

Sistem de gestiune a academiiilor sportive

Petre Vasile-Eduard

Grupa 243

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

Baza de date prezintă un sistem de gestiune a unei academii sportive. Prin intermediul tabelelor și a relațiilor definite între ele, baza de date permite gestionarea academiei sportive, înregistrarea sportivilor în echipe, asocierea lor cu categoriile de vîrstă și monitorizarea progresului lor prin intermediul notelor acordate la diferite aptitudini. De asemenea, permite urmărirea contractelor cu antrenorii și organizarea programelor de antrenament pentru fiecare categorie de vîrstă și poziție sportivă.

Tabela "ACADEMIE" stochează informații despre academie, cum ar fi numele, adresa, telefonul, adresa de e-mail și codul poștal. Această tabelă servește ca entitate principală pentru întregul sistem și este legată de alte tabele prin intermediul cheilor străine.

Tabela "ECHIPA" reprezintă echipele din cadrul academiei. Aceasta conține detalii despre numele echipei și data înființării. De asemenea, este conectată la tabela "ACADEMIE" prin intermediul unei chei străine pentru a indica apartenența echipei la o academie.

Tabela "SPORT" stochează informații despre sporturile practicate în cadrul academiei. Aceasta conține denumirea sportului și este asociată cu tabela "ACADEMIE" prin intermediul unei chei străine.

Tabela "CATEGORIEVARSTA" descrie categoriile de vîrstă pentru fiecare sport în parte. Aici se reține denumirea categoriei de vîrstă și este stabilită o relație cu tabela "SPORT" prin intermediul unei chei străine.

Tabela "SPORTIV" conține informații despre sportivi, cum ar fi nume, prenume, data nașterii, naționalitate și număr de telefon. De asemenea, tabela conține o cheie străină care face referire la tabela "ECHIPA" pentru a indica echipa la care este înscris sportivul.

Tabela "INSCRIERE" este o tabelă de asociere între sportivi și categorii de vîrstă, reținând datele de înregistrare ale sportivilor în categorii de vîrstă specifice.

Tabela "ANTRENOR" conține informații despre antrenori, inclusiv nume, prenume și număr de telefon.

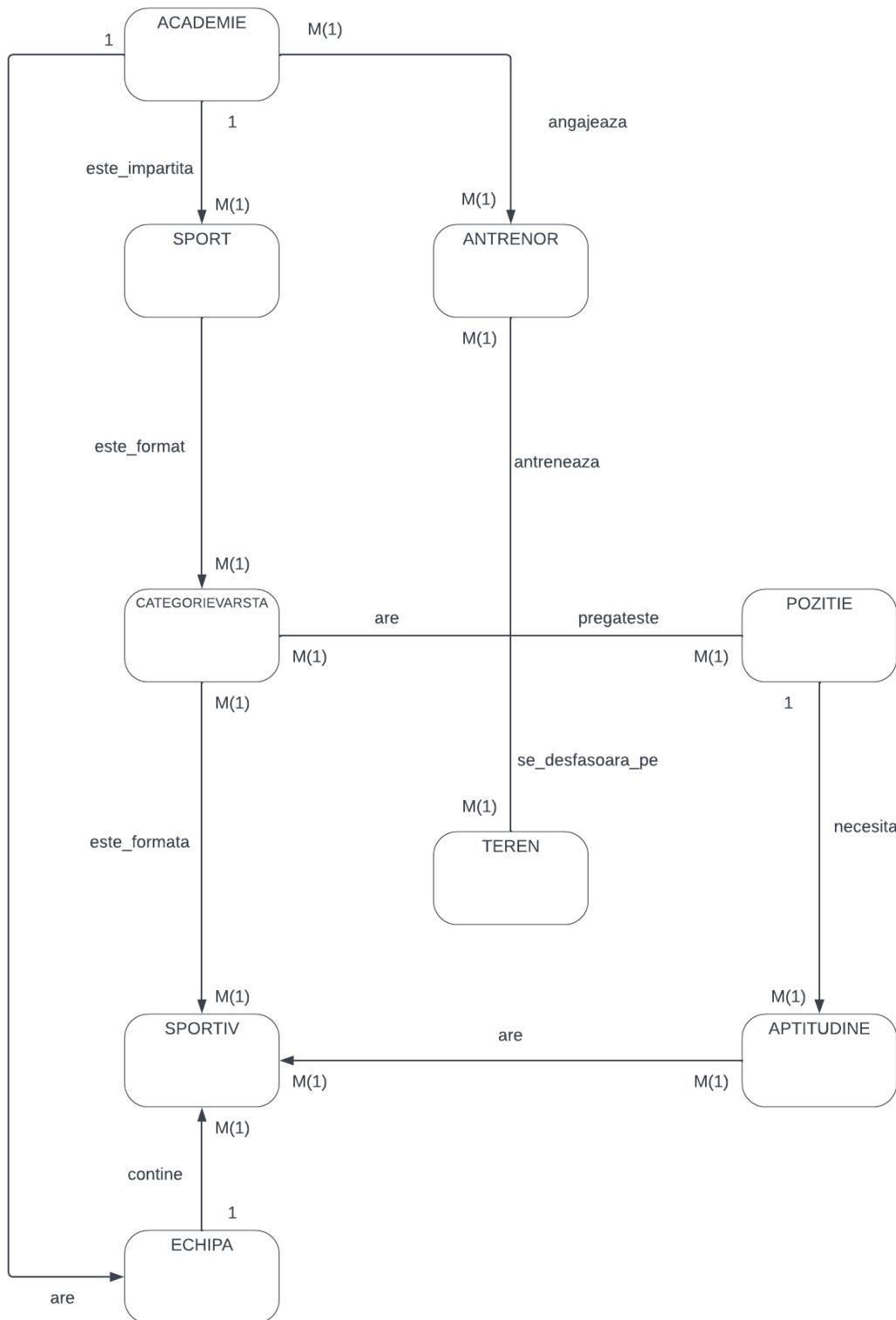
Tabela "CONTRACT" reprezintă contractele încheiate între academie și antrenori. Aici se stochează data încheierii contractului și salariul asociat. Tabela este conectată atât la tabela "ACADEMIE", cât și la tabela "ANTRENOR".

Tabela "APTITUDINE" descrie aptitudinile sportivilor, cum ar fi puterea, viteza sau agilitatea. Aceasta conține denumirea aptitudinii și este legată de tabela "POZITIE" prin intermediul unei chei străine.

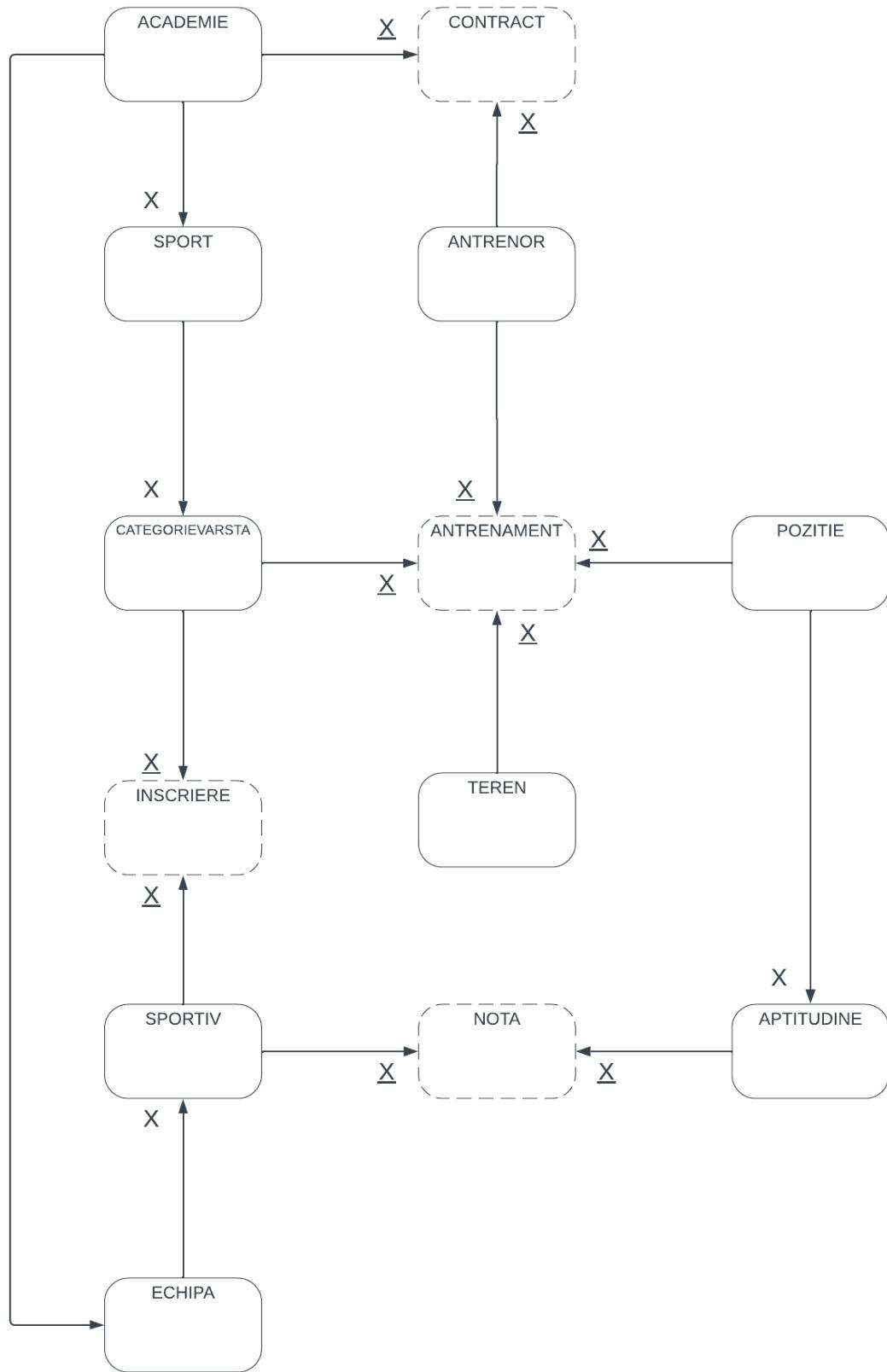
Tabela "NOTA" stochează notele acordate sportivilor pentru diferite aptitudini. Aceasta conține nota, precum și chei străine către sportivul și aptitudinea asociată.

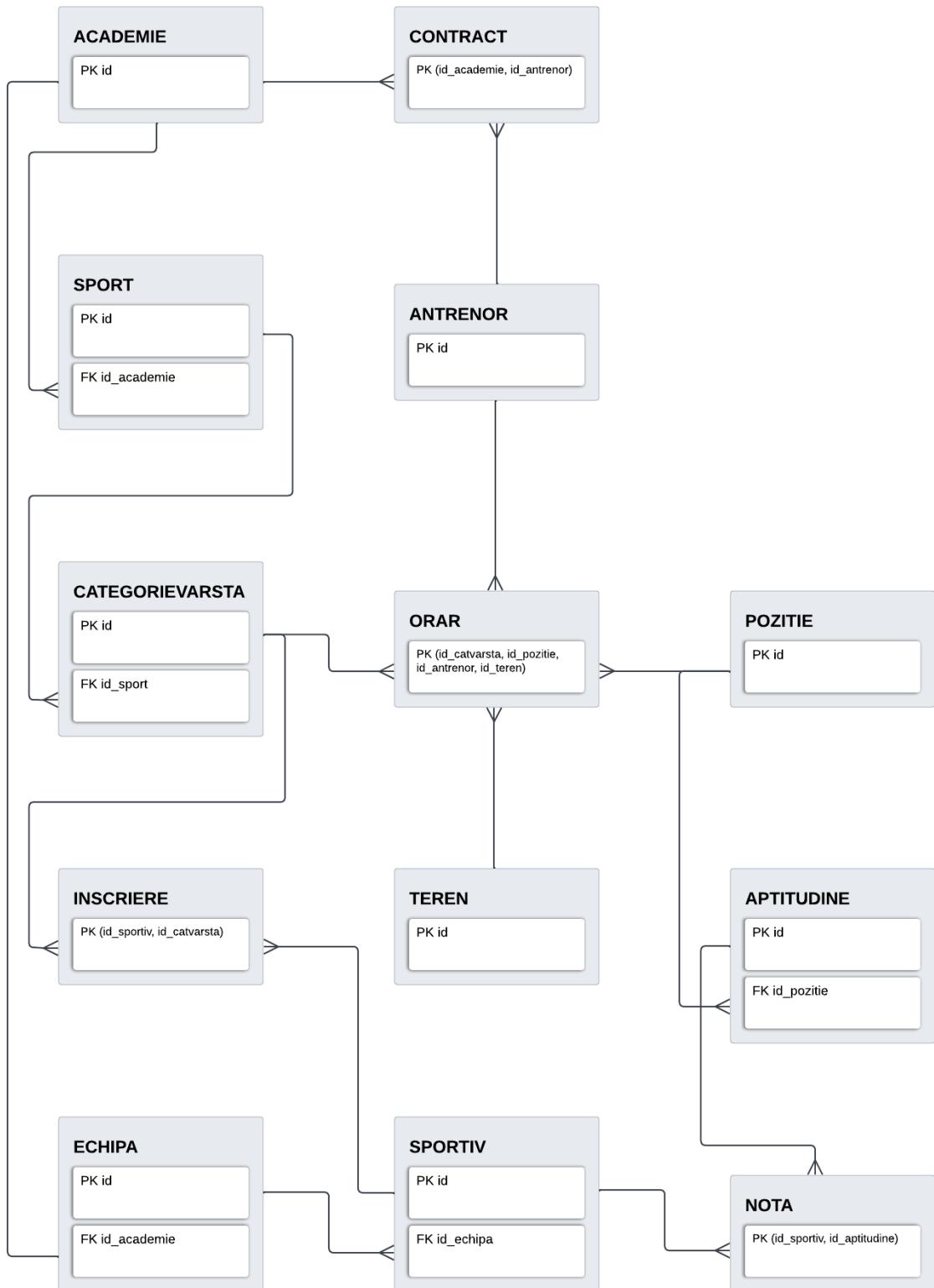
Tabela "ANTRENAMENT" reprezintă programul de antrenament și asociază categoriile de vîrstă, pozițiile, antrenorii și terenurile alocate. Este utilizată pentru a organiza și programa antrenamentele în cadrul academiei.

