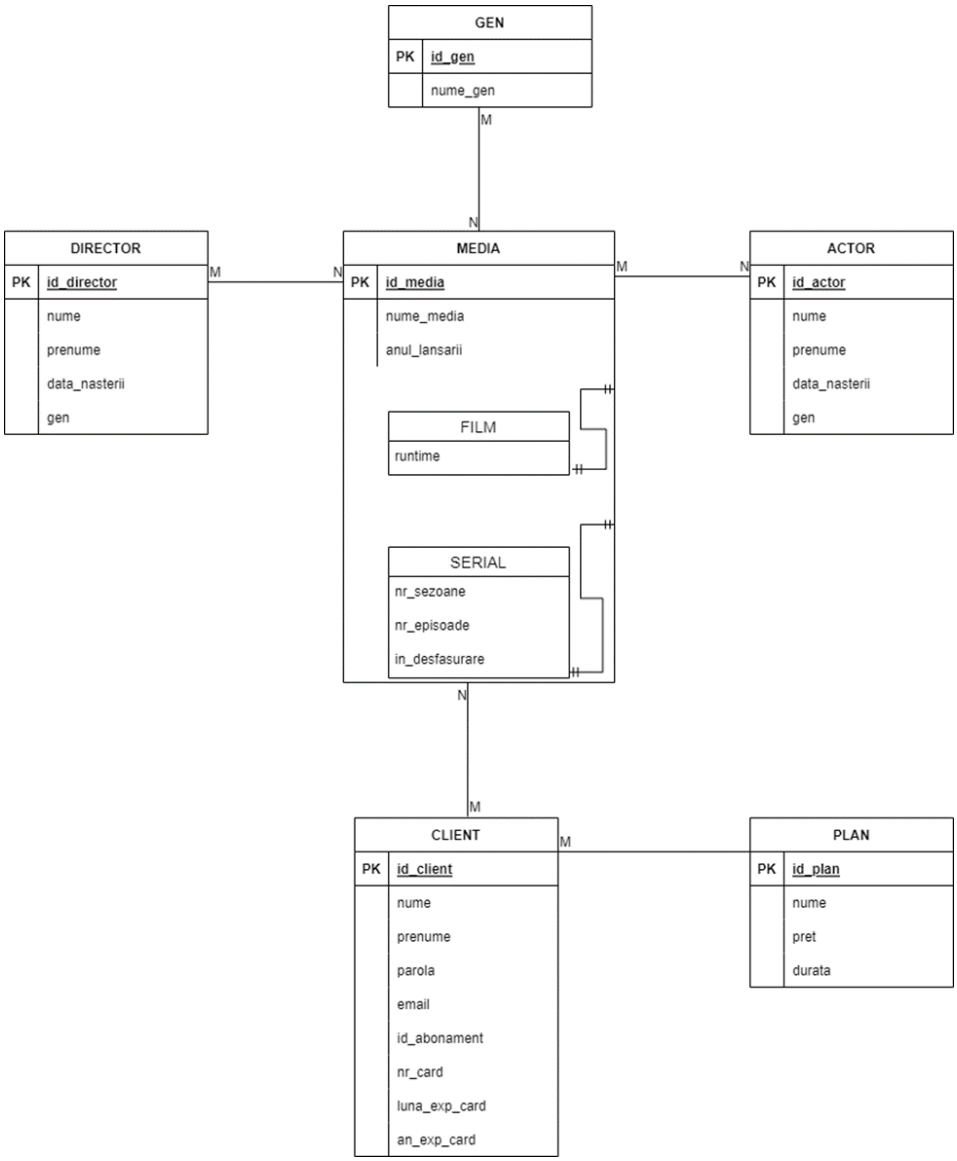
PROIECT SGBD - Fănică Narcis-Alexandru - 244

