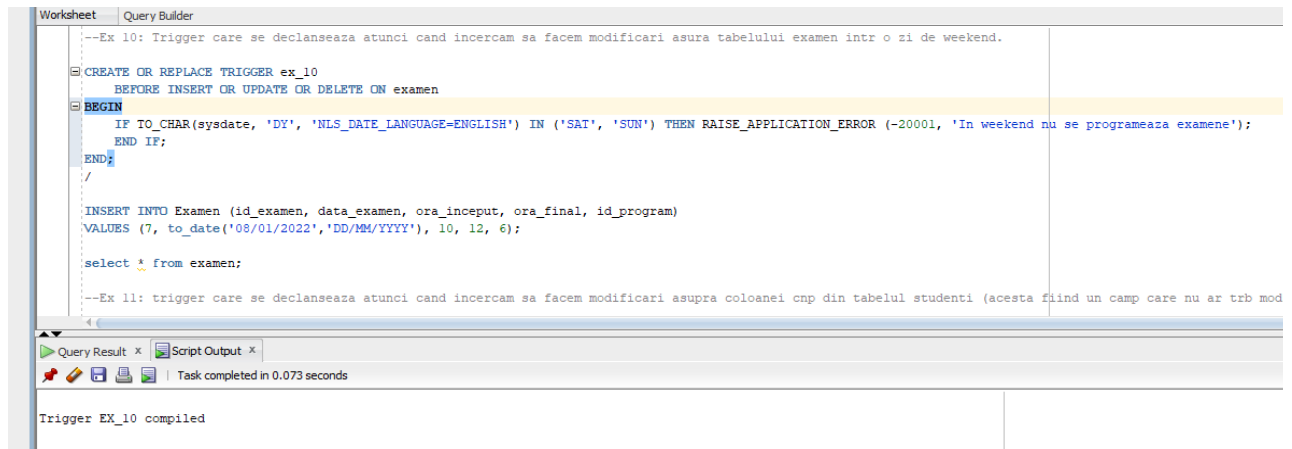
10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.

Trigger care se declanșează atunci când încercăm să facem modificări asupra tabelului examen într-o zi de weekend.

```
CREATE OR REPLACE TRIGGER ex_10
BEFORE INSERT OR UPDATE OR DELETE ON examen
BEGIN
    IF TO_CHAR(sysdate, 'DY', 'NLS_DATE_LANGUAGE=ENGLISH') IN ('SAT', 'SUN') THEN
        RAISE_APPLICATION_ERROR (-20001, 'In weekend nu se programeaza examene');
    END IF;
END;
/
```

```
INSERT INTO Examen (id_examen, data_examen, ora_inceput, ora_final, id_program)
VALUES (7, to_date('08/01/2022','DD/MM/YYYY'), 10, 12, 6);
```

```
select * from examen;
```



--am rulat trigger-ul cu joi si duminica

ID_EXAMEN	DATA_EXAMEN	ORA_INCEPUT	ORA_FINAL	ID_PROGRAM
1	5 21-JUN-21	11	14	18
2	1 03-JUN-21	10	12	3
3	2 08-JUN-21	12	15	2
4	3 14-JUN-21	9	12	4
5	4 27-JUN-21	8	10	1
6	6 08-JAN-22	10	12	5

după

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

Trigger care se declanșează atunci când încercăm să facem modificări asupra coloanei CNP din tabelul Studenti (acesta fiind un câmp care nu ar trebui modificat niciodată).

```
CREATE OR REPLACE TRIGGER ex_11
BEFORE UPDATE OF cnp ON student
FOR EACH ROW
WHEN (NEW.cnp <> OLD.cnp)
BEGIN
RAISE_APPLICATION_ERROR(-20002, 'nu puteti modifica cnp-ul unei persoane');
END;
/
```

```
UPDATE student SET cnp=0;
select * from student;
```

```
Error starting at line : 28 in command -
UPDATE student SET cnp=0
Error report -
ORA-20002: nu puteti modifica cnp-ul unei persoane
ORA-06512: at "ANDREEA.EX_11", line 2
ORA-04088: error during execution of trigger 'ANDREEA.EX_11'
```

The screenshot shows the SQL Developer interface. The top pane displays the trigger definition: `CREATE OR REPLACE TRIGGER ex_11 BEFORE UPDATE OF cnp ON student FOR EACH ROW WHEN (NEW.cnp <> OLD.cnp) BEGIN RAISE_APPLICATION_ERROR(-20002, 'nu puteti modifica cr END; /`. The bottom pane shows the execution results, including the error message: `ORA-20002: In weekend nu se programeaza examene ORA-06512: at "ANDREEA.EX_10", line 2 ORA-04088: error during execution of trigger 'ANDREEA.EX_10'`.