2. Diagrama ERD



3. Diagrama conceptuala





4. si 5. Tabele si inserare in tabele

```
CREATE TABLE ACADEMIE (
```

```
    id NUMBER(5) CONSTRAINT PKEY_ACADEMIE PRIMARY KEY,  
    nume VARCHAR(100) CONSTRAINT nume_academie NOT NULL,  
    adresa VARCHAR(100) CONSTRAINT adresa_academie NOT NULL,  
    telefon VARCHAR(20) CONSTRAINT telefon_academie NOT NULL,  
    mail VARCHAR(50) CONSTRAINT mail_academie UNIQUE NOT NULL,  
    cod VARCHAR(6) CONSTRAINT cod_postal_academie NOT NULL  
);
```

```
INSERT INTO ACADEMIE
```

```
VALUES (1, 'Academia Hagi', 'Str. Litoralului Nr. 201', '0238 459 862',  
'office@academiahagi.ro', '128956');
```

```
INSERT INTO ACADEMIE
```

```
VALUES (2, 'Academia Dinamo Bucuresti', 'Str. Crizantemelor Nr. 22', '0238 852 731',  
'office@academiadinamo.ro', '129634');
```

```
INSERT INTO ACADEMIE
```

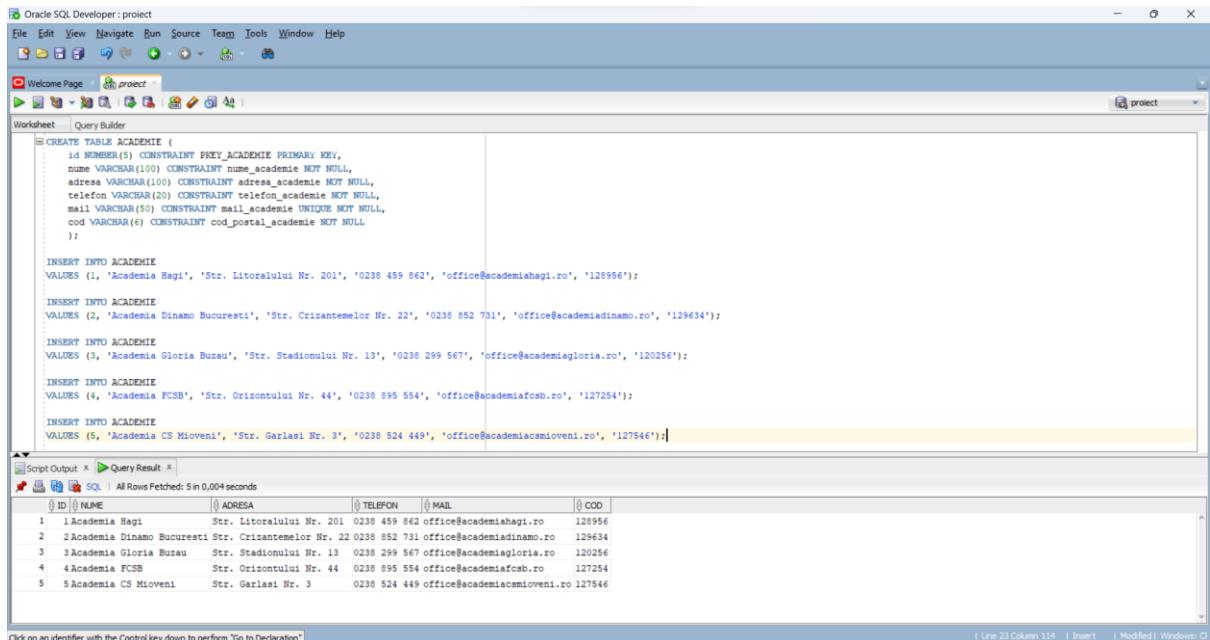
```
VALUES (3, 'Academia Gloria Buzau', 'Str. Stadionului Nr. 13', '0238 299 567',  
'office@academiagloria.ro', '120256');
```

```
INSERT INTO ACADEMIE
```

```
VALUES (4, 'Academia FCSB', 'Str. Orizontului Nr. 44', '0238 895 554',  
'office@academiafcsb.ro', '127254');
```

```
INSERT INTO ACADEMIE
```

```
VALUES (5, 'Academia CS Mioveni', 'Str. Garlasi Nr. 3', '0238 524 449',  
'office@academiacsmioveni.ro', '127546');
```



```

CREATE TABLE ACADEMIE (
    id NUMBER(5) CONSTRAINT PKEY_ACADEMIE PRIMARY KEY,
    nume VARCHAR(100) CONSTRAINT nume_academie NOT NULL,
    adresa VARCHAR(100) CONSTRAINT adresa_academie NOT NULL,
    telefon VARCHAR(20) CONSTRAINT telefon_academie NOT NULL,
    mail VARCHAR(50) CONSTRAINT mail_academie UNIQUE NOT NULL,
    cod VARCHAR(6) CONSTRAINT cod_postal_academie NOT NULL
);

INSERT INTO ACADEMIE
VALUES (1, 'Academia Hagi', 'Str. Litoralului Nr. 201', '0238 459 862', 'office@academiahagi.ro', '128956');

INSERT INTO ACADEMIE
VALUES (2, 'Academia Dinamo Bucuresti', 'Str. Crizantemelor Nr. 22', '0238 852 731', 'office@academadinamo.ro', '129634');

INSERT INTO ACADEMIE
VALUES (3, 'Academia Gloria Buzau', 'Str. Stadionului Nr. 13', '0238 299 567', 'office@academialgloria.ro', '120256');

INSERT INTO ACADEMIE
VALUES (4, 'Academia FCSEB', 'Str. Orizontului Nr. 44', '0238 895 554', 'office@academiacfcsb.ro', '127254');

INSERT INTO ACADEMIE
VALUES (5, 'Academia CS Mioveni', 'Str. Garlasi Nr. 3', '0238 524 449', 'office@academiacsmioveni.ro', '127546');

```

ID	NUME	ADRESA	TELEFON	MAIL	COD
1	Academia Hagi	Str. Litoralului Nr. 201	0238 459 862	office@academiahagi.ro	128956
2	Academia Dinamo Bucuresti	Str. Crizantemelor Nr. 22	0238 852 731	office@academadinamo.ro	129634
3	Academia Gloria Buzau	Str. Stadionului Nr. 13	0238 299 567	office@academialgloria.ro	120256
4	Academia FCSEB	Str. Orizontului Nr. 44	0238 895 554	office@academiacfcsb.ro	127254
5	Academia CS Mioveni	Str. Garlasi Nr. 3	0238 524 449	office@academiacsmioveni.ro	127546

Tabela ACADEMIE

CREATE TABLE ECHIPA (

```

id NUMBER(5) CONSTRAINT PKEY_ECHIPA PRIMARY KEY,
nume VARCHAR(100) CONSTRAINT nume_echipa NOT NULL,
data DATE CONSTRAINT data_infintare_echipa NOT NULL,
id_academie NUMBER(5), CONSTRAINT fk_echipa FOREIGN KEY(id_academie)
REFERENCES ACADEMIE(id)

);

```

INSERT INTO ECHIPA

```
VALUES(1, 'Farul U18', TO_DATE('2009-06-15','YYYY-MM-DD'), 1);
```

INSERT INTO ECHIPA

```
VALUES(2, 'Farul U17', TO_DATE('2009-06-15','YYYY-MM-DD'), 1);
```

INSERT INTO ECHIPA

```
VALUES(3, 'Farul U16', TO_DATE('2012-09-01','YYYY-MM-DD'), 1);
```

INSERT INTO ECHIPA

```
VALUES(4, 'Academia Hagi U15', TO_DATE('2012-09-01','YYYY-MM-DD'), 1);
```

INSERT INTO ECHIPA

```
VALUES(5, 'Academia Hagi U14', TO_DATE('2012-09-01','YYYY-MM-DD'), 1);
```

The screenshot shows the Oracle SQL Developer interface. In the 'Worksheet' tab, there is a script containing the creation of the ECHIPA table and the insertion of five rows. The table has columns: ID (primary key), NUME, DATA, and ID_ACADEMIE. The inserted rows are:

ID	NUME	DATA	ID_ACA...
1	Farul U18	15-JUN-09	1
2	Farul U17	15-JUN-09	1
3	Farul U16	01-SEP-12	1
4	Academia Hagi U15	01-SEP-12	1
5	Academia Hagi U14	01-SEP-12	1

Tabela ECHIPA

CREATE TABLE SPORT(

```
id NUMBER(5) CONSTRAINT PKEY_SPORT PRIMARY KEY,  
denumire VARCHAR(100) CONSTRAINT denumire_sport NOT NULL,  
id_academie NUMBER(5), CONSTRAINT fk_sport FOREIGN KEY(id_academie)  
REFERENCES ACADEMIE(id)  
);
```

INSERT INTO SPORT

```
VALUES(1, 'Fotbal', 1);
```

INSERT INTO SPORT

```
VALUES(2, 'Fotbal', 2);
```

INSERT INTO SPORT

```
VALUES(3, 'Handbal', 2);
```

INSERT INTO SPORT

```
VALUES(4, 'Baschet', 2);
```

INSERT INTO SPORT

```
VALUES(5, 'Fotbal', 3);
```

The screenshot shows the Oracle SQL Developer interface. In the 'Worksheet' tab, the following SQL code is visible:

```
CREATE TABLE SPORT (
    id NUMBER(5) CONSTRAINT PKH_Y_SPORT PRIMARY KEY,
    denumire VARCHAR(100) CONSTRAINT denumire_sport NOT NULL,
    id_academie NUMBER(5), CONSTRAINT fk_sport FOREIGN KEY(id_academie) REFERENCES ACADEMIE(id)
);

INSERT INTO SPORT
VALUES(1, 'Fotbal', 1);

INSERT INTO SPORT
VALUES(2, 'Fotbal', 2);

INSERT INTO SPORT
VALUES(3, 'Handbal', 2);

INSERT INTO SPORT
VALUES(4, 'Baschet', 2);

INSERT INTO SPORT
VALUES(5, 'Fotbal', 3);
```

In the 'Script Output' tab, the results of the insertions are shown:

ID	DENUMIRE	ID_AADEMIE
1	Fotbal	1
2	Fotbal	2
3	Handbal	2
4	Baschet	2
5	Fotbal	3

Tabela SPORT

```
CREATE TABLE CATEGORIEVARSTA (
    id NUMBER(5) CONSTRAINT PKEY_CATEGORIEVARSTA PRIMARY KEY,
    denumire VARCHAR(100) CONSTRAINT denumire_catvarsta NOT NULL,
    id_sport NUMBER(5), CONSTRAINT fk_catvarsta FOREIGN KEY(id_sport)
    REFERENCES SPORT(id)
);
```

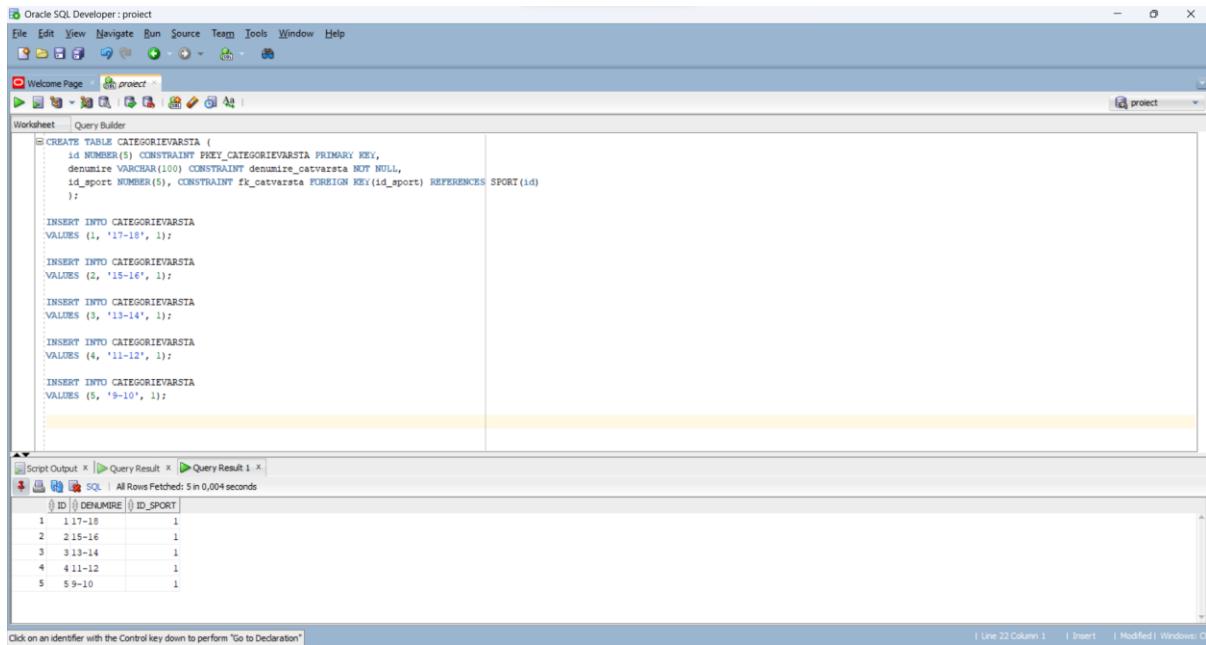
```
INSERT INTO CATEGORIEVARSTA
VALUES (1, '17-18', 1);
```

```
INSERT INTO CATEGORIEVARSTA
VALUES (2, '15-16', 1);
```

```
INSERT INTO CATEGORIEVARSTA
VALUES (3, '13-14', 1);
```

```
INSERT INTO CATEGORIEVARSTA
VALUES (4, '11-12', 1);
```

```
INSERT INTO CATEGORIEVARSTA
VALUES (5, '9-10', 1);
```



The screenshot shows the Oracle SQL Developer interface. In the 'Worksheet' tab, there is a SQL script for creating the 'CATEGORIEVARSTA' table and inserting five rows of data. The table has columns: ID (NUMBER(5)), DENUMIRE (VARCHAR(100)), and ID_SPORT (NUMBER(5)). The script inserts rows for age groups: 17-18, 15-16, 13-14, 11-12, and 9-10, all associated with sport ID 1.

```

CREATE TABLE CATEGORIEVARSTA (
    id NUMBER(5) CONSTRAINT PKEY_CATEGORIEVARSTA PRIMARY KEY,
    denumire VARCHAR(100) CONSTRAINT denumire_categorista NOT NULL,
    id_sport NUMBER(5), CONSTRAINT fk_categorista FOREIGN KEY(id_sport) REFERENCES SPORT(id)
);

INSERT INTO CATEGORIEVARSTA
VALUES (1, '17-18', 1);
INSERT INTO CATEGORIEVARSTA
VALUES (2, '15-16', 1);
INSERT INTO CATEGORIEVARSTA
VALUES (3, '13-14', 1);
INSERT INTO CATEGORIEVARSTA
VALUES (4, '11-12', 1);
INSERT INTO CATEGORIEVARSTA
VALUES (5, '9-10', 1);

```

In the 'Script Output' tab, the results of the insertions are shown:

ID	DENUMIRE	ID_SPORT
1	17-18	1
2	15-16	1
3	13-14	1
4	11-12	1
5	9-10	1

Tabela CATEGORIEVARSTA

CREATE TABLE SPORTIV(

```

        id NUMBER(5) CONSTRAINT PKEY_SPORTIV PRIMARY KEY,
        nume VARCHAR(100) CONSTRAINT nume_sportiv NOT NULL,
        prenume VARCHAR(100) CONSTRAINT prenume_sportiv NOT NULL,
        data DATE CONSTRAINT data_nastere_sportiv NOT NULL,
        nationalitate VARCHAR(30) CONSTRAINT nationalitate_sportiv NOT NULL,
        telefon VARCHAR(20) CONSTRAINT telefon_sportiv NOT NULL,
        id_echipa NUMBER(5), CONSTRAINT fk_sportiv FOREIGN KEY(id_echipa)
        REFERENCES ECHIPA(id)
    );

```

INSERT INTO SPORTIV

```

VALUES (1, 'Marian', 'Alex', TO_DATE('2002-06-15','YYYY-MM-DD'), 'roman', '0789 562
357', NULL);

```

INSERT INTO SPORTIV

```
VALUES (2, 'Florian', 'Petrescu', TO_DATE('2000-02-12','YYYY-MM-DD'), 'roman', '0752 668 895', NULL);
```

INSERT INTO SPORTIV

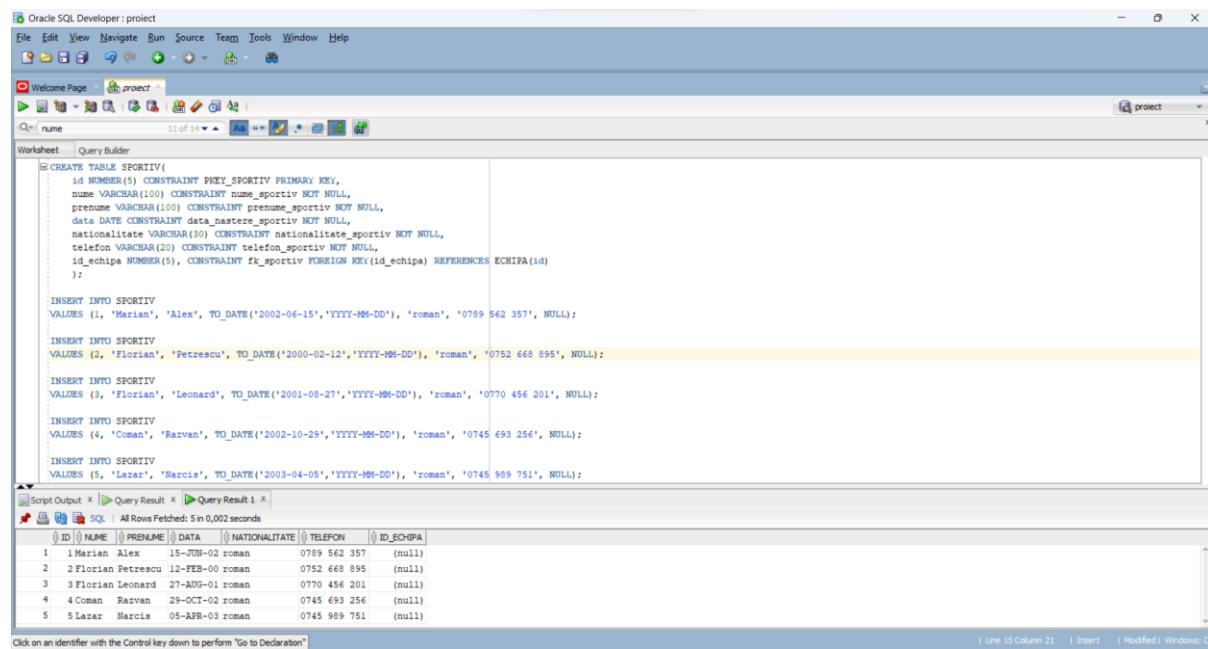
```
VALUES (3, 'Florian', 'Leonard', TO_DATE('2001-08-27','YYYY-MM-DD'), 'roman', '0770 456 201', NULL);
```

