

## PROIECT SGBD

### -CONCURS CANIN-

Pătrânjel David-George, FMI, grupa 251

#### 1. Prezențați pe scurt baza de date.

La concursul canin „Cățelul Anului 2023”, organizatorii își doresc gestiunea eficientă a resurselor necesare desfășurării competițiilor din cadrul concursului. La acesta se pot înscrie în competiție mai mulți câței, fiecare având o rasă, un nume, o vârstă, o talie (mică, mijlocie sau mare), o greutate și starea de vaccinare. Un câine este înscris de un proprietar, acesta putând să înscrie mai mulți câței.

Proprietarii sunt fie participanți experimanțați, pentru care se cunoaște rangul ocupat în clasamentul internațional, fie participanți noi, care pot concura doar la acest concurs sau care își doresc să concureze și la următoarele concursuri canine. Pentru fiecare participant se cunoaște numele, prenumele și vârsta. De asemenea, pentru participanții experimanțați se cunoaște istoricul participărilor acestora la alte concursuri, cât și eventualele premii obținute.

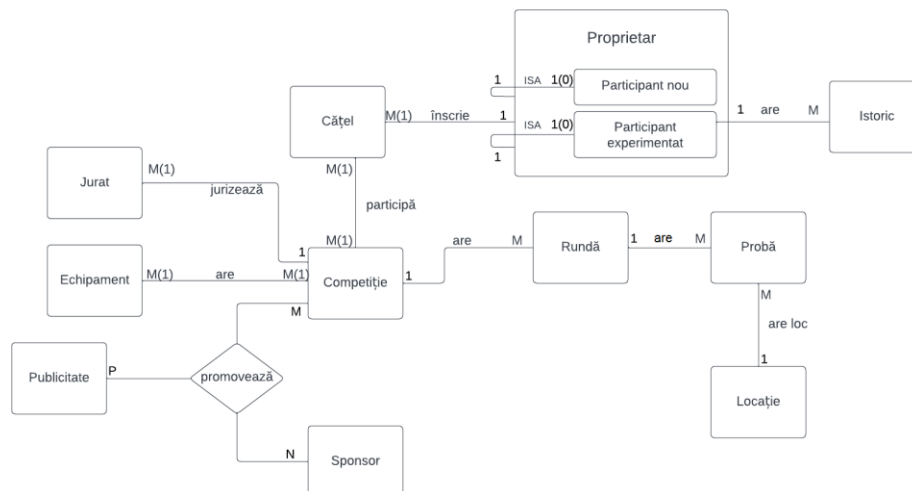
Concursul este compus din mai multe competiții la care câțelei pot concura, iar competițiile sunt compuse din mai multe runde, fiecare cu mai multe probe. Toate probele dintr-o rundă se desfășoară într-o singură zi, iar în funcție de tipul probei (acvatic, aspect, viteză, artistic, traseu) se alege o locație. La fiecare competiție jurizează mai mulți jurați, iar la fiecare competiție se vor folosi mai multe echipamente, ce pot fi refolosite și la alte competiții.

Fiecare competiție poate fi sponsorizată de firme care primesc promovare prin diverse materiale: afișe, banner, postări, etc. De asemenea, pe un material publicitar se pot regăsi mai multe firme, iar aceste materiale pot fi folosite la mai multe competiții.

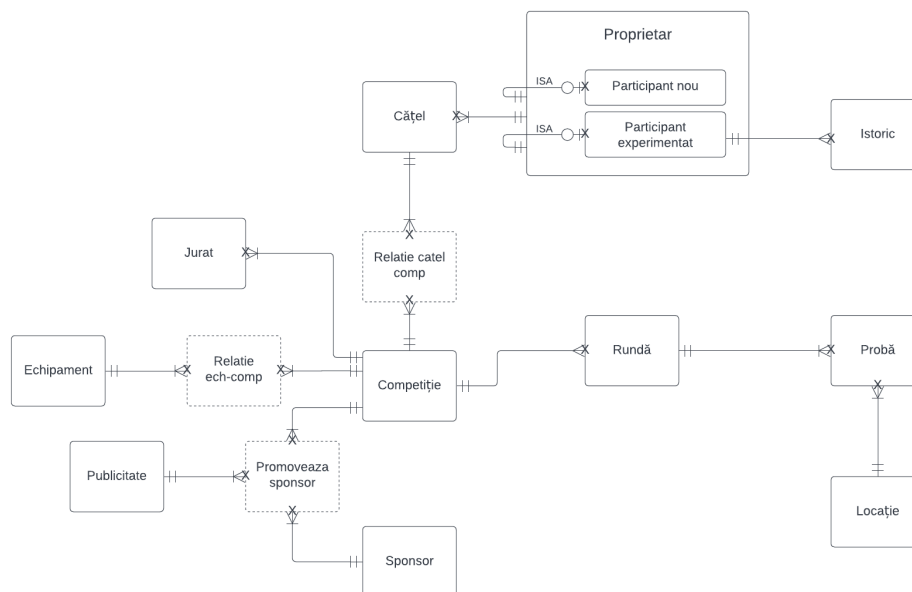
Constrângerile impuse asupra modelului sunt:

- Starea de vaccinare a unui câțel nu este obligatorie, iar la câțelei cu mai multe rase se va nota Metis
- Un proprietar poate fi fie un participant nou, fie un participant experimentat.
- O competiție poate fi sponsorizată de mai mulți sponsori, iar un sponsor poate sponsoriza una sau mai multe competiții.
- Tipul probei poate fi: acvatic, aspect, viteză, artistic, traseu
- Talia poate fi: mică, medie, mare
- Experianța juraților este un număr măsurat în ani
- Durata probelor este un număr măsurat în ore
- Atributele vaccinat și competitie\_unica pot lua doar valoarea T sau F.
- Proprietarii trebuie să aibă peste 18 ani, iar câțelei peste 2 ani.
- Toate premiile sunt de o valoare cuprinsă între 10.000 RON și 1.000.000 RON.
- Premiul obținut în istoricul unui participant experimantat poate să fie 1, 2, 3 sau null.

## 2. Realizați diagrama entitate-relație.



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



Atributele entităților prin intermediul schemelor relaționale:

- CĂȚEL(id\_catel (PK), id\_participant (FK), rasa, nume\_catel, varsta\_catel, talie, greutate, vaccinat)
- PROPRIETAR(id\_proprietar (PK), nume\_participant, prenume\_participant, data\_nasterii, data\_inreg, nr\_telefon)
- PARTICIPANT\_NOU(id\_participant (PK, FK), competitie\_unica)
- PARTICIPANT\_EXPERIMENTAT(id\_participant (PK, FK), rang)
- ISTORIC(id\_participare (PK), id\_participant\_exp (FK), data\_competitie, nume\_competitie, premiu)
- RELATIE\_CATEL\_COMP(id\_relatie\_cc (PK), id\_catel (FK), id\_competitie (FK))
- COMPETIȚIE(id\_competitie (PK), nume\_competitie, nr\_zile\_competitie, premiu\_competitie)
- ECHIPAMENT(id\_echipament (PK), nume\_echipament, material, greutate\_echipament)
- RELATIE\_ECH\_COMP(id\_relatie\_ec (PK), id\_echipament (FK), id\_competitie (FK))
- JURAT(id\_jurat (PK), id\_competitie (FK), nume\_jurat, prenume\_jurat, experienta\_jurat)
- RUNDĂ(id\_runda (PK), id\_competitie (FK), nume\_runda, data\_runda)
- PROBA ((id\_runda (FK), id\_proba) (PK), tip\_proba (FK), durata\_proba)
- LOCATIE(tip\_proba (PK), oras, strada)
- SPONSOR(id\_sponsor (PK), nume\_firma, oras\_firma, nume\_manager, prenume\_manager)
- PUBLICITATE(id\_publicitate (PK), tip\_publicitate, nr\_exemplare)
- PROMOVEAZĂ\_SPONSOR(id\_relatie\_pcs (PK), id\_publicitate (FK), id\_competitie (FK), id\_sponsor (FK))

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

ENTITATE: Produs

```
create table publicitate
(
    id_publicitate number(5) not null
        constraint PUBLICITATE_PK
            primary key,
    tip_publicitate varchar2(30) not null,
    nr_exemplare number(3) default 20 not null
);
```

ENTITATE: Locatie

```
create table locatie
(
    tip_proba varchar2(10) not null
        constraint LOCATIE_PK
            primary key,
    oras varchar2(20) not null,
    strada varchar2(20)
);
```

ENTITATE: Proprietar

```
create table proprietar
(
    id_proprietar number(5) not null
```

```
        constraint PROPRIETAR_PK
            primary key,
    nume_participant varchar2(30) not null,
    prenume_participant varchar2(40) not null,
    data_nasterii date not null,
    data_inreg date default sysdate,
    nr_telefon varchar2(13) not null
);

ALTER TABLE proprietar
ADD(
    CHECK((data_inreg - data_nasterii) / 365 >= 18)
);
```

#### ENTITATE: Participant nou

```
create table participant_nou
(
    id_participant number(5) not null
        constraint PARTICIPANT_NOU_PK
            primary key,
    competitie_unica varchar2(1) not null
);

ALTER TABLE participant_nou
ADD( CONSTRAINT part_nou_part_fk
        FOREIGN KEY (id_participant)
        REFERENCES PROPRIETAR(id_proprietar)
        ON DELETE CASCADE
);

ALTER TABLE participant_nou
ADD(
    CHECK ( competitie_unica = 'F' or competitie_unica = 'T')
);
```

#### ENTITATE: Participant experimentat

```
create table participant_experimentat
(
    id_participant number(5) not null
        constraint PARTICIPANT_EXPERIMENTAT_PK
            primary key,
    rang number(1) not null
);

ALTER TABLE participant_experimentat
ADD(
    CHECK ( rang > 0 and rang < 6)
);

ALTER TABLE participant_experimentat
ADD( CONSTRAINT part_exp_part_fk
        FOREIGN KEY (id_participant)
        REFERENCES PROPRIETAR(id_proprietar)
        ON DELETE CASCADE
);
```

#### ENTITATE: Catel

```
create table catel
(
    id_catel number(5) not null
```

```
        constraint CATEL_PK
            primary key,
id_participant number(5)    not null,
rasa            varchar2(50) not null,
nume_catel     varchar2(20),
varsta_catel   number(2),
talie          varchar2(5)  not null,
greutate      number(5, 2) not null,
vaccinat       varchar2(1)
);
ALTER TABLE catel
ADD(
    CHECK ( talie = 'mica' or talie = 'medie' or talie = 'mare')
);
ALTER TABLE catel
ADD(
    CHECK ( varsta_catel >= 2)
);
ALTER TABLE catel
ADD(
    CHECK ( vaccinat = 'F' or vaccinat = 'T')
);
ALTER TABLE catel
ADD ( CONSTRAINT part_catel_fk
        FOREIGN KEY (id_participant)
        REFERENCES PROPRIETAR(id_proprietar)
        ON DELETE CASCADE
    ) ;
```

### ENTITATE: Competiție

```
create table competitie
(
    id_competitie    number(5)    not null
        constraint COMPETITIE_PK
            primary key,
    nume_competitie  varchar2(60) not null,
    nr_zile_competitie number(3)  not null,
    premiu_competitie number(10)  not null
);

alter table competitie
add(
    check( premiu_competitie >= 10000 and premiu_competitie <= 1000000)
);
```

### ENTITATE: Runda

```
create table runda
(
    id_runda        number(5)    not null
        constraint RUNDA_PK
            primary key,
    id_competitie    number(5)    not null,
    nume_runda       varchar2(30) not null,
    data_runda       date         not null
);

ALTER TABLE runda
ADD ( CONSTRAINT competitie_runda_fk
        FOREIGN KEY (id_competitie)
        REFERENCES competitie(id_competitie)
```

```
        ON DELETE CASCADE  
    ) ;
```

### ENTITATE: Proba

```
create table proba  
(  
    id_proba      number(5)    not null,  
    id_runda      number(5),  
    tip_proba     varchar2(10),  
    durata_proba  number(3)    not null  
);  
  
ALTER TABLE proba  
ADD ( CONSTRAINT proba_pk  
        PRIMARY KEY (id_proba, id_runda)  
    ) ;  
ALTER TABLE proba  
ADD ( CONSTRAINT runda_proba_fk  
        FOREIGN KEY (id_runda)  
        REFERENCES runda(id_runda)  
        ON DELETE CASCADE  
    ) ;  
ALTER TABLE proba  
ADD ( CONSTRAINT locatie_proba_fk  
        FOREIGN KEY (tip_proba)  
        REFERENCES locatie(tip_proba)  
        ON DELETE SET NULL  
    ) ;
```

### ENTITATE: Jurat

```
create table jurat  
(  
    id_jurat      number(5)    not null  
        constraint JURAT_PK  
        primary key,  
    id_competitie number(5),  
    nume_jurat    varchar2(30) not null,  
    prenume_jurat varchar2(40) not null,  
    experianta_jurat number(2)  
);  
  
ALTER TABLE jurat  
ADD ( CONSTRAINT jurat_competitie_fk  
        FOREIGN KEY (id_competitie)  
        REFERENCES competitie(id_competitie)  
        ON DELETE SET NULL  
    ) ;
```