NR_MATRICOL	ID_SEMIGRUPA	NUME	PRENUME	CNP	DATA_NASTERE
1	42	1431 Gherghescu	Andreea	5010506442	27-SEP-01
2	45	1431 Rusu	Rares	6010506492	05-SEP-01
3	87	1432 Ionescu	Pavel	5270506448	14-OCT-01
4	12	1432 Popescu	Ion	5010516442	20-NOV-01
5	65	1511 Georgescu	Laura	6015506436	06-FEB-01
6	63	1511 Marinescu	Irina	5410506446	27-OCT-01
7	32	2131 Iliescu	Raluca	5019907442	21-APR-00
8	99	2121 Radulescu	Andrei	6980506412	22-MAY-00
9	74	1521 Mihailescu	Eduard	5010504140	01-SEP-00
10	44	2132 Petrescu	Alex	5018307142	08-JAN-01
11	23	2132 Teodorescu	Iulian	6077524135	19-DEC-00
12	10	20 Marin	George	5623458962	10-OCT-00
13	11	20 Stan	Alexandra	5248965326	30-AUG-01

după

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

Trigger care se declanșează de fiecare dată când se execută o operație de tip LDD inserând în tabela nou creată un log cu detaliile operației.

```
CREATE TABLE Audit_user (
nume_bd VARCHAR2(50),
```



```

user_logat VARCHAR2(50),
eveniment VARCHAR2(50),
tip_obiect_referit VARCHAR2(50),
nume_obiect_referit VARCHAR2(50),
data_exec TIMESTAMP(3)
);

```

```

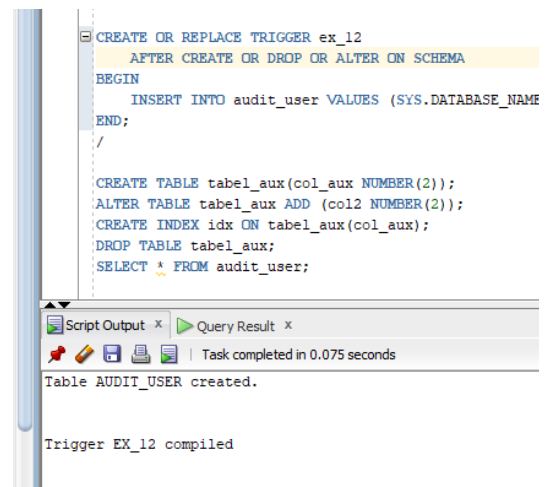
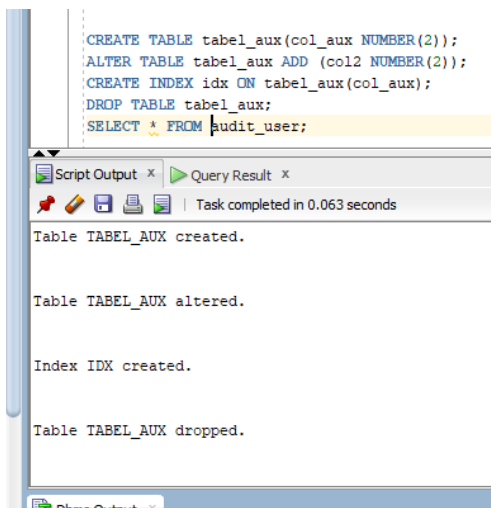
CREATE OR REPLACE TRIGGER ex_12
AFTER CREATE OR DROP OR ALTER ON SCHEMA
BEGIN
    INSERT INTO audit_user VALUES (SYS.DATABASE_NAME, SYS.LOGIN_USER,
    SYS.SYSEVENT, SYS.DICTIONARY_OBJ_TYPE, SYS.DICTIONARY_OBJ_NAME,
    SYSTIMESTAMP(3));
END;
/

```

```

CREATE TABLE tabel_aux(col_aux NUMBER(2));
ALTER TABLE tabel_aux ADD (col2 NUMBER(2));
CREATE INDEX idx ON tabel_aux(col_aux);
DROP TABLE tabel_aux;
SELECT * FROM audit_user;