INSERT INTO SPORTIV

```
VALUES (4, 'Coman', 'Razvan', TO_DATE('2002-10-29','YYYY-MM-DD'), 'roman', '0745 693 256', NULL);
```

INSERT INTO SPORTIV

```
VALUES (5, 'Lazar', 'Narcis', TO_DATE('2003-04-05','YYYY-MM-DD'), 'roman', '0745 989 751', NULL);
```



```
CREATE TABLE SPORTIV(
    id NUMBER(5) CONSTRAINT PKH_SPORTIV PRIMARY KEY,
    nume VARCHAR(100) CONSTRAINT name_sportiv NOT NULL,
    prenume VARCHAR(100) CONSTRAINT prenume_sportiv NOT NULL,
    data DATE CONSTRAINT data_nastere_sportiv NOT NULL,
    nationalitate VARCHAR(30) CONSTRAINT nationalitate_sportiv NOT NULL,
    telefon VARCHAR(20) CONSTRAINT telefon_sportiv NOT NULL,
    id_echipa NUMBER(5), CONSTRAINT fk_sportiv FOREIGN KEY(id_echipa) REFERENCES ECHIPA(id)
);

INSERT INTO SPORTIV
VALUES (1, 'Marian', 'Alex', TO_DATE('2002-06-15','YYYY-MM-DD'), 'roman', '0789 562 357', NULL);

INSERT INTO SPORTIV
VALUES (2, 'Florian', 'Petrescu', TO_DATE('2000-02-12','YYYY-MM-DD'), 'roman', '0752 668 895', NULL);

INSERT INTO SPORTIV
VALUES (3, 'Florian', 'Leonard', TO_DATE('2001-08-27','YYYY-MM-DD'), 'roman', '0770 456 201', NULL);

INSERT INTO SPORTIV
VALUES (4, 'Coman', 'Razvan', TO_DATE('2002-10-29','YYYY-MM-DD'), 'roman', '0745 693 256', NULL);

INSERT INTO SPORTIV
VALUES (5, 'Lazar', 'Narcis', TO_DATE('2003-04-05','YYYY-MM-DD'), 'roman', '0745 989 751', NULL);
```

ID	NUME	PRENUME	DATA	NATIONALITATE	TELEFON	ID_ECHIPA
1	Marian	Alex	15-JUN-02	roman	0789 562 357	(null)
2	Florian	Petrescu	12-FEB-00	roman	0752 668 895	(null)
3	Florian	Leonard	27-AUG-01	roman	0770 456 201	(null)
4	Coman	Razvan	29-OCT-02	roman	0745 693 256	(null)
5	Lazar	Narcis	05-APR-03	roman	0745 989 751	(null)

Tabela SPORTIV

```
CREATE TABLE INSCRIERE(  
    id_sportiv NUMBER(5) CONSTRAINT pk_id_sportiv REFERENCES  
    SPORTIV(id),  
    data DATE CONSTRAINT data_inscriere NOT NULL,  
    id_catvarsta NUMBER(5) CONSTRAINT pk_id_catvarsta REFERENCES  
    CATEGORIEVARSTA(id),  
    CONSTRAINT pk_compus_inscriere PRIMARY KEY(id_catvarsta, id_sportiv)  
);
```

```
INSERT INTO INSCRIERE  
VALUES (1, TO_DATE('2019-01-21','YYYY-MM-DD'), 1);
```

```
INSERT INTO INSCRIERE  
VALUES (2, TO_DATE('2017-04-30','YYYY-MM-DD'), 1);
```

```
INSERT INTO INSCRIERE  
VALUES (3, TO_DATE('2018-10-05','YYYY-MM-DD'), 1);
```

```
INSERT INTO INSCRIERE  
VALUES (4, TO_DATE('2020-09-12','YYYY-MM-DD'), 1);
```

```
INSERT INTO INSCRIERE  
VALUES (5, TO_DATE('2021-07-15','YYYY-MM-DD'), 1);
```

The screenshot shows the Oracle SQL Developer interface. In the top-left corner, it says "Oracle SQL Developer : project". The menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help. Below the menu is a toolbar with various icons. The main area has tabs for "Welcome Page" and "project". The "Query Builder" tab is active, displaying the following SQL code:

```
CREATE TABLE INSCRIERE(
    id_sportiv NUMBER(5) CONSTRAINT pk_id_sportiv REFERENCES SPORTIV(id),
    data DATE CONSTRAINT data_inscriere NOT NULL,
    id_catvarsta NUMBER(5) CONSTRAINT pk_id_catvarsta REFERENCES CATEGORIEVARSTA(id),
    CONSTRAINT pk_compusa_inscriere PRIMARY KEY(id_catvarsta, id_sportiv)
);

INSERT INTO INSCRIERE
VALUES (1, TO_DATE('2019-01-21','YYYY-MM-DD'), 1);

INSERT INTO INSCRIERE
VALUES (2, TO_DATE('2017-04-30','YYYY-MM-DD'), 1);

INSERT INTO INSCRIERE
VALUES (3, TO_DATE('2018-10-05','YYYY-MM-DD'), 1);

INSERT INTO INSCRIERE
VALUES (4, TO_DATE('2020-09-12','YYYY-MM-DD'), 1);

INSERT INTO INSCRIERE
VALUES (5, TO_DATE('2021-07-15','YYYY-MM-DD'), 1);
```

Below the code, the "Script Output" tab shows the results of the insertions:

ID_SPORTIV	DATA	ID_CATVARSTA
1	21-JAN-19	1
2	30-APR-17	1
3	05-OCT-18	1
4	12-SEP-20	1
5	15-JUL-21	1

At the bottom of the interface, there are buttons for "SQL" and "Script", and a status bar with "Friends - Discord".

Tabela INSCRIERE

CREATE TABLE ANTRENOR(

```
id NUMBER(5) CONSTRAINT PKEY_ANTRENOR PRIMARY KEY,  
nume VARCHAR(100) CONSTRAINT nume_antrenor NOT NULL,  
prenume VARCHAR(100) CONSTRAINT prenume_antrenor NOT NULL,  
telefon VARCHAR(20) CONSTRAINT telefon_antrenor NOT NULL  
);
```

INSERT INTO ANTRENOR

```
VALUES (1, 'Gheorghe', 'Hagi', '0789 564 589');
```

INSERT INTO ANTRENOR

```
VALUES (2, 'Razvan', 'Stan', '0756 852 323');
```

INSERT INTO ANTRENOR

```
VALUES (3, 'Marius', 'Popescu', '0742 210 301');
```

```
INSERT INTO ANTRENOR
VALUES (4, 'Florin', 'Bogdan', '0765 458 892');
```

```
INSERT INTO ANTRENOR
VALUES (5, 'Rares', 'Popa', '0712 541 132');
```

The screenshot shows the Oracle SQL Developer interface. In the 'Worksheet' tab, there is a 'Query Builder' window containing the following SQL code:

```
CREATE TABLE ANTRENOR (
    id NUMBER(5) CONSTRAINT pk_antrenor PRIMARY KEY,
    nume VARCHAR(100) CONSTRAINT nume_antrenor NOT NULL,
    prenume VARCHAR(100) CONSTRAINT prenume_antrenor NOT NULL,
    telefon VARCHAR(20) CONSTRAINT telefon_antrenor NOT NULL,
);

INSERT INTO ANTRENOR
VALUES (1, 'Gheorghe', 'Magi', '0789 564 589');

INSERT INTO ANTRENOR
VALUES (2, 'Ravan', 'Stan', '0756 852 323');

INSERT INTO ANTRENOR
VALUES (3, 'Marius', 'Popescu', '0742 210 301');

INSERT INTO ANTRENOR
VALUES (4, 'Florin', 'Bogdan', '0765 458 892');

INSERT INTO ANTRENOR
VALUES (5, 'Rares', 'Popa', '0712 541 132');
```

Below the worksheet, the 'Query Result' tab displays the data inserted into the table:

ID	NUME	PRENUME	TELEFON
1	Gheorghe	Magi	0789 564 589
2	Ravan	Stan	0756 852 323
3	Marius	Popescu	0742 210 301
4	Florin	Bogdan	0765 458 892
5	Rares	Popa	0712 541 132

Tabela ANTRENOR

```
CREATE TABLE CONTRACT(
    id_academie NUMBER(5) CONSTRAINT pk_id_academie REFERENCES
ACADEMIE(id),
    data DATE CONSTRAINT data_contract NOT NULL,
    id_antrenor NUMBER(5) CONSTRAINT pk_id_antrenor REFERENCES
ANTRENOR(id),
    salariu NUMBER CONSTRAINT salariu_contract NOT NULL,
    CONSTRAINT pk_compus_contract PRIMARY KEY(id_academie, id_antrenor)
);
```

INSERT INTO CONTRACT

```
VALUES (1, TO_DATE('2009-06-15','YYYY-MM-DD'), 1, 9800);
```

INSERT INTO CONTRACT

```
VALUES (2, TO_DATE('2009-07-20','YYYY-MM-DD'), 2, 7800);
```

INSERT INTO CONTRACT

```
VALUES (3, TO_DATE('2010-01-05','YYYY-MM-DD'), 1, 8500);
```

INSERT INTO CONTRACT

```
VALUES (4, TO_DATE('2010-04-30','YYYY-MM-DD'), 1, 7900);
```

INSERT INTO CONTRACT

```
VALUES (5, TO_DATE('2010-07-01','YYYY-MM-DD'), 1, 8250);
```

```
CREATE TABLE CONTRACT (
    id_academie NUMBER(5) CONSTRAINT pk_id_academie REFERENCES ACADEMIE(id),
    data DATE CONSTRAINT data_contract NOT NULL,
    id_antrenor NUMBER(5) CONSTRAINT pk_id_antrenor REFERENCES ANTRENOR(id),
    salariu NUMBER CONSTRAINT salariu_contract NOT NULL,
    CONSTRAINT pk_compas_contract PRIMARY KEY(id_academie, id_antrenor)
);

INSERT INTO CONTRACT
VALUES (1, TO_DATE('2009-06-15','YYYY-MM-DD'), 1, 9800);

INSERT INTO CONTRACT
VALUES (2, TO_DATE('2009-07-20','YYYY-MM-DD'), 2, 7800);

INSERT INTO CONTRACT
VALUES (3, TO_DATE('2010-01-05','YYYY-MM-DD'), 1, 8500);

INSERT INTO CONTRACT
VALUES (4, TO_DATE('2010-04-30','YYYY-MM-DD'), 1, 7900);

INSERT INTO CONTRACT
VALUES (5, TO_DATE('2010-07-01','YYYY-MM-DD'), 1, 8250);
```

ID_ACADEMIE	DATA	ID_ANTRENO	SALARIU
1	1 15-JUN-09	1	9800
2	2 20-JUL-09	2	7800
3	3 05-JAN-10	1	8500
4	4 30-APR-10	1	7900
5	5 01-JUL-10	1	8250

Tabela CONTRACT

```
CREATE TABLE TEREN(  
    id NUMBER(5) CONSTRAINT PKEY_TEREN PRIMARY KEY,  
    denumire VARCHAR(30) CONSTRAINT denumire_teren NOT NULL  
);
```

```
INSERT INTO TEREN  
VALUES (1, 'Teren Fotbal 1');
```

```
INSERT INTO TEREN  
VALUES (2, 'Teren Fotbal 2');
```

```
INSERT INTO TEREN  
VALUES (3, 'Teren Fotbal 3');
```

```
INSERT INTO TEREN  
VALUES (4, 'Teren Fotbal 4');
```

```
INSERT INTO TEREN  
VALUES (5, 'Teren Fotbal 5');
```

The screenshot shows the Oracle SQL Developer interface. In the 'Worksheet' tab, there is a SQL script window containing the following code:

```
CREATE TABLE TEREN(
    id NUMBER(5) CONSTRAINT PKEY_TEREN PRIMARY KEY,
    denumire VARCHAR(30) CONSTRAINT denumire_teren NOT NULL
);

INSERT INTO TEREN
VALUES (1, 'Teren Fotbal 1');

INSERT INTO TEREN
VALUES (2, 'Teren Fotbal 2');

INSERT INTO TEREN
VALUES (3, 'Teren Fotbal 3');

INSERT INTO TEREN
VALUES (4, 'Teren Fotbal 4');

INSERT INTO TEREN
VALUES (5, 'Teren Fotbal 5');
```

Below the script, the 'Script Output' tab shows the results of the insertions:

ID	DENUMIRE
1	Teren Fotbal 1
2	Teren Fotbal 2
3	Teren Fotbal 3
4	Teren Fotbal 4
5	Teren Fotbal 5

Tabela TEREN

CREATE TABLE POZITIE(

```
    id NUMBER(5) CONSTRAINT PKEY_POZITIE PRIMARY KEY,
    nume VARCHAR(50) CONSTRAINT nume_pozitie NOT NULL
);
```

INSERT INTO POZITIE

```
VALUES(1, 'CF');
```

INSERT INTO POZITIE

```
VALUES(2, 'LW/RW');
```

INSERT INTO POZITIE

```
VALUES(3, 'CM');
```

INSERT INTO POZITIE

```
VALUES(4, 'CB');
```

INSERT INTO POZITIE
VALUES(5, 'GK');

The screenshot shows the Oracle SQL Developer interface. In the 'Worksheet' tab, there is a code editor containing the following SQL script:

```
CREATE TABLE POZITIE(
    id NUMBER(5) CONSTRAINT PKEY_POZITIE PRIMARY KEY,
    nume VARCHAR(50) CONSTRAINT nume_pozitie NOT NULL
);

INSERT INTO POZITIE
VALUES(1, 'CF');

INSERT INTO POZITIE
VALUES(2, 'LM/RM');

INSERT INTO POZITIE
VALUES(3, 'CM');

INSERT INTO POZITIE
VALUES(4, 'CB');

INSERT INTO POZITIE
VALUES(5, 'GK');
```

Below the code editor is a 'Script Output' tab showing the results of the execution:

ID	NUME
1	CF
2	LM/RM
3	CM
4	CB
5	GK

A message at the bottom of the interface says: "Click on an identifier with the Control key down to perform "Go to Declaration".