### ENTITATE: Sponsor

```
create table sponsor  
(  
    id_sponsor    number(5)    not null  
        constraint SPONSOR_PK  
        primary key,  
    nume_firma    varchar2(50) not null,  
    oras_firma    varchar2(50) not null,  
    nume_manager  varchar2(30) not null,  
    prenume_manager varchar2(40) not null  
);
```

### ENTITATE: Relatie\_Catel\_Comp

```
create table relatie_catel_comp
(
    id_relatie_cc number(5) not null
        constraint RELATIE_CATEL_COMP_PK
            primary key,
    id_catel number(5) not null,
    id_competitie number(5) not null
);

ALTER TABLE relatie_catel_comp
ADD ( CONSTRAINT relatie_catel_comp1_fk
      FOREIGN KEY (id_competitie)
      REFERENCES competitie(id_competitie)
      ON DELETE CASCADE
    ) ;

ALTER TABLE relatie_catel_comp
ADD ( CONSTRAINT relatie_catel_comp2_fk
      FOREIGN KEY (id_catel)
      REFERENCES CATEL(id_catel)
      ON DELETE CASCADE
    ) ;
```

### ENTITATE: Echipament

```
create table echipament
(
    id_echipament number(5) not null
        constraint ECHIPAMENT_PK
            primary key,
    nume_echipament varchar2(30) not null,
    material varchar2(20),
    greutate_echipament number(4)
);
```

### ENTITATE: Relatie\_ech\_comp

```
create table relatie_ech_comp
(
    id_relatie_ec number(5) not null
        constraint RELATIE_ECH_COMP_PK
            primary key,
    id_echipament number(5) not null,
    id_competitie number(5) not null
);

ALTER TABLE relatie_ech_comp
ADD ( CONSTRAINT relatie_ech_comp1_fk
      FOREIGN KEY (id_echipament)
      REFERENCES echipament(id_echipament)
      ON DELETE CASCADE
    ) ;

ALTER TABLE relatie_ech_comp
ADD ( CONSTRAINT relatie_ech_comp2_fk
      FOREIGN KEY (id_competitie)
      REFERENCES competitie(id_competitie)
    ) ;
```

```
ON DELETE CASCADE  
)
```

#### ENTITATE: Promoveaza\_sponsor

```
create table promoveaza_sponsor  
(  
    id_relatie_pcs number(5) not null  
        constraint PROMOVEAZA_SPONSOR_PK  
        primary key,  
    id_publicitate number(5) not null,  
    id_competitie number(5) not null,  
    id_sponsor number(5) not null  
);  
  
ALTER TABLE promoveaza_sponsor  
ADD ( CONSTRAINT promoveaza_sponsor1_fk  
        FOREIGN KEY (id_competitie)  
        REFERENCES competitie(id_competitie)  
        ON DELETE CASCADE  
    );  
  
ALTER TABLE promoveaza_sponsor  
ADD ( CONSTRAINT promoveaza_sponsor2_fk  
        FOREIGN KEY (id_sponsor)  
        REFERENCES sponsor(id_sponsor)  
        ON DELETE CASCADE  
    );  
ALTER TABLE promoveaza_sponsor  
ADD ( CONSTRAINT promoveaza_sponsor3_fk  
        FOREIGN KEY (id_publicitate)  
        REFERENCES publicitate(id_publicitate)  
        ON DELETE CASCADE  
    );
```

#### ENTITATE: Istoric

```
create table istoric  
(  
    id_participare number(5) not null  
        constraint istoric_pk  
        primary key,  
    id_participant_exp number(5) not null,  
    data_competitie date not null,  
    nume_competitie varchar2(30) not null,  
    premiu number(1) not null  
);  
  
ALTER TABLE istoric  
ADD ( CONSTRAINT istoric_fk  
        FOREIGN KEY (id_participant_exp)  
        REFERENCES PARTICIPANT_EXPERIMENTAT(ID_PARTICIPANT)  
        ON DELETE CASCADE  
    );  
  
ALTER TABLE istoric  
ADD(  
    CHECK(premiu is null or (premiu >=1 and premiu <= 3))  
);
```



## 5. Adăugați informații coerente în tabelele create.

ENTITATE: Proprietar

```
INSERT INTO PROPRIETAR VALUES
(1001, 'Popescu', 'Mihai', TO_DATE('09-JUL-1984', 'dd-MON-yyyy'), DEFAULT, '0712121213');
INSERT INTO PROPRIETAR VALUES
(1002, 'Francis', 'Alexandru', TO_DATE('11-JUN-1976', 'dd-MON-yyyy'), DEFAULT, '0712121213');
INSERT INTO PROPRIETAR VALUES
(1003, 'Mihaita', 'Ana-Maria', TO_DATE('21-FEB-1989', 'dd-MON-yyyy'), DEFAULT, '0712121213');
INSERT INTO PROPRIETAR VALUES
(1004, 'Petrescu', 'Alexandra', TO_DATE('19-OCT-1990', 'dd-MON-yyyy'), DEFAULT, '0712121213');
INSERT INTO PROPRIETAR VALUES
(1005, 'Ivanovici', 'Mircea', TO_DATE('30-MAY-2003', 'dd-MON-yyyy'), DEFAULT, '0712121213');
INSERT INTO PROPRIETAR VALUES
(1006, 'Atanasiu', 'Elena', TO_DATE('21-JUN-2002', 'dd-MON-yyyy'), DEFAULT, '0712121213');
```

console [Concurs Canin SGBD]PROPRIETAR [Concurs Canin SGBD]

6 rows

SQL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

DDL

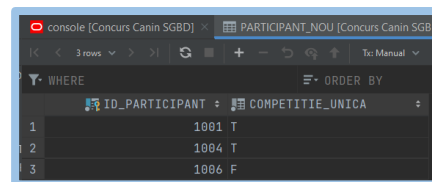
DDL

DDL

DDL

ENTITATE: Participant\_nou

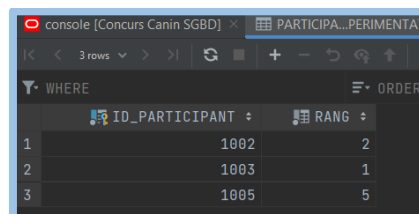
```
INSERT INTO PARTICIPANT_NOU VALUES
(1001, 'T');
INSERT INTO PARTICIPANT_NOU VALUES
(1004, 'T');
INSERT INTO PARTICIPANT_NOU VALUES
(1006, 'F');
```



ID_PARTICIPANT	COMPETITIE_UNICA
1	1001 T
2	1004 T
3	1006 F

ENTITATE: Participant\_experimental

```
INSERT INTO PARTICIPANT_EXPERIMENTAT VALUES
(1002, 2);
INSERT INTO PARTICIPANT_EXPERIMENTAT VALUES
(1003, 1);
INSERT INTO PARTICIPANT_EXPERIMENTAT VALUES
(1005, 5);
```



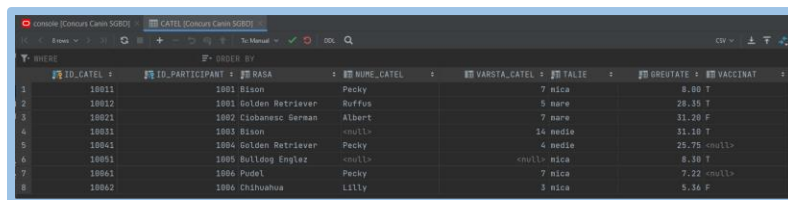
ID_PARTICIPANT	RANG
1	1002 2
2	1003 1
3	1005 5

ENTITATE: Catel

## SGBD – Grupa 251

### Pătrânjel David-George

```
INSERT INTO CATEL VALUES
(10011, 1001, 'Bison', 'Pecky', 7, 'mica', 8, 'T');
INSERT INTO CATEL VALUES
(10012, 1001, 'Golden Retriever', 'Ruffus', 5, 'mare', 28.35, 'T');
INSERT INTO CATEL VALUES
(10021, 1002, 'Ciobanesc German', 'Albert', 7, 'mare', 31.2, 'F');
INSERT INTO CATEL VALUES
(10031, 1003, 'Bison', NULL, 14, 'medie', 31.1, 'T');
INSERT INTO CATEL VALUES
(10041, 1004, 'Golden Retriever', 'Pecky', 4, 'medie', 25.75, NULL);
INSERT INTO CATEL VALUES
(10051, 1005, 'Bulldog Englez', NULL, NULL, 'mica', 8.3, 'T');
INSERT INTO CATEL VALUES
(10061, 1006, 'Pudel', 'Pecky', 7, 'mica', 7.22, NULL);
INSERT INTO CATEL VALUES
(10062, 1006, 'Chihuahua', 'Lilly', 3, 'mica', 5.36, 'F');
```

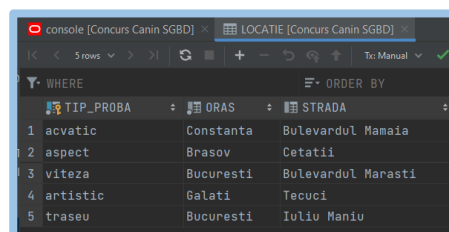


The screenshot shows a database console window titled 'CATEL [Concurs Canin SGBD]'. It displays a table with 8 rows and 7 columns. The columns are: ID\_CATEL, ID\_PARTICIPANT, RASA, NUME\_CATEL, VARSTA\_CATEL, TALIE, and GREUTATE. The data is as follows:

ID_CATEL	ID_PARTICIPANT	RASA	NUME_CATEL	VARSTA_CATEL	TALIE	GREUTATE
1	10011	1001 Bison	Pecky	7	nica	8.00 T
2	10012	1001 Golden Retriever	Ruffus	5	mare	28.35 T
3	10021	1002 Ciobanesc German	Albert	7	mare	31.20 F
4	10031	1003 Bison	NULL	14	medie	31.10 T
5	10041	1004 Golden Retriever	Pecky	4	medie	25.75 NULL
6	10051	1005 Bulldog Englez	NULL	NULL	nica	8.30 T
7	10061	1006 Pudel	Pecky	7	nica	7.22 NULL
8	10062	1006 Chihuahua	Lilly	3	nica	5.36 F

### ENTITATE: Locatie

```
INSERT INTO LOCATIE VALUES
('acvatic', 'Constanta', 'Bulevardul Mamaia');
INSERT INTO LOCATIE VALUES
('aspect', 'Brasov', 'Cetatii');
INSERT INTO LOCATIE VALUES
('viteza', 'Bucuresti', 'Bulevardul Marasti');
INSERT INTO LOCATIE VALUES
('artistic', 'Galati', 'Tecuci');
INSERT INTO LOCATIE VALUES
('traseu', 'Bucuresti', 'Iuliu Maniu');
```

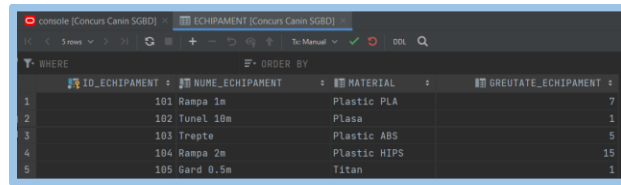


The screenshot shows a database console window titled 'LOCATIE [Concurs Canin SGBD]'. It displays a table with 5 rows and 3 columns. The columns are: TIP\_PROBA, ORAS, and STRADA. The data is as follows:

TIP_PROBA	ORAS	STRADA
1 acvatic	Constanta	Bulevardul Mamaia
2 aspect	Brasov	Cetatii
3 viteza	Bucuresti	Bulevardul Marasti
4 artistic	Galati	Tecuci
5 traseu	Bucuresti	Iuliu Maniu

### ENTITATE: Echipament

```
INSERT INTO ECHIPAMENT VALUES
(101, 'Rampa 1m', 'Plastic PLA', 7);
INSERT INTO ECHIPAMENT VALUES
(102, 'Tunel 10m', 'Plasa', 1);
INSERT INTO ECHIPAMENT VALUES
(103, 'Trepte', 'Plastic ABS', 5);
INSERT INTO ECHIPAMENT VALUES
(104, 'Rampa 2m', 'Plastic HIPS', 15);
INSERT INTO ECHIPAMENT VALUES
(105, 'Gard 0.5m', 'Titan', 1);
```

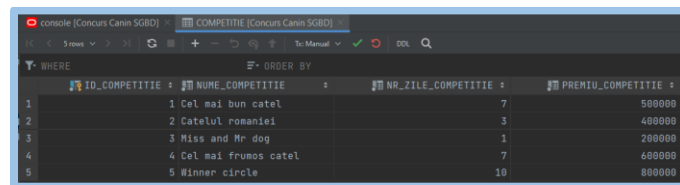


The screenshot shows a database console window with the 'ECHIPAMENT' table selected. The table has four columns: ID\_ECHIPAMENT, NUME\_ECHIPAMENT, MATERIAL, and GREUTATE\_ECHIPAMENT. It contains five rows of data.

