Gestiunea unui Turneu de Tenis

Danciu Cosmin Alexandru
Seria 25, Grupa 251

Anul Universitar 2025-2026
Cuprins

Introducere

Tema aleasă pentru proiect este “Gestiunea unui turneu de tenis”. Modelul de date va gestiona informații legate de organizarea și desfășurarea unui turneu de tenis.

Am optat pentru această temă deoarece tenisul de câmp este un sport care implică un volum mare de date interconectate și reguli stricte de organizare, fiind un candidat ideal pentru modelarea într-o bază de date relațională.

Sistemul propus are ca scop digitalizarea și centralizarea informațiilor necesare desfășurării competițiilor, acoperind aspecte logistice (locații, programarea meciurilor pe zile), resurse umane (jucători, arbitri, staff, voluntari) și comerciale (gestionarea biletelor și a spectatorilor). Baza de date permite menținerea integrității datelor (de exemplu: un meci nu poate exista fără jucători sau fără un arbitru, iar un bilet trebuie să fie valid pentru o zi existentă a turneului), eliminând redundanța și facilitând interogarea rapidă a rezultatelor și a statisticilor.

Infrastructura utilizată:

- **Versiune SGBD:** Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
- **Interfața de dezvoltare (IDE):** Visual Studio Code + Oracle SQL Developer
- **Sistem de operare:** Microsoft Windows 11
- **Utilizarea unei mașini virtuale:** NU

Cerința 1

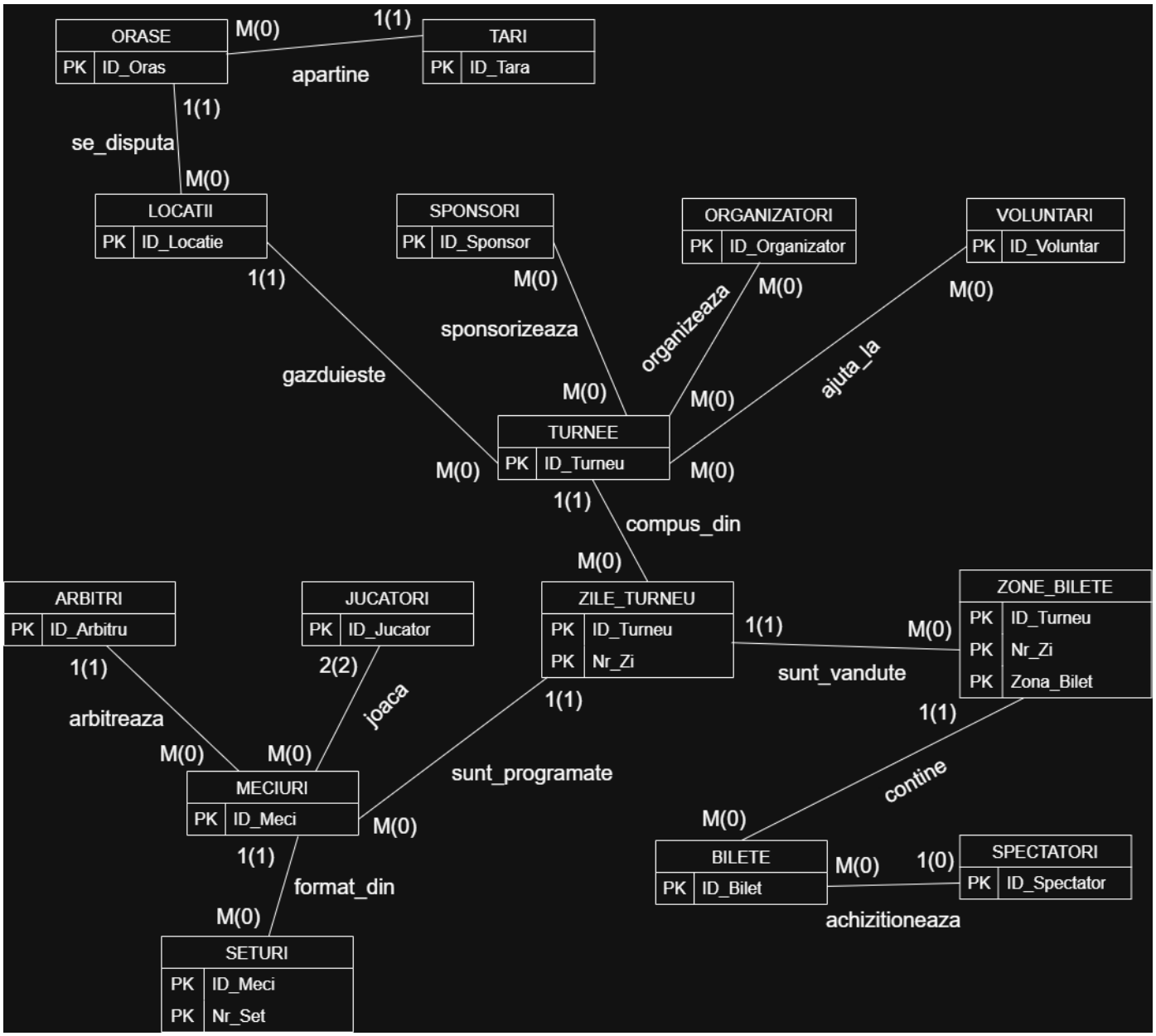
Prezentați pe scurt baza de date (utilitatea ei).

Această bază de date a fost creată pentru a ușura munca organizatorilor de turnee de tenis, punând la un loc toate informațiile necesare desfășurării competiției. Principalul ei rol este să țină o evidență clară a jucătorilor și a meciurilor, dar face mai mult decât o simplă listă de rezultate. Sistemul este capabil să memoreze scorul în detaliu, set cu set, și să gestioneze corect situațiile reale din tenis, cum ar fi momentele în care un jucător se accidentează și abandonează sau situațiile în care nu se prezintă deloc la meci.

Pe lângă partea sportivă, aplicația rezolvă problemele legate de programare și logistică. Totul este organizat strict pe zile, astfel încât să se știe exact pe ce teren se joacă un meci și cine îl arbitrează, evitând astfel greșelile de programare. De asemenea, sistemul include o parte comercială care se ocupă de vânzarea biletelor către spectatori, permițând organizatorilor să vadă oricând câte locuri au mai rămas libere, dar ține și evidența sponsorilor și a voluntarilor care ajută la buna desfășurare a evenimentului.

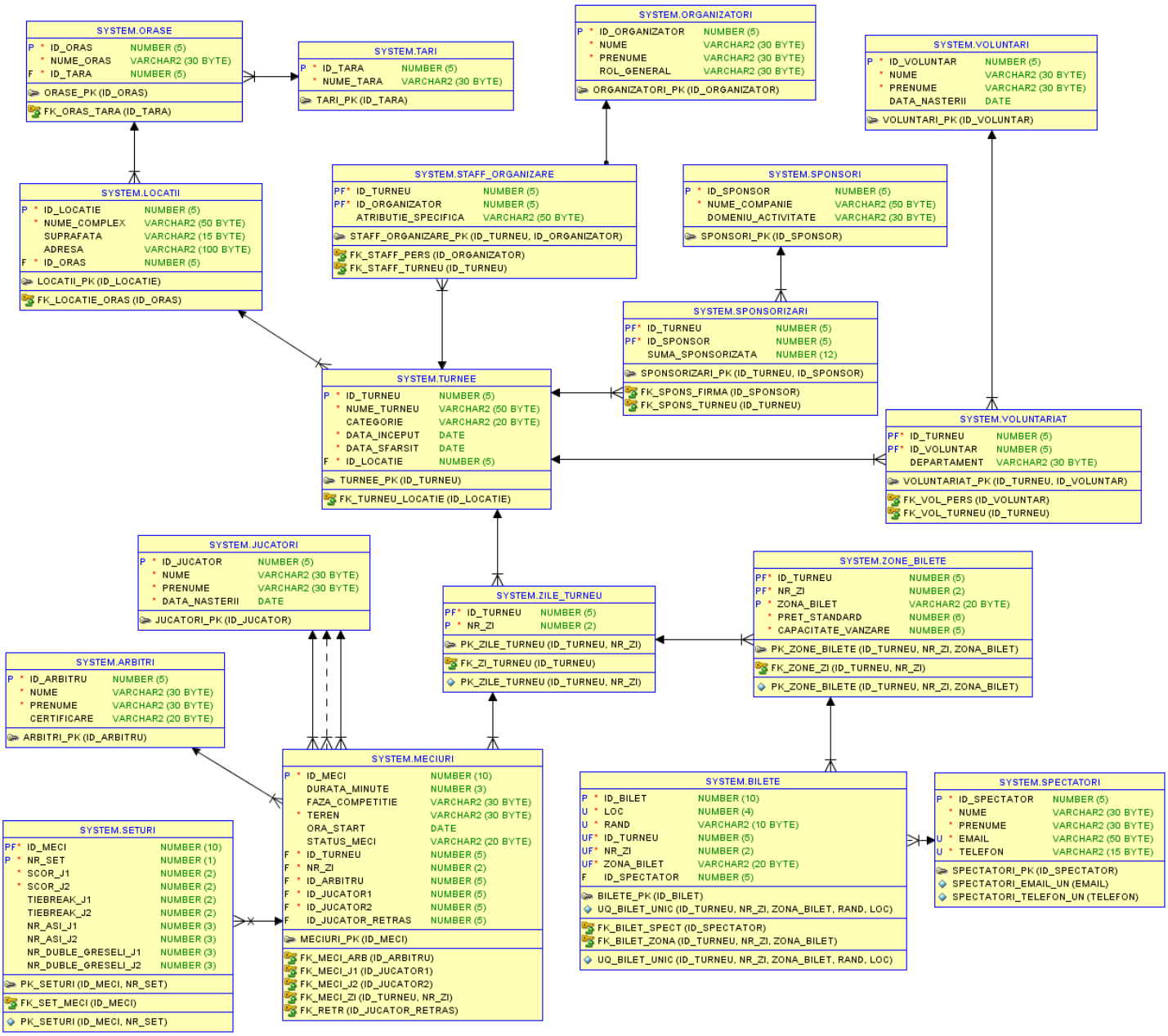
Cerința 2

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



Cerința 3

Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare: entitățile, relațiile și atributele trebuie definite în limba română.



Entitățile și atributele lor

- **TARI**: Lista țărilor unde au loc evenimentele.