```



SQL | All Rows Fetched: 4 in 0.009 seconds

	NUME_BD	USER_LOGAT	EVENIMENT	TIP_OBIECT_REFERIT	NUME_OBIECT_REFERIT	DATA_EXEC
1	XE	ANDREEA	CREATE	TABLE	TABEL_AUX	06-JAN-22 01.20.56.542000000 PM
2	XE	ANDREEA	ALTER	TABLE	TABEL_AUX	06-JAN-22 01.21.00.716000000 PM
3	XE	ANDREEA	CREATE	INDEX	IDX	06-JAN-22 01.21.05.740000000 PM
4	XE	ANDREEA	DROP	TABLE	TABEL_AUX	06-JAN-22 01.21.15.309000000 PM

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

```

CREATE OR REPLACE PACKAGE Proiect
AS
    PROCEDURE ex_6 (uni_nr universitate.id_universitate%TYPE);
    PROCEDURE ex_7 (opt NUMBER);
    FUNCTION ex_8 (nrc NUMBER) RETURN NUMBER;
    PROCEDURE ex_9 (nume_l VARCHAR2);
END Proiect;
/
CREATE OR REPLACE PACKAGE BODY Proiect
AS

```

```
PROCEDURE ex_6 (uni_nr universitate.id_universitate%TYPE)
IS
```

```
    TYPE ids IS VARRAY(6) OF universitate.id_universitate%TYPE;
    universitati ids;
```

```
    TYPE fac IS TABLE OF facultate%ROWTYPE;
    facultati fac := fac();
```

```
    cnt_uni NUMBER(2);
```

```
    NEGATIVE_NUMBER EXCEPTION;
    NO_DATA_FOUND EXCEPTION;
```

```
BEGIN
```

```
    IF uni_nr <= 0 THEN
        RAISE NEGATIVE_NUMBER;
    END IF;
```

```
    SELECT id_universitate
    BULK COLLECT INTO universitati
    FROM universitate;
```

```
    SELECT COUNT(*) INTO cnt_uni
    FROM universitate;
```

```
    SELECT *
    BULK COLLECT INTO facultati
    FROM facultate
    WHERE id_universitate = universitati(uni_nr);
```

```
    IF SQL%NOTFOUND THEN
        RAISE NO_DATA_FOUND;
    END IF;
```

```
    FOR i IN facultati.FIRST..facultati.LAST LOOP
        DBMS_OUTPUT.PUT_LINE(facultati(i).nume);
    END LOOP;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista facultati in universitatea de pe pozitia ' ||
```

```
uni_nr);
```

```
    WHEN NEGATIVE_NUMBER THEN
        DBMS_OUTPUT.PUT_LINE('Nu sunt permise valori negative');
```

```
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE ('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE ('Mesajul erorii: ' || SQLERRM);
```

```
END ex_6;
```

```
PROCEDURE ex_7 (opt NUMBER)
IS
```

```
Worksheet  Query Builder
3 CREATE OR REPLACE PACKAGE Project
4 AS
5     PROCEDURE ex_6 (uni_nr universitate.id_universitate%TYPE);
6     PROCEDURE ex_7 (opt NUMBER);
7     FUNCTION ex_8 (nrc NUMBER) RETURN NUMBER;
8     PROCEDURE ex_9 (nume_1 VARCHAR2);
9 END Project;
10 /
11 CREATE OR REPLACE PACKAGE BODY Project
12 AS
13     PROCEDURE ex_6 (uni_nr universitate.id_universitate%TYPE)
14     IS
15         TYPE ids IS VARRAY(6) OF universitate.id_universitate%TYPE;
16         universitati ids;
17
18         TYPE fac IS TABLE OF facultate%ROWTYPE;
19         facultati fac := fac();
20
21         cnt_uni NUMBER(2);
22
23         NEGATIVE_NUMBER EXCEPTION;
24         NO_DATA_FOUND EXCEPTION;
25     BEGIN
26         IF uni_nr <= 0 THEN
27             RAISE NEGATIVE_NUMBER;
```

Script Output x

Task completed in 0.063 seconds

Package Body COLECTIE compiled

PL/SQL procedure successfully completed.