ID_ECHIPAMENT	NUME_ECHIPAMENT	MATERIAL	GREUTATE_ECHIPAMENT
1	101 Rampa 1m	Plastic PLA	7
2	102 Tunel 10m	Plasa	1
3	103 Trepte	Plastic ABS	5
4	104 Rampa 2m	Plastic HIPS	15
5	105 Gard 0.5m	Titan	1

## ENTITATE: Competitie

```
INSERT INTO COMPETITIE VALUES
(1, 'Cel mai bun catel', 7, 500000);
INSERT INTO COMPETITIE VALUES
(2, 'Catelul romaniei', 3, 400000);
INSERT INTO COMPETITIE VALUES
(3, 'Miss and Mr dog', 1, 200000);
INSERT INTO COMPETITIE VALUES
(4, 'Cel mai frumos catel', 7, 600000);
INSERT INTO COMPETITIE VALUES
(5, 'Winner circle', 10, 800000);
```

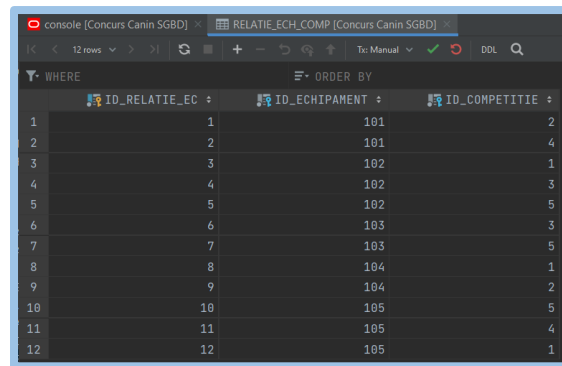


The screenshot shows a database console window with the 'COMPETITIE' table selected. The table has four columns: ID\_COMPETITIE, NUME\_COMPETITIE, NR\_ZILE\_COMPETITIE, and PREMIU\_COMPETITIE. It contains five rows of data.

ID_COMPETITIE	NUME_COMPETITIE	NR_ZILE_COMPETITIE	PREMIU_COMPETITIE
1	1 Cel mai bun catel	7	500000
2	2 Catelul romaniei	3	400000
3	3 Miss and Mr dog	1	200000
4	4 Cel mai frumos catel	7	600000
5	5 Winner circle	10	800000

## ENTITATE: Relatie\_ech\_comp

```
INSERT INTO RELATIE_ECH_COMP VALUES
(1, 101, 2);
INSERT INTO RELATIE_ECH_COMP VALUES
(2, 101, 4);
INSERT INTO RELATIE_ECH_COMP VALUES
(3, 102, 1);
INSERT INTO RELATIE_ECH_COMP VALUES
(4, 102, 3);
INSERT INTO RELATIE_ECH_COMP VALUES
(5, 102, 5);
INSERT INTO RELATIE_ECH_COMP VALUES
(6, 103, 3);
INSERT INTO RELATIE_ECH_COMP VALUES
(7, 103, 5);
INSERT INTO RELATIE_ECH_COMP VALUES
(8, 104, 1);
INSERT INTO RELATIE_ECH_COMP VALUES
(9, 104, 2);
INSERT INTO RELATIE_ECH_COMP VALUES
(10, 105, 5);
INSERT INTO RELATIE_ECH_COMP VALUES
(11, 105, 4);
INSERT INTO RELATIE_ECH_COMP VALUES
(12, 105, 1);
```

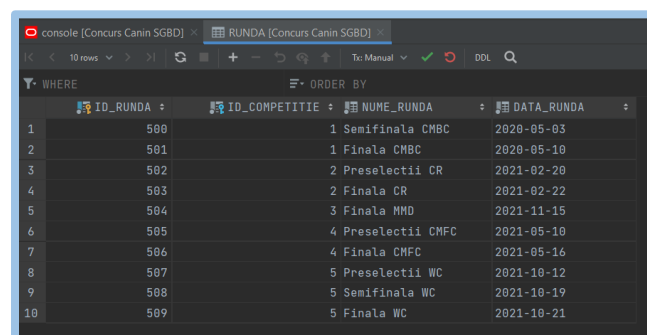


The screenshot shows a database console window with the table RELATIE\_ECH\_COMP selected. The table has three columns: ID\_RELATIE\_EC, ID\_ECHIPAMENT, and ID\_COMPETITIE. The data is as follows:

ID_RELATIE_EC	ID_ECHIPAMENT	ID_COMPETITIE
1	101	2
2	101	4
3	102	1
4	102	3
5	102	5
6	103	3
7	103	5
8	104	1
9	104	2
10	105	5
11	105	4
12	105	1

## ENTITATE: Runda

```
INSERT INTO RUNDA VALUES
(500, 1, 'Semifinala CMBC', TO_DATE('03-MAY-2020', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(501, 1, 'Finala CMBC', TO_DATE('10-MAY-2020', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(502, 2, 'Preselectii CR', TO_DATE('20-FEB-2021', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(503, 2, 'Finala CR', TO_DATE('22-FEB-2021', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(504, 3, 'Finala MMD', TO_DATE('15-NOV-2021', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(505, 4, 'Preselectii CMFC', TO_DATE('10-MAY-2021', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(506, 4, 'Finala CMFC', TO_DATE('16-MAY-2021', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(507, 5, 'Preselectii WC', TO_DATE('12-OCT-2021', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(508, 5, 'Semifinala WC', TO_DATE('19-OCT-2021', 'dd-MON-yyyy'));
INSERT INTO RUNDA VALUES
(509, 5, 'Finala WC', TO_DATE('21-OCT-2021', 'dd-MON-yyyy'));
```



The screenshot shows a database console window with the table RUNDA selected. The table has four columns: ID\_RUNDA, ID\_COMPETITIE, NUME\_RUNDA, and DATA\_RUNDA. The data is as follows:

ID_RUNDA	ID_COMPETITIE	NUME_RUNDA	DATA_RUNDA
500	1	Semifinala CMBC	2020-05-03
501	1	Finala CMBC	2020-05-10
502	2	Preselectii CR	2021-02-20
503	2	Finala CR	2021-02-22
504	3	Finala MMD	2021-11-15
505	4	Preselectii CMFC	2021-05-10
506	4	Finala CMFC	2021-05-16
507	5	Preselectii WC	2021-10-12
508	5	Semifinala WC	2021-10-19
509	5	Finala WC	2021-10-21

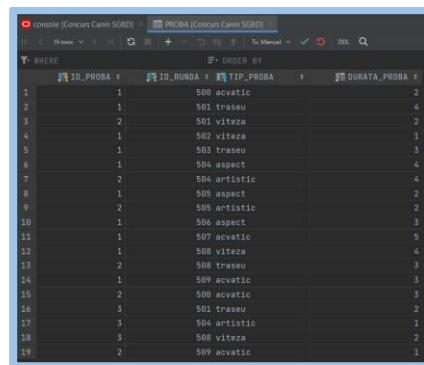
## ENTITATE: Proba

```
INSERT INTO PROBA VALUES
(1, 500, 'acvatic', 2);
INSERT INTO PROBA VALUES
(1, 501, 'traseu', 4);
INSERT INTO PROBA VALUES
(2, 501, 'viteza', 2);
```

```

INSERT INTO PROBA VALUES
(1, 502, 'viteza', 1);
INSERT INTO PROBA VALUES
(1, 503, 'traseu', 3);
INSERT INTO PROBA VALUES
(1, 504, 'aspect', 4);
INSERT INTO PROBA VALUES
(2, 504, 'artistic', 4);
INSERT INTO PROBA VALUES
(1, 505, 'aspect', 2);
INSERT INTO PROBA VALUES
(2, 505, 'artistic', 2);
INSERT INTO PROBA VALUES
(1, 506, 'aspect', 3);
INSERT INTO PROBA VALUES
(1, 507, 'acvatic', 5);
INSERT INTO PROBA VALUES
(1, 508, 'viteza', 4);
INSERT INTO PROBA VALUES
(2, 508, 'traseu', 3);
INSERT INTO PROBA VALUES
(1, 509, 'acvatic', 3);
INSERT INTO PROBA VALUES
(2, 500, 'acvatic', 3);
INSERT INTO PROBA VALUES
(3, 501, 'traseu', 2);
INSERT INTO PROBA VALUES
(3, 504, 'artistic', 1);
INSERT INTO PROBA VALUES
(3, 508, 'viteza', 2);
INSERT INTO PROBA VALUES
(2, 509, 'acvatic', 1);

```



ID_PROBA	ID_RUNDA	TIP_PROBA	DURATA_PROBA
1	1	500 acvatic	2
2	1	501 traseu	4
3	2	501 viteza	2
4	1	502 viteza	1
5	1	503 traseu	3
6	1	504 aspect	4
7	2	505 artistic	4
8	1	505 aspect	2
9	2	505 artistic	2
10	1	506 aspect	3
11	1	507 acvatic	5
12	1	508 viteza	4
13	2	508 traseu	3
14	1	509 acvatic	3
15	2	500 acvatic	3
16	1	501 traseu	2
17	3	504 artistic	1
18	3	508 viteza	2
19	2	509 acvatic	1

## ENTITATE: Jurat

```

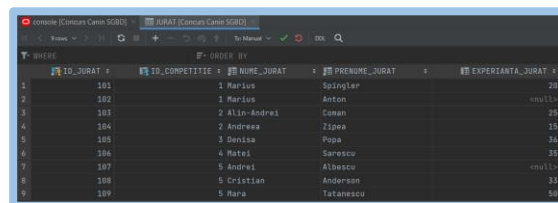
INSERT INTO JURAT VALUES
(101, 1, 'Marius', 'Spingler', 20);
INSERT INTO JURAT VALUES
(102, 1, 'Marius', 'Anton', NULL);
INSERT INTO JURAT VALUES
(103, 2, 'Alin-Andrei', 'Coman', 25);
INSERT INTO JURAT VALUES
(104, 2, 'Andreea', 'Zipea', 15);

```

## SGBD – Grupa 251

### Pătrânjel David-George

```
INSERT INTO JURAT VALUES
(105, 3, 'Denisa', 'Popa', 36);
INSERT INTO JURAT VALUES
(106, 4, 'Matei', 'Sarescu', 35);
INSERT INTO JURAT VALUES
(107, 5, 'Andrei', 'Albescu', NULL);
INSERT INTO JURAT VALUES
(108, 5, 'Cristian', 'Anderson', 33);
INSERT INTO JURAT VALUES
(109, 5, 'Mara', 'Tatanescu', 50);
```

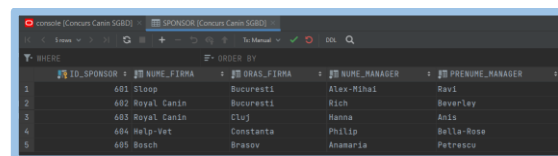


The screenshot shows a database console window with the JURAT table selected. The table contains 9 rows of data, including columns for ID, competition ID, name, surname, and experience.

ID_JURAT	ID_COMPETITIE	NUME_JURAT	PRENUME_JURAT	EXPERIENTA_JURAT
101	1	Marius	Spingler	20
102	1	Marius	Anton	<null>
103	2	Alin-Andrei	Coman	25
104	2	Andreea	Zipea	15
105	3	Denisa	Popa	36
106	4	Matei	Sarescu	35
107	5	Andrei	Albescu	<null>
108	5	Cristian	Anderson	33
109	5	Mara	Tatanescu	50

### ENTITATE: Sponsor

```
INSERT INTO SPONSOR VALUES
(601, 'Sloop', 'Bucuresti', 'Alex-Mihai', 'Ravi');
INSERT INTO SPONSOR VALUES
(602, 'Royal Canin', 'Bucuresti', 'Rich', 'Beverley');
INSERT INTO SPONSOR VALUES
(603, 'Royal Canin', 'Cluj', 'Hanna', 'Anis');
INSERT INTO SPONSOR VALUES
(604, 'Help-Vet', 'Constanta', 'Philip', 'Bella-Rose');
INSERT INTO SPONSOR VALUES
(605, 'Bosch', 'Brasov', 'Anamaria', 'Petrescu');
```

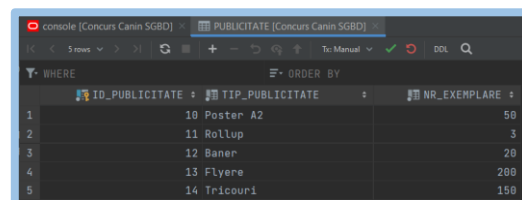


The screenshot shows a database console window with the SPONSOR table selected. The table contains 5 rows of data, including columns for ID, name, city, manager name, and manager surname.