    - ID_TARA: Numărul unic al țării.
    - NUME_TARA: Denumirea statului.
- **ORASE**: Localitățile din cadrul țărilor.
    - ID_ORAS: Numărul unic al orașului.
    - NUME_ORAS: Numele orașului.
    - ID_TARA: Legătura către țara de care aparține.
- **LOCATII**: Complexele sportive unde se țin turneele.
    - ID_LOCATIE: Codul unic al bazei sportive.
    - NUME_COMPLEX: Denumirea bazei.
    - SUPRAFATA: Tipul de teren (Zgura, Iarba, Hard).
    - ADRESA: Localizarea exactă.
    - ID_ORAS: Orașul în care se află baza.
- **JUCATORI**: Tenismenii care participă la meciuri.
    - ID_JUCATOR: Codul unic al sportivului.
    - NUME, PRENUME, DATA_NASTERII: Datele de identificare.
- **ARBITRI**: Cei care conduc meciurile.
    - ID_ARBITRU: Codul unic al arbitrului.
    - NUME, PRENUME, CERTIFICARE: Identitatea și gradul de calificare.
- **TURNEE**: Evenimentele mari (competițiile).
    - ID_TURNEU: Codul unic al turneului.
    - NUME_TURNEU: Titlul competiției.
    - CATEGORIE: Tipul (ex: Grand Slam, ATP 500).
    - DATA_INCEPUT, DATA_SFARSIT: Perioada turneului.
    - ID_LOCATIE: Locul unde se desfășoară tot turneul.
- **ZILE_TURNEU**: Segmentarea turneului pe zile (Ziua 1, Ziua 2...).
    - ID_TURNEU, NR_ZI: Identificarea zilei în cadrul unui turneu specific.
- **MECIURI**: Jocurile propriu-zise.
    - ID_MECI: Codul unic al meciului.
    - DURATA_MINUTE, FAZA_COMPETITIE, TEREN, ORA_START: Detalii logistice.
    - STATUS_MECI: Starea (Programat, In Desfasurare, Terminat etc.).

    - ID_TURNEU, NR_ZI: Ziua de turneu în care este programat.
    - ID_ARBITRU, ID_JUCATOR1, ID_JUCATOR2, ID_JUCATOR_RETRAS: Legăturile cu oamenii implicați.
- **SETURI**: Scorul detaliat pe fiecare set în parte.
    - ID_MECI, NR_SET: Identificarea setului pentru un meci anume.
    - SCOR_J1, SCOR_J2, TIEBREAK, ASI, DUBLE_GRESELI: Statisticile de joc.
- **SPONSORI / ORGANIZATORI / VOLUNTARI**: Entitățile care susțin și gestionează evenimentul.
    - ID_SPONSOR, ID_ORGANIZATOR, ID_VOLUNTAR: Codurile unice.
    - NUME_COMPANIE / NUME, PRENUME: Identificarea entităților.
- **SPONSORIZARI / STAFF_ORGANIZARE / VOLUNTARIAT**: Tabele care fac legătura între ajutoare și turnee.
    - Stochează cine, la ce turneu participă și ce rol sau sumă are.
- **SPECTATORI**: Oamenii care cumpără bilete.
    - ID_SPECTATOR, NUME, PRENUME, EMAIL, TELEFON: Datele de contact.
- **ZONE_BILETE**: Sectoarele din arenă și prețurile lor.
    - ID_TURNEU, NR_ZI, ZONA_BILET: Identificarea zonei într-o zi anume.
    - PRET_STANDARD, CAPACITATE_VANZARE: Detaliile comerciale.
- **BILETE**: Tichetele individuale vândute spectatorilor.
    - ID_BILET, LOC, RAND: Poziția pe scaun.
    - ID_SPECTATOR: Cine deține biletul.

Detalierea relațiilor

- **apartine (TARI - ORASE)**: Leagă orașele de țara lor.
- **se_disputa (ORASE - LOCATII)**: Indică orașul în care este construit complexul.
- **gazduieste (LOCATII - TURNEE)**: Indică complexul unde are loc un turneu întreg.
- **compus_din (TURNEE - ZILE_TURNEU)**: Împarte un turneu în zile calendaristice.

- **sunt_programate (ZILE_TURNEU - MECIURI)**: Stabilește în ce zi a turneului se joacă un meci.
- **arbitreaza (ARBITRI - MECIURI)**: Desemnează arbitrul pentru un meci.
- **joaca (JUCATORI - MECIURI)**: Leagă cei doi sportivi de meciul respectiv.
- **format_din (MECIURI - SETURI)**: Descompune scorul final în punctaj pe seturi.
- **sponsorizeaza / organizeaza / ajuta_la (SPONSORI/ORGANIZATORI/VOLUNTARI - TURNEE)**: Relațiile de suport pentru eveniment.
- **sunt_vandute (ZILE_TURNEU – ZONE_BILETE)**: Împarte biletele dintr-o zi pe zone.
- **contine (ZONE_BILETE - BILETE)**: Leagă scaunul individual de categoria de preț și ziua turneului.
- **achizitioneaza (SPECTATORI - BILETE)**: Atribuie biletul unei persoane.

Detalierea cardinalităților

- **TARI -> ORASE (1:M)**: O țară are mai multe orașe (M), dar un oraș aparține de o singură țară (1).
- **ORASE -> LOCATII (1:M)**: Un oraș poate avea mai multe complexe sportive (M), dar un complex se află într-un singur oraș (1).
- **LOCATII -> TURNEE (1:M)**: Într-o locație se pot ține mai multe turnee (M), dar un turneu se desfășoară într-o singură locație (1).
- **TURNEE -> ZILE_TURNEU (1:M)**: Un turneu are mai multe zile de concurs (M), dar o zi aparține unui singur turneu (1).
- **ZILE_TURNEU -> MECIURI (1:M)**: Într-o zi de turneu se joacă mai multe meciuri (M), dar un meci este programat într-o singură zi (1).
- **MECIURI -> SETURI (1:M)**: Un meci are mai multe seturi (M), dar un set aparține unui singur meci (1).
- **ARBITRI -> MECIURI (1:M)**: Un arbitru poate arbitra mai multe meciuri (M), dar un meci are un singur arbitru principal (1).

- **JUCATORI -> MECIURI (2:M)**: Un meci are obligatoriu 2 jucători, iar un jucător poate juca în mai multe meciuri pe parcursul turneului (M).
- **ZILE_TURNEU -> ZONE_BILETE (1:M)**: În fiecare zi de turneu există mai multe zone de preț (M), dar o zonă dintr-o zi aparține acelei zile (1).
- **ZONE_BILETE -> BILETE (1:M)**: O zonă conține mai multe locuri/bilete (M), dar un bilet aparține unei singure zone (1).
- **SPECTATORI -> BILETE (1:M)**: Un spectator poate cumpăra mai multe bilete (M), dar un bilet este deținut de un singur spectator (1) sau de niciunul dacă nu e vândut (0).
- **SPONSORI/ORGANIZATORI/VOLUNTARI -> TURNEE (M:M)**: Relații de tip mulți-la-mulți. Un sponsor poate susține mai multe turnee, iar un turneu poate avea mai mulți sponsori. Un organizator poate organiza mai multe turnee, iar un turneu poate avea mai mulți organizatori. Un voluntar poate participa la mai multe turnee, iar un turneu poate avea mai mulți voluntari.

Cerința 4

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

```
CREATE TABLE TARI (
  ID_Tara NUMBER(5) PRIMARY KEY,
  Nume_Tara VARCHAR2(30) NOT NULL
);

CREATE TABLE ORASE (
  ID_Oras NUMBER(5) PRIMARY KEY,
  Nume_Oras VARCHAR2(30) NOT NULL,
  ID_Tara NUMBER(5) NOT NULL,
  CONSTRAINT fk_oras_tara FOREIGN KEY (ID_Tara) REFERENCES TARI(ID_Tara)
);

CREATE TABLE LOCATII (
  ID_Locatie NUMBER(5) PRIMARY KEY,
  Nume_Complex VARCHAR2(50) NOT NULL,
  Suprafata VARCHAR2(15) CHECK (Suprafata IN ('Zgura', 'Iarba', 'Hard')),
  Adresa VARCHAR2(100),
  ID_Oras NUMBER(5) NOT NULL,
  CONSTRAINT fk_locatie_oras FOREIGN KEY (ID_Oras) REFERENCES ORASE(ID_Oras)
);

CREATE TABLE SPONSORI (
```

```
  ID_Sponsor NUMBER(5) PRIMARY KEY,
  Nume_Companie VARCHAR2(50) NOT NULL,
  Domeniu_Activitate VARCHAR2(30)
);

CREATE TABLE ORGANIZATORI (
  ID_Organizator NUMBER(5) PRIMARY KEY,
  Nume VARCHAR2(30) NOT NULL,
  Prenume VARCHAR2(30) NOT NULL,
  Rol_General VARCHAR2(30)
);

CREATE TABLE VOLUNTARI (
  ID_Voluntar NUMBER(5) PRIMARY KEY,
  Nume VARCHAR2(30) NOT NULL,
  Prenume VARCHAR2(30) NOT NULL,
  Data_Nasterii DATE
);

CREATE TABLE ARBITRI (
  ID_Arbitru NUMBER(5) PRIMARY KEY,
  Nume VARCHAR2(30) NOT NULL,
  Prenume VARCHAR2(30) NOT NULL,
  Certificare VARCHAR2(20)
);

CREATE TABLE JUCATORI (
  ID_Jucator NUMBER(5) PRIMARY KEY,
```