Package PROIECT compiled

Package Body PROIECT compiled

```

TYPE student_tip IS REF CURSOR RETURN student%ROWTYPE;
std student_tip;
stud student%ROWTYPE;
BEGIN
  IF opt = 1 THEN
    OPEN std FOR SELECT *
      FROM student;
  ELSIF opt = 2 THEN
    OPEN std FOR SELECT *
      FROM student
      WHERE lower(ume) >= 'a' AND lower(ume) < 'j';
  ELSIF opt = 3 THEN
    OPEN std FOR SELECT *
      FROM student
      WHERE lower(pname) like 'r%';
  ELSE
    DBMS_OUTPUT.PUT_LINE('Optiune incorecta');
  END IF;

  LOOP
    FETCH std INTO stud;
    EXIT WHEN std%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE(stud.ume || ' ' || stud.pname);
  END LOOP;

  DBMS_OUTPUT.PUT_LINE ('Au fost procesate ' || std%ROWCOUNT || ' linii');
  CLOSE std;
END;

FUNCTION ex_8 (nrc NUMBER) RETURN NUMBER
IS
  nr_s NUMBER;
  TYPE tabel IS TABLE OF cadru_didactic%ROWTYPE INDEX BY PLS_INTEGER;
  aux tabel;
  tip_cadru cadru_didactic.tip_cadru%TYPE;
  NEGATIVE_NUMBER EXCEPTION;
  WRONG_DATA EXCEPTION;
  NO_DATA_FOUND1 EXCEPTION;
  NO_DATA_FOUND2 EXCEPTION;
BEGIN
  if nrc < 0 THEN
    RAISE NEGATIVE_NUMBER;
  END IF;
  SELECT * BULK COLLECT INTO aux FROM cadru_didactic WHERE id_cadru = nrc;
  IF SQL%NOTFOUND THEN
    RAISE NO_DATA_FOUND1;
  END IF;

  SELECT COUNT (ser.cod_serie) INTO nr_s

```

```

FROM serie ser JOIN sala sal ON (ser.cod_serie = sal.cod_serie)
      JOIN cadru_didactic c ON (c.id_cadru = sal.id_cadru)
WHERE c.id_cadru = nrc;

```

```

SELECT tip_cadru INTO tip_cadru
FROM cadru_didactic
WHERE id_cadru = nrc;

```

```

IF nr_s = 0 THEN
    RAISE NO_DATA_FOUND2;
ELSIF tip_cadru != 'lector' THEN
    RAISE WRONG_DATA;
ELSE
    RETURN nr_s;
END IF;

```

```

EXCEPTION
    WHEN NO_DATA_FOUND1 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista cadru didactic cu id-ul ' || nrc);
        RETURN -1;
    WHEN NO_DATA_FOUND2 THEN
        DBMS_OUTPUT.PUT_LINE('Lectorul nu preda la nicio serie');
        RETURN -1;
    WHEN NEGATIVE_NUMBER THEN
        DBMS_OUTPUT.PUT_LINE('Nu sunt permise valori negative');
        RETURN -1;
    WHEN WRONG_DATA THEN
        DBMS_OUTPUT.PUT_LINE('Cadrul didactic cu id-ul ' || nrc || ' nu este lector');
        RETURN -1;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE ('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE ('Mesajul erorii: ' || SQLERRM);
END;

```

```

PROCEDURE ex_9 (nume_l VARCHAR2)
IS
    TYPE tabel_ind_s IS TABLE OF semigrupa.id_semigrupa%TYPE INDEX BY
PLS_INTEGER;
    sgr tabel_ind_s;
    aux cadru_didactic.id_cadru%TYPE;
    NO_DATA_FOUND1 EXCEPTION;
    NO_DATA_FOUND2 EXCEPTION;
    WRONG_DATA EXCEPTION;
BEGIN
    SELECT id_cadru INTO aux
    FROM cadru_didactic
    WHERE cadru_didactic.nume = nume_l AND cadru_didactic.tip_cadru = 'lector';

    SELECT id_semigrupa BULK COLLECT INTO sgr

```

```

FROM semigrupa sem JOIN grupa g ON (sem.cod_grupa = g.cod_grupa)
    JOIN serie ser ON (g.cod_serie = ser.cod_serie)
    JOIN sala sal ON (ser.cod_serie = sal.cod_serie)
    JOIN cadru_didactic c ON (c.id_cadru = sal.id_cadru)
WHERE c.nume = nume_l;