ID_SPONSOR	NUME_FIRMA	ORAS_FIRMA	NUME_MANAGER	PRENUME_MANAGER
601	Sloop	Bucuresti	Alex-Mihai	Ravi
602	Royal Canin	Bucuresti	Rich	Beverley
603	Royal Canin	Cluj	Hanna	Anis
604	Help-Vet	Constanta	Philip	Bella-Rose
605	Bosch	Brasov	Anamaria	Petrescu

### ENTITATE: Publicitate

```
INSERT INTO PUBLICITATE VALUES
(10, 'Poster A2', 50);
INSERT INTO PUBLICITATE VALUES
(11, 'Rollup', 3);
INSERT INTO PUBLICITATE VALUES
(12, 'Baner', default);
INSERT INTO PUBLICITATE VALUES
(13, 'Flyere', 200);
INSERT INTO PUBLICITATE VALUES
(14, 'Tricouri', 150);
```



The screenshot shows a database console window with the PUBLICITATE table selected. The table contains 5 rows of data, including columns for ID, type, and number of examples.

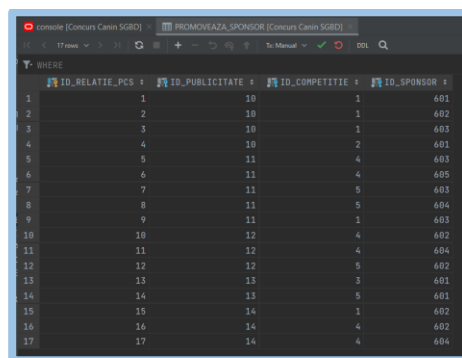
ID_PUBLICITATE	TIP_PUBLICITATE	NR_EXEMPLARE
10	Poster A2	50
11	Rollup	3
12	Baner	20
13	Flyere	200
14	Tricouri	150

### ENTITATE: Promoveaza\_sponsor

## SGBD – Grupa 251

Pătrânjel David-George

```
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(1, 10, 1, 601);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(2, 10, 1, 602);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(3, 10, 1, 603);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(4, 10, 2, 601);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(5, 11, 4, 603);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(6, 11, 4, 605);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(7, 11, 5, 603);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(8, 11, 5, 604);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(9, 11, 1, 603);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(10, 12, 4, 602);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(11, 12, 4, 604);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(12, 12, 5, 602);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(13, 13, 3, 601);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(14, 13, 5, 601);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(15, 14, 1, 602);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(16, 14, 4, 602);
INSERT INTO PROMOVEAZA_SPONSOR VALUES
(17, 14, 4, 604);
```



ID_RELATIE_PCS	ID_PUBLICITATE	ID_COMPETITIE	ID_SPONSOR
1	10	1	601
2	10	1	602
3	10	1	603
4	10	2	601
5	11	4	603
6	11	4	605
7	11	5	603
8	11	5	604
9	11	1	603
10	12	4	602
11	12	4	604
12	12	5	602
13	13	3	601
14	13	5	601
15	14	1	602
16	14	4	602
17	14	4	604

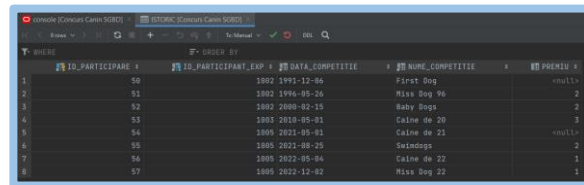
## ENTITATE: Istoric

```
INSERT INTO ISTORIC VALUES
(50, 1002, TO_DATE('06-DEC-1991', 'dd-MON-yyyy'), 'First Dog', null);
INSERT INTO ISTORIC VALUES
(51, 1002, TO_DATE('26-MAY-1996', 'dd-MON-yyyy'), 'Miss Dog 96', 2);
INSERT INTO ISTORIC VALUES
(52, 1002, TO_DATE('15-FEB-2000', 'dd-MON-yyyy'), 'Baby Dogs', 2);
INSERT INTO ISTORIC VALUES
(53, 1003, TO_DATE('01-MAY-2010', 'dd-MON-yyyy'), 'Caine de 20', 3);
INSERT INTO ISTORIC VALUES
(54, 1005, TO_DATE('01-MAY-2021', 'dd-MON-yyyy'), 'Caine de 21', null);
INSERT INTO ISTORIC VALUES
(55, 1005, TO_DATE('25-AUG-2021', 'dd-MON-yyyy'), 'Swimdogs', 2);
```

## SGBD – Grupa 251

### Pătrânjel David-George

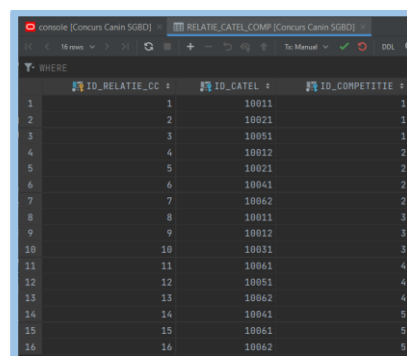
```
INSERT INTO ISTORIC VALUES
(56, 1005, TO_DATE('04-MAY-2022', 'dd-MON-yyyy'), 'Caine de 22', 1);
INSERT INTO ISTORIC VALUES
(57, 1005, TO_DATE('02-DEC-2022', 'dd-MON-yyyy'), 'Miss Dog 22', 1);
```



ID_PARTICIPARE	ID_PARTICIPANT_EXP	DATA_COMPETITIE	NUME_COMPETITIE	PREMIU
50	1002	1991-12-06	First Dog	<null>
51	1002	1996-05-26	Miss Dog 96	2
52	1002	2000-02-19	Baby Dogs	2
53	1003	2019-09-01	Caine de 20	3
54	1005	2021-09-01	Caine de 21	<null>
55	1005	2021-08-25	Swindogs	2
56	1005	2022-05-04	Caine de 22	1
57	1005	2022-12-02	Miss Dog 22	1

### ENTITATE: Relatie\_catel\_comp

```
INSERT INTO RELATIE_CATEL_COMP VALUES
(1, 10011, 1);
INSERT INTO RELATIE_CATEL_COMP VALUES
(2, 10021, 1);
INSERT INTO RELATIE_CATEL_COMP VALUES
(3, 10051, 1);
INSERT INTO RELATIE_CATEL_COMP VALUES
(4, 10012, 2);
INSERT INTO RELATIE_CATEL_COMP VALUES
(5, 10021, 2);
INSERT INTO RELATIE_CATEL_COMP VALUES
(6, 10041, 2);
INSERT INTO RELATIE_CATEL_COMP VALUES
(7, 10062, 2);
INSERT INTO RELATIE_CATEL_COMP VALUES
(8, 10011, 3);
INSERT INTO RELATIE_CATEL_COMP VALUES
(9, 10012, 3);
INSERT INTO RELATIE_CATEL_COMP VALUES
(10, 10031,3);
INSERT INTO RELATIE_CATEL_COMP VALUES
(11, 10061 ,4);
INSERT INTO RELATIE_CATEL_COMP VALUES
(12, 10051 ,4);
INSERT INTO RELATIE_CATEL_COMP VALUES
(13, 10062, 4);
INSERT INTO RELATIE_CATEL_COMP VALUES
(14, 10041,5);
INSERT INTO RELATIE_CATEL_COMP VALUES
(15, 10061,5);
INSERT INTO RELATIE_CATEL_COMP VALUES
(16, 10062 ,5);
```



ID_RELATIE_CC	ID_CATEL	ID_COMPETITIE
1	10011	1
2	10021	1
3	10051	1
4	10012	2
5	10021	2
6	10041	2
7	10062	2
8	10011	3
9	10012	3
10	10031	3
11	10061	4
12	10051	4
13	10062	4
14	10041	5
15	10061	5
16	10062	5



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

**Enunț:** Definim un câțel ca fiind n-activ dacă are cel mai mic număr nenul de participări la primele n competiții, după numărul de probe tip traseu pe care le conține. Dacă sunt mai multe competiții cu același număr de probe tip traseu, acestea vor fi ordonate alfabetic după numele competiției. Dorim să determinăm numele și idul câțelilor n-activi, pentru un n dat.

```
CREATE OR REPLACE PROCEDURE findLeastNActiveDog
(v_nrcomp NUMBER DEFAULT 1)
IS
    /*Record*/
    type dog_info is record (
        id_catel CATEL.id_catel%TYPE,
        nume_catel CATEL.nume_catel%TYPE,
        id_comp RELATIE_CATEL_COMP.id_competitie%TYPE
    );
    /*Tablou imbricat*/
    type tablou_imbricat is table of dog_info;
    v_list_dogs tablou_imbricat := tablou_imbricat();
    v_old_dog CATEL.id_catel%TYPE;
    v_nume CATEL.nume_catel%TYPE;
    /*Tablou indexat*/
    type tablou_indexat is table of competitie.id_competitie%TYPE index by pls_integer;
    v_list_comps tablou_indexat;
    v_allcomps number;
    /*Vector*/
    type vector is varray(150) of catel.id_catel%TYPE;
    v_list_dogsids vector := vector();
    i number;
    j number;
    minim number;
    contor number;
    too_many_comps exception;
    too_many_comps_traseu exception;
BEGIN
    select count(*)
    into v_allcomps
    from COMPETITIE;

    if(v_nrcomp > v_allcomps) then
        raise too_many_comps;
    end if;

    select CAUX.ID_COMPETITIE
    bulk collect into v_list_comps
    from COMPETITIE CAUX
    join (select C.ID_COMPETITIE, count(*) as nr_probe_tr
          from COMPETITIE C
          join RUNDA R2 on C.ID_COMPETITIE = R2.ID_COMPETITIE
          join PROBA P on R2.ID_RUNDA = P.ID_RUNDA
          where Lower(p.TIP_PROBA) = 'traseu'
          GROUP BY C.ID_COMPETITIE, C.NUME_COMPETITIE) D on D.ID_COMPETITIE = CAUX.ID_COMPETITIE
    order by D.nr_probe_tr desc, CAUX.NUME_COMPETITIE;

    if(v_nrcomp > v_list_comps.COUNT) then
        raise too_many_comps_traseu;
    end if;

    select c2.ID_CATEL, c2.NUME_CATEL, rcc.ID_COMPETITIE
```

```
bulk collect into v_list_dogs
from RELATIE_CATEL_COMP rcc
join CATEL C2 on C2.ID_CATEL = rcc.ID_CATEL
order by c2.ID_CATEL;

v_old_dog := v_list_dogs(1).id_catel;
contor := 0;
minim := v_list_dogs.count;
for i in v_list_dogs.first..v_list_dogs.last loop
    if(v_old_dog <> v_list_dogs(i).id_catel) then
        if(minim = contor and contor <> 0) then
            v_list_dogsids.extend;
            v_list_dogsids(v_list_dogsids.last) := v_old_dog;
        elsif(minim > contor and contor <> 0) then
            v_list_dogsids.delete;
            v_list_dogsids.extend;
            v_list_dogsids(v_list_dogsids.last) := v_old_dog;
            minim := contor;
        end if;
        v_old_dog := v_list_dogs(i).id_catel;
        contor:= 0;
    end if;
    for j in 1..v_nrcomp loop
        if(v_list_dogs(i).id_comp = v_list_comps(j)) then
            contor := contor + 1;
        end if;
    end loop;
end loop;
if(minim = contor and contor <> 0) then
    v_list_dogsids.extend;
    v_list_dogsids(v_list_dogsids.last) := v_old_dog;
elsif(minim > contor and contor <> 0) then
    v_list_dogsids.delete;
    v_list_dogsids.extend;
    v_list_dogsids(v_list_dogsids.last) := v_old_dog;
    minim := contor;
end if;

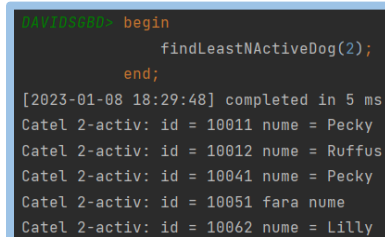
for i in v_list_dogsids.first..v_list_dogsids.last loop
    select NUME_CATEL
    into v_nume
    from CATEL
    where ID_CATEL = v_list_dogsids(i);

    if(v_nume is not null) then
        DBMS_OUTPUT.PUT_LINE('Catel ' || v_nrcomp || '-activ: id = ' || v_list_dogsids(i) || ' nume = '
|| v_nume);
    else
        DBMS_OUTPUT.PUT_LINE('Catel ' || v_nrcomp || '-activ: id = ' || v_list_dogsids(i) || ' fara
nume');
    end if;
end loop;

EXCEPTION
when too_many_comps then
    RAISE_APPLICATION_ERROR(-20001,'Prea multe competitii cerute');
when too_many_comps_traseu then
    RAISE_APPLICATION_ERROR(-20002,'Prea multe competitii tip traseu cerute');
when TOO_MANY_ROWS then
    RAISE_APPLICATION_ERROR(-20003,'Prea multi catei gasiti cu acelasi id');
```

```
        WHEN NO_DATA_FOUND THEN
            RAISE_APPLICATION_ERROR(-20004,'Nu s-au gasit datele');
        WHEN OTHERS THEN
            RAISE_APPLICATION_ERROR(-20000,'Alta eroare!');

    END findLeastNActiveDog;
/
begin
    findLeastNActiveDog(2);
end;
/
```



```
DAVID@DB00> begin
        findLeastNActiveDog(2);
    end;
[2023-01-08 18:29:48] completed in 5 ms
Catel 2-activ: id = 10011 nume = Pecky
Catel 2-activ: id = 10012 nume = Ruffus
Catel 2-activ: id = 10041 nume = Pecky
Catel 2-activ: id = 10051 fara nume
Catel 2-activ: id = 10062 nume = Lilly
```