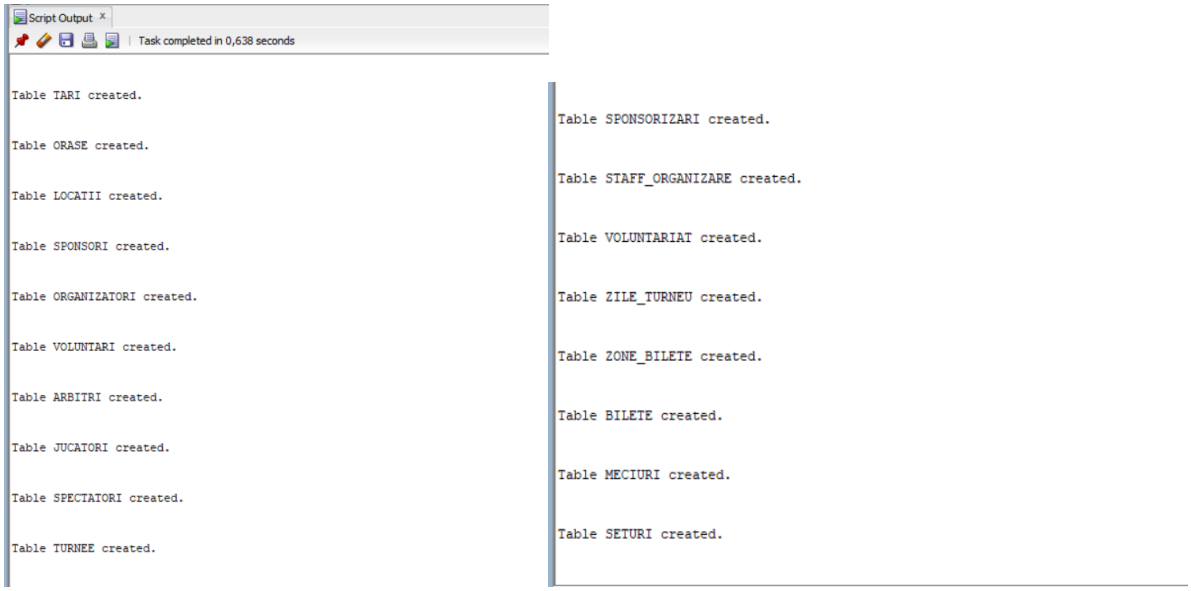
```
  Nume VARCHAR2(30) NOT NULL,
  Prenume VARCHAR2(30) NOT NULL,
  Data_Nasterii DATE NOT NULL
);

CREATE TABLE SPECTATORI (
  ID_Spectator NUMBER(5) PRIMARY KEY,
  Nume VARCHAR2(30) NOT NULL,
  Prenume VARCHAR2(30) NOT NULL,
  Email VARCHAR2(50) UNIQUE NOT NULL,
  Telefon VARCHAR2(15) UNIQUE NOT NULL
);

CREATE TABLE TURNEE (
  ID_Turneu NUMBER(5) PRIMARY KEY,
  Nume_Turneu VARCHAR2(50) NOT NULL,
  Categorie VARCHAR2(20) CHECK (Categorie IN ('Grand Slam Masculin', 'ATP 1000',
'ATP 500', 'ATP 250', 'Challenger', 'WTA 1000', 'WTA 500', 'WTA 250', 'Grand Slam
Feminin')),
  Data_Inceput DATE NOT NULL,
  Data_Sfarsit DATE NOT NULL,
  ID_Locatie NUMBER(5) NOT NULL,
  CONSTRAINT fk_turneu_locatie FOREIGN KEY (ID_Locatie) REFERENCES
LOCATII(ID_Locatie),
  CONSTRAINT chk_date_turneu CHECK (Data_Sfarsit >= Data_Inceput)
);

CREATE TABLE SPONSORIZARI (
  ID_Turneu NUMBER(5),
```

```
  ID_Sponsor NUMBER(5),
  Suma_Sponsorizata NUMBER(12),
  PRIMARY KEY (ID_Turneu, ID_Sponsor),
  CONSTRAINT fk_spons_turneu FOREIGN KEY (ID_Turneu) REFERENCES
TURNEE(ID_Turneu),
  CONSTRAINT fk_spons_firma FOREIGN KEY (ID_Sponsor) REFERENCES
SPONSORI(ID_Sponsor),
  CONSTRAINT chk_suma_pozitiva CHECK (Suma_Sponsorizata > 0)
);

CREATE TABLE STAFF_ORGANIZARE (
  ID_Turneu NUMBER(5),
  ID_Organizator NUMBER(5),
  Atributie_Specifica VARCHAR2(50),
  PRIMARY KEY (ID_Turneu, ID_Organizator),
  CONSTRAINT fk_staff_turneu FOREIGN KEY (ID_Turneu) REFERENCES
TURNEE(ID_Turneu),
  CONSTRAINT fk_staff_pers FOREIGN KEY (ID_Organizator) REFERENCES
ORGANIZATORI(ID_Organizator)
);

CREATE TABLE VOLUNTARIAT (
  ID_Turneu NUMBER(5),
  ID_Voluntar NUMBER(5),
  Departament VARCHAR2(30),
  PRIMARY KEY (ID_Turneu, ID_Voluntar),
  CONSTRAINT fk_vol_turneu FOREIGN KEY (ID_Turneu) REFERENCES
TURNEE(ID_Turneu),
  CONSTRAINT fk_vol_pers FOREIGN KEY (ID_Voluntar) REFERENCES
VOLUNTARI(ID_Voluntar)
```

```
);

CREATE TABLE ZILE_TURNEU (
  ID_Turneu NUMBER(5),
  Nr_Zi NUMBER(2),
  CONSTRAINT pk_zile_turneu PRIMARY KEY (ID_Turneu, Nr_Zi),
  CONSTRAINT fk_zi_turneu FOREIGN KEY (ID_Turneu) REFERENCES
TURNEE(ID_Turneu)
);

CREATE TABLE ZONE_BILETE (
  ID_Turneu NUMBER(5),
  Nr_Zi NUMBER(2),
  Zona_Bilet VARCHAR2(20) CHECK (Zona_Bilet IN ('VIP 1', 'VIP 2', 'Tribuna 1', 'Tribuna
2', 'Tribuna 3', 'Peluza 1', 'Peluza 2')),
  Pret_Standard NUMBER(6) NOT NULL,
  Capacitate_Vanzare NUMBER(5) NOT NULL,
  CONSTRAINT pk_zone_bilete PRIMARY KEY (ID_Turneu, Nr_Zi, Zona_Bilet),
  CONSTRAINT fk_zone_zi FOREIGN KEY (ID_Turneu, Nr_Zi) REFERENCES
ZILE_TURNEU(ID_Turneu, Nr_Zi),
  CONSTRAINT chk_pret_std_poz CHECK (Pret_Standard > 0),
  CONSTRAINT chk_cap_poz CHECK (Capacitate_Vanzare > 0)
);

CREATE TABLE BILETE (
  ID_Bilet NUMBER(10) PRIMARY KEY,
  Loc NUMBER(4) NOT NULL,
  Rand VARCHAR2(10) NOT NULL,
  ID_Turneu NUMBER(5) NOT NULL,
```

```
  Nr_Zi NUMBER(2) NOT NULL,
  Zona_Bilet VARCHAR2(20) NOT NULL,
  ID_Spectator NUMBER(5),
  CONSTRAINT fk_bilet_zona FOREIGN KEY (ID_Turneu, Nr_Zi, Zona_Bilet)
    REFERENCES ZONE_BILETE(ID_Turneu, Nr_Zi, Zona_Bilet),
  CONSTRAINT fk_bilet_spect FOREIGN KEY (ID_Spectator) REFERENCES
SPECTATORI(ID_Spectator),
  CONSTRAINT uq_bilet_unic UNIQUE (ID_Turneu, Nr_Zi, Zona_Bilet, Rand, Loc),
  CONSTRAINT chk_loc_poz CHECK (Loc > 0)
);

CREATE TABLE MECIURI (
  ID_Meci NUMBER(10) PRIMARY KEY,
  Durata_Minute NUMBER(3),
  Faza_Competitie VARCHAR2(30),
  Teren VARCHAR2(30) NOT NULL,
  Ora_Start DATE,
  Status_Meci VARCHAR2(20) DEFAULT 'Programat'
    CHECK (Status_Meci IN ('Programat', 'In Desfasurare', 'Terminat', 'Abandon',
'Amanat')),
  ID_Turneu NUMBER(5) NOT NULL,
  Nr_Zi NUMBER(2) NOT NULL,
  ID_Arbitru NUMBER(5) NOT NULL,
  ID_Jucator1 NUMBER(5) NOT NULL,
  ID_Jucator2 NUMBER(5) NOT NULL,
  ID_Jucator_Retras NUMBER(5),
  CONSTRAINT fk_meci_zi FOREIGN KEY (ID_Turneu, Nr_Zi)
    REFERENCES ZILE_TURNEU(ID_Turneu, Nr_Zi),
```

```
  CONSTRAINT fk_meci_arb FOREIGN KEY (ID_Arbitru) REFERENCES
ARBITRI(ID_Arbitru),
  CONSTRAINT fk_meci_j1 FOREIGN KEY (ID_Jucator1) REFERENCES
JUCATORI(ID_Jucator),
  CONSTRAINT fk_meci_j2 FOREIGN KEY (ID_Jucator2) REFERENCES
JUCATORI(ID_Jucator),
  CONSTRAINT fk_retr FOREIGN KEY (ID_Jucator_Retras) REFERENCES
JUCATORI(ID_Jucator),
  CONSTRAINT chk_jucatori_diferiti CHECK (ID_Jucator1 <> ID_Jucator2)
);

CREATE TABLE SETURI (
  ID_Meci NUMBER(10),
  Nr_Set NUMBER(1) CHECK (Nr_Set BETWEEN 1 AND 5),
  Scor_J1 NUMBER(2) NOT NULL,
  Scor_J2 NUMBER(2) NOT NULL,
  Tiebreak_J1 NUMBER(2),
  Tiebreak_J2 NUMBER(2),
  Nr_Asi_J1 NUMBER(3) DEFAULT 0,
  Nr_Asi_J2 NUMBER(3) DEFAULT 0,
  Nr_Duble_Greseli_J1 NUMBER(3) DEFAULT 0,
  Nr_Duble_Greseli_J2 NUMBER(3) DEFAULT 0,
  CONSTRAINT pk_seturi PRIMARY KEY (ID_Meci, Nr_Set),
  CONSTRAINT fk_set_meci FOREIGN KEY (ID_Meci)
    REFERENCES MECIURI(ID_Meci) ON DELETE CASCADE,
  CONSTRAINT chk_scor_j1 CHECK (Scor_J1 >= 0),
  CONSTRAINT chk_scor_j2 CHECK (Scor_J2 >= 0),
  CONSTRAINT chk_asi_pozitiv CHECK (Nr_Asi_J1 >= 0 AND Nr_Asi_J2 >= 0)
);
```



Cerinta 5

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

INSERT INTO TARI VALUES (1, 'Romania');

INSERT INTO TARI VALUES (2, 'Spania');

INSERT INTO TARI VALUES (3, 'Serbia');

INSERT INTO TARI VALUES (4, 'Italia');

INSERT INTO TARI VALUES (5, 'Franta');

INSERT INTO TARI VALUES (6, 'SUA');

INSERT INTO TARI VALUES (7, 'Germania');

INSERT INTO TARI VALUES (8, 'Grecia');

INSERT INTO TARI VALUES (9, 'Polonia');

INSERT INTO TARI VALUES (10, 'Australia');

INSERT INTO TARI VALUES (11, 'Elvetia');