Tabela POZITIE

CREATE TABLE APTITUDINE(
 id NUMBER(5) CONSTRAINT PKEY_APTITUDINE PRIMARY KEY,
 denumire VARCHAR(50) CONSTRAINT denumire_aptitudine NOT NULL,
 id_pozitie NUMBER(5), CONSTRAINT fk_aptitudine FOREIGN KEY(id_pozitie)
 REFERENCES POZITIE(id)
);

INSERT INTO APTITUDINE
VALUES (1, 'Sut', 1);

INSERT INTO APTITUDINE
VALUES (2, 'Viteza', 2);

INSERT INTO APTITUDINE
VALUES (3, 'Pase', 3);

INSERT INTO APTITUDINE
VALUES (4, 'Defensiva', 4);

INSERT INTO APTITUDINE
VALUES (5, 'Aparare', 5);

```
CREATE TABLE APTITUDINE(
    id NUMBER(5) CONSTRAINT PK_APTITUDINE PRIMARY KEY,
    denumire VARCHAR(50) CONSTRAINT denumire_aptitudine NOT NULL,
    id_pozitie NUMBER(5), CONSTRAINT fk_aptitudine FOREIGN KEY(id_pozitie) REFERENCES POZITIE(id)
);

INSERT INTO APTITUDINE
VALUES (1, 'Sut', 1);

INSERT INTO APTITUDINE
VALUES (2, 'Viteza', 2);

INSERT INTO APTITUDINE
VALUES (3, 'Pase', 3);

INSERT INTO APTITUDINE
VALUES (4, 'Defensiva', 4);

INSERT INTO APTITUDINE
VALUES (5, 'Aparare', 5);
```

Tabela APTITUDINE

CREATE TABLE NOTA(

```
nota NUMBER(5,2) CONSTRAINT nota_aptitudine NOT NULL,
id_sportiv NUMBER(5) CONSTRAINT pk_id_notaSportiv REFERENCES
SPORTIV(id),
id_aptitudine NUMBER(5) CONSTRAINT pk_id_notaAptitudine REFERENCES
APTITUDINE(id),
CONSTRAINT pk_compus_nota PRIMARY KEY (id_sportiv, id_aptitudine));
```

INSERT INTO NOTA

VALUES (9, 1, 1);

INSERT INTO NOTA

VALUES (8, 1, 2);

INSERT INTO NOTA

VALUES (8, 1, 3);

INSERT INTO NOTA

VALUES (7, 1, 4);

INSERT INTO NOTA

VALUES (3, 1, 5);

```
CREATE TABLE NOTA (
    nota NUMBER(6,2) CONSTRAINT nota_apititudine NOT NULL,
    id_sportiv NUMBER(5) CONSTRAINT pk_id_notaSportiv REFERENCES SPORTIV(id),
    id_apititudine NUMBER(5) CONSTRAINT pk_id_notaApititudine REFERENCES APITUDINE(id),
    CONSTRAINT pk_nota PRIMARY KEY (id_sportiv, id_apititudine)
);

INSERT INTO NOTA
VALUES (9, 1, 1);

INSERT INTO NOTA
VALUES (8, 1, 2);

INSERT INTO NOTA
VALUES (8, 1, 3);

INSERT INTO NOTA
VALUES (7, 1, 4);

INSERT INTO NOTA
VALUES (3, 1, 5);
```

Script Output | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | All Rows Fetched: 5 in 0,004 seconds

ID_NOTA	ID_SPORTIV	ID_APITUDINE
1	9	1
2	8	1
3	8	1
4	7	1
5	3	1

Tabela NOTA

```
CREATE TABLE ANTRENAMENT(
    id_catvarsta NUMBER(5) CONSTRAINT pk_id_catvarstaAntrenament
    REFERENCES CATEGORIEVARSTA(id),
    id_pozitie NUMBER(5) CONSTRAINT pk_id_pozitieAntrenament REFERENCES
    POZITIE(id),
    id_antrenor NUMBER(5) CONSTRAINT pk_id_antrenorAntrenament REFERENCES
    ANTRENOR(id),
    id_teren NUMBER(5) CONSTRAINT pk_id_terenAntrenament REFERENCES
    TEREN(id),
    CONSTRAINT pk_compus_antrenament PRIMARY KEY(id_catvarsta, id_pozitie,
    id_antrenor, id_teren)
);
```

```
INSERT INTO ANTRENAMENT
VALUES (1, 1, 1, 1);
```

```
INSERT INTO ANTRENAMENT
VALUES (1, 3, 1, 1);
```

```
INSERT INTO ANTRENAMENT
VALUES (1, 4, 2, 2);
```

```
INSERT INTO ANTRENAMENT
VALUES (1, 5, 3, 3);
```

```
INSERT INTO ANTRENAMENT
VALUES (2, 1, 1, 1);
```

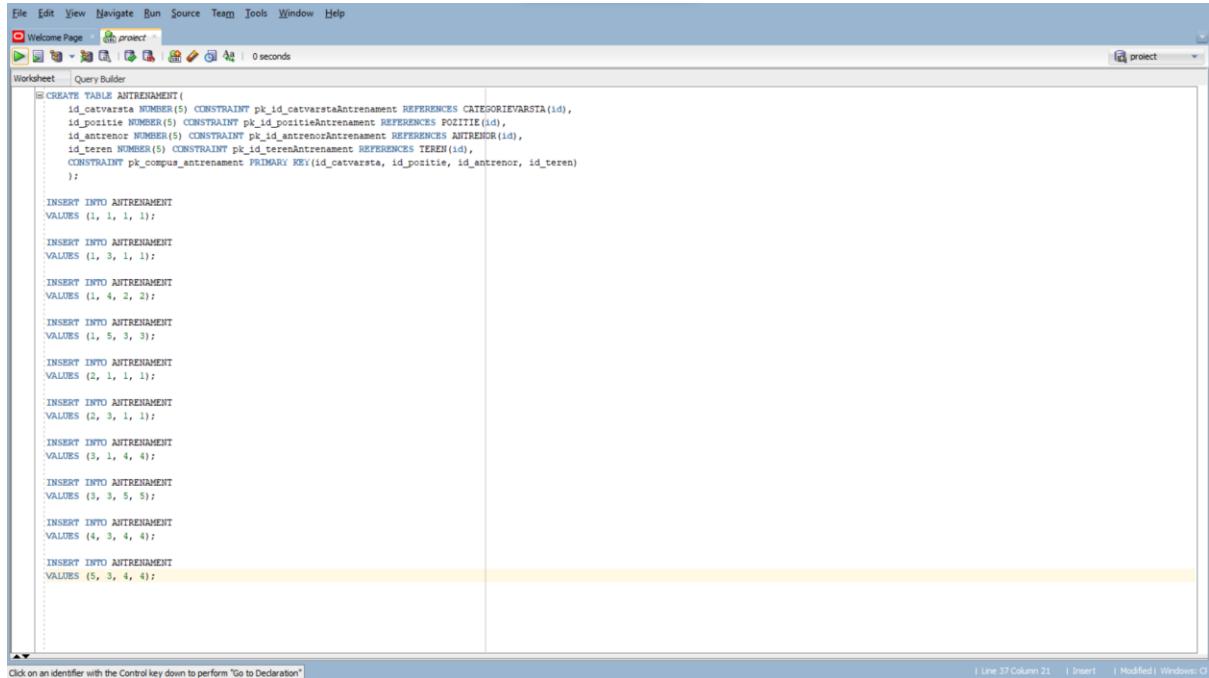
```
INSERT INTO ANTRENAMENT
VALUES (2, 3, 1, 1);
```

INSERT INTO ANTRENAMENT
VALUES (3, 1, 4, 4);

INSERT INTO ANTRENAMENT
VALUES (3, 3, 5, 5);

INSERT INTO ANTRENAMENT
VALUES (4, 3, 4, 4);

INSERT INTO ANTRENAMENT
VALUES (5, 3, 4, 4);



The screenshot shows a MySQL Workbench interface with the following code in the Worksheet tab:

```
CREATE TABLE ANTRENAMENT (
    id_catvarsta NUMBER(5) CONSTRAINT pk_id_catvarstaAntrenament REFERENCES CATEGORIEVARSTA(id),
    id_positie NUMBER(5) CONSTRAINT pk_id_positieAntrenament REFERENCES POZITIE(id),
    id_antrenor NUMBER(5) CONSTRAINT pk_id_antrenorAntrenament REFERENCES ANTRENOR(id),
    id_teren NUMBER(5) CONSTRAINT pk_id_terenAntrenament REFERENCES TEREN(id),
    CONSTRAINT pk_compoz_antrenament PRIMARY KEY(id_catvarsta, id_positie, id_antrenor, id_teren)
);

INSERT INTO ANTRENAMENT
VALUES (1, 1, 1, 1);

INSERT INTO ANTRENAMENT
VALUES (1, 3, 1, 1);

INSERT INTO ANTRENAMENT
VALUES (1, 4, 2, 2);

INSERT INTO ANTRENAMENT
VALUES (1, 5, 3, 3);

INSERT INTO ANTRENAMENT
VALUES (2, 1, 1, 1);

INSERT INTO ANTRENAMENT
VALUES (2, 3, 1, 1);

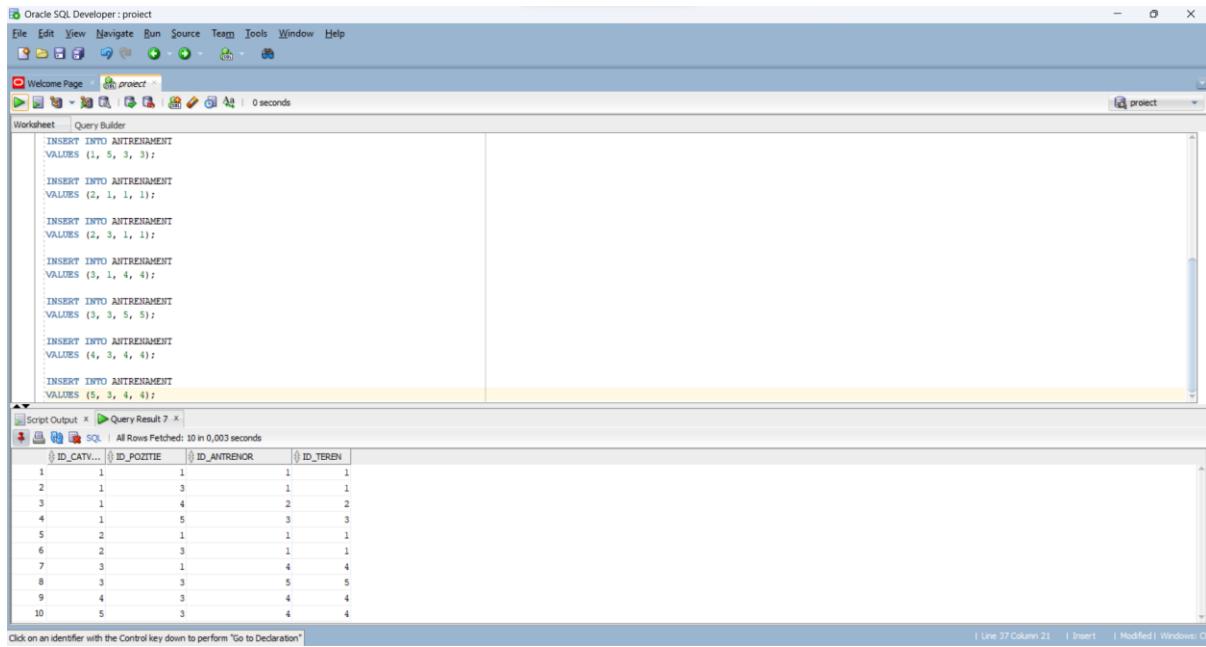
INSERT INTO ANTRENAMENT
VALUES (3, 1, 4, 4);

INSERT INTO ANTRENAMENT
VALUES (3, 3, 5, 5);

INSERT INTO ANTRENAMENT
VALUES (4, 3, 4, 4);

INSERT INTO ANTRENAMENT
VALUES (5, 3, 4, 4);
```

Tabela ANTRENAMENT



```
INSERT INTO ANTRENAMENT
VALUES (1, 5, 3, 3);

INSERT INTO ANTRENAMENT
VALUES (2, 1, 1, 1);

INSERT INTO ANTRENAMENT
VALUES (2, 3, 1, 1);

INSERT INTO ANTRENAMENT
VALUES (3, 1, 4, 4);

INSERT INTO ANTRENAMENT
VALUES (3, 3, 5, 5);

INSERT INTO ANTRENAMENT
VALUES (4, 3, 4, 4);

INSERT INTO ANTRENAMENT
VALUES (5, 3, 4, 4);
```

ID_CATEGORIE	ID_POZITIE	ID_ANTRENOR	ID_TEREN
1	1	1	1
2	1	3	1
3	1	4	2
4	1	5	3
5	2	1	1
6	2	3	1
7	3	1	4
8	3	3	5
9	4	3	4
10	5	3	4

Tabela ANTRENAMENT

6. Un subprogram care să utilizeze două tipuri diferite de colecții studiate

- pentru o academie al carei nume este dat pentru fiecare categorie de varsta afisati pozitiile pentru care se antreneaza cu ce antrenor,
- unde se antreneaza si elevii care fac parte din categoria de varsta

CREATE OR REPLACE PROCEDURE ex6 (numeAcademie Academie.nume%TYPE)

AS

 TYPE tablouIndexat IS TABLE OF POZITIE%ROWTYPE INDEX BY PLS_INTEGER;

 pozitii tablouIndexat;

 TYPE tablouImbricat IS TABLE OF CATEGORIEVARSTA%ROWTYPE;

 categoriivarsta tablouImbricat := tablouImbricat();

 TYPE output IS TABLE OF VARCHAR(200) INDEX BY PLS_INTEGER;

sportivi output;
desfAntrenament output;

BEGIN

 SELECT *

 BULK COLLECT INTO pozitii
 FROM pozitie;

-- IF pozitii.count = 0 THEN
-- DBMS_OUTPUT.PUT_LINE('Nu exista pozitii');
-- END IF;

--
-- IF pozitii.count > 0 THEN
-- FOR i IN pozitii.first..pozitii.last LOOP
-- DBMS_OUTPUT.PUT_LINE(pozitii(i).nume);
-- END LOOP;
-- END IF;

SELECT catv.id, catv.denumire, catv.id_sport
BULK COLLECT INTO categoriivarsta
FROM CATEGORIEVARSTA catv, SPORT s, ACADEMIE a
WHERE catv.id_sport = s.id and s.id_academie = a.id
AND UPPER(a.nume) LIKE UPPER(numeAcademie);

-- IF categoriivarsta.count = 0 THEN
-- DBMS_OUTPUT.PUT_LINE('Nu exista categorii de varsta');
-- END IF;