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

**Enunț:** Competițiile cu un premiu mai mare de 500.000 RON sunt considerate competiții de clasă înaltă. Astfel, toți proprietarii care dețin în total mai mult de  $n$  căței trebuie să îi vaccineze pe cei înscrși în aceste competiții de clasă și care nu erau vaccinați înainte. Să se afișeze pentru fiecare participant ce deține mai mult de  $n$  căței numărul de căței care trebuie vaccinați și să se actualizeze acest lucru în baza de date.

```
CREATE OR REPLACE PROCEDURE proc1
    (v_in NUMBER DEFAULT 0)
IS
    contor number;
    flag number := 0;
    cursor c is
        select P.ID_PROPRIETAR, count(*) nr_dogs
        from PROPRIETAR P
        join CATEL C2 on P.ID_PROPRIETAR = C2.ID_PARTICIPANT
        group by P.ID_PROPRIETAR
        having count(*) >= v_in;

    cursor u(v_idowner PROPRIETAR.id_proprietar%type) is
        select *
        from CATEL
        where ID_PARTICIPANT = v_idowner and ID_CATEL in (
            select ID_CATEL
            from RELATIE_CATEL_COMP
            join COMPETITIE C3 on C3.ID_COMPETITIE = RELATIE_CATEL_COMP.ID_COMPETITIE
            where c3.PREMIU_COMPETITIE >= 500000
        )
        for update of VACCINAT nowait ;
    no_owners exception;

BEGIN
    for owner in c loop
        flag := 1;
```

```
        contor := 0;

        for j in u(owner.ID_PROPRIETAR) loop
            if(lower(j.VACCINAT) <> 't' or j.VACCINAT is null) then
                contor := contor + 1;

                update CATEL
                set VACCINAT = 'T'
                where current of u;
            end if;
        end loop;

        DBMS_OUTPUT.PUT('Proprietarul ' || owner.ID_PROPRIETAR);
        if(contor = 0) then
            DBMS_OUTPUT.PUT_LINE(' nu si-a vaccinat niciun catel');
        else
            DBMS_OUTPUT.PUT_LINE(' si-a vaccinat ' || contor || ' catei');
        end if;

        rollback;
    end loop;
    if(flag = 0) then
        raise no_owners;
    end if;

EXCEPTION
    when INVALID_CURSOR then
        RAISE_APPLICATION_ERROR(-20001, 'Cursorul este inchis');
    when CURSOR_ALREADY_OPEN then
        RAISE_APPLICATION_ERROR(-20002, 'Cursor deja deschis');
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20003, 'Nu s-au gasit date');
    WHEN no_owners THEN
        RAISE_APPLICATION_ERROR(-20004, 'Nu sunt proprietari cu acest nr minim de catei');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000, 'Alta eroare!');
END proc1;
/
DECLARE
    v_x number := &p;
BEGIN
    proc1(v_x);
END;
/
```

```
DAVIDSGR@> DECLARE
        v_x number := 2;
    BEGIN
        proc1(v_x);
    END;

[2023-01-08 23:27:38] completed in 7 ms
Proprietarul 1001 nu si-a vaccinat niciun catel
Proprietarul 1006 si-a vaccinat 2 catei
```

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

**Enunț:** În anul 2021 a apărut o epidemie canină. Astfel, la anumite competiții cu runde în 2021, cățelei nevaccinați din anumite rase trebuie să poarte o zgardă de protecție de 0.5kg. Să se afișeze toți cățelei care fac parte din rasa specificată care participă la competiția dată, să se afle numărul de căței care au primit zgarda de protecție, și să se actualizeze greutatea cățelei cu zgardă. Să se trateze toate excepțiile ce pot apărea.

```
CREATE OR REPLACE FUNCTION dogs2022
    (vin_rasa IN VARCHAR2,
     vin_idcomp IN NUMBER)
RETURN NUMBER
IS
    nr_err number;
    ans number;
    aux number;
    my_no_data_found exception;
    no_dogs_to_vaccinate exception;
    vals_modif number := 0;
    type tablou is table of catel.ID_CATEL%TYPE;
    v_list_dogs tablou := tablou();
begin

    select count(*)
    into aux
    from CATEL c2
    where Lower(c2.RASA) = Lower(vin_rasa);
    if aux = 0 then
        nr_err := 1;
        raise my_no_data_found;
    end if;

    select count(*)
    into aux
    from COMPETITIE c3
    where C3.ID_COMPETITIE = vin_idcomp;
    if aux = 0 then
        nr_err := 2;
        raise my_no_data_found;
    end if;

    select count(*)
    into aux
    from COMPETITIE c3
    join RUNDA R2 on C3.ID_COMPETITIE = R2.ID_COMPETITIE
    where C3.ID_COMPETITIE = vin_idcomp and
           TO_CHAR(r2.DATA_RUNDA, 'YYYY') = 2021;
    if aux = 0 then
        nr_err := 3;
        raise my_no_data_found;
    end if;

    select distinct C2.ID_CATEL
    bulk collect into v_list_dogs
    from RELATIE_CATEL_COMP
    join CATEL C2 on C2.ID_CATEL = RELATIE_CATEL_COMP.ID_CATEL
    join COMPETITIE C3 on C3.ID_COMPETITIE = RELATIE_CATEL_COMP.ID_COMPETITIE
    join RUNDA R2 on C3.ID_COMPETITIE = R2.ID_COMPETITIE
```

```
where Lower(c2.RASA) = Lower(vin_rasa) and
       C3.ID_COMPETITIE = vin_idcomp and
       TO_CHAR(r2.DATA_RUNDA, 'YYYY') = 2021;

for i in v_list_dogs.first..v_list_dogs.last loop
    DBMS_OUTPUT.PUT_LINE(v_list_dogs(i));
    update CATEL C
    set C.GREUTATE = C.GREUTATE + 0.5
    where (Lower(C.VACCINAT) = 'f' or C.VACCINAT is null) and C.ID_CATEL = v_list_dogs(i);
    vals_modif := vals_modif + SQL%ROWCOUNT;
end loop;

if vals_modif = 0 then
    raise no_dogs_to_vaccinate;
end if;

return vals_modif;
exception
    WHEN my_no_data_found THEN
        if nr_err = 1 then
            DBMS_OUTPUT.PUT_LINE('Nu s-a gasit niciun catel cu rasa ' || vin_rasa);
        elsif nr_err = 2 then
            DBMS_OUTPUT.PUT_LINE('Nu s-a gasit nicio competitie cu idul ' || vin_idcomp);
        elsif nr_err = 3 then
            DBMS_OUTPUT.PUT_LINE('Nu sunt runde in anul 2021 in competitia ' || vin_idcomp);
        end if;
        RETURN -20001;
    WHEN no_dogs_to_vaccinate THEN
        DBMS_OUTPUT.PUT_LINE('No dogs need vaccination');
        RETURN -20002;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Alta eroare');
        DBMS_OUTPUT.PUT_LINE('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE('Mesajul erorii: ' || SQLERRM);
        RETURN -20000;
end dogs2022;
/
begin
    DBMS_OUTPUT.PUT_LINE('Result ' || dogs2022('Husky', 1));
    DBMS_OUTPUT.PUT_LINE('Result ' || dogs2022('Bison', 10));
    DBMS_OUTPUT.PUT_LINE('Result ' || dogs2022('Bison', 1));
    DBMS_OUTPUT.PUT_LINE('Result ' || dogs2022('Bison', 3));
    UPDATE CATEL
    SET VACCINAT = 'F'
    WHERE ID_CATEL = 10011;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Result ' || dogs2022('Bison', 3));
    COMMIT;
end;
/
```

```
[2023-01-13 03:27:23] completed in 14 ms
Nu s-a gasit niciun catel cu rasa Husky
Result -20001
Nu s-a gasit nicio competitie cu idul 10
Result -20001
Nu sunt runde in anul 2021 in competitia 1
Result -20001
10011
10031
No dogs need vaccination
Result -20002
10011
10031
Result 1
```

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

**Enunț:** Pentru un k dat, să se afișeze toți participanții experimentați ce au participat la ultima competiție cu cel mult k ani înainte de data curentă și care participă la o competiție jurizată de un jurat al cărui nume este dat.

```
CREATE OR REPLACE PROCEDURE participant_exp
(vin_n in number,
 vin_name in varchar2)
IS
    nr_err number;
    ind number := 0;
    id_prop PARTICIPANT_EXPERIMENTAT.ID_PARTICIPANT%TYPE;
    type tablou is table of PARTICIPANT_EXPERIMENTAT.ID_PARTICIPANT%TYPE;
    v_list_part tablou := tablou();
    type date_part is record(
        id_proprietar proprietar.id_proprietar%TYPE,
        nume_proprietar proprietar.nume_participant%TYPE,
        prenume_proprietar proprietar.prenume_participant%TYPE
    );
    type tablou21 is table of date_part;
    v_list_part_fin tablou21 := tablou21();
    type tablou2 is table of date_part INDEX BY PLS_INTEGER;
    v_list_part_aux tablou2;
    type tablou3 is table of jurat.nume_jurat%TYPE INDEX BY PLS_INTEGER;
    v_aux_jurat tablou3;

    negative exception;
    my_no_data_found exception;
    no_participants_incase exception;
    multi_jurati exception;

BEGIN
    if vin_n < 0 then
        raise negative;
    end if;

    select NUME_JURAT
    bulk collect into v_aux_jurat
    from JURAT
    where NUME_JURAT = vin_name;

    if sql%rowcount > 1 then
        raise multi_jurati;
    elsif sql%notfound then
        nr_err := 1;
```

```

        raise my_no_data_found;
    end if;

    select PE.ID_PARTICIPANT
    bulk collect into v_list_part
    from PARTICIPANT_EXPERIMENTAT PE
    join (
        select i.ID_PARTICIPANT_EXP, MAX(DATA_COMPETITIE) last_comp
        from ISTORIC i
        group by i.ID_PARTICIPANT_EXP

    ) aux on PE.ID_PARTICIPANT = aux.ID_PARTICIPANT_EXP
    where ROUND(MONTHS_BETWEEN(SYSDATE, aux.last_comp)) <= vin_n * 12;

    if v_list_part.count = 0 then
        nr_err := 2;
        raise my_no_data_found;
    end if;

    for i in v_list_part.first..v_list_part.last loop
        id_prop := v_list_part(i);

        select
            P.ID_PROPRIETAR,
            P.NUME_PARTICIPANT,
            P.PRENUME_PARTICIPANT
        bulk collect into v_list_part_aux
        from PROPRIETAR P
        where P.ID_PROPRIETAR = id_prop and P.ID_PROPRIETAR in(
            select distinct
                C1.ID_PARTICIPANT
            from RELATIE_CATEL_COMP R
            join COMPETITIE C2 on R.ID_COMPETITIE = C2.ID_COMPETITIE
            join CATEL C1 on R.ID_CATEL = C1.ID_CATEL
            where C2.ID_COMPETITIE in(
                select
                    C3.ID_COMPETITIE
                from COMPETITIE C3
                join JURAT J on C3.ID_COMPETITIE = J.ID_COMPETITIE
                where Lower(J.NUME_JURAT) = Lower(vin_name))
        );

        if SQL%ROWCOUNT = 1 then
            v_list_part_fin.extend();
            v_list_part_fin(v_list_part_fin.last) := v_list_part_aux(v_list_part_aux.first);
        end if;
    end loop;

    if v_list_part_fin.count = 0 then
        raise no_participants_incase;
    end if;

    for i in v_list_part_fin.first..v_list_part_fin.last loop
        ind := ind + 1;
        DBMS_OUTPUT.PUT_LINE('Participant ' || ind || ') id: ' || v_list_part_fin(i).id_proprietar || '
nume: '
        || v_list_part_fin(i).nume_proprietar || ' ' || v_list_part_fin(i).prenume_proprietar);
    end loop;