INSERT INTO TARI VALUES (12, 'Marea Britanie');

```
INSERT INTO TARI VALUES (13, 'Canada');
INSERT INTO TARI VALUES (14, 'Japonia');
INSERT INTO TARI VALUES (15, 'Bulgaria');

INSERT INTO ORASE VALUES (1, 'Bucuresti', 1);
INSERT INTO ORASE VALUES (2, 'Madrid', 2);
INSERT INTO ORASE VALUES (3, 'Belgrad', 3);
INSERT INTO ORASE VALUES (4, 'Roma', 4);
INSERT INTO ORASE VALUES (5, 'Paris', 5);
INSERT INTO ORASE VALUES (6, 'New York', 6);
INSERT INTO ORASE VALUES (7, 'Berlin', 7);
INSERT INTO ORASE VALUES (8, 'Atena', 8);
INSERT INTO ORASE VALUES (9, 'Varsovia', 9);
INSERT INTO ORASE VALUES (10, 'Melbourne', 10);
INSERT INTO ORASE VALUES (11, 'Geneva', 11);
INSERT INTO ORASE VALUES (12, 'Londra', 12);
INSERT INTO ORASE VALUES (13, 'Toronto', 13);
INSERT INTO ORASE VALUES (14, 'Tokyo', 14);
INSERT INTO ORASE VALUES (15, 'Sofia', 15);

INSERT INTO LOCATII VALUES (1, 'Arenele BNR', 'Zgura', 'Str. Staicovici 42', 1);
INSERT INTO LOCATII VALUES (2, 'Caja Magica', 'Hard', 'Camino de Perales', 2);
INSERT INTO LOCATII VALUES (3, 'Novak Tennis Center', 'Zgura', 'Dunavski Kej', 3);
INSERT INTO LOCATII VALUES (4, 'Foro Italico', 'Zgura', 'Viale dei Gladiatori', 4);
INSERT INTO LOCATII VALUES (5, 'Roland Garros', 'Zgura', '2 Avenue Gordon Bennett',
5);
INSERT INTO LOCATII VALUES (6, 'Flushing Meadows', 'Hard', 'Queens', 6);
INSERT INTO LOCATII VALUES (7, 'Steffi Graf Stadion', 'Zgura', 'Hundekehlestrasse', 7);
```

```
INSERT INTO LOCATII VALUES (8, 'OAKA Center', 'Hard', 'Marousi', 8);
INSERT INTO LOCATII VALUES (9, 'Legia Tennis Club', 'Zgura', 'Mysliwiecka 4', 9);
INSERT INTO LOCATII VALUES (10, 'Rod Laver Arena', 'Hard', 'Olympic Blvd', 10);
INSERT INTO LOCATII VALUES (11, 'Palexpo', 'Hard', 'Route Francois-Peyrot', 11);
INSERT INTO LOCATII VALUES (12, 'Wimbledon Club', 'Iarba', 'Church Road', 12);
INSERT INTO LOCATII VALUES (13, 'Aviva Centre', 'Hard', 'Shoreham Dr', 13);
INSERT INTO LOCATII VALUES (14, 'Ariake Coliseum', 'Hard', 'Ariake', 14);
INSERT INTO LOCATII VALUES (15, 'Botev Park', 'Zgura', 'Kozloduy', 15);

INSERT INTO SPONSORI VALUES (1, 'Dedeman', 'Retail');
INSERT INTO SPONSORI VALUES (2, 'Banca Transilvania', 'Bancar');
INSERT INTO SPONSORI VALUES (3, 'Emirates', 'Aviatie');
INSERT INTO SPONSORI VALUES (4, 'Rolex', 'Lux');
INSERT INTO SPONSORI VALUES (5, 'Nike', 'Echipament');
INSERT INTO SPONSORI VALUES (6, 'Adidas', 'Echipament');
INSERT INTO SPONSORI VALUES (7, 'Wilson', 'Sport');
INSERT INTO SPONSORI VALUES (8, 'KIA', 'Auto');
INSERT INTO SPONSORI VALUES (9, 'Lacoste', 'Moda');
INSERT INTO SPONSORI VALUES (10, 'Oracle', 'Tech');
INSERT INTO SPONSORI VALUES (11, 'SAP', 'Software');
INSERT INTO SPONSORI VALUES (12, 'Peugeot', 'Auto');
INSERT INTO SPONSORI VALUES (13, 'Hublot', 'Lux');
INSERT INTO SPONSORI VALUES (14, 'RedBull', 'Bauturi');
INSERT INTO SPONSORI VALUES (15, 'Mastercard', 'Plati');

INSERT INTO JUCATORI VALUES (1, 'Djokovic', 'Novak', TO_DATE('22-05-1987', 'DD-MM-
YYYY'));
```

```
INSERT INTO JUCATORI VALUES (2, 'Nadal', 'Rafael', TO_DATE('03-06-1986', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (3, 'Alcaraz', 'Carlos', TO_DATE('05-05-2003', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (4, 'Sinner', 'Jannik', TO_DATE('16-08-2001', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (5, 'Tsitsipas', 'Stefanos', TO_DATE('12-08-1998', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (6, 'Medvedev', 'Daniil', TO_DATE('11-02-1996', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (7, 'Zverev', 'Alexander', TO_DATE('20-04-1997', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (8, 'Ruud', 'Casper', TO_DATE('22-12-1998', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (9, 'Rublev', 'Andrey', TO_DATE('20-10-1997', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (10, 'Hurkacz', 'Hubert', TO_DATE('11-02-1997', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (11, 'Dimitrov', 'Grigor', TO_DATE('16-05-1991', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (12, 'Fritz', 'Taylor', TO_DATE('28-10-1997', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (13, 'Rune', 'Holger', TO_DATE('29-04-2003', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (14, 'Shelton', 'Ben', TO_DATE('09-10-2002', 'DD-MM-YYYY'));

INSERT INTO JUCATORI VALUES (15, 'Monfils', 'Gael', TO_DATE('01-09-1986', 'DD-MM-YYYY'));


INSERT INTO ARBITRI VALUES (1, 'Lahyani', 'Mohamed', 'Gold Badge');

INSERT INTO ARBITRI VALUES (2, 'Bernardes', 'Carlos', 'Gold Badge');

INSERT INTO ARBITRI VALUES (3, 'Dumusois', 'Damien', 'Gold Badge');
```

```
INSERT INTO ARBITRI VALUES (4, 'Steiner', 'Marija', 'Gold Badge');

INSERT INTO ARBITRI VALUES (5, 'Mourier', 'Cedric', 'Gold Badge');

INSERT INTO ARBITRI VALUES (6, 'Asderaki', 'Eva', 'Gold Badge');

INSERT INTO ARBITRI VALUES (7, 'Nouni', 'Kader', 'Gold Badge');

INSERT INTO ARBITRI VALUES (8, 'Ramos', 'Carlos', 'Gold Badge');

INSERT INTO ARBITRI VALUES (9, 'Keothavong', 'James', 'Gold Badge');

INSERT INTO ARBITRI VALUES (10, 'Cicak', 'Marijana', 'Gold Badge');

INSERT INTO ARBITRI VALUES (11, 'Earley', 'Wayne', 'Silver Badge');

INSERT INTO ARBITRI VALUES (12, 'Tourte', 'Aurelie', 'Gold Badge');

INSERT INTO ARBITRI VALUES (13, 'Zhang', 'Juan', 'Gold Badge');

INSERT INTO ARBITRI VALUES (14, 'Blom', 'Richard', 'Silver Badge');

INSERT INTO ARBITRI VALUES (15, 'Engzell', 'Louise', 'Gold Badge');


INSERT INTO SPECTATORI VALUES (1, 'Ionescu', 'Mihai', 'mihai@test.ro', '0722111221');

INSERT INTO SPECTATORI VALUES (2, 'Popescu', 'Ana', 'ana@test.ro', '0722333442');

INSERT INTO SPECTATORI VALUES (3, 'Enache', 'George', 'george@test.ro',
'0744111223');

INSERT INTO SPECTATORI VALUES (4, 'Marin', 'Elena', 'elena@test.ro', '0755666774');

INSERT INTO SPECTATORI VALUES (5, 'Vasile', 'Dan', 'dan@test.ro', '0788999005');

INSERT INTO SPECTATORI VALUES (6, 'Dima', 'Andra', 'andra@test.ro', '0722111226');

INSERT INTO SPECTATORI VALUES (7, 'Constantin', 'Paul', 'paul@test.ro',
'0722333447');

INSERT INTO SPECTATORI VALUES (8, 'Miron', 'Ioan', 'ioan@test.ro', '0744111228');

INSERT INTO SPECTATORI VALUES (9, 'Radu', 'Sofia', 'sofia@test.ro', '0755666779');

INSERT INTO SPECTATORI VALUES (10, 'Stancu', 'Victor', 'victor@test.ro',
'0788999010');

INSERT INTO SPECTATORI VALUES (11, 'Lazar', 'Geta', 'geta@test.ro', '0722111211');

INSERT INTO SPECTATORI VALUES (12, 'Voinea', 'Ion', 'ionv@test.ro', '0722333412');

INSERT INTO SPECTATORI VALUES (13, 'Sava', 'Alina', 'alina@test.ro', '0744111213');
```

```
INSERT INTO SPECTATORI VALUES (14, 'Matei', 'Mihnea', 'mihnea@test.ro',
'0755666714');

INSERT INTO SPECTATORI VALUES (15, 'Toma', 'Bianca', 'bianca@test.ro',
'0788999015');


INSERT INTO TURNEE VALUES (1, 'Bucharest Open', 'ATP 250', TO_DATE('15-04-2024',
'DD-MM-YYYY'), TO_DATE('21-04-2024', 'DD-MM-YYYY'), 1);

INSERT INTO TURNEE VALUES (2, 'Roland Garros', 'Grand Slam Masculin',
TO_DATE('26-05-2024', 'DD-MM-YYYY'), TO_DATE('09-06-2024', 'DD-MM-YYYY'), 5);

INSERT INTO TURNEE VALUES (3, 'Wimbledon', 'Grand Slam Masculin', TO_DATE('01-
07-2024', 'DD-MM-YYYY'), TO_DATE('14-07-2024', 'DD-MM-YYYY'), 12);

INSERT INTO TURNEE VALUES (4, 'Monte Carlo Masters', 'ATP 1000', TO_DATE('07-04-
2024', 'DD-MM-YYYY'), TO_DATE('14-04-2024', 'DD-MM-YYYY'), 1);

INSERT INTO TURNEE VALUES (5, 'US Open', 'Grand Slam Masculin', TO_DATE('26-08-
2024', 'DD-MM-YYYY'), TO_DATE('08-09-2024', 'DD-MM-YYYY'), 6);


INSERT INTO SPONSORIZARI VALUES (1, 1, 100000);

INSERT INTO SPONSORIZARI VALUES (1, 2, 80000);

INSERT INTO SPONSORIZARI VALUES (2, 3, 500000);

INSERT INTO SPONSORIZARI VALUES (2, 4, 450000);

INSERT INTO SPONSORIZARI VALUES (3, 5, 600000);

INSERT INTO SPONSORIZARI VALUES (3, 6, 550000);

INSERT INTO SPONSORIZARI VALUES (4, 7, 200000);

INSERT INTO SPONSORIZARI VALUES (4, 8, 180000);

INSERT INTO SPONSORIZARI VALUES (5, 9, 700000);

INSERT INTO SPONSORIZARI VALUES (5, 10, 650000);

INSERT INTO SPONSORIZARI VALUES (1, 11, 50000);

INSERT INTO SPONSORIZARI VALUES (2, 12, 150000);

INSERT INTO SPONSORIZARI VALUES (3, 13, 300000);

INSERT INTO SPONSORIZARI VALUES (4, 14, 100000);
```