--
-- IF categoriivarsta.count > 0 THEN
-- FOR i IN categoriivarsta.first..categoriivarsta.last LOOP

```
--      DBMS_OUTPUT.PUT_LINE(categoriivarsta(i).denumire);
--      END LOOP;
--      END IF;

FOR i IN categoriivarsta.first..categoriivarsta.last LOOP
    DBMS_OUTPUT.PUT_LINE(categoriivarsta(i).denumire || ' ani');
    DBMS_OUTPUT.PUT_LINE(");

        FOR j IN pozitii.first..pozitii.last LOOP
            DBMS_OUTPUT.PUT_LINE(pozitii(j).nume || ':');
            SELECT t.denumire || ' - ' || ant.nume || ' ' || ant.prenume
            BULK COLLECT INTO desfAntrenament
            FROM ANTRENAMENT a, TEREN t, ANTRENOR ant
            WHERE a.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(j).id AND
            a.id_teren = t.id AND a.id_antrenor = ant.id;

            IF desfAntrenament.count > 0 THEN
                SELECT s.nume || ' ' || s.prenume
                BULK COLLECT INTO sportivi
                FROM SPORTIV s, INSCRIERE insc, ANTRENAMENT a
                WHERE s.id = insc.id_sportiv AND insc.id_catvarsta = categoriivarsta(i).id AND
                a.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(j).id;

                FOR k in desfAntrenament.first..desfAntrenament.last LOOP
                    DBMS_OUTPUT.PUT_LINE(' - ' || desfAntrenament(k));
                    IF sportivi.count > 0 THEN
                        FOR l IN sportivi.first..sportivi.last LOOP
                            DBMS_OUTPUT.PUT_LINE(sportivi(l));
                        END LOOP;
                    END LOOP;
                END LOOP;
            END IF;
        END LOOP;
    END LOOP;
END IF;
```

```
        ELSE DBMS_OUTPUT.PUT_LINE('Nu exista sportivi inregistrati pentru acest
        antrenament');

        END IF;

        END LOOP;

        DBMS_OUTPUT.PUT_LINE('-----');

        END IF;

IF desfAntrenament.count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista');
    DBMS_OUTPUT.PUT_LINE('-----');
END IF;

DBMS_OUTPUT.PUT_LINE(");

END LOOP;

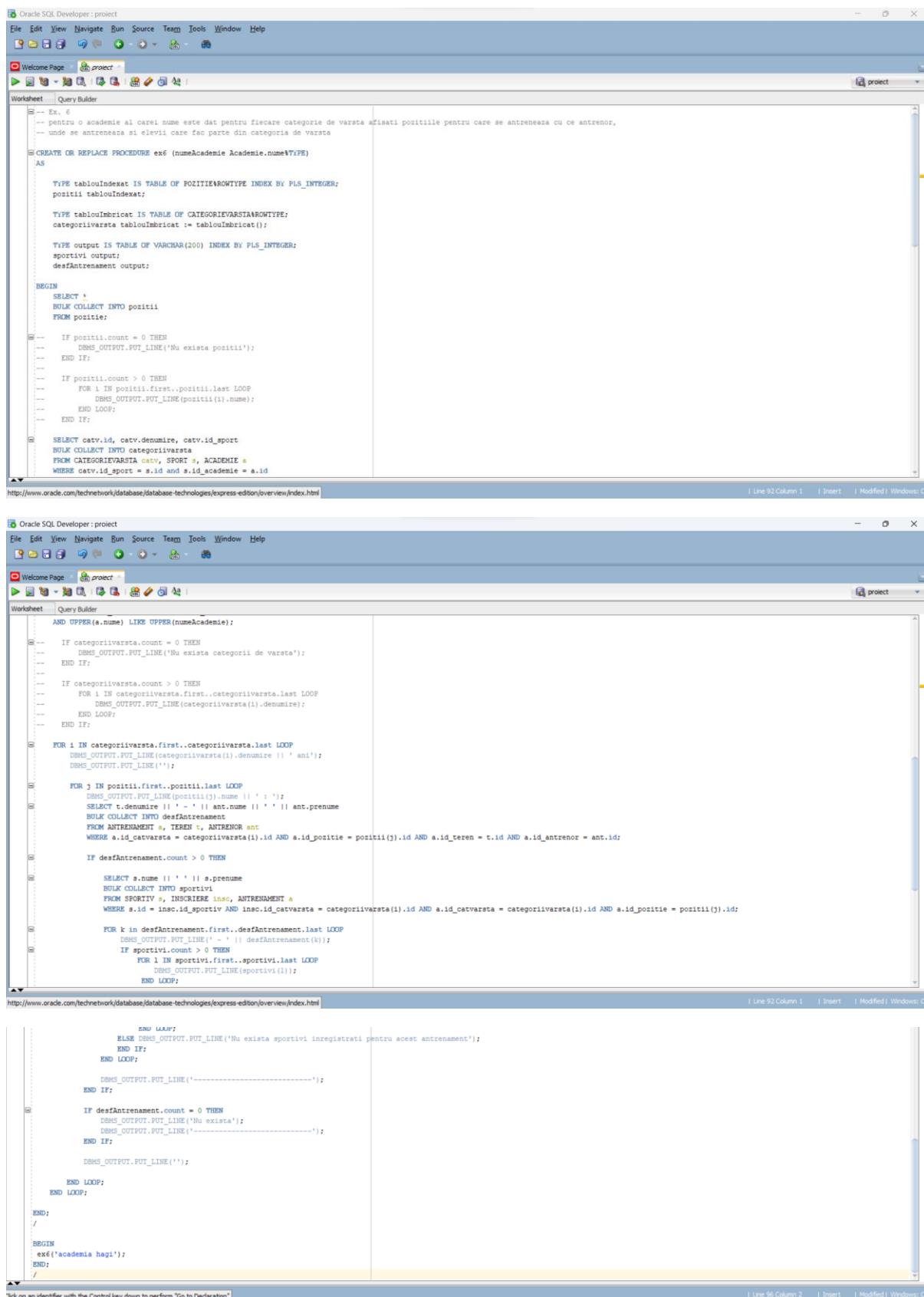
END LOOP;

END;
/

BEGIN
ex6('academia hagi');

END;
/
```

Sisteme de gestiune a bazelor de date
 Petre Vasile-Eduard
 Grupa 243



```

-- Ex. 6
-- pentru o academie al carei nume este dat pentru fiecare categorie de varsta afisati pozitiile pentru care se antrenaza cu ce antrenor,
-- unde se antrenaza si elevii care fac parte din categoria de varsta

CREATE OR REPLACE PROCEDURE ex6 (numeAcademie Academie.nume%TYPE)
AS

    TYPE tablouIndexat IS TABLE OF POZITIE%ROWTYPE INDEX BY PLS_INTEGER;
    pozitii tablouIndexat;

    TYPE tablouImbricat IS TABLE OF CATEGORIEVARSTA%ROWTYPE;
    categoriivarsta tablouImbricat := tablouImbricat();

    TYPE output IS TABLE OF VARCHAR(200) INDEX BY PLS_INTEGER;
    sportivi output;
    desfintrenament output;

BEGIN
    SELECT *
    BULK COLLECT INTO pozitii
    FROM pozitie;

    IF pozitii.count = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista pozitii');
    END IF;

    IF pozitii.count > 0 THEN
        FOR i IN pozitii.first..pozitii.last LOOP
            DBMS_OUTPUT.PUT_LINE(pozitii(i).nume);
        END LOOP;
    END IF;

    SELECT catv.id, catv.denumire, catv.id_sport
    BULK COLLECT INTO categoriivarsta
    FROM CATEGORIEVARSTA catv, SPORT s, ACADEMIE a
    WHERE catv.id_sport = s.id AND s.id_academie = a.id

```



```

AND UPPER(a.name) LIKE UPPER(numeAcademie);

IF categoriivarsta.count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista categorii de varsta');
END IF;

IF categoriivarsta.count > 0 THEN
    FOR i IN categoriivarsta.first..categoriivarsta.last LOOP
        DBMS_OUTPUT.PUT_LINE(categoriivarsta(i).denumire || ' ani');
        DBMS_OUTPUT.PUT_LINE('');
    END LOOP;
END IF;

FOR i IN pozitii.first..pozitii.last LOOP
    DBMS_OUTPUT.PUT_LINE(pozitii(i).nume || ' ');
    SELECT t.denumire || ' ' || ant.nume || ' ' || ant.prenume
    BULK COLLECT INTO desfintrenament
    FROM ANTRENAMENTE a, TEREN t, ANTRENOR ant
    WHERE a.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(i).id AND a.id_teren = t.id AND a.id_antrenor = ant.id;

    IF desfintrenament.count > 0 THEN
        SELECT a.name || ' ' || s.prenume
        BULK COLLECT INTO sportivi
        FROM SPORTIV s, INSCRIERE insc, ANTRENAMENT a
        WHERE s.id = insc.id_sportiv AND insc.id_catvarsta = categoriivarsta(i).id AND a.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(i).id;

        FOR k IN desfintrenament.first..desfintrenament.last LOOP
            DBMS_OUTPUT.PUT_LINE(' - ' || desfintrenament(k));
            IF sportivi.count > 0 THEN
                FOR l IN sportivi.first..sportivi.last LOOP
                    DBMS_OUTPUT.PUT_LINE(sportivi(l));
                END LOOP;
            END IF;
        END LOOP;
    END IF;

```



```

        ELSE DBMS_OUTPUT.PUT_LINE('Nu exista sportivi inregistrati pentru acest antrenament');
        END IF;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('-----');
END IF;

IF desfintrenament.count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista');
    DBMS_OUTPUT.PUT_LINE('-----');
END IF;

DBMS_OUTPUT.PUT_LINE('');

END LOOP;
END LOOP;
END;
/

BEGIN
    ex6('academia hagi');
END;
/

```

Sisteme de gestiune a bazelor de date
Petre Vasile-Eduard
Grupa 243

17-18 ani

CF :
- Teren Fotbal 1 - Gheorghe Hagi
Marian Alex
Florian Petrescu
Florian Leonard
Coman Razvan
Lazar Narcis

LW/RW :
Nu exista

CH :
- Teren Fotbal 1 - Gheorghe Hagi
Marian Alex
Florian Petrescu
Florian Leonard
Coman Razvan
Lazar Narcis

CB :
- Teren Fotbal 2 - Razvan Stan
Marian Alex
Florian Petrescu
Florian Leonard

Coman Razvan

Lazar Narcis

15-16 ani

CF :
- Teren Fotbal 1 - Gheorghe Hagi
Nu exista sportivi inregistrati pentru acest antrenament

LW/RW :
Nu exista

CM :
- Teren Fotbal 1 - Gheorghe Hagi
Nu exista sportivi inregistrati pentru acest antrenament

CB :
Nu exista

GK :
Nu exista

13-14 ani

CF :
- Teren Fotbal 4 - Florin Bogdan
Nu exista sportivi inregistrati pentru acest antrenament

LW/RW :
Nu exista

CH :
- Teren Fotbal 5 - Rares Popa
Nu exista sportivi inregistrati pentru acest antrenament

CB :
Nu exista

GK :
Nu exista

11-12 ani

CF :
Nu exista

LW/RW :
Nu exista

CM :
- Teren Fotbal 4 - Florin Bogdan
Nu exista sportivi inregistrati pentru acest antrenament

CB :
Nu exista

GK :
Nu exista

9-10 ani

CF :
Nu exista

LW/RW :
Nu exista

CH :
- Teren Fotbal 4 - Florin Bogdan
Nu exista sportivi inregistrati pentru acest antrenament

CB :
Nu exista

GK :
Nu exista

7. Un subprogram care să utilizeze două tipuri diferite de cursoare studiate
-- pentru un sportiv al carui nume este dat afisati pozitiile pentru care s-ar incadra in functie
de notele obtinute pentru aptitudinile sale

CREATE OR REPLACE PROCEDURE Ex7 (nume SPORTIV.nume%TYPE, prenume
SPORTIV.prenume%TYPE)

AS

```
CURSOR note (idSportiv SPORTIV.id%TYPE) IS
    SELECT n.id_aptitudine
    FROM NOTA n
    WHERE n.id_sportiv = idSportiv AND n.nota >=8;
```

```
CURSOR sportivi IS
    SELECT id, nume, prenume
    FROM SPORTIV;
```

```
CURSOR pozitii(idAptitudine APTITUDINE.id%TYPE) IS
    SELECT p.nume as output
    FROM APTITUDINE a, POZITIE p
    WHERE a.id = idAptitudine AND a.id_pozitie = p.id;
```

```
    idAptitudine APTITUDINE.id%TYPE;
```

BEGIN

```
    FOR sportiv in sportivi LOOP
        IF UPPER(sportiv.nume) LIKE UPPER(nume) AND UPPER(sportiv.prenume) LIKE
        UPPER(prenume) THEN
            DBMS_OUTPUT.PUT_LINE('Nume: ' || UPPER(nume) || ' ' || UPPER(prenume));
        END IF;
    OPEN note(sportiv.id);
```

```
DBMS_OUTPUT.PUT_LINE('Pentru aptitudinile sale, pozitiile sugerate sunt
urmatoarele: ');

LOOP
    FETCH note INTO idAptitudine;
    EXIT WHEN note%NOTFOUND;

    FOR pozitie IN pozitii(idAptitudine) LOOP
        DBMS_OUTPUT.PUT_LINE(' - ' || pozitie.output);
    END LOOP;
    CLOSE note;

    END IF;
END LOOP;

END;
/
BEGIN
    ex7('marian', 'alex');
END;
/
```

```

-- Ex. 7
-- pentru un sportiv al carui nume este dat afisati pozitiile pentru care s-ar incafra in functie de notele obtinute pentru aptitudinile sale

CREATE OR REPLACE PROCEDURE Ex7 (nume SPORTIV.nume%TYPE, prenume SPORTIV.prenume%TYPE)
AS
    CURSOR note (idSportiv SPORTIV.id%TYPE) IS
        SELECT n.id_aptitudine
        FROM NOTA n
        WHERE n.id_sportiv = idSportiv AND n.nota >=8;

    CURSOR sportivi IS
        SELECT id, nume, prenume
        FROM SPORTIV;

    CURSOR pozitii(idAptitudine APITITUDINE.id%TYPE) IS
        SELECT p.nume AS output
        FROM APITITUDINE a, POZITIE p
        WHERE a.id = idAptitudine AND a.id_pozitie = p.id;

    idAptitudine APITITUDINE.id%TYPE;

BEGIN
    FOR sportiv IN sportivi LOOP
        IF UPPER(sportiv.nume) LIKE UPPER(nume) AND UPPER(sportiv.prenume) LIKE UPPER(prenume) THEN
            DBMS_OUTPUT.PUT_LINE('Name: ' || UPPER(nume) || ' ' || UPPER(prenume));
            OPEN note(sportiv.id);
            DBMS_OUTPUT.PUT_LINE('Pentru aptitudinile sale, pozitiile sugerate sunt urmatoarele: ');
            LOOP
                FETCH note INTO idAptitudine;
                EXIT WHEN note%NOTFOUND;
                FOR pozitie IN pozitii(idAptitudine) LOOP
                    DBMS_OUTPUT.PUT_LINE(' - ' || pozitie.output);
                END LOOP;
            END LOOP;
        END IF;
        END LOOP;
    END;
/
BEGIN
    ex7('marian', 'alex');
END;
/

```

Click on an identifier with the Control key down to perform "Go to Declaration"

Click on an identifier with the Control key down to perform "Go to Declaration"

```

PL/SQL procedure successfully completed.

Name: MARIAN ALEX
Pentru aptitudinile sale, pozitiile sugerate sunt urmatoarele:
- CF
- LW/RW
- CH

PL/SQL procedure successfully completed.

```

8. Un subprogram de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 exceptii

-- pentru un antrenor al carui id este dat si o categorie de varsta a carei nume este dat, returnati numarul de pozitii pentru care au loc antrenamentele

-- exceptii

-- Campul pentru nume este gol

-- Id-ul pentru teren este negativ

```
CREATE OR REPLACE FUNCTION ex8(numeCatVarsta
CATEGORIEVARSTA.denumire%TYPE, idAntrenor ANTRENOR.id%TYPE) RETURN
NUMBER

IS
    nrPozitii NUMBER(3);

    IS_EMPTY EXCEPTION;
    INVALID_ID EXCEPTION;

BEGIN
    IF numeCatVarsta IS NULL THEN
        RAISE IS_EMPTY;
    END IF;

    IF idAntrenor <= 0 THEN
        RAISE INVALID_ID;
    END IF;

    SELECT COUNT(p.id)
    INTO nrPozitii
    FROM POZITIE p
    JOIN ANTRENAMENT a ON (p.id = a.id_pozitie)
    JOIN CATEGORIEVARSTA c ON(a.id_catvarsta = c.id)
    WHERE UPPER(c.denumire) LIKE UPPER(numeCatVarsta) AND a.id_antrenor = id_ant;

    IF nrPozitii <= 0 THEN
        RAISE NO_DATA_FOUND;
    ELSE RETURN nrPozitii;
    END IF;
```

EXCEPTION

WHEN IS_EMPTY THEN

DBMS_OUTPUT.PUT_LINE('Campul pentru numele categoriei de varsta nu poate fi gol!');

RETURN -1;

WHEN INVALID_ID THEN

DBMS_OUTPUT.PUT_LINE('ID Invalid!');

RETURN -1;

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('Nu au fost gasite categorii de varsta!');

RETURN -1;

END;