```



```
EXCEPTION
    WHEN my_no_data_found THEN
        if nr_err = 1 then
            DBMS_OUTPUT.PUT_LINE('Nu exista jurat cu numele ' || vin_name);
        elsif nr_err = 2 then
            DBMS_OUTPUT.PUT_LINE('Nu sunt participanti experimentati care au participat la ultimul concurs
cu cel mult ' || vin_n || ' ani');
        end if;
        RAISE_APPLICATION_ERROR(-20001, 'Nu s-au gasit date');
    WHEN no_participants_incase THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun participant experimentat care sa respecte toate conditiile');
        RAISE_APPLICATION_ERROR(-20002, 'Nu exista niciun participant experimentat care sa respecte toate
conditiile');
    WHEN multi_jurati THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multi jurati cu acest nume');
        RAISE_APPLICATION_ERROR(-20003, 'Exista mai multi jurati cu acest nume!');
    WHEN negative THEN
        DBMS_OUTPUT.PUT_LINE('Numarul de ani este unul pozitiv');
        RAISE_APPLICATION_ERROR(-20004, 'Numarul de ani trebuie sa fie unul pozitiv!');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Alta eroare');
        DBMS_OUTPUT.PUT_LINE ('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE ('Mesajul erorii: ' || SQLERRM);
        RAISE_APPLICATION_ERROR(-20000, 'Alta eroare!');
END participant_exp;
/
begin
    /*participant_exp(-2, 'Matei');*/
    /*participant_exp(20, 'Anamaria');*/
    /*participant_exp(20, 'Marius');*/
    /*participant_exp(0, 'Andreea');*/
    /*participant_exp(20, 'Andreea');*/
    /*Actualizam datele pentru a vedea mai multe afisari*/
    update RELATIE_CATEL_COMP
    set ID_COMPETITIE = 4
    where ID_RELATIE_CC = 5;
    commit;

    participant_exp(30, 'Matei');

    update RELATIE_CATEL_COMP
    set ID_COMPETITIE = 2
    where ID_RELATIE_CC = 5;
    commit;
end;
/
```

```
DAVIDSGBD> begin
    participant_exp(-2, 'Matei');
end;
[2023-01-13 13:43:42] [72000][20004]
[2023-01-13 13:43:42] ORA-20004: Numarul de ani trebuie sa fie unul pozitiv!
[2023-01-13 13:43:42] ORA-06512: at "DAVIDSGBD.PARTICIPANT_EXP", line 115
[2023-01-13 13:43:42] ORA-06512: at line 2
[2023-01-13 13:43:42] Position: 0
Numarul de ani este unul pozitiv
```

```
DAVIDSGBD> begin
    participant_exp(20, 'Anamaria');
end;
[2023-01-13 13:44:32] [72000][20001]
[2023-01-13 13:44:32] ORA-20001: Nu s-au gasit date
[2023-01-13 13:44:32] ORA-06512: at "DAVIDSGBD.PARTICIPANT_EXP", line 106
[2023-01-13 13:44:32] ORA-06512: at line 2
[2023-01-13 13:44:32] Position: 0
Nu exista jurat cu numele Anamaria
```

```
DAVIDSGBD> begin
    participant_exp(20, 'Marius');
end;
[2023-01-13 13:45:05] [72000][20003]
[2023-01-13 13:45:05] ORA-20003: Exista mai multi jurati cu acest nume!
[2023-01-13 13:45:05] ORA-06512: at "DAVIDSGBD.PARTICIPANT_EXP", line 112
[2023-01-13 13:45:05] ORA-06512: at line 2
[2023-01-13 13:45:05] Position: 0
Exista mai multi jurati cu acest nume
```

```
DAVIDSGBD> begin
    participant_exp(0, 'Andreea');
end;
[2023-01-13 13:45:40] [72000][20001]
[2023-01-13 13:45:40] ORA-20001: Nu s-au gasit date
[2023-01-13 13:45:40] ORA-06512: at "DAVIDSGBD.PARTICIPANT_EXP", line 106
[2023-01-13 13:45:40] ORA-06512: at line 2
[2023-01-13 13:45:40] Position: 0
Nu sunt participantii experimentati care au participat la ultimul concurs cu cel mult 0 ani
```

```
DAVIDSGBD> begin
    participant_exp(20, 'Andreea');
end;
[2023-01-13 13:46:01] [72000][20002]
[2023-01-13 13:46:01] ORA-20002: Nu exista niciun participant experimentat care sa respecte toate conditiile
[2023-01-13 13:46:01] ORA-06512: at "DAVIDSGBD.PARTICIPANT_EXP", line 109
[2023-01-13 13:46:01] ORA-06512: at line 2
[2023-01-13 13:46:01] Position: 0
Nu exista niciun participant experimentat care sa respecte toate conditiile
```

```
DAVIDSGBD> begin
    update RELATIE_CATEL_COMP
    set ID_COMPETITIE = 4
    where ID_RELATIE_CC = 5;
    commit;

    participant_exp(30, 'Matei');

    update RELATIE_CATEL_COMP
    set ID_COMPETITIE = 2
    where ID_RELATIE_CC = 5;
    commit;
end;
[2023-01-13 13:46:48] completed in 12 ms
Participant 1) id: 1002 nume: Francis Alexandru
Participant 2) id: 1005 nume: Ivanovici Mircea
```

## 10 + 11. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.

**Enunț:** Runderle dintr-o competiție se pot anula din cauza vremii. Însă, runderle care au două sau mai multe probe acvatice își desfășoară toate probele la un complex acoperit, asigurând desfășurarea probelor. Astfel, să se interzică, prin intermediul unui trigger, ștergerea unei runde cu două sau mai multe probe acvatice. În cazul în care s-a reușit ștergerea runde, să se afișeze probele șterse din cadrul acestei runde.

```
create or replace trigger rundaprobat2
for delete
```

```
on runda
referencing old as val_veche
compound trigger

n number;
m number;
TYPE tip_proba IS RECORD (
    id_proba proba.ID_PROBA%TYPE,
    id_runda proba.ID_RUNDA%TYPE,
    tip_proba proba.TIP_PROBA%TYPE
);
TYPE vect_p IS TABLE OF tip_proba INDEX BY PLS_INTEGER;
list_probe vect_p;
TYPE vect_q IS TABLE OF proba.ID_PROBA%TYPE INDEX BY PLS_INTEGER;
list_fin_probe vect_q;
no_acvatic exception;
before statement is
begin
    select p.ID_PROBA, p.ID_RUNDA, p.TIP_PROBA
    bulk collect into list_probe
    from PROBA p;
end before statement;

before each row is
begin
    DBMS_OUTPUT.PUT_LINE('Vreau sa sterg ' || :val_veche.id_runda);
    n := 0;
    m := 0;
    for i in list_probe.first..list_probe.last loop
        if :val_veche.ID_RUNDA = list_probe(i).id_runda then
            if 'acvatic' = list_probe(i).tip_proba then
                n := n + 1;
            end if;
            list_fin_probe(m) := list_probe(i).id_proba;
            m := m + 1;
        end if;
    end loop;

    if n > 1 then
        DBMS_OUTPUT.PUT_LINE('Stergere esuata!');
        RAISE_APPLICATION_ERROR(-20000,'Prea multe probe acvatice!');
    end if;

end before each row;

after statement is
begin
    DBMS_OUTPUT.PUT_LINE('S-a sters runda si cele ' || m || ' probe:');
    for i in list_fin_probe.first..list_fin_probe.last loop
        DBMS_OUTPUT.PUT(list_fin_probe(i) || ' ');
    end loop;
    DBMS_OUTPUT.PUT_LINE(' ');
end after statement;
end;

delete RUNDA where ID_RUNDA = 501;
rollback;
delete RUNDA where ID_RUNDA = 500;
```

```
delete RUNDIA where ID_RUNDIA = 501
[2023-01-12 04:19:01] 1 row affected in 7 ms
Vreau sa sterg 501
Nu sters runda si cele 3 probe:
4 2 5
rollback
[2023-01-12 04:19:09] completed in 8 ms
delete RUNDIA where ID_RUNDIA = 500
[2023-01-12 04:19:11] [72000][10000]
[2023-01-12 04:19:11] ORA-20000: Prea multe probe activate!
[2023-01-12 04:19:11] ORA-04012: at "DAVIDSGEO.RUNDAPROBAT2", line 39
[2023-01-12 04:19:11] ORA-04089: error during execution of trigger "DAVIDSGEO.RUNDAPROBAT2"
[2023-01-12 04:19:11] Position: 7
Vreau sa sterg 500
Stergere esuata!
```

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

**Enunț:** Înainte de crearea sau ștergerea unui obiect din baza de date, să se verifice dacă toate constrângerile FK sunt activate. În caz contrar, să se declanșeze o eroare în care să se menționeze numele constrângerii și tabela pe care se află constrângerea dezactivată.

```
create or replace trigger verifyfks
before create or drop on database
declare
    v_tabela VARCHAR2(50);
    v_contrangere VARCHAR2(30);
    v_status VARCHAR2(10);
BEGIN
    for i in (select table_name, constraint_name, status
              from user_constraints
              where constraint_type = 'R') loop
        /*R = referential integrity*/
        v_tabela := i.table_name;
        v_contrangere := i.constraint_name;
        v_status := i.status;

        if v_status <> 'ENABLED' then
            RAISE_APPLICATION_ERROR(-20000, 'Constrangerea fk ' || v_contrangere || ' pe tabela ' || v_tabela ||
            ' nu este activat.');
```

```
        end if;
    end loop;
end;

/*TESTARE CREARE CU ARUNCAREA ERORII*/

create table ii
(
    v varchar2(30)
    constraint iipk
        primary key
);

create table iii
(
    vi varchar2(50)
        constraint iipk
            primary key,
    vii varchar2(30)
);

alter table iii
add CONSTRAINT iifk
    FOREIGN KEY (vii)
        REFERENCES ii(v)
    ON DELETE SET NULL;
```

## SGBD – Grupa 251

### Pătrânel David-George

```
alter table iii
disable constraint iifk;

drop table iii;
```

```
DAVIDSGBD> drop table iii
[2023-01-12 22:35:14] [42000][4088]
[2023-01-12 22:35:14] ORA-04088: error during execution of trigger 'DAVIDSGBD.VERIFYFKS'
[2023-01-12 22:35:14] ORA-00604: error occurred at recursive SQL level 1
[2023-01-12 22:35:14] ORA-20000: Constrangerea fk IIFK pe tabela III nu este activat.
[2023-01-12 22:35:14] ORA-06512: at line 15
[2023-01-12 22:35:14] ORA-06512: at line 15
[2023-01-12 22:35:14] Position: 0
```

```
/*TESTARE STERGERE FARA ARUNCARE DE EROARE*/
```

```
alter table iii
enable constraint iifk;
```

```
drop table iii;
drop table ii;
```

```
DAVIDSGBD> alter table iii
            enable constraint iifk
[2023-01-12 22:35:41] completed in 13 ms
DAVIDSGBD> drop table iii
[2023-01-12 22:35:41] completed in 17 ms
DAVIDSGBD> drop table ii
[2023-01-12 22:35:41] completed in 17 ms
```