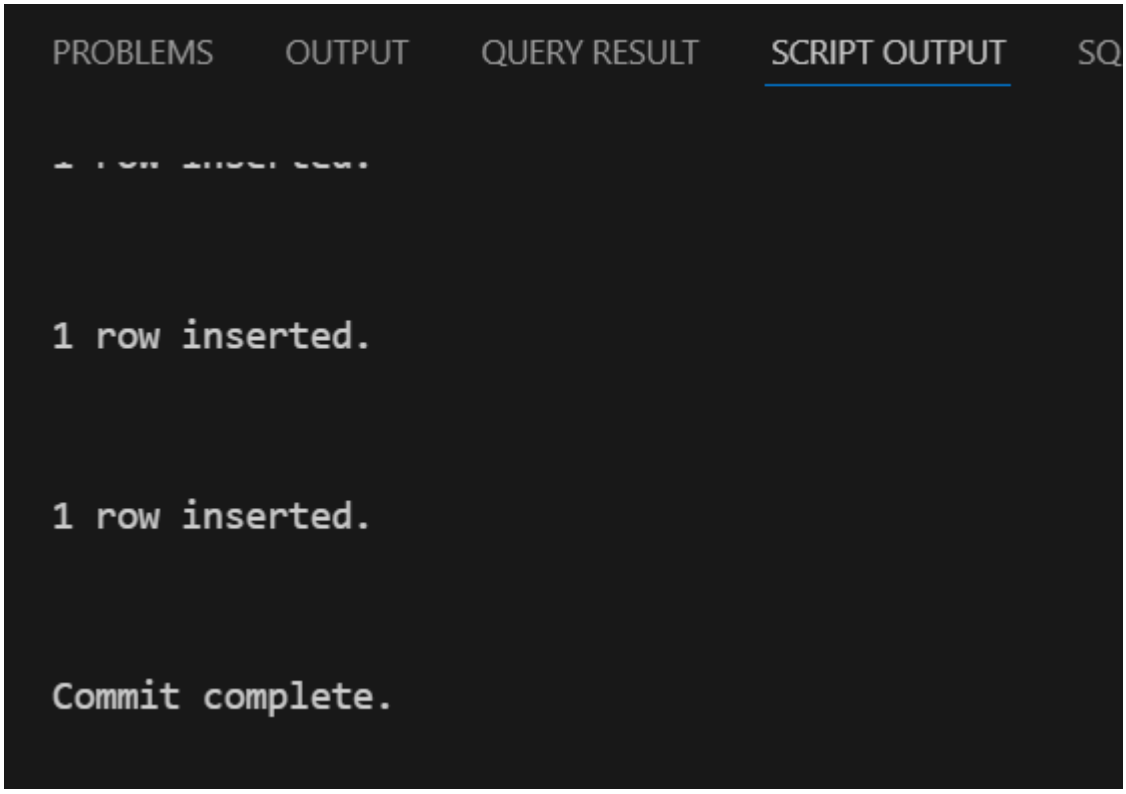
```
INSERT INTO SPONSORIZARI VALUES (5, 15, 400000);


INSERT INTO ZILE_TURNEU VALUES (1, 1);
INSERT INTO ZILE_TURNEU VALUES (1, 2);
INSERT INTO ZILE_TURNEU VALUES (1, 3);
INSERT INTO ZILE_TURNEU VALUES (2, 1);
INSERT INTO ZILE_TURNEU VALUES (2, 2);
INSERT INTO ZILE_TURNEU VALUES (2, 3);
INSERT INTO ZILE_TURNEU VALUES (3, 1);
INSERT INTO ZILE_TURNEU VALUES (3, 2);
INSERT INTO ZILE_TURNEU VALUES (3, 3);
INSERT INTO ZILE_TURNEU VALUES (4, 1);
INSERT INTO ZILE_TURNEU VALUES (4, 2);
INSERT INTO ZILE_TURNEU VALUES (4, 3);
INSERT INTO ZILE_TURNEU VALUES (5, 1);
INSERT INTO ZILE_TURNEU VALUES (5, 2);
INSERT INTO ZILE_TURNEU VALUES (5, 3);


INSERT INTO ZONE_BILETE VALUES (1, 1, 'VIP 1', 500, 100);
INSERT INTO ZONE_BILETE VALUES (1, 1, 'Tribuna 1', 150, 500);
INSERT INTO ZONE_BILETE VALUES (2, 1, 'VIP 1', 1000, 200);
INSERT INTO ZONE_BILETE VALUES (2, 1, 'Tribuna 1', 300, 1000);
INSERT INTO ZONE_BILETE VALUES (3, 1, 'VIP 2', 1200, 150);
INSERT INTO ZONE_BILETE VALUES (3, 1, 'Tribuna 2', 400, 1200);
INSERT INTO ZONE_BILETE VALUES (4, 1, 'VIP 1', 800, 100);
INSERT INTO ZONE_BILETE VALUES (4, 1, 'Peluza 1', 100, 800);
INSERT INTO ZONE_BILETE VALUES (5, 1, 'VIP 1', 1500, 300);
INSERT INTO ZONE_BILETE VALUES (5, 1, 'Tribuna 1', 500, 2000);
```

```
INSERT INTO ZONE_BILETE VALUES (1, 2, 'Peluza 2', 50, 600);
INSERT INTO ZONE_BILETE VALUES (2, 2, 'Peluza 1', 80, 1500);
INSERT INTO ZONE_BILETE VALUES (3, 2, 'Tribuna 3', 200, 800);
INSERT INTO ZONE_BILETE VALUES (4, 2, 'VIP 2', 600, 50);
INSERT INTO ZONE_BILETE VALUES (5, 2, 'Peluza 1', 120, 3000);

INSERT INTO BILETE VALUES (1, 1, 'A', 1, 1, 'VIP 1', 1);
INSERT INTO BILETE VALUES (2, 2, 'A', 1, 1, 'VIP 1', 2);
INSERT INTO BILETE VALUES (3, 1, 'B', 2, 1, 'Tribuna 1', 3);
INSERT INTO BILETE VALUES (4, 2, 'B', 2, 1, 'Tribuna 1', 4);
INSERT INTO BILETE VALUES (5, 1, 'C', 3, 1, 'VIP 2', 5);
INSERT INTO BILETE VALUES (6, 2, 'C', 3, 1, 'VIP 2', 6);
INSERT INTO BILETE VALUES (7, 1, 'D', 4, 1, 'Peluza 1', 7);
INSERT INTO BILETE VALUES (8, 2, 'D', 4, 1, 'Peluza 1', 8);
INSERT INTO BILETE VALUES (9, 1, 'E', 5, 1, 'Tribuna 1', 9);
INSERT INTO BILETE VALUES (10, 2, 'E', 5, 1, 'Tribuna 1', 10);
INSERT INTO BILETE VALUES (11, 10, 'A', 1, 2, 'Peluza 2', 11);
INSERT INTO BILETE VALUES (12, 15, 'B', 2, 2, 'Peluza 1', 12);
INSERT INTO BILETE VALUES (13, 20, 'C', 3, 2, 'Tribuna 3', 13);
INSERT INTO BILETE VALUES (14, 5, 'D', 4, 2, 'VIP 2', 14);
INSERT INTO BILETE VALUES (15, 50, 'E', 5, 2, 'Peluza 1', 15);

INSERT INTO MECIURI VALUES (1, 120, 'Finala', 'Central', TO_DATE('16:00', 'HH24:MI'),
'Terminat', 1, 1, 1, 1, 2, NULL);
INSERT INTO MECIURI VALUES (2, 150, 'Tur 1', 'Chatrier', TO_DATE('11:00', 'HH24:MI'),
'Terminat', 2, 1, 2, 3, 4, NULL);
INSERT INTO MECIURI VALUES (3, 180, 'Tur 1', 'Centre Court', TO_DATE('13:00',
'HH24:MI'), 'Terminat', 3, 1, 3, 5, 6, NULL);
```

```
INSERT INTO MECIURI VALUES (4, 90, 'Tur 1', 'Rainier III', TO_DATE('10:00', 'HH24:MI'), 'Terminat', 4, 1, 4, 7, 8, NULL);

INSERT INTO MECIURI VALUES (5, 200, 'Tur 1', 'Ashe', TO_DATE('19:00', 'HH24:MI'), 'Terminat', 5, 1, 5, 9, 10, NULL);

INSERT INTO MECIURI VALUES (6, 100, 'Tur 2', 'Central', TO_DATE('12:00', 'HH24:MI'), 'Terminat', 1, 2, 6, 11, 12, NULL);

INSERT INTO MECIURI VALUES (7, 130, 'Tur 2', 'Chatrier', TO_DATE('14:00', 'HH24:MI'), 'Terminat', 2, 2, 7, 13, 14, NULL);

INSERT INTO MECIURI VALUES (8, 110, 'Tur 2', 'Court 1', TO_DATE('12:00', 'HH24:MI'), 'Terminat', 3, 2, 8, 15, 1, NULL);

INSERT INTO MECIURI VALUES (9, 95, 'Tur 2', 'Rainier III', TO_DATE('11:00', 'HH24:MI'), 'Terminat', 4, 2, 9, 2, 3, NULL);

INSERT INTO MECIURI VALUES (10, 140, 'Tur 2', 'Armstrong', TO_DATE('14:00', 'HH24:MI'), 'Terminat', 5, 2, 10, 4, 5, NULL);

INSERT INTO MECIURI VALUES (11, 85, 'Tur 3', 'Central', TO_DATE('14:00', 'HH24:MI'), 'Abandon', 1, 3, 11, 6, 7, 6);

INSERT INTO MECIURI VALUES (12, 160, 'Tur 3', 'Lenglen', TO_DATE('15:00', 'HH24:MI'), 'Terminat', 2, 3, 12, 8, 9, NULL);

INSERT INTO MECIURI VALUES (13, 125, 'Tur 3', 'Court 2', TO_DATE('11:00', 'HH24:MI'), 'Terminat', 3, 3, 13, 10, 11, NULL);

INSERT INTO MECIURI VALUES (14, 115, 'Tur 3', 'Court 2', TO_DATE('13:00', 'HH24:MI'), 'Terminat', 4, 3, 14, 12, 13, NULL);

INSERT INTO MECIURI VALUES (15, 155, 'Tur 3', 'Grandstand', TO_DATE('11:00', 'HH24:MI'), 'Terminat', 5, 3, 15, 14, 15, NULL);


INSERT INTO SETURI VALUES (1, 1, 6, 4, NULL, NULL, 5, 2, 1, 3);

INSERT INTO SETURI VALUES (2, 1, 6, 2, NULL, NULL, 4, 1, 0, 2);

INSERT INTO SETURI VALUES (3, 1, 7, 5, NULL, NULL, 6, 4, 1, 1);

INSERT INTO SETURI VALUES (4, 1, 6, 1, NULL, NULL, 3, 0, 0, 4);

INSERT INTO SETURI VALUES (5, 1, 4, 6, NULL, NULL, 2, 8, 3, 0);

INSERT INTO SETURI VALUES (6, 1, 6, 3, NULL, NULL, 5, 2, 1, 2);

INSERT INTO SETURI VALUES (7, 1, 6, 0, NULL, NULL, 4, 0, 0, 5);
```

```
INSERT INTO SETURI VALUES (8, 1, 7, 6, 7, 5, 8, 9, 2, 1);
INSERT INTO SETURI VALUES (9, 1, 6, 4, NULL, NULL, 3, 2, 1, 3);
INSERT INTO SETURI VALUES (10, 1, 6, 2, NULL, NULL, 5, 1, 0, 4);
INSERT INTO SETURI VALUES (11, 1, 3, 0, NULL, NULL, 2, 0, 0, 2);
INSERT INTO SETURI VALUES (12, 1, 6, 4, NULL, NULL, 4, 3, 1, 1);
INSERT INTO SETURI VALUES (13, 1, 6, 3, NULL, NULL, 5, 2, 1, 2);
INSERT INTO SETURI VALUES (14, 1, 6, 1, NULL, NULL, 3, 0, 0, 3);
INSERT INTO SETURI VALUES (15, 1, 7, 5, NULL, NULL, 6, 4, 2, 1);

INSERT INTO ORGANIZATORI VALUES (1, 'Popa', 'Andrei', 'Director Turneu');
INSERT INTO ORGANIZATORI VALUES (2, 'Ionescu', 'Maria', 'Coordonator Logistica');
INSERT INTO ORGANIZATORI VALUES (3, 'Stanciu', 'Cristian', 'Manager Marketing');
INSERT INTO ORGANIZATORI VALUES (4, 'Dumitru', 'Elena', 'Relatii Publice');
INSERT INTO ORGANIZATORI VALUES (5, 'Radu', 'George', 'Seful Securitatii');


INSERT INTO STAFF_ORGANIZARE VALUES (1, 3, 'Coordonator Marketing Local');
INSERT INTO STAFF_ORGANIZARE VALUES (2, 5, 'Responsabil Securitate Terenuri');
INSERT INTO STAFF_ORGANIZARE VALUES (3, 1, 'Manager Protocol VIP');
INSERT INTO STAFF_ORGANIZARE VALUES (3, 2, 'Supervizor Logistica');
INSERT INTO STAFF_ORGANIZARE VALUES (4, 4, 'Responsabil Comunicare Presa');
INSERT INTO STAFF_ORGANIZARE VALUES (5, 3, 'Manager Operational');
INSERT INTO STAFF_ORGANIZARE VALUES (5, 4, 'Coordonator Transport Jucatori');
INSERT INTO STAFF_ORGANIZARE VALUES (1, 3, 'Coordonator Bilete');
INSERT INTO STAFF_ORGANIZARE VALUES (2, 5, 'Supervizor Arbitri');
INSERT INTO STAFF_ORGANIZARE VALUES (3, 1, 'Manager Protocol');
INSERT INTO STAFF_ORGANIZARE VALUES (4, 4, 'Relatii Sponsori');
INSERT INTO STAFF_ORGANIZARE VALUES (5, 3, 'Comunicare Presa');
```

```
INSERT INTO VOLUNTARI VALUES (1, 'Vasile', 'Ionut', TO_DATE('12-05-2000', 'DD-MM-YYYY'));

INSERT INTO VOLUNTARI VALUES (2, 'Nica', 'Andreea', TO_DATE('25-08-2001', 'DD-MM-YYYY'));

INSERT INTO VOLUNTARI VALUES (3, 'Stoica', 'Marius', TO_DATE('10-01-1999', 'DD-MM-YYYY'));

INSERT INTO VOLUNTARI VALUES (4, 'Lupu', 'Raluca', TO_DATE('15-03-2002', 'DD-MM-YYYY'));

INSERT INTO VOLUNTARI VALUES (5, 'Matei', 'Cosmin', TO_DATE('30-11-2000', 'DD-MM-YYYY'));


INSERT INTO VOLUNTARIAT VALUES (1, 1, 'Copii de mingi');

INSERT INTO VOLUNTARIAT VALUES (1, 2, 'Asistenta Spectatori');

INSERT INTO VOLUNTARIAT VALUES (2, 3, 'Copii de mingi');

INSERT INTO VOLUNTARIAT VALUES (3, 4, 'Verificare Bilete');

INSERT INTO VOLUNTARIAT VALUES (4, 5, 'Copii de mingi');

INSERT INTO VOLUNTARIAT VALUES (5, 1, 'Ghidare Media');

INSERT INTO VOLUNTARIAT VALUES (2, 2, 'Protocol');

INSERT INTO VOLUNTARIAT VALUES (1, 3, 'Asistent Scor');

INSERT INTO VOLUNTARIAT VALUES (2, 4, 'Suport Tehnic Teren');

INSERT INTO VOLUNTARIAT VALUES (3, 1, 'Distributie Apa Jucatori');

INSERT INTO VOLUNTARIAT VALUES (3, 2, 'Orientare Spectatori');

INSERT INTO VOLUNTARIAT VALUES (4, 2, 'Asistent Arbitru Scaun');


COMMIT;
```