/

BEGIN

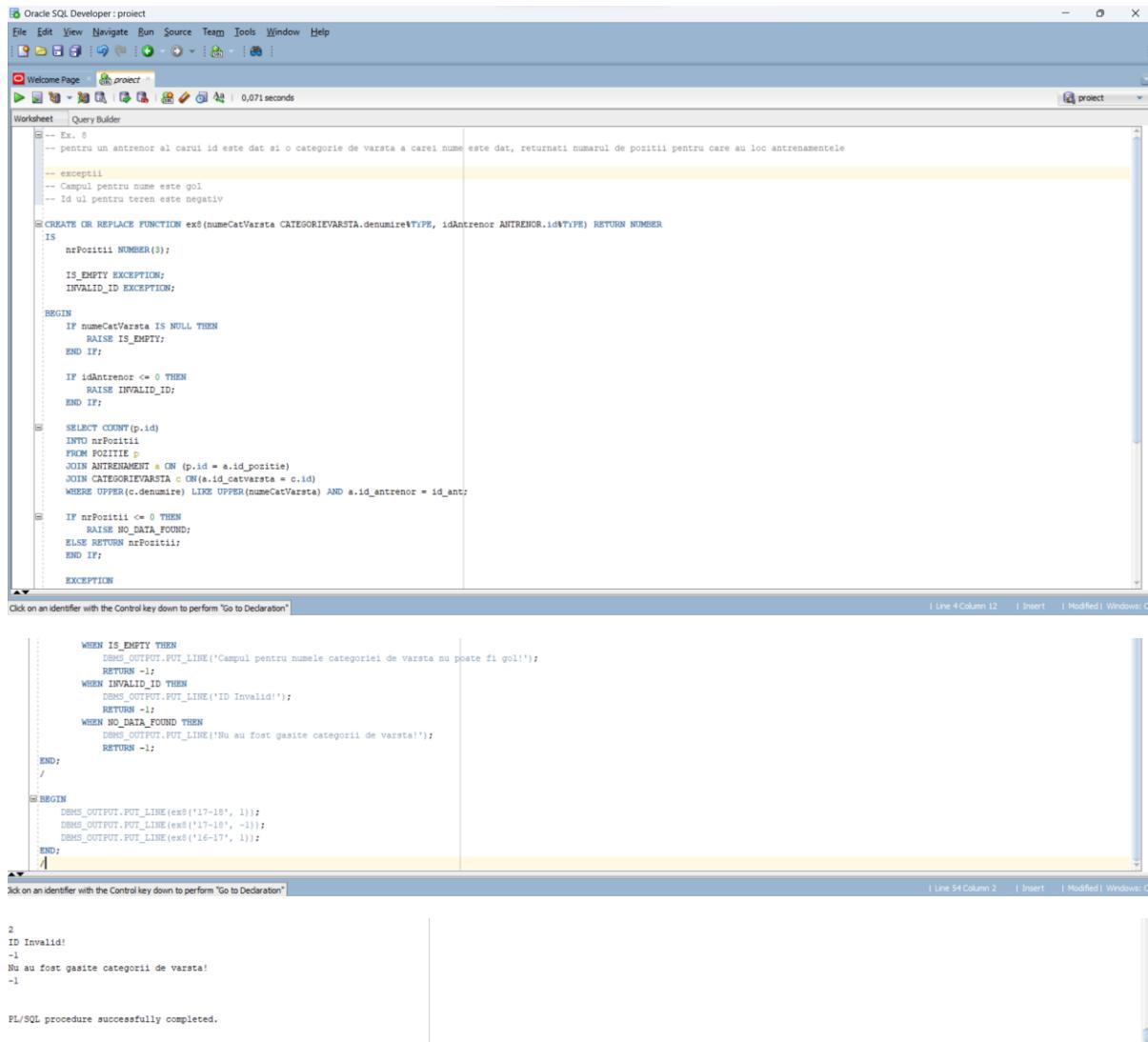
DBMS_OUTPUT.PUT_LINE(ex8('17-18', 1));

DBMS_OUTPUT.PUT_LINE(ex8('17-18', -1));

DBMS_OUTPUT.PUT_LINE(ex8('16-17', 1));

END;

/



```

-- Ex. 8
-- pentru un antrenor al carui id este dat si o categorie de varsta a carei nume este dat, returnati numarul de pozitii pentru care au loc antrenamentele
-- exceptii
-- Campul pentru nume este gol
-- Id ul pentru teren este negativ

CREATE OR REPLACE FUNCTION ex8(numeCatVarsta CATEGORIEVARSTA.denumire%TYPE, idAntrenor ANTRENOR.id%TYPE) RETURN NUMBER
IS
    nrPozitii NUMBER(3);
    IS_EMPTY EXCEPTION;
    INVALID_ID EXCEPTION;

BEGIN
    IF numeCatVarsta IS NULL THEN
        RAISE IS_EMPTY;
    END IF;

    IF idAntrenor <= 0 THEN
        RAISE INVALID_ID;
    END IF;

    SELECT COUNT(p.id)
    INTO nrPozitii
    FROM POZITIE p
    JOIN ANTRENAMENT a ON (p.id = a.id_pozitie)
    JOIN CATEGORIEVARSTA c ON (a.id_catvarsta = c.id)
    WHERE UPPER(c.denumire) LIKE UPPER(numeCatVarsta) AND a.id_antrenor = id_ant;

    IF nrPozitii <= 0 THEN
        RAISE NO_DATA_FOUND;
    ELSE RETURN nrPozitii;
    END IF;

EXCEPTION
    WHEN IS_EMPTY THEN
        DBMS_OUTPUT.PUT_LINE('Campul pentru numele categoriei de varsta nu poate fi gol!');
        RETURN -1;
    WHEN INVALID_ID THEN
        DBMS_OUTPUT.PUT_LINE('ID Invalid!');
        RETURN -1;
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu au fost gasite categorii de varsta!');
        RETURN -1;
END;
/

```

```

2
ID Invalid!
-1
Nu au fost gasite categorii de varsta!
-1

PL/SQL procedure successfully completed.

```

9. Un subprogram 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

-- pentru un sportiv al carui nume este dat specificați ce aptitudine principala încearcă să își
 -- imbunătățească în cadrul unui antrenament doar dacă nota la acea aptitudine este mai
 -- mică decât 8

-- exceptii
 -- Campul pentru nume este gol

-- NO_DATA_FOUND - nu exista sportivi cu acest nume sau nu exista aptitudini care sa indeplineasca conditiile

-- TOO_MANY_ROWS - exista mai multi sportivi cu acelasi nume

```
CREATE OR REPLACE PROCEDURE ex9(numeSportiv SPORTIV.nume%TYPE)
```

```
AS
```

```
    TYPE tablouAptitudini IS TABLE OF APTITUDINE.denumire%TYPE INDEX BY  
    PLS_INTEGER;
```

```
    aptitudini tablouAptitudini;
```

```
    sportivi SPORTIV%ROWTYPE;
```

```
    IS_EMPTY EXCEPTION;
```

```
BEGIN
```

```
    IF numeSportiv IS NULL THEN
```

```
        RAISE IS_EMPTY;
```

```
    END IF;
```

```
    SELECT *
```

```
        INTO sportivi
```

```
        FROM SPORTIV s
```

```
        WHERE UPPER(s.nume) LIKE UPPER(numeSportiv);
```

```
    SELECT apt.denumire
```

```
        BULK COLLECT INTO aptitudini
```

```
        FROM APTITUDINE apt
```

```
        JOIN POZITIE p ON (p.id = apt.id_pozitie)
```

```
        JOIN ANTRENAMENT a ON(a.id_pozitie = p.id)
```

```
        JOIN INSCRIERE ins ON (ins.id_catvarsta = a.id_catvarsta)
```

```
JOIN SPORTIV s ON (s.id = ins.id_sportiv)
JOIN NOTA n ON (n.id_sportiv = s.id AND n.id_aptitudine = apt.id)
WHERE UPPER(s.nume) LIKE UPPER(numeSportiv) AND n.nota <= 8;
IF SQL%NOTFOUND THEN
    RAISE NO_DATA_FOUND;
END IF;

DBMS_OUTPUT.PUT_LINE('Sportivul ' || UPPER(numeSportiv) || ' inca lucreaza la
urmatoarele aptitudini: ');
FOR i IN aptitudini.first..aptitudini.last LOOP
    DBMS_OUTPUT.PUT_LINE(' - ' || aptitudini(i));
END LOOP;

EXCEPTION
    WHEN IS_EMPTY THEN
        DBMS_OUTPUT.PUT_LINE('Campul pentru nume nu poate fi gol!');
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Sportivul cu acest nume nu exista sau nu au fost gasite
aptitudini care sa indeplineasca conditiile!');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Există mai mulți sportivi cu același nume!');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Codul de eroare: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE('Mesajul erorii: ' || SQLERRM);
END;
/
BEGIN
    ex9();

```

```
ex9('marian');

ex9('florian');

ex9('coman');

ex9('petre');

END;

/

--SELECT *

--FROM POZITIE p

--JOIN APTITUDINE apt ON (apt.id_pozitie = p.id)

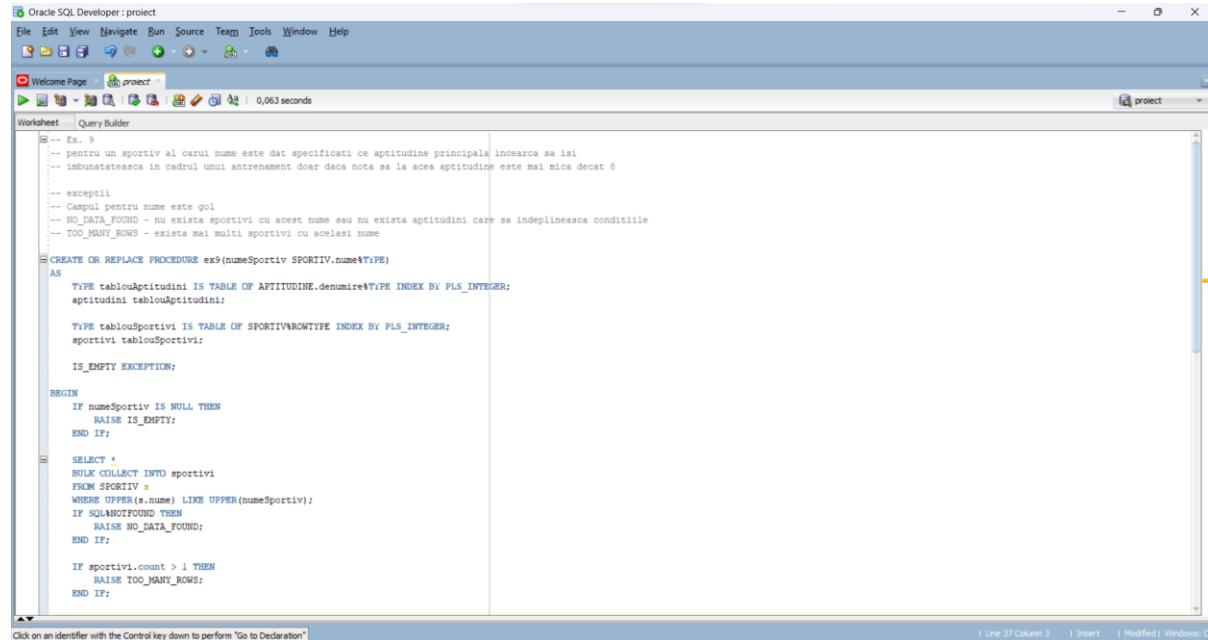
--JOIN NOTA n ON (n.id_aptitudine = apt.id)

--JOIN SPORTIV s ON (s.id = n.id_sportiv)

--JOIN ANTRENAMENT a ON (a.id_pozitie = p.id)

--JOIN INSCRIERE ins ON (ins.id_catvarsta = a.id_catvarsta AND ins.id_sportiv = s.id)

--WHERE UPPER(s.nume) LIKE UPPER('marian') AND n.nota<=8;
```



The screenshot shows the Oracle SQL Developer interface with the following details:

- Project:** project
- File Menu:** File Edit View Navigate Run Source Team Tools Window Help
- Toolbar:** Standard toolbar with icons for New, Open, Save, Run, Stop, etc.
- Header:** Welcome Page project | 0,063 seconds
- Worksheet:** The main area contains the PL/SQL code for Ex. 9. The code includes comments explaining the logic for handling multiple sportsmen with the same name and ensuring no more than one sport is registered per activity.
- Bottom Status Bar:** Click on an identifier with the Control key down to perform "Go to Declaration". Line 37 Column 3 | Insert | Modified | Windows: 0

```

SELECT apt.denumire
BULK COLLECT INTO aptitudini
FROM APTITUDINE apt
JOIN POZITIE p ON (p.id = apt.id_pozitie)
JOIN ANTRENAMENT a ON (a.id_pozitie = p.id)
JOIN INSCHRIERE ins CN (ins.id_catvarsta = a.id_catvarsta)
JOIN SPORTIV s ON (s.id = ins.id_sportiv)
JOIN NOTA n ON (n.id_sportiv = s.id AND n.id_aptitudine = apt.id)
WHERE UPPER(s.nume) LIKE UPPER(numarSportiv) AND n.nota <= 8;
IF SQLNOTFOUND THEN
    RAISE NO_DATA_FOUND;
END IF;

DBMS_OUTPUT.PUT_LINE('Sportivul ' || UPPER(numarSportiv) || ' inca lucreaza la urmatoarele aptitudini: ');
FOR i IN aptitudini.first..aptitudini.last LOOP
    DBMS_OUTPUT.PUT_LINE(' - ' || aptitudini(i));
END LOOP;

EXCEPTION
    WHEN IS_EMPTY THEN
        DBMS_OUTPUT.PUT_LINE('Campul pentru nume nu poate fi gol!');
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Sportivul cu acest nume nu exista sau nu au fost gasite aptitudini care sa indeplineasca conditiile!');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Există mai multi sportivi cu același nume!');
END;
/
BEGIN
    ex9('marijan');
    ex9('florian');
    ex9('coman');
END;
/

```

Sportivul MARIAN inca lucreaza la urmatoarele sale aptitudini:
 - Pase
 - Defensiva
 - Aparare
 Există mai multi sportivi cu același nume!
 Sportivul cu acest nume nu exista sau nu au fost gasite aptitudini ale acestuia!

PL/SQL procedure successfully completed.

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

CREATE OR REPLACE TRIGGER ex10

BEFORE INSERT OR DELETE OR UPDATE ON ANTRENAMENT

BEGIN

IF TO_CHAR(SYSDATE,'MM') NOT BETWEEN 3 AND 12 OR
 TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 24 THEN

IF INSERTING THEN

RAISE_APPLICATION_ERROR(-20001,'Inserarea in tabel este interzisa in afara
 sezonului sau pe parcursul noptii! Odihna face parte din antrenament!');

ELSIF DELETING THEN

RAISE_APPLICATION_ERROR(-20002,'Stergerea din tabel este interzisa in afara
 sezonului sau pe parcursul noptii! Odihna face parte din antrenament!');

ELSE

RAISE_APPLICATION_ERROR(-20003,'Actualizările in tabel sunt interzise in afara
 sezonului sau pe parcursul noptii! Odihna face parte din antrenament!');

END IF;

END IF;

END;

/

UPDATE ANTRENAMENT SET id_teren = 5 WHERE id_catvarsta = 5;

The screenshot shows the Oracle SQL Developer interface. In the central workspace, there is a code editor window containing a PL/SQL trigger definition named 'EX10'. The trigger is defined as follows:

```
-- Ex. 10
CREATE OR REPLACE TRIGGER ex10
BEFORE INSERT OR UPDATE OR DELETE ON ANTRENAMENT
BEGIN
    IF TO_CHAR(SYSDATE,'MM') NOT BETWEEN 3 AND 12 OR TO_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 24 THEN
        RAISE_APPLICATION_ERROR(-20001,'Inserarea in tabel este interzisa in afara sezonului sau pe parcursul noptii! Odihna face parte din antrenament!');
    ELSIF DELETING THEN
        RAISE_APPLICATION_ERROR(-20002,'Stergerea din tabel este interzisa in afara sezonului sau pe parcursul noptii! Odihna face parte din antrenament!');
    ELSE
        RAISE_APPLICATION_ERROR(-20003,'Actualizările in tabel sunt interzise in afara sezonului sau pe parcursul noptii! Odihna face parte din antrenament!');
    END IF;
END;
/
UPDATE ANTRENAMENT SET id_teren = 5 WHERE id_catvarsta = 5;
```

Below the code editor, the 'Script Output' tab shows an error message:

```
Error starting at line : 18 in command -
UPDATE ANTRENAMENT SET id_teren = 5 WHERE id_catvarsta = 5
Error report -
ORA-20003: Actualizările in tabel sunt interzise in afara sezonului sau pe parcursul noptii! Odihna face parte din antrenament!
ORA-06512: at "SYSTEM.EX10", line 8
ORA-04080: error during execution of trigger 'SYSTEM.EX10'
```

The status bar at the bottom indicates 'Task completed in 0,032 seconds'.