```

```

IF sgr.COUNT = 0 THEN
    RAISE NO_DATA_FOUND2;
END IF;

```

```

DBMS_OUTPUT.PUT_LINE('Semigrupele la care preda lectorul cautat: ');
FOR i IN sgr.FIRST..sgr.LAST LOOP
    DBMS_OUTPUT.PUT_LINE(sgr(i));
END LOOP;

```

```

EXCEPTION
    WHEN NO_DATA_FOUND2 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista date de afisat');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multi lectori cu numele ' || nume_l);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE('Mesajul erorii: ' || SQLERRM);
END;

```

```

END Proiect;
/

```

```

BEGIN
    Proiect.ex_9('Anghel');
END;
/

```

```

186
187 BEGIN
188     Proiect.ex_9('Anghel');
189 END;
190 /
191
192
193
194
195
196
197 --Ex 14:

```

Script Output x

Task completed in 0.048 seconds

Package Body PROIECT compiled

PL/SQL procedure successfully completed.

Dbms Output x

Buffer Size: 20000

PROIECT x

Semigrupele la care preda lectorul cautat:

1311

1312

1331

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).

Am definit un pachet care reține date deja existente despre note într-o colecție. Se pot adăuga, scoate sau afișa toate, prima sau ultima notă.

- tipuri de date complexe: înregistrare, tabel imbricat de înregistrări
- proceduri: creare, adauga, scoate, afisare
- funcții: prima, ultima

```
CREATE OR REPLACE PACKAGE Colectie
IS
```

```
    TYPE item IS RECORD (
        calificativ NUMBER(3),
        materie VARCHAR2(50),
        data_obtinere DATE
    );
    aux item;
    TYPE tabel IS TABLE OF item;
    note tabel := tabel();
    PROCEDURE creare;
    PROCEDURE adauga(n NUMBER, m nota.disciplina%TYPE, d nota.data_nota%TYPE);
    PROCEDURE scoate;
    PROCEDURE afisare;
    FUNCTION prima RETURN NUMBER;
    FUNCTION ultima RETURN NUMBER;
```

```
END Colectie;
/
```

```
CREATE OR REPLACE PACKAGE BODY Colectie
AS
```

```
    PROCEDURE creare
    IS
    BEGIN
        SELECT nota, disciplina, data_nota
        BULK COLLECT INTO note
        FROM nota;
    END;
```

```
    PROCEDURE adauga(n NUMBER, m nota.disciplina%TYPE, d nota.data_nota%TYPE)
    IS
        NEGATIVE_VALUE EXCEPTION;
    BEGIN
        IF n > 0 AND n <= 10 THEN
            aux.calificativ := n;
            aux.materie := m;
            aux.data_obtinere := d;
            note.extend;
            note(note.last) := aux;
```

```

ELSE
    RAISE NEGATIVE_VALUE;
END IF;
EXCEPTION
    WHEN NEGATIVE_VALUE THEN
        DBMS_OUTPUT.PUT_LINE ('Nu se pot acorda calificative negative');
END;

PROCEDURE scoate
IS
    NO_ELEMENTS EXCEPTION;
BEGIN
    IF note.COUNT > 0 THEN
        aux := note(note.LAST);
        DBMS_OUTPUT.PUT_LINE ('S-a scos nota ' || aux.calificativ || ' la disciplina ' || aux.materie || '
din data de ' || aux.data_obtinere);
        note.TRIM;
    ELSE
        RAISE NO_ELEMENTS;
        --dbms_output.put_line ('nu exista date');
    END IF;
EXCEPTION
    WHEN NO_ELEMENTS THEN
        DBMS_OUTPUT.PUT_LINE ('Nu exista date in tabel');
END;

PROCEDURE afisare
IS
BEGIN
    FOR i IN note.FIRST..note.LAST LOOP
        aux := note(i);
        dbms_output.put_line('Nota ' || aux.calificativ || ' la disciplina ' || aux.materie || ' din data de ' ||
aux.data_obtinere);
    END LOOP;
END;

FUNCTION prima RETURN NUMBER
IS
    NO_ELEMENTS EXCEPTION;
BEGIN
    IF note.COUNT > 0 THEN
        aux := note(note.FIRST);
        RETURN aux.calificativ;
    ELSE
        RAISE NO_ELEMENTS;
        --dbms_output.put_line ('nu exista date');
    END IF;
EXCEPTION
    WHEN NO_ELEMENTS THEN