PROBLEMS
OUTPUT
QUERY RESULT
SCRIPT OUTPUT
SQ
1 row inserted.
1 row inserted.
Commit complete.

Cerința 6

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate. Apelați subprogramul.

Să se realizeze un subprogram stocat care clasifică jucătorii unui turneu în funcție de eficiența serviciului, calculată pe baza numărului total de ași.

Să se afișeze numele fiecărui jucător împreuna cu numărul său de ași de la un turneu anume. Pentru fiecare jucător in parte va apărea în ce categorie intră (Server Excelent, Server Bun, Server OK). La final se vor afișa jucătorii care excelează pe serviciu (au servit mai mult de 10 ași).

```
CREATE OR REPLACE TYPE t_categorii_serviciu AS VARRAY(3) OF VARCHAR2(50);
/

CREATE OR REPLACE TYPE t_lista_top_serveri AS TABLE OF NUMBER;
/

CREATE OR REPLACE PROCEDURE Clasifica_Serveri_Turneu (p_id_turneu IN NUMBER)
IS
   v_nivele     t_categorii_serviciu := t_categorii_serviciu('Server Excelent', 'Server Bun',
'Server OK');

   v_top_jucatori t_lista_top_serveri := t_lista_top_serveri();

   TYPE t_stat_asi IS TABLE OF NUMBER INDEX BY BINARY_INTEGER;
   v_lista_asi    t_stat_asi;
```

```
  v_idx       NUMBER;
  v_nume_j    VARCHAR2(100);
  v_status    VARCHAR2(50);
BEGIN
  DBMS_OUTPUT.PUT_LINE('CLASIFICARE SERVICIU TURNEU CU ID-ul: ' ||
p_id_turneu);

  FOR rec IN (SELECT J.ID_Jucator,
             NVL(SUM(S.Nr_Asi_J1), 0) +
             NVL((SELECT SUM(Nr_Asi_J2) FROM SETURI S2 JOIN MECIURI M2 ON
S2.ID_Meci = M2.ID_Meci
                 WHERE M2.ID_Jucator2 = J.ID_Jucator AND M2.ID_Turneu = p_id_turneu),
0) as total_asi
          FROM JUCATORI J
          JOIN MECIURI M ON (J.ID_Jucator = M.ID_Jucator1)
          JOIN SETURI S ON M.ID_Meci = S.ID_Meci
          WHERE M.ID_Turneu = p_id_turneu
          GROUP BY J.ID_Jucator) LOOP

    v_lista_asi(rec.ID_Jucator) := rec.total_asi;

    IF rec.total_asi > 10 THEN
      v_top_jucatori.EXTEND;
      v_top_jucatori(v_top_jucatori.LAST) := rec.ID_Jucator;
    END IF;
  END LOOP;

  v_idx := v_lista_asi.FIRST;
  WHILE v_idx IS NOT NULL LOOP
```

```
    SELECT Prenume || ' ' || Nume INTO v_nume_j FROM JUCATORI WHERE ID_Jucator =
v_idx;

    IF v_lista_asi(v_idx) >= 20 THEN v_status := v_nivele(1);
    ELSIF v_lista_asi(v_idx) >= 10 THEN v_status := v_nivele(2);
    ELSE v_status := v_nivele(3);
    END IF;

    DBMS_OUTPUT.PUT_LINE('Jucator: ' || v_nume_j || ' | Asi: ' || v_lista_asi(v_idx) || ' |
Categorie: ' || v_status);
    v_idx := v_lista_asi.NEXT(v_idx);
  END LOOP;

  DBMS_OUTPUT.PUT_LINE('Jucatori care exceleaza pe serviciu:');
  FOR i IN 1..v_top_jucatori.COUNT LOOP
    SELECT Nume INTO v_nume_j FROM JUCATORI WHERE ID_Jucator =
v_top_jucatori(i);
    DBMS_OUTPUT.PUT_LINE('ID ' || v_top_jucatori(i) || ': ' || v_nume_j);
  END LOOP;

EXCEPTION
  WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('Eroare: ' || SQLERRM);
END;
/

SET SERVEROUTPUT ON;
BEGIN
  Clasifica_Serveri_Turneu(3);
END;
```

/

```
CLASIFICARE SERVICIU TURNEU CU ID-ul: 3
Jucator: Stefanos Tsitsipas | Asi: 26 | Categorie: Server Excelent
Jucator: Hubert Hurkacz | Asi: 5 | Categorie: Server OK
Jucator: Gael Monfils | Asi: 8 | Categorie: Server OK
Jucatori care exceleaza pe serviciu:
ID 5: Tsitsipas


PL/SQL procedure successfully completed.
```

Cerința 7

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

Să se realizeze un subprogram stocat care calculează banii încasați pe biletele vândute pe fiecare zonă de bilete de la fiecare turneu. Să se afișeze atât totalul grupat pe zone, cât și totalul încasat pe turneu.

```
CREATE OR REPLACE PROCEDURE Raport_Vanzari_Cursoare IS
  CURSOR c_turnee IS
    SELECT ID_Turneu, Nume_Turneu FROM TURNEE;

  CURSOR c_detalii_bilete(p_id_turneu NUMBER) IS
    SELECT ZB.Zona_Bilet,
        COUNT(B.ID_Bilet) as bucati,
        SUM(ZB.Pret_Standard) as suma
    FROM BILETE B
    JOIN ZONE_BILETE ZB ON B.ID_Turneu = ZB.ID_Turneu
                AND B.Nr_Zi = ZB.Nr_Zi
                AND B.Zona_Bilet = ZB.Zona_Bilet
    WHERE B.ID_Turneu = p_id_turneu
    GROUP BY ZB.Zona_Bilet;

  v_id_t     TURNEE.ID_Turneu%TYPE;
```

```
  v_nume_t   TURNEE.Nume_Turneu%TYPE;
  v_zona     VARCHAR2(50);
  v_buc      NUMBER;
  v_val      NUMBER;
  v_total_t  NUMBER;
BEGIN
  DBMS_OUTPUT.PUT_LINE('RAPORT FINANCIAR TURNEE');

  OPEN c_turnee;
  LOOP
    FETCH c_turnee INTO v_id_t, v_nume_t;
    EXIT WHEN c_turnee%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('TURNEU: ' || v_nume_t);
    v_total_t := 0;

    OPEN c_detalii_bilete(v_id_t);
    LOOP
      FETCH c_detalii_bilete INTO v_zona, v_buc, v_val;
      EXIT WHEN c_detalii_bilete%NOTFOUND;

      DBMS_OUTPUT.PUT_LINE('Zona: ' || v_zona || ' | Bilete: ' || v_buc || ' | Venit: ' ||
NVL(v_val, 0));
      v_total_t := v_total_t + NVL(v_val, 0);
    END LOOP;
    CLOSE c_detalii_bilete;

    DBMS_OUTPUT.PUT_LINE('TOTAL TURNEU: ' || v_total_t || ' LEI');
```

```
    END LOOP;
    CLOSE c_turnee;
END;
/
```

```
SET SERVEROUTPUT ON;
EXEC Raport_Vanzari_Cursoare;
```

```
PL/SQL procedure successfully completed.

RAPORT FINANCIAR TURNEE
TURNEU: Bucharest Open
Zona: VIP 1 | Bilete: 2 | Venit: 1000
Zona: Peluza 2 | Bilete: 1 | Venit: 50
TOTAL TURNEU: 1050 LEI
TURNEU: Roland Garros
Zona: Tribuna 1 | Bilete: 2 | Venit: 600
Zona: Peluza 1 | Bilete: 1 | Venit: 80
TOTAL TURNEU: 680 LEI
TURNEU: Wimbledon
Zona: VIP 2 | Bilete: 2 | Venit: 2400
Zona: Tribuna 3 | Bilete: 1 | Venit: 200
TOTAL TURNEU: 2600 LEI
TURNEU: Monte Carlo Masters
Zona: Peluza 1 | Bilete: 2 | Venit: 200
Zona: VIP 2 | Bilete: 1 | Venit: 600
TOTAL TURNEU: 800 LEI
TURNEU: US Open
Zona: Tribuna 1 | Bilete: 2 | Venit: 1000
Zona: Peluza 1 | Bilete: 1 | Venit: 120
TOTAL TURNEU: 1120 LEI
```