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

CREATE OR REPLACE TRIGGER ex11

BEFORE INSERT OR UPDATE OF data ON INSCRIERE

FOR EACH ROW

WHEN (NEW.data < TO_DATE('2009-06-15', 'YYYY-MM-DD'))

BEGIN

RAISE_APPLICATION_ERROR (-20000, 'Nu este posibila inscrierea sportivilor la o data mai veche decat infiintarea academiei!');

END;

/

UPDATE INSCRIERE SET data = TO_DATE('2008-03-21', 'YYYY-MM-DD') WHERE id_sportiv = 5;

The screenshot shows the Oracle SQL Developer interface. In the 'Query Builder' worksheet, there is a trigger creation script:

```
-- Ex. 11
CREATE OR REPLACE TRIGGER ex11
  BEFORE INSERT OR UPDATE OF data ON INSCRIERE
  FOR EACH ROW
  WHEN (NEW.data < TO_DATE('2009-06-15', 'YYYY-MM-DD'))
BEGIN
  RAISE_APPLICATION_ERROR (-20000, 'Nu este posibila inscrierea sportivilor la o data mai veche decat inaintarea academiei!');
END;
/
UPDATE INSCRIERE SET data = TO_DATE('2008-03-21', 'YYYY-MM-DD') WHERE id_sportiv = 5;
```

In the 'Query Result' tab, the output shows an error:

```
Error starting at line : 12 in command -
UPDATE INSCRIERE SET data = TO_DATE('2008-03-21', 'YYYY-MM-DD') WHERE id_sportiv = 5
Error report -
ORA-20000: Nu este posibila inscrierea sportivilor la o data mai veche decat inaintarea academiei!
ORA-06512: at "SYSTEM.EX11", line 2
ORA-04081: error during execution of trigger 'SYSTEM.EX11'
```

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

```
CREATE TABLE tabelEx12(
    databasen VARCHAR2(10),
    usern VARCHAR2(10),
    statementn VARCHAR2(10),
    data TIMESTAMP(1)
);
```

```
--DROP TRIGGER ex12;
--DROP TABLE tabelEx12;
```

```
CREATE OR REPLACE TRIGGER ex12
AFTER CREATE OR DROP ON SCHEMA
```

BEGIN

 INSERT INTO tabelEx12

 VALUES (SYS.DATABASE_NAME, SYS.LOGIN_USER, SYS.SYSEVENT,
 SYSTIMESTAMP(3));

END;

/

CREATE TABLE testEx12 (x varchar(200));

INSERT INTO testEx12 VALUES ("Test ex12");

DROP TABLE testEx12;

SELECT * FROM tabelEx12;

The screenshot shows the Oracle SQL Developer interface. The main area displays a script named 'Ex_12' containing the provided SQL code. Below the script, the 'Query Result' tab is active, showing the execution log:

DATA BASEN	USERN	STATEMENTN	DATA
1 XE	SYSTEM	CREATE	23-MAY-23 04.46.24.60000000 PM
2 XE	SYSTEM	DROP	23-MAY-23 04.46.24.60000000 PM

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

CREATE OR REPLACE PACKAGE ex13 AS

 PROCEDURE ex6(numeAcademie ACADEMIE.nume%TYPE);

 PROCEDURE ex7(nume SPORTIV.nume%TYPE, prenume SPORTIV.prenume%TYPE);

```
FUNCTION ex8(numeCatVarsta CATEGORIEVARSTA.denumire%TYPE, idAntrenor
ANTRENOR.id%TYPE) RETURN NUMBER;
PROCEDURE ex9(numeSportiv SPORTIV.nume%TYPE);
END ex13;
/
```

```
CREATE OR REPLACE PACKAGE BODY ex13
```

```
AS
```

```
-- pentru o academie al carei nume este dat pentru fiecare categorie de varsta afisati pozitiile pentru care se antreneaza cu ce antrenor,
```

```
-- unde se antreneaza si elevii care fac parte din categoria de varsta
```

```
PROCEDURE ex6 (numeAcademie Academie.nume%TYPE)
```

```
AS
```

```
TYPE tablouIndexat IS TABLE OF POZITIE%ROWTYPE INDEX BY
PLS_INTEGER;
```

```
pozitii tablouIndexat;
```

```
TYPE tablouImbricat IS TABLE OF CATEGORIEVARSTA%ROWTYPE;
```

```
categoriivarsta tablouImbricat := tablouImbricat();
```

```
TYPE output IS TABLE OF VARCHAR(200) INDEX BY PLS_INTEGER;
```

```
sportivi output;
```

```
desfAntrenament output;
```

```
BEGIN
```

```
SELECT *
```

```
BULK COLLECT INTO pozitii
```

FROM pozitie;

```
-- IF pozitii.count = 0 THEN
--   DBMS_OUTPUT.PUT_LINE('Nu exista pozitii');
-- END IF;

--
-- IF pozitii.count > 0 THEN
--   FOR i IN pozitii.first..pozitii.last LOOP
--     DBMS_OUTPUT.PUT_LINE(pozitii(i).nume);
--   END LOOP;
-- END IF;
```

```
SELECT catv.id, catv.denumire, catv.id_sport
BULK COLLECT INTO categoriivarsta
FROM CATEGORIEVARSTA catv, SPORT s, ACADEMIE a
WHERE catv.id_sport = s.id and s.id_academie = a.id
AND UPPER(a.nume) LIKE UPPER(numeAcademie);
```

```
-- IF categoriivarsta.count = 0 THEN
--   DBMS_OUTPUT.PUT_LINE('Nu exista categorii de varsta');
-- END IF;

--
-- IF categoriivarsta.count > 0 THEN
--   FOR i IN categoriivarsta.first..categoriivarsta.last LOOP
--     DBMS_OUTPUT.PUT_LINE(categoriivarsta(i).denumire);
--   END LOOP;
-- END IF;
```

```
FOR i IN categoriivarsta.first..categoriivarsta.last LOOP
  DBMS_OUTPUT.PUT_LINE(categoriivarsta(i).denumire || ' ani');
```

```
DBMS_OUTPUT.PUT_LINE(");

FOR j IN pozitii.first..pozitii.last LOOP
    DBMS_OUTPUT.PUT_LINE(pozitii(j).nume || ':' );
    SELECT t.denumire || ' - ' || ant.nume || ' ' || ant.prenume
    BULK COLLECT INTO desfAntrenament
    FROM ANTRENAMENT a, TEREN t, ANTRENOR ant
    WHERE a.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(j).id AND
    a.id_teren = t.id AND a.id_antrenor = ant.id;

    IF desfAntrenament.count > 0 THEN
        SELECT s.nume || ' ' || s.prenume
        BULK COLLECT INTO sportivi
        FROM SPORTIV s, INSCRIERE insc, ANTRENAMENT a
        WHERE s.id = insc.id_sportiv AND insc.id_catvarsta = categoriivarsta(i).id
        AND a.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(j).id;

        FOR k in desfAntrenament.first..desfAntrenament.last LOOP
            DBMS_OUTPUT.PUT_LINE(' - ' || desfAntrenament(k));
            IF sportivi.count > 0 THEN
                FOR l IN sportivi.first..sportivi.last LOOP
                    DBMS_OUTPUT.PUT_LINE(sportivi(l));
                END LOOP;
            ELSE DBMS_OUTPUT.PUT_LINE('Nu exista sportivi inregistrati pentru
            acest antrenament');
            END IF;
            END LOOP;

            DBMS_OUTPUT.PUT_LINE('-----');
        END IF;
```

```
IF desfAntrenament.count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista');
    DBMS_OUTPUT.PUT_LINE('-----');
    END IF;

DBMS_OUTPUT.PUT_LINE(");

END LOOP;
END LOOP;

END ex6;
```

-- pentru un sportiv al carui nume este dat afisati pozitiile pentru care s-ar incadra in functie de notele obtinute pentru aptitudinile sale

```
PROCEDURE ex7 (nume SPORTIV.nume%TYPE, prenume SPORTIV.prenume%TYPE)
AS
CURSOR note (idSportiv SPORTIV.id%TYPE) IS
    SELECT n.id_aptitudine
    FROM NOTA n
    WHERE n.id_sportiv = idSportiv AND n.nota >=8;

CURSOR sportivi IS
    SELECT id, nume, prenume
    FROM SPORTIV;

CURSOR pozitii(idAptitudine APTITUDINE.id%TYPE) IS
```

```
SELECT p.nume as output
FROM APTITUDINE a, POZITIE p
WHERE a.id = idAptitudine AND a.id_pozitie = p.id;

idAptitudine APTITUDINE.id%TYPE;

BEGIN

FOR sportiv in sportivi LOOP
    IF UPPER(sportiv.nume) LIKE UPPER(nume) AND UPPER(sportiv.prenume) LIKE
UPPER(prenume) THEN
        DBMS_OUTPUT.PUT_LINE('Nume: ' || UPPER(nume) || ' ' || UPPER(prenume));

        OPEN note(sportiv.id);
        DBMS_OUTPUT.PUT_LINE('Pentru aptitudinile sale, pozitiile sugerate sunt
urmatoarele: ');
        LOOP
            FETCH note INTO idAptitudine;
            EXIT WHEN note%NOTFOUND;

        FOR pozitie IN pozitii(idAptitudine) LOOP
            DBMS_OUTPUT.PUT_LINE(' - ' || pozitie.output);
        END LOOP;
        CLOSE note;

    END IF;
END LOOP;

END ex7;
```

-- pentru un antrenor al carui id este dat si o categorie de varsta a carei nume este dat,
returnati numarul de pozitii pentru care au loc antrenamentele

```
FUNCTION ex8(numeCatVarsta CATEGORIEVARSTA.denumire%TYPE, idAntrenor
ANTRENOR.id%TYPE) RETURN NUMBER

IS
    nrPozitii NUMBER(3);

    IS_EMPTY EXCEPTION;
    INVALID_ID EXCEPTION;

BEGIN
    IF numeCatVarsta IS NULL THEN
        RAISE IS_EMPTY;
    END IF;

    IF idAntrenor <= 0 THEN
        RAISE INVALID_ID;
    END IF;

    SELECT COUNT(p.id)
    INTO nrPozitii
    FROM POZITIE p
    JOIN ANTRENAMENT a ON (p.id = a.id_pozitie)
    JOIN CATEGORIEVARSTA c ON(a.id_catvarsta = c.id)
    WHERE UPPER(c.denumire) LIKE UPPER(numeCatVarsta) AND a.id_antrenor =
idAntrenor;

    IF nrPozitii <= 0 THEN
```

```
RAISE NO_DATA_FOUND;  
ELSE RETURN nrPozitii;  
END IF;  
  
EXCEPTION  
    WHEN IS_EMPTY THEN  
        DBMS_OUTPUT.PUT_LINE('Campul pentru numele categoriei de varsta nu poate fi gol!');  
        RETURN -1;  
    WHEN INVALID_ID THEN  
        DBMS_OUTPUT.PUT_LINE('ID Invalid!');  
        RETURN -1;  
    WHEN NO_DATA_FOUND THEN  
        DBMS_OUTPUT.PUT_LINE('Nu au fost gasite categorii de varsta!');  
        RETURN -1;  
END ex8;
```

-- pentru un sportiv al carui nume este dat specificati ce aptitudine principala incerca sa isi
-- imbunatareasca in cadrul unui antrenament doar daca nota sa la acea aptitudine este mai
mica decat 8

-- exceptii
-- Campul pentru nume este gol
-- NO_DATA_FOUND - nu exista sportivi cu acest nume sau nu exista aptitudini care sa
indeplineasca conditiile
-- TOO_MANY_ROWS - exista mai multi sportivi cu acelasi nume

```
PROCEDURE ex9(numeSportiv SPORTIV.nume%TYPE)  
AS
```

```
TYPE tablouAptitudini IS TABLE OF APTITUDINE.denumire%TYPE INDEX BY
PLS_INTEGER;
aptitudini tablouAptitudini;

sportivi SPORTIV%ROWTYPE;

IS_EMPTY EXCEPTION;

BEGIN
IF numeSportiv IS NULL THEN
    RAISE IS_EMPTY;
END IF;

SELECT *
INTO sportivi
FROM SPORTIV s
WHERE UPPER(s.nume) LIKE UPPER(numeSportiv);

SELECT apt.denumire
BULK COLLECT INTO aptitudini
FROM APTITUDINE apt
JOIN POZITIE p ON (p.id = apt.id_pozitie)
JOIN ANTRENAMENT a ON(a.id_pozitie = p.id)
JOIN INSCRIERE ins ON (ins.id_catvarsta = a.id_catvarsta)
JOIN SPORTIV s ON (s.id = ins.id_sportiv)
JOIN NOTA n ON (n.id_sportiv = s.id AND n.id_aptitudine = apt.id)
WHERE UPPER(s.nume) LIKE UPPER(numeSportiv) AND n.nota <= 8;
IF SQL%NOTFOUND THEN
    RAISE NO_DATA_FOUND;
END IF;
```

```
DBMS_OUTPUT.PUT_LINE('Sportivul ' || UPPER(numeSportiv) || ' inca lucreaza la
urmatoarele aptitudini: ');
```

```
FOR i IN aptitudini.first..aptitudini.last LOOP
    DBMS_OUTPUT.PUT_LINE(' - ' || aptitudini(i));
END LOOP;
```

```
EXCEPTION
```

```
WHEN IS_EMPTY THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Campul pentru nume nu poate fi gol!');
```

```
WHEN NO_DATA_FOUND THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Sportivul cu acest nume nu exista sau nu au fost
gasite aptitudini care sa indeplineasca conditiile!');
```

```
WHEN TOO_MANY_ROWS THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Există mai mulți sportivi cu același nume!');
```

```
WHEN OTHERS THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Codul de eroare: ' || SQLCODE);
```

```
    DBMS_OUTPUT.PUT_LINE('Mesajul erorii: ' || SQLERRM);
```

```
END ex9;
```

```
END ex13;
```

```
/
```

```
EXECUTE ex13.ex6('academia hagi');
```

```
EXECUTE ex13.ex7('marian', 'alex');
```