```

```

        DBMS_OUTPUT.PUT_LINE ('Nu exista date in tabel');
        RETURN -1;
    END;

    FUNCTION ultima RETURN NUMBER
    IS
        NO_ELEMENTS EXCEPTION;
    BEGIN
        IF note.COUNT > 0 THEN
            aux := note(note.LAST);
            RETURN aux.calificativ;
        ELSE
            RAISE NO_ELEMENTS;
            -- dbms_output.put_line ('nu exista date');
        END IF;
    EXCEPTION
        WHEN NO_ELEMENTS THEN
            DBMS_OUTPUT.PUT_LINE ('Nu exista date in tabel');
            RETURN -1;
    END;
END Colectie;
/

DECLARE
    aux NUMBER;
BEGIN
    --Colectie.creare;
    --Colectie.afisare;
    --Colectie.adauga(3, 'plsql', to_date('06/01/2022', 'DD/MM/YYYY'));
    --Colectie.adauga(10, 'so', to_date('07/01/2022', 'DD/MM/YYYY'));
    --Colectie.adauga(-2, 'so', to_date('15/04/2021',
'DD/MM/YYYY')); --NEGATIVE_VALUE
    --aux := Colectie.prima;
    --aux := Colectie.ultima;
    --dbms_output.put_line(aux);
    Colectie.scoate;
END;
/

```

```

231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251

```

```

PROCEDURE adauga(n NUMBER, m nota.disciplina%TYPE, d r
IS
    NEGATIVE_VALUE EXCEPTION;
BEGIN
    IF n > 0 AND n <= 10 THEN
        aux.calificativ := n;
        aux.materie := m;
        aux.data_obtinere := d;
        note.extend;
        note(note.last) := aux;
    ELSE
        RAISE NEGATIVE_VALUE;
    END IF;
EXCEPTION
    WHEN NEGATIVE_VALUE THEN
        DBMS_OUTPUT.PUT_LINE ('Nu se pot acorda califi
END;

PROCEDURE scoate
IS

```

Script Output x

Task completed in 0.057 seconds

Package COLECTIE compiled

Package Body COLECTIE compiled

```

313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330

```

```

DECLARE
    aux NUMBER;
BEGIN
    Colectie.creare;
    --Colectie.afisare;
    --Colectie.adauga(3, 'plsql', to_date('06/01/
    --Colectie.adauga(10, 'so', to_date('07/01/20
    --Colectie.adauga(-2, 'so', to_date('15/04/20
    --aux := Colectie.prima;
    --aux := Colectie.ultima;
    --dbms_output.put_line(aux);
    --Colectie.scoate;
END;
/

```

Script Output x

Task completed in 0.052 seconds

Package Body COLECTIE compiled

PL/SQL procedure successfully completed.


```

312
313 DECLARE
314     aux NUMBER;
315 BEGIN
316     --Colectie.creare;
317     --Colectie.afisare;
318     --Colectie.adauga(3, 'plsql', to_date('06/01/2022
319     --Colectie.adauga(10, 'so', to_date('07/01/2022',
320     --Colectie.adauga(-2, 'so', to_date('15/04/2021',
321     --aux := Colectie.prima;
322     aux := Colectie.ultima;
323     dbms_output.put_line(aux);
324     --Colectie.scoate;
325 END;
326 /
327
328
329
330

```

Script Output x
Task completed in 0.053 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x
Buffer Size: 20000

PROIECT x

1312
1331

7

```

313 DECLARE
314     aux NUMBER;
315 BEGIN
316     --Colectie.creare;
317     --Colectie.afisare;
318     --Colectie.adauga(3, 'plsql', to_date('06/01/2022', 'DD/MM/YYYY'));
319     --Colectie.adauga(10, 'so', to_date('07/01/2022', 'DD/MM/YYYY'));
320     --Colectie.adauga(-2, 'so', to_date('15/04/2021', 'DD/MM/YYYY')); --NEG
321     --aux := Colectie.prima;
322     --aux := Colectie.ultima;
323     --dbms_output.put_line(aux);
324     Colectie.scoate;
325 END;
326 /
327
328
329
330

```

Script Output x
Task completed in 0.045 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output x
Buffer Size: 20000

PROIECT x

7

S-a scos nota 7 la disciplina Logica matematica din data de 12-DEC-20