Cerința 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 create. Tratați toate excepțiile care pot apărea, incluzând excepțiile predefinite NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Să se realizeze un subprogram stocat de tip funcție care afișează data nașterii celui mai mare voluntar aflat la un turneu dat printr-un ID anume. Să se afișeze, de asemenea, numele său și departamentul din care face parte.

```
CREATE OR REPLACE FUNCTION Info_Voluntar_Varsta_Maxima (p_id_turneu NUMBER)
RETURN VARCHAR2 IS
  v_nume        VARCHAR2(100);
  v_prenume     VARCHAR2(100);
  v_departament  VARCHAR2(100);
  v_data_n      DATE;
BEGIN
  SELECT v.Nume, v.Prenume, vt.Departament, v.Data_Nasterii
  INTO v_nume, v_prenume, v_departament, v_data_n
  FROM VOLUNTARI v
```

```
  JOIN VOLUNTARIAT vt ON v.ID_Voluntar = vt.ID_Voluntar
  JOIN TURNEE t ON vt.ID_Turneu = t.ID_Turneu
  WHERE t.ID_Turneu = p_id_turneu
   AND v.Data_Nasterii = (
     SELECT MIN(v2.Data_Nasterii)
     FROM VOLUNTARI v2
     JOIN VOLUNTARIAT vt2 ON v2.ID_Voluntar = vt2.ID_Voluntar
     WHERE vt2.ID_Turneu = p_id_turneu
   );

  RETURN 'Voluntar: ' || v_nume || ' ' || v_prenume ||
       ' | Data nasterii: ' || TO_CHAR(v_data_n, 'DD-MM-YYYY') ||
       ' | Departament: ' || v_departament;

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RETURN 'Nu exista voluntari inregistrati la acest turneu.';
  WHEN TOO_MANY_ROWS THEN
    RETURN 'Exista mai multi voluntari cu varsta maxima la acest turneu.';
  WHEN OTHERS THEN
    RETURN 'EROARE: ' || SQLERRM;
END;
/

SET SERVEROUTPUT ON;
DECLARE
  v_rezultat VARCHAR2(500);
BEGIN
```

```
    v_rezultat := Info_Voluntar_Varsta_Maxima(2);
    DBMS_OUTPUT.PUT_LINE(v_rezultat);

    v_rezultat := Info_Voluntar_Varsta_Maxima(1);
    DBMS_OUTPUT.PUT_LINE(v_rezultat);

    v_rezultat := Info_Voluntar_Varsta_Maxima(6);
    DBMS_OUTPUT.PUT_LINE(v_rezultat);

    v_rezultat := Info_Voluntar_Varsta_Maxima(-1);
    DBMS_OUTPUT.PUT_LINE(v_rezultat);
END;
/
```

```
Voluntar: Stoica Marius | Data nasterii: 10-01-1999 | Departament: Copii de mingi


PL/SQL procedure successfully completed.
```

FARA ERORI

```
PL/SQL procedure successfully completed.

Exista mai multi voluntari cu varsta maxima la acest turneu.
```

TOO_MANY_ROWS (mai multi voluntari cu varsta maxima)

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

```
PL/SQL procedure successfully completed.

Nu exista voluntari inregistrati la acest turneu.


PL/SQL procedure successfully completed.
```

NO_DATA_FOUND (nu exista voluntari la turneu)

```
No errors.
ID-ul introdus ca parametru este invalid

        ID_INVALID (exceptia definita de mine)
PL/SQL procedure successfully completed.
```

Cerința 9

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să aibă minim 2 parametri și să utilizeze într-o singură comandă SQL 5 dintre tabelele create. Definiți minim 2 excepții proprii, altele decât cele predefinite la nivel de sistem. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.

Să se realizeze un subprogram stocat de tip procedură care primește ca parametri ID-ul unui turneu și ID-ul unui jucător și afișează numărul de meciuri câștigate de jucătorul respectiv în cadrul turneului, cât și totalul game-urilor câștigate de acesta.

În cazul în care jucătorul nu are victorii în acel turneu se va afișa o eroare, la fel în cazul în care turneul nu are momentan zile înregistrate în baza de date.

```
CREATE OR REPLACE PROCEDURE Statistica_Completa_Jucator_Turneu (
   p_id_turneu NUMBER,
   p_id_jucator NUMBER
) IS
   v_nume_jucator     JUCATORI.Nume%TYPE;
   v_prenume_jucator  JUCATORI.Prenume%TYPE;
   v_nume_turneu      TURNEE.Nume_Turneu%TYPE;

   v_meciuri_castigate NUMBER := 0;
   v_gameuri_castigate NUMBER := 0;
   v_nr_zile          NUMBER;
```

```
  TURNEU_FARA_ZILE_INREGISTRATE EXCEPTION;
  NICIUN_MECI_CASTIGAT        EXCEPTION;
BEGIN
  SELECT COUNT(*) INTO v_nr_zile
  FROM ZILE_TURNEU
  WHERE ID_Turneu = p_id_turneu;

  IF v_nr_zile = 0 THEN
    RAISE TURNEU_FARA_ZILE_INREGISTRATE;
  END IF;

  SELECT Nume, Prenume, Nume_Turneu,
      SUM(CASE
        WHEN ID_Retras IS NOT NULL THEN
          CASE
            WHEN ID_Retras != p_id_jucator THEN 1
            ELSE 0
          END
        ELSE
          CASE
            WHEN Seturi_Pro > Seturi_Contra THEN 1
            ELSE 0
          END
```

```
    END),
    SUM(Gameuri_Pro)
INTO v_nume_jucator, v_prenume_jucator, v_nume_turneu,
   v_meciuri_castigate, v_gameuri_castigate
FROM (
  SELECT
    j.Nume, j.Prenume, t.Nume_Turneu,
    m.ID_Meci, m.ID_Jucator_Retras as ID_Retras,
    SUM(CASE
      WHEN (m.ID_Jucator1 = p_id_jucator AND s.Scor_J1 > s.Scor_J2) OR
         (m.ID_Jucator2 = p_id_jucator AND s.Scor_J2 > s.Scor_J1)
      THEN 1 ELSE 0 END) as Seturi_Pro,
    SUM(CASE
      WHEN (m.ID_Jucator1 = p_id_jucator AND s.Scor_J1 < s.Scor_J2) OR
         (m.ID_Jucator2 = p_id_jucator AND s.Scor_J2 < s.Scor_J1)
      THEN 1 ELSE 0 END) as Seturi_Contra,
    SUM(CASE
      WHEN m.ID_Jucator1 = p_id_jucator THEN s.Scor_J1
      WHEN m.ID_Jucator2 = p_id_jucator THEN s.Scor_J2
      ELSE 0 END) as Gameuri_Pro
  FROM TURNEE t
  JOIN ZILE_TURNEU zt ON t.ID_Turneu = zt.ID_Turneu
  JOIN MECIURI m     ON zt.ID_Turneu = m.ID_Turneu AND zt.Nr_Zi = m.Nr_Zi
  JOIN SETURI s      ON m.ID_Meci = s.ID_Meci
```

```
    JOIN JUCATORI j     ON (m.ID_Jucator1 = j.ID_Jucator OR m.ID_Jucator2 =
j.ID_Jucator)

    WHERE t.ID_Turneu = p_id_turneu

     AND j.ID_Jucator = p_id_jucator

    GROUP BY j.Nume, j.Prenume, t.Nume_Turneu, m.ID_Meci, m.ID_Jucator_Retras

  )

  GROUP BY Nume, Prenume, Nume_Turneu;


  IF v_meciuri_castigate = 0 THEN

    RAISE NICIUN_MECI_CASTIGAT;

  END IF;


  DBMS_OUTPUT.PUT_LINE('RAPORT ' || v_prenume_jucator || ' ' || v_nume_jucator || '
LA TURNEUL ' || v_nume_turneu);

  DBMS_OUTPUT.PUT_LINE('Meciuri castigate: ' || v_meciuri_castigate);

  DBMS_OUTPUT.PUT_LINE('Game-uri castigate: ' || v_gameuri_castigate);


EXCEPTION

  WHEN TURNEU_FARA_ZILE_INREGISTRATE THEN

    DBMS_OUTPUT.PUT_LINE('Turneul ' || p_id_turneu || ' nu are zile inregistrate.');


  WHEN NICIUN_MECI_CASTIGAT THEN

    DBMS_OUTPUT.PUT_LINE('Jucatorul ' || v_prenume_jucator || ' ' || v_nume_jucator ||
' a participat dar nu a castigat niciun meci la acest turneu.');
```

```
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Jucatorul ' || p_id_jucator || ' nu a participat la acest turneu.');

  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('EROARE: ' || SQLERRM);
END;
/

SET SERVEROUTPUT ON;

BEGIN
  Statistica_Completa_Jucator_Turneu(1, 1);
  Statistica_Completa_Jucator_Turneu(1, 2);
  Statistica_Completa_Jucator_Turneu(1, 10);
  Statistica_Completa_Jucator_Turneu(10, 1);
END;
/
```

```
RAPORT Novak Djokovic LA TURNEUL Bucharest Open
Meciuri castigate: 1
Game-uri castigate: 13
```

FARA ERORI

```
Jucatorul Rafael Nadal a participat dar nu a castigat niciun meci la acest turneu.
```

NICIUN_MECI_CASTIGAT (Jucatorul nu are victorii la turneu)

TURNEU_FARA_ZILE_INREGISTRATE

(Turneul nu are zile inregistrate)

NO_DATA_FOUND (Jucatorul nu a participat la turneu)

Cerința 10

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

```
SET SERVEROUTPUT ON;


CREATE OR REPLACE TRIGGER TRG_LMD_MECIURI
BEFORE INSERT OR UPDATE OR DELETE ON MECIURI
BEGIN
  IF DELETING THEN
    RAISE_APPLICATION_ERROR(-20001, 'STERGERE: Este strict interzisa stergerea meciurilor din turneu!');

  ELSIF UPDATING('ID_Jucator1') OR UPDATING('ID_Jucator2') THEN
    RAISE_APPLICATION_ERROR(-20002, 'MODIFICARE JUCATORI: Nu puteti schimba jucatorii unui meci deja stabilit! Creati un meci nou.');

  ELSIF INSERTING THEN
    DBMS_OUTPUT.PUT_LINE('Inserare cu succes.');
  END IF;
END;
/
```

INSERT INTO MECIURI VALUES (99, 90, 'Semifinala', 'Teren 1', TO_DATE('14:00', 'HH24:MI'), 'Terminat', 1, 2, 1, 1, 2, NULL);

UPDATE MECIURI SET ID_Jucator1 = 5 WHERE ID_Meci = 1;

DELETE FROM MECIURI WHERE ID_Meci = 1;

```
Inserare cu succes.


1 row inserted.
```

```
Error starting at line : 1 in command -
UPDATE MECIURI SET ID_Jucator1 = 5 WHERE ID_Meci = 1
Error report -
ORA-20002: MODIFICARE JUCATORI: Nu puteti schimba jucatorii unui meci deja stabilit! Creati un meci nou.
ORA-06512: at "SYSTEM.TRG_LMD_MECIURI", line 6
ORA-04088: error during execution of trigger 'SYSTEM.TRG_LMD_MECIURI'

https://docs.oracle.com/error-help/db/ora-20002/

More Details :
https://docs.oracle.com/error-help/db/ora-20002/
https://docs.oracle.com/error-help/db/ora-06512/
https://docs.oracle.com/error-help/db/ora-04088/
```

```
Error starting at line : 1 in command -
DELETE FROM MECIURI WHERE ID_Meci = 1
Error at Command Line : 1 Column : 13
Error report -
SQL Error: ORA-20001: STERGERE: Este strict interzisa stergerea meciurilor din turneu!
ORA-06512: at "SYSTEM.TRG_LMD_MECIURI", line 3
ORA-04088: error during execution of trigger 'SYSTEM.TRG_LMD_MECIURI'

https://docs.oracle.com/error-help/db/ora-20001/

More Details :
https://docs.oracle.com/error-help/db/ora-20001/
https://docs.oracle.com/error-help/db/ora-06512/
https://docs.oracle.com/error-help/db/ora-04088/
```