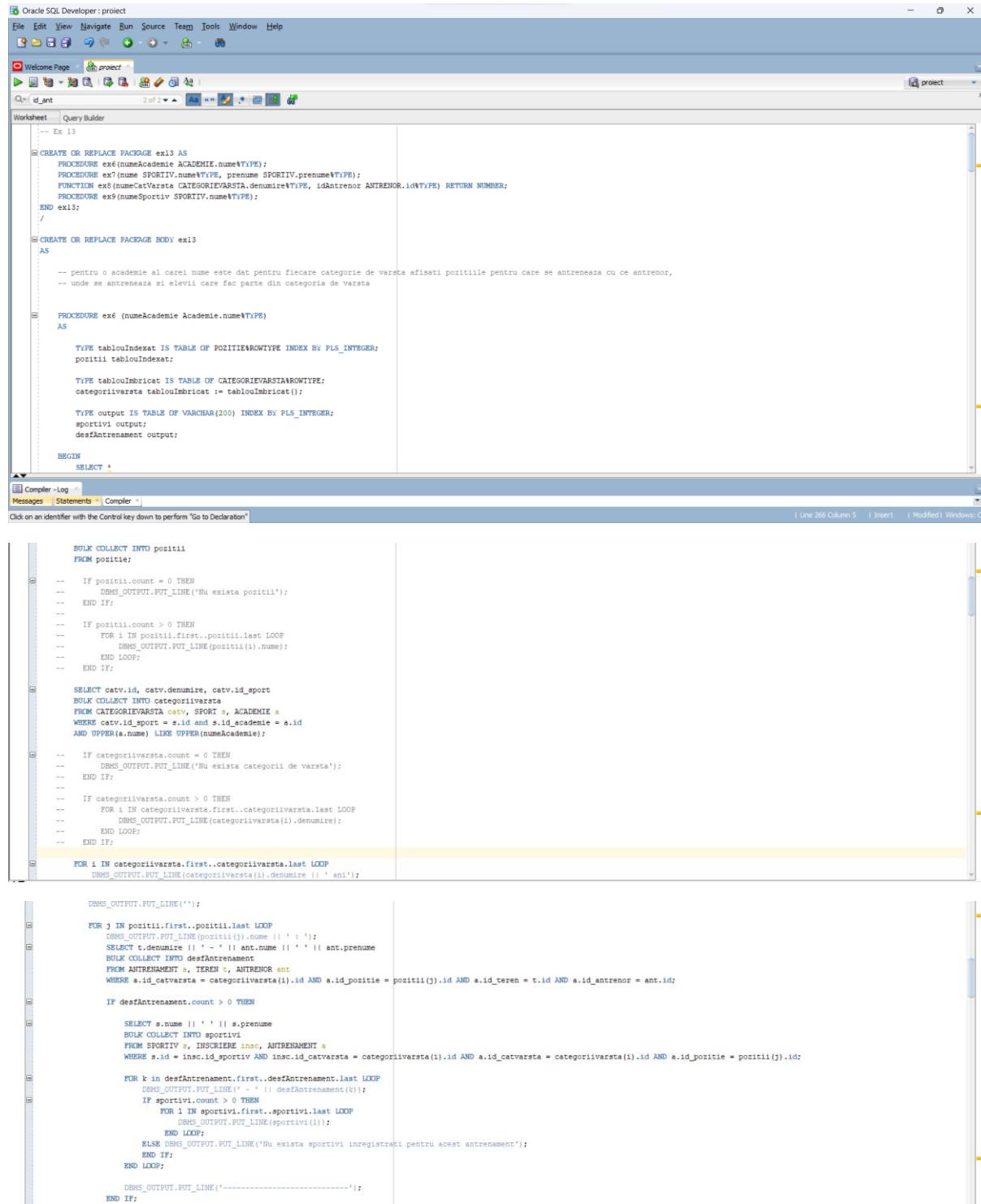
```
BEGIN
```

```
    DBMS_OUTPUT.PUT_LINE(ex13.ex8('17-18', 1));
```

```
END;
```

/

EXECUTE ex13.ex9('marian');



```

CREATE OR REPLACE PACKAGE ex13 AS
    PROCEDURE ex5(numeAcademie ACADEMIE.nume%TYPE);
    PROCEDURE ex7(nume SPORTIV.nume%TYPE, prenume SPORTIV.prenume%TYPE);
    FUNCTION ex9(catVarsta CATEGORIEVARSTA.denumire%TYPE, idAntrenor ANTRENOR.id%TYPE) RETURN NUMBER;
    PROCEDURE ex10(numeSportiv SPORTIV.nume%TYPE);
    END ex13;
    /
CREATE OR REPLACE PACKAGE BODY ex13 AS
    -- pentru o academie al carei nume este dat pentru fiecare categorie de varsta afisati pozitiile pentru care se antrenaza cu ce antrenor,
    -- unde se antrenaza si elevii care fac parte din categoria de varsta
    PROCEDURE ex6 (numeAcademie Academie.nume%TYPE)
    AS
        TYPE tablouIndexat IS TABLE OF POZITIE%ROWTYPE INDEX BY PLS_INTEGER;
        pozitii tablouIndexat;
        TYPE tablouImbricat IS TABLE OF CATEGORIEVARSTA%ROWTYPE;
        categoriivarsta tablouImbricat := tablouImbricat();
        TYPE output IS TABLE OF VARCHAR(200) INDEX BY PLS_INTEGER;
        sportivi output;
        desfActrenament output;
    BEGIN
        SELECT *
        BULK COLLECT INTO pozitii
        FROM pozitii;
        IF pozitii.count = 0 THEN
            DBMS_OUTPUT.PUT_LINE('Nu exista pozitii');
        END IF;
        IF pozitii.count > 0 THEN
            FOR i IN pozitii.first..pozitii.last LOOP
                DBMS_OUTPUT.PUT_LINE(pozitii(i).nume);
            END LOOP;
        END IF;
        SELECT catv.id, catv.denumire, catv.id_sport
        BULK COLLECT INTO categoriivarsta
        FROM CATEGORIEVARSTA catv, SPORT $, ACADEMIE a
        WHERE catv.id_sport = s.id AND s.id_academie = a.id
        AND UPPER(a.nume) LIKE UPPER(numAcademie);
        IF categoriivarsta.count = 0 THEN
            DBMS_OUTPUT.PUT_LINE('Nu exista categorii de varsta');
        END IF;
        IF categoriivarsta.count > 0 THEN
            FOR i IN categoriivarsta.first..categoriivarsta.last LOOP
                DBMS_OUTPUT.PUT_LINE(categoriivarsta(i).denumire);
            END LOOP;
        END IF;
        FOR i IN categoriivarsta.first..categoriivarsta.last LOOP
            DBMS_OUTPUT.PUT_LINE(categoriivarsta(i).denumire || ' ani');
        END LOOP;
        DBMS_OUTPUT.PUT_LINE('');
        FOR j IN pozitii.first..pozitii.last LOOP
            DBMS_OUTPUT.PUT_LINE(pozitii(j).nume || ' ');
            SELECT t.denumire || '-' || ant.nume || ' ' || ant.prenume
            BULK COLLECT INTO desfAntrenament
            FROM ANTRENAMENT $, TEREN $, ANTRENOR ant
            WHERE s.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(j).id AND a.id_teren = t.id AND a.id_antrenor = ant.id;
            IF desfAntrenament.count > 0 THEN
                SELECT t.nume || ' ' || s.prenume
                BULK COLLECT INTO sportivi
                FROM SPORTIV $, INSCRIERE insc, ANTRENAMENT a
                WHERE s.id = insc.id_sportiv AND insc.id_catvarsta = categoriivarsta(i).id AND a.id_catvarsta = categoriivarsta(i).id AND a.id_pozitie = pozitii(j).id;
                FOR k IN desfAntrenament.first..desfAntrenament.last LOOP
                    DBMS_OUTPUT.PUT_LINE(' - ' || desfAntrenament(k));
                    IF sportivi.count > 0 THEN
                        FOR l IN sportivi.first..sportivi.last LOOP
                            DBMS_OUTPUT.PUT_LINE(sportivi(l));
                        END LOOP;
                    ELSE DBMS_OUTPUT.PUT_LINE('Nu exista sportivi inregistrati pentru acest antrenament');
                    END IF;
                END LOOP;
            DBMS_OUTPUT.PUT_LINE('-----');
        END IF;
    END;
END ex13;
/

```

Sisteme de gestiune a bazelor de date

Petre Vasile-Eduard

Grupa 243

```
IF desfAntrenament.count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista');
    DBMS_OUTPUT.PUT_LINE('-----');
END IF;

DBMS_OUTPUT.PUT_LINE('');

END LOOP;
END LOOP;

END ex6;

-- pentru un sportiv al carui nume este dat afisati pozitiile pentru care s-ar incadra in functie de notele obtinute pentru aptitudinile sale

PROCEDURE ex7 (nume SPORTIV.nume%TYPE, prenume SPORTIV.prenume%TYPE)
AS
    CURSOR note (idSportiv SPORTIV.id%TYPE) IS
        SELECT n.id_aptitudine
        FROM NOTA n
        WHERE n.id_sportiv = idSportiv AND n.nota >=8;

    CURSOR sportivi IS
        SELECT id, nume, prenume
        FROM SPORTIV;

    CURSOR pozitii(idAptitudine APITITUDINE.id%TYPE) IS
        SELECT p.nume as output
        FROM APITITUDINE a, POZITIE p
        WHERE a.id = idAptitudine AND a.id_pozitie = p.id;

    idAptitudine APITITUDINE.id%TYPE;
BEGIN

FOR sportiv IN sportivi LOOP
    IF UPPER(sportiv.nume) LIKE UPPER(nume) AND UPPER(sportiv.prenume) LIKE UPPER(prenume) THEN
        DBMS_OUTPUT.PUT_LINE('Pentru aptitudinile sale, pozitiile superate sunt urmatoarele: ');
        OPEN note (sportiv.id);
        DBMS_OUTPUT.PUT_LINE('Fentru aptitudinile sale, pozitiile superate sunt urmatoarele: ');
        LOOP
            FETCH note INTO idAptitudine;
            EXIT WHEN note%NOTFOUND;

            FOR pozitie IN pozitii(idAptitudine) LOOP
                DBMS_OUTPUT.PUT_LINE(' - ' || pozitie.output);
            END LOOP;
        END LOOP;
        CLOSE note;
    END IF;
END LOOP;

END ex7;

FUNCTION ex8(nameCatVarsta CATEGORIEVARSTA.denumire%TYPE, idAntrenor ANTRENOR.id%TYPE) RETURN NUMBER
IS
    nrPozitii NUMBER(3);

    IS_EMPTY EXCEPTION;
    INVALID_ID EXCEPTION;

BEGIN
    IF nameCatVarsta IS NULL THEN
        RAISE IS_EMPTY;
    END IF;

    IF idAntrenor <= 0 THEN
        RAISE INVALID_ID;
    END IF;

    SELECT COUNT(p.id)
    INTO nrPozitii
    FROM POZITIE p
    JOIN ANTRERENTAMENT a ON (p.id = a.id_pozitie)
    JOIN CATEGORIEVARSTA c ON (a.id_catvarsta = c.id)
    WHERE UPPER(c.denumire) LIKE UPPER(nameCatVarsta) AND a.id_antrenor = idAntrenor;

    IF nrPozitii <= 0 THEN
        RAISE NO_DATA_FOUND;
    ELSE
        RETURN nrPozitii;
    END IF;

    EXCEPTION
        WHEN IS_EMPTY THEN
            DBMS_OUTPUT.PUT_LINE('Campul pentru numele categoriei de varsta nu poate fi gol!');
        WHEN INVALID_ID THEN
            DBMS_OUTPUT.PUT_LINE('ID Invalid!');
            RETURN -1;
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Nu au fost gasite categorii de varsta!');
            RETURN -1;
    END ex8;

-- pentru un sportiv al carui nume este dat specificati ce aptitudine principala incerca sa isi
-- imbunatatescă în cadrul unui antrenament doar daca nota sa la aceea aptitudine este mai mica decat 8

-- exceptie
-- Campul pentru nume este gol
-- NO_DATA_FOUND - nu exista sportivi cu acest nume sau nu exista aptitudini care sa indeplineasca conditiile
-- TOO_MANY_ROWS - exista mai multi sportivi cu acelasi nume

PROCEDURE ex9(numeSportiv SPORTIV.nume%TYPE)
AS
    TYPE tablouAptitudini IS TABLE OF APITITUDINE.denumire%TYPE INDEX BY PLS_INTEGER;
    aptitudini tablouAptitudini;

    TYPE tablouSportivi IS TABLE OF SPORTIV%ROWTYPE INDEX BY PLS_INTEGER;
    sportivi tablouSportivi;

    IS_EMPTY EXCEPTION;
```

Sisteme de gestiune a bazelor de date
 Petre Vasile-Eduard
 Grupa 243

```

BEGIN
    IF numeSportiv IS NULL THEN
        RAISE IS_EMPTY;
    END IF;

    SELECT *
    BULK COLLECT INTO sportivi
    FROM SPORTIV s
    WHERE UPPER(s.name) LIKE UPPER(numeSportiv);
    IF SQLNOTFOUND THEN
        RAISE NO_DATA_FOUND;
    END IF;

    IF sportivi.count > 1 THEN
        RAISE TOO_MANY_ROWS;
    END IF;

    SELECT apt.denumire
    BULK COLLECT INTO aptitudini
    FROM APITUDINI apt
    JOIN POSITIE p ON (p.id = apt.id_positie)
    JOIN SPOTIFI s ON (s.id_spotif = p.id)
    JOIN INSCRIERE ins ON (ins.id_cetvarsta = s.id)
    JOIN SPORTIV s ON (s.id = ins.id_sportiv)
    JOIN NOTA n ON (n.id_sportiv = s.id AND n.id_apitudine = apt.id)
    WHERE UPPER(s.name) LIKE UPPER(numeSportiv) AND n.nota <= 8;
    IF SQLNOTFOUND THEN
        RAISE NO_DATA_FOUND;
    END IF;
END;
    
```



```

DBMS_OUTPUT.PUT_LINE('Sportivul ' || UPPER(numeSportiv) || ' inca lucraea la urmatoarele apititudini:');
FOR i IN aptitudini.first..aptitudini.last LOOP
    DBMS_OUTPUT.PUT_LINE(' - ' || aptitudini(i));
END LOOP;

EXCEPTION
    WHEN IS_EMPTY THEN
        DBMS_OUTPUT.PUT_LINE('Campul pentru nume nu poate fi gol!');
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Sportivul cu acest nume nu exista sau nu au fost gasite apititudini care sa indeplineasca conditiile!');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Există mai multi sportivi cu același nume!');
END ex9;
END ex13;
/
EXECUTE ex13.ex6('academia hagi');
EXECUTE ex13.ex7('marian', 'alex');
BEGIN
    DBMS_OUTPUT.PUT_LINE(ex13.ex8('17-18', 1));
END;
/
EXECUTE ex13.ex9('marian');

PL/SQL procedure successfully completed.

17-18 ani
CF :
- Teren Fotbal 1 - Gheorghe Hagi
Marian Alex
Florian Petrescu
Florian Leonard
Coman Razvan
Lazar Narcis
-----
LM/RW :
Nu exista
-----

CM :
- Teren Fotbal 1 - Gheorghe Hagi
Marian Alex
Florian Petrescu
Florian Leonard
Coman Razvan
Lazar Narcis
-----
CB :
- Teren Fotbal 2 - Razvan Stan
Marian Alex
Florian Petrescu
-----
Florian Leonard
Coman Razvan
Lazar Narcis
-----
GK :
- Teren Fotbal 3 - Marius Popescu
Marian Alex
Florian Petrescu
Florian Leonard
Coman Razvan
Lazar Narcis
-----
15-16 ani
CF :
- Teren Fotbal 1 - Gheorghe Hagi
Nu exista sportivi inregistrati pentru acest antrenament
-----
LM/RW :
Nu exista
-----
CM :
- Teren Fotbal 1 - Gheorghe Hagi
Nu exista sportivi inregistrati pentru acest antrenament
-----
```

Sisteme de gestiune a bazelor de date
Petre Vasile-Eduard
Grupa 243

```
CB :  
Nu exista  
-----  
  
GK :  
Nu exista  
-----  
  
13-14 ani  
  
CF :  
- Teren Fotbal 4 - Florin Bogdan  
Nu exista sportivi inregistrati pentru acest antrenament  
-----  
  
LM/RW :  
Nu exista  
-----  
  
CM :  
- Teren Fotbal 5 - Bares Popa  
Nu exista sportivi inregistrati pentru acest antrenament  
-----  
  
CB :  
Nu exista  
-----  
  
GK :  
Nu exista
```

```
11-12 ani  
  
CF :  
Nu exista  
-----  
  
LM/RW :  
Nu exista  
-----  
  
CM :  
- Teren Fotbal 4 - Florin Bogdan  
Nu exista sportivi inregistrati pentru acest antrenament  
-----  
  
CB :  
Nu exista  
-----  
  
GK :  
Nu exista  
-----  
  
9-10 ani  
  
CF :  
Nu exista  
-----
```

```
LM/RW :  
Nu exista  
-----  
  
CM :  
- Teren Fotbal 4 - Florin Bogdan  
Nu exista sportivi inregistrati pentru acest antrenament  
-----  
  
CB :  
Nu exista  
-----  
  
GK :  
Nu exista  
-----
```

```
PL/SQL procedure successfully completed.  
  
Name: MARIAN ALEX  
Pentru aptitudinile sale, pozitiile sugerate sunt urmatoarele:  
- CF  
- LM/RW  
- CM
```

```
PL/SQL procedure successfully completed.
```

```
2  
  
PL/SQL procedure successfully completed.
```

```
Sportivul MARIAN incă lucrează la urmatoarele aptitudini:  
- Fara  
- Defensiva  
- Aparare
```

```
PL/SQL procedure successfully completed.
```