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

```
create or replace package catei13 is
    PROCEDURE findLeastNActiveDog
        (v_nrcomp NUMBER DEFAULT 1);
    PROCEDURE proc1
        (v_in NUMBER DEFAULT 0);
    FUNCTION dogs2022
        (vin_rasa IN VARCHAR2,
         vin_idcomp IN NUMBER)
        RETURN NUMBER;
    PROCEDURE participant_exp
        (vin_n in number,
         vin_name in varchar2);
end catei13;
```

```
create or replace package body catei13 is
    PROCEDURE findLeastNActiveDog
        (v_nrcomp NUMBER DEFAULT 1)
    IS
        /*Record*/
        type dog_info is record (
            id_catel CATEL.id_catel%TYPE,
            nume_catel CATEL.nume_catel%TYPE,
            id_comp RELATIE_CATEL_COMP.id_competitie%TYPE
        );
        /*Tablou imbricat*/
        type tablou_imbricat is table of dog_info;
```

```
v_list_dogs tablou_imbricat := tablou_imbricat();
v_old_dog CATEL.id_catel%TYPE;
v_nume CATEL.nume_Catel%TYPE;
/*Tablou indexat*/
type tablou_indexat is table of competitie.id_competitie%TYPE index by pls_integer;
v_list_comps tablou_indexat;
v_allcomps number;
/*Vector*/
type vector is varray(150) of catel.id_catel%TYPE;
v_list_dogsids vector := vector();
i number;
j number;
minim number;
contor number;
too_many_comps exception;
too_many_comps_traseu exception;
BEGIN
select count(*)
into v_allcomps
from COMPETITIE;

if(v_nrcomp > v_allcomps) then
    raise too_many_comps;
end if;

select CAUX.ID_COMPETITIE
bulk collect into v_list_comps
from COMPETITIE CAUX
join (select C.ID_COMPETITIE, count(*) as nr_probe_tr
      from COMPETITIE C
      join RUNDA R2 on C.ID_COMPETITIE = R2.ID_COMPETITIE
      join PROBA P on R2.ID_RUNDA = P.ID_RUNDA
      where Lower(p.TIP_PROBA) = 'traseu'
      GROUP BY C.ID_COMPETITIE, C.NUME_COMPETITIE) D on D.ID_COMPETITIE = CAUX.ID_COMPETITIE
order by D.nr_probe_tr desc, CAUX.NUME_COMPETITIE;

if(v_nrcomp > v_list_comps.COUNT) then
    raise too_many_comps_traseu;
end if;

select c2.ID_CATEL, c2.NUME_CATEL, rcc.ID_COMPETITIE
bulk collect into v_list_dogs
from RELATIE_CATEL_COMP rcc
join CATEL C2 on C2.ID_CATEL = rcc.ID_CATEL
order by c2.ID_CATEL;

v_old_dog := v_list_dogs(1).id_catel;
contor := 0;
minim := v_list_dogs.count;
for i in v_list_dogs.first..v_list_dogs.last loop
    if(v_old_dog <> v_list_dogs(i).id_catel) then
        if(minim = contor and contor <> 0) then
            v_list_dogsids.extend;
            v_list_dogsids(v_list_dogsids.last) := v_old_dog;
        elsif(minim > contor and contor <> 0) then
            v_list_dogsids.delete;
            v_list_dogsids.extend;
            v_list_dogsids(v_list_dogsids.last) := v_old_dog;
            minim := contor;
        end if;
        v_old_dog := v_list_dogs(i).id_catel;
```

```

        contor:= 0;
    end if;
    for j in 1..v_nrcomp loop
        if(v_list_dogs(i).id_comp = v_list_comps(j)) then
            contor := contor + 1;
        end if;
    end loop;
end loop;
if(minim = contor and contor <> 0) then
    v_list_dogsids.extend;
    v_list_dogsids(v_list_dogsids.last) := v_old_dog;
elsif(minim > contor and contor <> 0) then
    v_list_dogsids.delete;
    v_list_dogsids.extend;
    v_list_dogsids(v_list_dogsids.last) := v_old_dog;
    minim := contor;
end if;

for i in v_list_dogsids.first..v_list_dogsids.last loop
    select NUME_CATEL
    into v_nume
    from CATEL
    where ID_CATEL = v_list_dogsids(i);

    if(v_nume is not null) then
        DBMS_OUTPUT.PUT_LINE('Catel ' || v_nrcomp || '-activ: id = ' || v_list_dogsids(i) || ' nume = '
|| v_nume);
    else
        DBMS_OUTPUT.PUT_LINE('Catel ' || v_nrcomp || '-activ: id = ' || v_list_dogsids(i) || ' fara
nume');
    end if;
end loop;

EXCEPTION
    when too_many_comps then
        RAISE_APPLICATION_ERROR(-20001,'Prea multe competitii cerute');
    when too_many_comps_traseu then
        RAISE_APPLICATION_ERROR(-20002,'Prea multe competitii tip traseu cerute');
    when TOO_MANY_ROWS then
        RAISE_APPLICATION_ERROR(-20003,'Prea multi catei gasiti cu acelasi id');
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20004,'Nu s-au gasit datele');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20000,'Alta eroare!');

END findLeastNActiveDog;

PROCEDURE proc1
(v_in NUMBER DEFAULT 0)
IS
    contor number;
    flag number := 0;
    cursor c is
        select P.ID_PROPRIETAR, count(*) nr_dogs
        from PROPRIETAR P
        join CATEL C2 on P.ID_PROPRIETAR = C2.ID_PARTICIPANT
        group by P.ID_PROPRIETAR
        having count(*) >= v_in;

```

```
cursor u(v_idowner PROPRIETAR.id_proprietar%type) is
select *
from CATEL
where ID_PARTICIPANT = v_idowner and ID_CATEL in (
    select ID_CATEL
    from RELATIE_CATEL_COMP
    join COMPETITIE C3 on C3.ID_COMPETITIE = RELATIE_CATEL_COMP.ID_COMPETITIE
    where c3.PREMIU_COMPETITIE >= 500000
)
for update of VACCINAT nowait ;
no_owners exception;
BEGIN
for owner in c loop
    flag := 1;
    contor := 0;

    for j in u(owner.ID_PROPRIETAR) loop
        if(lower(j.VACCINAT) <> 't' or j.VACCINAT is null) then
            contor := contor + 1;

            update CATEL
            set VACCINAT = 'T'
            where current of u;
        end if;
    end loop;

    DBMS_OUTPUT.PUT('Proprietarul ' || owner.ID_PROPRIETAR);
    if(contor = 0) then
        DBMS_OUTPUT.PUT_LINE(' nu si-a vaccinat niciun catei');
    else
        DBMS_OUTPUT.PUT_LINE(' si-a vaccinat ' || contor || ' catei');
    end if;

    rollback;
end loop;
if(flag = 0) then
    raise no_owners;
end if;

EXCEPTION
when INVALID_CURSOR then
    RAISE_APPLICATION_ERROR(-20001, 'Cursorul este inchis');
when CURSOR_ALREADY_OPEN then
    RAISE_APPLICATION_ERROR(-20002, 'Cursor deja deschis');
WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20003, 'Nu s-au gasit date');
WHEN no_owners THEN
    RAISE_APPLICATION_ERROR(-20004, 'Nu sunt proprietari cu acest nr minim de catei');
WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20000, 'Alta eroare!');
END proc1;

FUNCTION dogs2022
(vin_rasa IN VARCHAR2,
vin_idcomp IN NUMBER)
RETURN NUMBER
IS
    nr_err number;
    ans number;
    aux number;
    my_no_data_found exception;
```



```
no_dogs_to_vaccinate exception;
vals_modif number := 0;
type tablou is table of catel.ID_CATEL%TYPE;
v_list_dogs tablou := tablou();
begin

    select count(*)
    into aux
    from CATEL c2
    where lower(c2.RASA) = lower(vin_rasa);
    if aux = 0 then
        nr_err := 1;
        raise my_no_data_found;
    end if;

    select count(*)
    into aux
    from COMPETITIE c3
    where C3.ID_COMPETITIE = vin_idcomp;
    if aux = 0 then
        nr_err := 2;
        raise my_no_data_found;
    end if;

    select count(*)
    into aux
    from COMPETITIE c3
    join RUNDA R2 on C3.ID_COMPETITIE = R2.ID_COMPETITIE
    where C3.ID_COMPETITIE = vin_idcomp and
        TO_CHAR(r2.DATA_RUNDA, 'YYYY') = 2021;
    if aux = 0 then
        nr_err := 3;
        raise my_no_data_found;
    end if;

    select distinct C2.ID_CATEL
    bulk collect into v_list_dogs
    from RELATIE_CATEL_COMP
    join CATEL C2 on C2.ID_CATEL = RELATIE_CATEL_COMP.ID_CATEL
    join COMPETITIE C3 on C3.ID_COMPETITIE = RELATIE_CATEL_COMP.ID_COMPETITIE
    join RUNDA R2 on C3.ID_COMPETITIE = R2.ID_COMPETITIE
    where lower(c2.RASA) = lower(vin_rasa) and
        C3.ID_COMPETITIE = vin_idcomp and
        TO_CHAR(r2.DATA_RUNDA, 'YYYY') = 2021;

    for i in v_list_dogs.first..v_list_dogs.last loop
        DBMS_OUTPUT.PUT_LINE(v_list_dogs(i));
        update CATEL C
        set C.GREUTATE = C.GREUTATE + 0.5
        where (lower(C.VACCINAT) = 'f' or C.VACCINAT is null) and C.ID_CATEL = v_list_dogs(i);
        vals_modif := vals_modif + SQL%ROWCOUNT;
    end loop;

    if vals_modif = 0 then
        raise no_dogs_to_vaccinate;
    end if;

    return vals_modif;
exception
    WHEN my_no_data_found THEN
```

```
        if nr_err = 1 then
            DBMS_OUTPUT.PUT_LINE('Nu s-a gasit niciun catel cu rasa ' || vin_rasa);
        elsif nr_err = 2 then
            DBMS_OUTPUT.PUT_LINE('Nu s-a gasit nicio competitie cu idul ' || vin_idcomp);
        elsif nr_err = 3 then
            DBMS_OUTPUT.PUT_LINE('Nu sunt runde in anul 2021 in competitia ' || vin_idcomp);
        end if;
        RETURN -20001;
    WHEN no_dogs_to_vaccinate THEN
        DBMS_OUTPUT.PUT_LINE('No dogs need vaccination');
        RETURN -20002;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Alta eroare');
        DBMS_OUTPUT.PUT_LINE('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE('Mesajul erorii: ' || SQLERRM);
        RETURN -20000;
end dogs2022;

PROCEDURE participant_exp
(vin_n in number,
 vin_name in varchar2)
IS
    nr_err number;
    ind number := 0;
    id_prop PARTICIPANT_EXPERIMENTAT.ID_PARTICIPANT%TYPE;
    type tablou is table of PARTICIPANT_EXPERIMENTAT.ID_PARTICIPANT%TYPE;
    v_list_part tablou := tablou();
    type date_part is record(
        id_proprietar proprietar.id_proprietar%TYPE,
        nume_proprietar proprietar.nume_participant%TYPE,
        prenume_proprietar proprietar.prenume_participant%TYPE
    );
    type tablou21 is table of date_part;
    v_list_part_fin tablou21 := tablou21();
    type tablou2 is table of date_part INDEX BY PLS_INTEGER;
    v_list_part_aux tablou2;
    type tablou3 is table of jurat.nume_jurat%TYPE INDEX BY PLS_INTEGER;
    v_aux_jurat tablou3;

    negative exception;
    my_no_data_found exception;
    no_participants_incase exception;
    multi_jurati exception;

BEGIN
    if vin_n < 0 then
        raise negative;
    end if;

    select NUME_JURAT
    bulk collect into v_aux_jurat
    from JURAT
    where NUME_JURAT = vin_name;

    if sql%rowcount > 1 then
        raise multi_jurati;
    elsif sql%notfound then
        nr_err := 1;
        raise my_no_data_found;
    end if;
```

```

select PE.ID_PARTICIPANT
bulk collect into v_list_part
from PARTICIPANT_EXPERIMENTAT PE
join (
    select i.ID_PARTICIPANT_EXP, MAX(DATA_COMPETITIE) last_comp
    from ISTORIC i
    group by i.ID_PARTICIPANT_EXP

) aux on PE.ID_PARTICIPANT = aux.ID_PARTICIPANT_EXP
where ROUND(MONTHS_BETWEEN(SYSDATE, aux.last_comp)) <= vin_n * 12;

if v_list_part.count = 0 then
    nr_err := 2;
    raise my_no_data_found;
end if;

for i in v_list_part.first..v_list_part.last loop
    id_prop := v_list_part(i);

    select
        P.ID_PROPRIETAR,
        P.NUME_PARTICIPANT,
        P.PRENUME_PARTICIPANT
    bulk collect into v_list_part_aux
    from PROPRIETAR P
    where P.ID_PROPRIETAR = id_prop and P.ID_PROPRIETAR in(
        select distinct
            C1.ID_PARTICIPANT
        from RELATIE_CATEL_COMP R
        join COMPETITIE C2 on R.ID_COMPETITIE = C2.ID_COMPETITIE
        join CATEL C1 on R.ID_CATEL = C1.ID_CATEL
        where C2.ID_COMPETITIE in(
            select
                C3.ID_COMPETITIE
            from COMPETITIE C3
            join JURAT J on C3.ID_COMPETITIE = J.ID_COMPETITIE
            where Lower(J.NUME_JURAT) = Lower(vin_name))
    );

    if SQL%ROWCOUNT = 1 then
        v_list_part_fin.extend();
        v_list_part_fin(v_list_part_fin.last) := v_list_part_aux(v_list_part_aux.first);
    end if;
end loop;

if v_list_part_fin.count = 0 then
    raise no_participants_incise;
end if;

for i in v_list_part_fin.first..v_list_part_fin.last loop
    ind := ind + 1;
    DBMS_OUTPUT.PUT_LINE('Participant ' || ind || ') id: ' || v_list_part_fin(i).id_proprietar || '
nume: '
        || v_list_part_fin(i).nume_proprietar || ' ' || v_list_part_fin(i).prenume_proprietar);
end loop;

EXCEPTION
    WHEN my_no_data_found THEN
        if nr_err = 1 then
            DBMS_OUTPUT.PUT_LINE('Nu exista jurat cu numele ' || vin_name);