Cerința 11

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

```
SET SERVEROUTPUT ON;


CREATE OR REPLACE TRIGGER TRG_LMD_JUCATOR_RETRAS
BEFORE INSERT OR UPDATE OF ID_Jucator_Retras ON MECIURI
FOR EACH ROW
BEGIN
  IF :NEW.ID_Jucator_Retras IS NOT NULL THEN


    IF :NEW.ID_Jucator_Retras != :NEW.ID_Jucator1 AND :NEW.ID_Jucator_Retras !=
:NEW.ID_Jucator2 THEN


      RAISE_APPLICATION_ERROR(-20003,
        'EROARE: Jucatorul cu ID-ul ' || :NEW.ID_Jucator_Retras ||
        ' nu participa la acest meci! (Joaca ' || :NEW.ID_Jucator1 || ' vs ' ||
:NEW.ID_Jucator2 || ')');
    END IF;


  END IF;
END;
/
```

UPDATE MECIURI

SET ID_Jucator_Retras = 99

WHERE ID_Meci = 1;

```
Trigger TRG_LMD_JUCATOR_RETRAS compiled

No errors.

Error starting at line : 1 in command -
UPDATE MECIURI
SET ID_Jucator_Retras = 99
WHERE ID_Meci = 100
Error at Command Line : 1 Column : 8
Error report -
SQL Error: ORA-20003: EROARE: Jucatorul cu ID-ul 99 nu participa la acest meci! (Joaca 1 vs 2)
ORA-06512: at "SYSTEM.TRG_LMD_JUCATOR_RETRAS", line 6
ORA-04088: error during execution of trigger 'SYSTEM.TRG_LMD_JUCATOR_RETRAS'

https://docs.oracle.com/error-help/db/ora-20003/
```

Cerința 12

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

```
CREATE OR REPLACE TRIGGER TRG_LDD_DROP
BEFORE DROP ON SCHEMA
BEGIN
  IF ORA_DICT_OBJ_TYPE = 'TABLE' THEN
    RAISE_APPLICATION_ERROR(-20005, 'STOP! Nu ai voie sa stergi tabelele!');
  END IF;
END;
/

DROP TABLE TARI;
```

```
Trigger TRG_LDD_DROP compiled

No errors.

Error starting at line : 1 in command -
DROP TABLE TARI
Error report -
ORA-04088: error during execution of trigger 'SYSTEM.TRG_LDD_DROP'
ORA-00604: error occurred at recursive SQL level 1
ORA-20005: STOP! Nu ai voie sa stergi tabelele!
ORA-06512: at line 3
```

Cerința 13

Formulați în limbaj natural o problemă pe care să o rezolvați folosind un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).

Să se realizeze un sistem de gestionare a logisticii turneelor de tenis care să permită generarea automată a programului de joc zilnic și mutarea sigură a meciurilor între terenuri. Sistemul trebuie să formateze numele jucătorilor pentru transmisiunea TV (ex. 'Nadal R.'), să calculeze automat data calendaristică a fiecărui meci în funcție de ziua turneului și să blocheze mutarea meciurilor pe terenuri care au deja o încărcare prea mare (peste 5 meciuri/zi), garantând astfel integritatea fluxului operațional.

```
CREATE OR REPLACE PACKAGE PKG_LOGISTICA_TURNEU AS

  TYPE Rec_Detaliu_Afisaj IS RECORD (
    Ora_Start     VARCHAR2(10),
    Nume_Teren    VARCHAR2(50),
    Duel_Jucatori  VARCHAR2(100)
  );

  TYPE Tab_Program_Zilnic IS TABLE OF Rec_Detaliu_Afisaj;

  FUNCTION Formateaza_Nume_TV(p_id_jucator IN NUMBER) RETURN VARCHAR2;

  FUNCTION Verifica_Incarcare_Teren(p_nume_teren IN VARCHAR2, p_data IN DATE)
RETURN NUMBER;

  PROCEDURE Genereaza_Order_Of_Play(p_data_program IN DATE);
```

```
  PROCEDURE Muta_Meci_Urgent(p_id_meci IN NUMBER, p_teren_nou IN VARCHAR2);

END PKG_LOGISTICA_TURNEU;
/

CREATE OR REPLACE PACKAGE BODY PKG_LOGISTICA_TURNEU AS

  v_lista_program Tab_Program_Zilnic := Tab_Program_Zilnic();

  FUNCTION Formateaza_Nume_TV(p_id_jucator IN NUMBER) RETURN VARCHAR2 IS
    v_nume JUCATORI.Nume%TYPE;
    v_prenume JUCATORI.Prenume%TYPE;
  BEGIN
    SELECT Nume, Prenume INTO v_nume, v_prenume
    FROM JUCATORI
    WHERE ID_Jucator = p_id_jucator;

    RETURN v_nume || ' ' || SUBSTR(v_prenume, 1, 1) || '.';
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      RETURN 'TBA';
  END Formateaza_Nume_TV;

  FUNCTION Verifica_Incarcare_Teren(p_nume_teren IN VARCHAR2, p_data IN DATE)
RETURN NUMBER IS
    v_count NUMBER;
  BEGIN
    SELECT COUNT(*) INTO v_count
```

```
    FROM MECIURI m
    JOIN TURNEE t ON m.ID_Turneu = t.ID_Turneu
    WHERE m.Teren = p_nume_teren
    AND TRUNC(t.Data_Inceput + m.NR_ZI - 1) = TRUNC(p_data);

    RETURN v_count;
END Verifica_Incarcare_Teren;

PROCEDURE Genereaza_Order_Of_Play(p_data_program IN DATE) IS
    CURSOR c_meciuri IS
        SELECT m.ID_Meci, m.Teren, m.Ora_Start, m.ID_Jucator1, m.ID_Jucator2
        FROM MECIURI m
        JOIN TURNEE t ON m.ID_Turneu = t.ID_Turneu
        WHERE TRUNC(t.Data_Inceput + m.NR_ZI - 1) = TRUNC(p_data_program)
        ORDER BY m.Ora_Start;

    v_nume1 VARCHAR2(50);
    v_nume2 VARCHAR2(50);
    i NUMBER := 1;
BEGIN
    v_lista_program.DELETE;

    FOR r_meci IN c_meciuri LOOP
        v_lista_program.EXTEND;

        v_nume1 := Formateaza_Nume_TV(r_meci.ID_Jucator1);
        v_nume2 := Formateaza_Nume_TV(r_meci.ID_Jucator2);
```

```
        v_lista_program(i).Ora_Start := TO_CHAR(r_meci.Ora_Start, 'HH24:MI');
        v_lista_program(i).Nume_Teren := r_meci.Teren;
        v_lista_program(i).Duel_Jucatori := v_nume1 || ' vs ' || v_nume2;

        i := i + 1;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('ORDER OF PLAY : ' || TO_CHAR(p_data_program, 'DD-
MON-YYYY'));

    IF v_lista_program.COUNT = 0 THEN
       DBMS_OUTPUT.PUT_LINE('Nu exista meciuri programate pentru aceasta data.');
    ELSE
      FOR k IN v_lista_program.FIRST .. v_lista_program.LAST LOOP
        DBMS_OUTPUT.PUT_LINE('[' || v_lista_program(k).Ora_Start || '] ' ||
                    v_lista_program(k).Nume_Teren || ' : ' ||
                    v_lista_program(k).Duel_Jucatori);
      END LOOP;
    END IF;

  END Genereaza_Order_Of_Play;

  PROCEDURE Muta_Meci_Urgent(p_id_meci IN NUMBER, p_teren_nou IN VARCHAR2)
IS
    v_nr_meciuri NUMBER;
    v_data_calculata DATE;
  BEGIN
    SELECT (t.Data_Inceput + m.NR_ZI - 1)
```

```
    INTO v_data_calculata
    FROM MECIURI m
    JOIN TURNEE t ON m.ID_Turneu = t.ID_Turneu
    WHERE m.ID_Meci = p_id_meci;

    v_nr_meciuri := Verifica_Incarcare_Teren(p_teren_nou, v_data_calculata);

    IF v_nr_meciuri >= 5 THEN
      RAISE_APPLICATION_ERROR(-20020, 'Terenul ' || p_teren_nou || ' este deja supraincarcat in acea zi.');
    END IF;

    UPDATE MECIURI
    SET Teren = p_teren_nou
    WHERE ID_Meci = p_id_meci;

    DBMS_OUTPUT.PUT_LINE('Meciul cu ID-ul ' || p_id_meci || ' a fost mutat pe ' || p_teren_nou);

  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('Meciul nu exista sau nu este asociat corect unui turneu.');
  END Muta_Meci_Urgent;

END PKG_LOGISTICA_TURNEU;
/

SET SERVEROUTPUT ON;
```

```
BEGIN
  PKG_LOGISTICA_TURNEU.Genereaza_Order_Of_Play(TO_DATE('15-04-2024', 'DD-MM-YYYY'));

  DBMS_OUTPUT.PUT_LINE('');

  PKG_LOGISTICA_TURNEU.Muta_Meci_Urgent(1, 'Terenul 2');

  DBMS_OUTPUT.PUT_LINE('');

  PKG_LOGISTICA_TURNEU.Genereaza_Order_Of_Play(TO_DATE('15-04-2024', 'DD-MM-YYYY'));

  PKG_LOGISTICA_TURNEU.Muta_Meci_Urgent(1, 'Central');
END;
/
```

```
Package Body PKG_LOGISTICA_TURNEU compiled

ORDER OF PLAY : 15-APR-2024
[09:00] Arena 1 : Tsitsipas S. vs Medvedev D.
[10:30] Arena 2 : Zverev A. vs Ruud C.
[12:00] Arena 3 : Rublev A. vs Hurkacz H.
[16:00] Central : Djokovic N. vs Nadal R.

Meciul cu ID-ul 1 a fost mutat pe Terenul 2

ORDER OF PLAY : 15-APR-2024
[09:00] Arena 1 : Tsitsipas S. vs Medvedev D.
[10:30] Arena 2 : Zverev A. vs Ruud C.
[12:00] Arena 3 : Rublev A. vs Hurkacz H.
[16:00] Terenul 2 : Djokovic N. vs Nadal R.
Meciul cu ID-ul 1 a fost mutat pe Central


PL/SQL procedure successfully completed.
```

```
l

t at line 1:
20020: Terenul Arena 1 este deja supraincarcat in acea zi.
06512: at "SYSTEM.PKG_LOGISTICA_TURNEU", line 85
06512: at line 6

s://docs.oracle.com/error-help/db/ora-20020/


 Details :
s://docs.oracle.com/error-help/db/ora-20020/
s://docs.oracle.com/error-help/db/ora-06512/
```