```

```
        elsif nr_err = 2 then
            DBMS_OUTPUT.PUT_LINE('Nu sunt participanti experimentati care au participat la ultimul
concurș cu cel mult ' || vin_n || ' ani');
        end if;
        RAISE_APPLICATION_ERROR(-20001, 'Nu s-au gasit date');
    WHEN no_participants_incase THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun participant experimentat care sa respecte toate
conditiile');
        RAISE_APPLICATION_ERROR(-20002, 'Nu exista niciun participant experimentat care sa respecte
toate conditiile');
    WHEN multi_jurati THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multi jurati cu acest nume');
        RAISE_APPLICATION_ERROR(-20003, 'Exista mai multi jurati cu acest nume!');
    WHEN negative THEN
        DBMS_OUTPUT.PUT_LINE('Numarul de ani este unul pozitiv');
        RAISE_APPLICATION_ERROR(-20004, 'Numarul de ani trebuie sa fie unul pozitiv!');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Alta eroare');
        DBMS_OUTPUT.PUT_LINE ('Codul erorii: ' || SQLCODE);
        DBMS_OUTPUT.PUT_LINE ('Mesajul erorii: ' || SQLERRM);
        RAISE_APPLICATION_ERROR(-20000, 'Alta eroare!');
END participant_exp;
end catei13;
/
declare
    v_x number := &p;
begin
    catei13.findLeastNActiveDog(2);

    catei13.proc1(v_x);

    UPDATE CATEL
    SET VACCINAT = 'F'
    WHERE ID_CATEL = 10011;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Result ' || catei13.dogs2022('Bison', 3));
    COMMIT;
    UPDATE CATEL
    SET VACCINAT = 'T'
    WHERE ID_CATEL = 10011;
    COMMIT;

    update RELATIE_CATEL_COMP
    set ID_COMPETITIE = 4
    where ID_RELATIE_CC = 5;
    commit;

    catei13.participant_exp(30, 'Matei');

    update RELATIE_CATEL_COMP
    set ID_COMPETITIE = 2
    where ID_RELATIE_CC = 5;
    commit;
end;
```

```
[2023-01-13 13:56:04] completed in 54 ms
Catei 2-activ: id = 10011 nume = Pecky
Catei 2-activ: id = 10012 nume = Ruffus
Catei 2-activ: id = 10021 nume = Albert
Catei 2-activ: id = 10041 nume = Pecky
Catei 2-activ: id = 10051 fara nume
Catei 2-activ: id = 10062 nume = Lilly
Proprietarul 1001 nu si-a vaccinat niciun catei
Proprietarul 1006 si-a vaccinat 2 catei
10011
10031
Result 1
Participant 1) id: 1002 nume: Francis Alexandru
Participant 2) id: 1005 nume: Ivanovici Mircea
```

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

##### Enunț:

Mi-am propus să simulez atribuirea unui echipament unei competiții, cu anumite condiții. O competiție poate să primească un echipament nou doar dacă are doi sau mai mulți sponsori. În cazul în care s-a reușit atribuirea echipamentului la competiție, ne dorim să afișăm care este cel mai influent sponsor pentru acea competiție (sponsorul care apare pe cele mai multe materiale publicitare în acea competiție).

Totodată, pentru acest sponsor, dacă este unic pentru competiție, vrem să știm numărul total de materiale publicitare pe care apare, iar dacă nu este unic, să afișăm tipul de materiale promoționale și numărul de exemplare pe care apare acest sponsor.

```
create or replace package catei14 is
  PROCEDURE init;
  PROCEDURE add_equipment
    (v_id_equip IN relatie_ech_comp.id_echipament%TYPE,
     v_id_competitie IN relatie_ech_comp.id_competitie%TYPE);
  FUNCTION adaugare_valida
    (v_id_competitie IN relatie_ech_comp.id_competitie%TYPE,
     v_id_sponsor_max OUT sponsor.id_sponsor%TYPE)
    RETURN BOOLEAN;
  PROCEDURE afisare_detalii_sponsor
    (v_id_sponsor IN sponsor.id_sponsor%TYPE);
  FUNCTION verif_sponsor
    (v_id_sponsor IN sponsor.id_sponsor%TYPE)
    RETURN BOOLEAN;
end catei14;
/
create or replace package body catei14 is
  initializare boolean := false;
  type sponsor_data is record(
    id_competitie promoveaza_sponsor.id_competitie%TYPE,
    id_sponsor promoveaza_sponsor.id_sponsor%TYPE,
    nr_publ number
  );
  type tablou1 is table of sponsor_data;
  type tablou11 is table of tablou1;
  v_init_comps tablou11 := tablou11();

  PROCEDURE init
  IS
    v_sponsori tablou1 := tablou1();
    id_comp number;
  BEGIN
```

```
for j in (select ID_COMPETITIE
           from COMPETITIE) loop
    id_comp := j.ID_COMPETITIE;

    select ID_COMPETITIE, ID_SPONSOR, COUNT(*) freqv_sponsor
    bulk collect into v_sponsori
    from PROMOVEAZA_SPONSOR
    group by ID_COMPETITIE, ID_SPONSOR
    having ID_COMPETITIE = id_comp
    order by freqv_sponsor desc, ID_SPONSOR;

    if sql%rowcount >= 1 then
        v_init_comps.extend();
        v_init_comps(v_init_comps.last) := v_sponsori;
    end if;

end loop;

/*
for i in v_init_comps.first..v_init_comps.last loop
    for j in v_init_comps(i).first..v_init_comps(i).last loop
        DBMS_OUTPUT.PUT_LINE(v_init_comps(i)(j).id_competitie || ' ' || v_init_comps(i)(j).id_sponsor || ' '
|| v_init_comps(i)(j).nr_publ);
    end loop;
    DBMS_OUTPUT.PUT_LINE(' ');
end loop;*/
initializare := true;
END init;

PROCEDURE add_equipment
(v_id_equip IN relatie_ech_comp.id_echipament%TYPE,
v_id_competitie IN relatie_ech_comp.id_competitie%TYPE)
IS
    nr_err number;
    aux_comp_id competitie.id_competitie%TYPE;
    aux_echipament_id echipament.id_echipament%TYPE;
    type tablou2 is table of RELATIE_ECH_COMP.id_relatie_ec%TYPE;
    v_echs tablou2 := tablou2();
    last_id number;
    id_sponsor_max sponsor.id_sponsor%TYPE;

    echipament_existent exception;
    venit_insuficient exception;
    fara_initializare exception;
    sponsori_insuficienti exception;
BEGIN
    initializare := false;
    init();

    if not initializare then
        raise fara_initializare;
    end if;

    nr_err := 1;
    select ID_COMPETITIE
    into aux_comp_id
    from COMPETITIE
    where ID_COMPETITIE = v_id_competitie;

    nr_err := 2;
    select ID_ECHIPAMENT
```

```
into aux_echipament_id
from ECHIPAMENT
where ID_ECHIPAMENT = v_id_equip;

select ID_RELATIE_EC
bulk collect into v_echs
from RELATIE_ECH_COMP
where ID_ECHIPAMENT = v_id_equip and ID_COMPETITIE = v_id_competitie;

if SQL%ROWCOUNT >=1 THEN
    raise echipament_existent;
end if;

if not adaugare_valida(v_id_competitie, id_sponsor_max) then
    raise sponsori_insuficienti;
end if;

select nvl(max(ID_RELATIE_EC), 0)
into last_id
from RELATIE_ECH_COMP r;

insert into RELATIE_ECH_COMP values
(last_id + 1, v_id_equip, v_id_competitie);
DBMS_OUTPUT.PUT_LINE('Echipament adaugat la competitie');
DBMS_OUTPUT.PUT_LINE('Cel mai mare sponsor este ' || id_sponsor_max);
afisare_detalii_sponsor(id_sponsor_max);
rollback;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        if nr_err = 1 then
            DBMS_OUTPUT.PUT_LINE('Nu exista competitie cu id = ' || v_id_competitie);
        elsif nr_err = 2 then
            DBMS_OUTPUT.PUT_LINE('Nu exista echipamentul cu id = ' || v_id_equip);
        end if;
        RAISE_APPLICATION_ERROR(-20001, 'Nu s-au gasit date');
    WHEN echipament_existent THEN
        DBMS_OUTPUT.PUT_LINE('Competitia ' || v_id_competitie || ' are deja echipamentul ' ||
v_id_equip);
        RAISE_APPLICATION_ERROR(-20002, 'Competitia are acest echipament!');
    WHEN fara_initalizare THEN
        DBMS_OUTPUT.PUT_LINE('A aparut o eroare la initializare');
        RAISE_APPLICATION_ERROR(-20003, 'Eroare la initializare!');
    WHEN sponsori_insuficienti THEN
        DBMS_OUTPUT.PUT_LINE('Competitia nu are mai mult de un sponsor');
        RAISE_APPLICATION_ERROR(-20004, 'Sponsori insuficienti!');

END add_equipment;

FUNCTION adaugare_valida
(v_id_competitie IN relatie_ech_comp.id_competitie%TYPE,
v_id_sponsor_max OUT sponsor.id_sponsor%TYPE)
RETURN BOOLEAN
IS
    v_nr_sponsori number;
    flag boolean := true;
    poz number;

    no_sponsors exception;
```

```
BEGIN
    for i in v_init_comps.first..v_init_comps.last loop
        if v_init_comps(i)(v_init_comps(i).first).id_competitie = v_id_competitie then
            v_nr_sponsori := v_init_comps(i).count;
            flag := false;
            poz := i;
            exit;
        end if;
    end loop;

    if flag then
        return false;
    end if;

    if v_nr_sponsori < 2 then
        return false;
    end if;

    v_id_sponsor_max := v_init_comps(poz)(v_init_comps(poz).FIRST).id_sponsor;
    return true;
EXCEPTION
    WHEN no_sponsors THEN
        DBMS_OUTPUT.PUT_LINE('Competitia ' || v_id_competitie || ' nu are niciun sponsor.');
```

return false;

```
END adaugare_valida;
```

PROCEDURE afisare\_detalii\_sponsor  
(v\_id\_sponsor IN sponsor.id\_sponsor%TYPE)

IS

TYPE pubref IS REF CURSOR;  
v\_pub pubref;  
v\_tip\_publ PUBLICITATE.TIP\_PUBLICITATE%TYPE;  
v\_nr\_exec PUBLICITATE.NR\_EXEMPLARE%TYPE;  
suma number := 0;  
flag boolean;

BEGIN

if verif\_sponsor(v\_id\_sponsor) then  
 flag := true;  
else  
 flag := false;  
end if;

/\*cursor deschis cu ajutorul unui șir dinamic\*/  
OPEN v\_pub FOR  
 'SELECT P.TIP\_PUBLICITATE, P.NR\_EXEMPLARE ' ||  
 'FROM PUBLICITATE P ' ||  
 'JOIN( ' ||  
 'SELECT DISTINCT ID\_PUBLICITATE, ID\_SPONSOR ' ||  
 'FROM PROMOVEAZA\_SPONSOR ' ||  
 'WHERE ID\_SPONSOR = :bind\_var) aux ON P.ID\_PUBLICITATE = aux.ID\_PUBLICITATE'  
USING v\_id\_sponsor;

loop  
 fetch v\_pub into v\_tip\_publ, v\_nr\_exec;  
 exit when v\_pub%notfound;  
 if flag then  
 suma := suma + v\_nr\_exec;  
 else  
 DBMS\_OUTPUT.PUT\_LINE(v\_tip\_publ || ' cu ' || v\_nr\_exec || ' exemplare.');

end if;  
end loop;



```
        if flag then
            DBMS_OUTPUT.PUT_LINE('Sponsorul ' || v_id_sponsor || ' apare pe ' || suma || ' materiale.');
```

end if;

END afisare\_detalii\_sponsor;

FUNCTION verif\_sponsor  
 (v\_id\_sponsor IN sponsor.id\_sponsor%TYPE)  
 RETURN BOOLEAN  
IS  
 nr number;  
 BEGIN  
 select count(\*)  
 into nr  
 from(  
 select distinct ID\_COMPETITIE, ID\_SPONSOR  
 from PROMOVEAZA\_SPONSOR  
 where ID\_SPONSOR = v\_id\_sponsor);

if nr = 1 then  
 return true;  
 else  
 return false;  
 end if;

END verif\_sponsor;

end catei14;  
/  
  
begin  
 insert into PROMOVEAZA\_SPONSOR  
 values (18, 10, 4, 605);  
 insert into PROMOVEAZA\_SPONSOR  
 values (19, 12, 4, 605);  
 commit;  
 catei14.add\_equipment(102, 4);  
end;  
/

```
DAVID@SGBD> begin
                catei14.add_equipment(101, 5);
            end;
[2023-01-13 20:44:18] completed in 9 ms
Echipament adaugat la competitie
Cel mai mare sponsor este 601
Poster A2 cu 50 exemplare.
Flyere cu 200 exemplare.
```

```
DAVIDSGBD> begin
    insert into PROMOVEAZA_SPONSOR
    values (18, 10, 4, 605);
    insert into PROMOVEAZA_SPONSOR
    values (19, 12, 4, 605);
    commit;
    catei14.add_equipment(102, 4);
end;
[2023-01-13 21:14:47] completed in 9 ms
Echipament adaugat la competitie
Cel mai mare sponsor este 605
Sponsorul 605 apare pe 73 materiale.
```

```
DAVIDSGBD> begin
    catei14.add_equipment(109, 5);
    /*catei14.add_equipment(101, 5);*/
end;
[2023-01-13 20:49:01] [72000][20001]
[2023-01-13 20:49:01] ORA-20001: Nu s-au gasit date
[2023-01-13 20:49:01] ORA-06512: at "DAVIDSGBD.CATEI14", line 104
[2023-01-13 20:49:01] ORA-06512: at line 2
[2023-01-13 20:49:01] Position: 0
Nu exista echipamentul cu id = 109
```