GESTIONARE NETFLIX

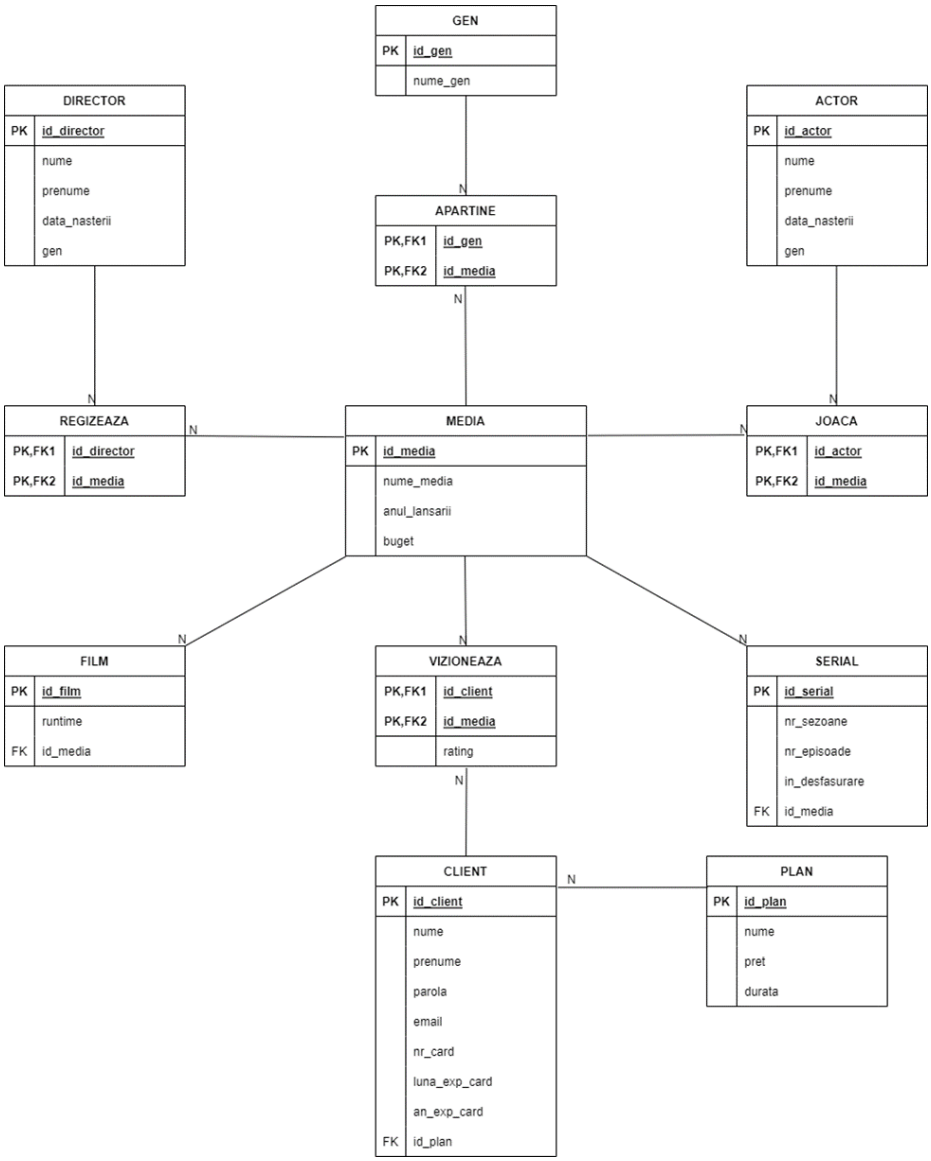
Exercitiul 1 - Descrierea modelului real, a utilității acestuia și a regulilor de funcționare.

Netflix reprezintă un serviciu de transmitere al filmelor și serialelor contra costul unui abonament plătit lunar. În baza acestui abonament, pot fi redate oricâte filme și seriale se află la un moment dat pe platformă. Filmele au o durată determinată exprimată în minute, în timp ce serialele au un număr variabil de sezoane și episoade la un moment dat. Acestea pot fi incheiate, însemnând că nu vor mai fi lansate noi episoade sau pot fi în desfășurare, ceea ce înseamnă că noi episoade pot apărea periodic.
Toate filmele și serialele au unul sau mai mulți actori precum și unul sau mai mulți directori ce au lucrat la regizarea acestora.
În plus, fiecare film și serial apartine unuia sau mai multor genuri.
Pentru a avea acces la aceste filme și seriale, clienții sunt nevoiți să își creeze un abonament lunar în baza unui plan pentru care vor plăti o anumită sumă predeterminată în funcție de abonamentul ales și de facilitățile acestuia.

Exercitiul 2 - Realizați diagrama entitate-relație (ERD).



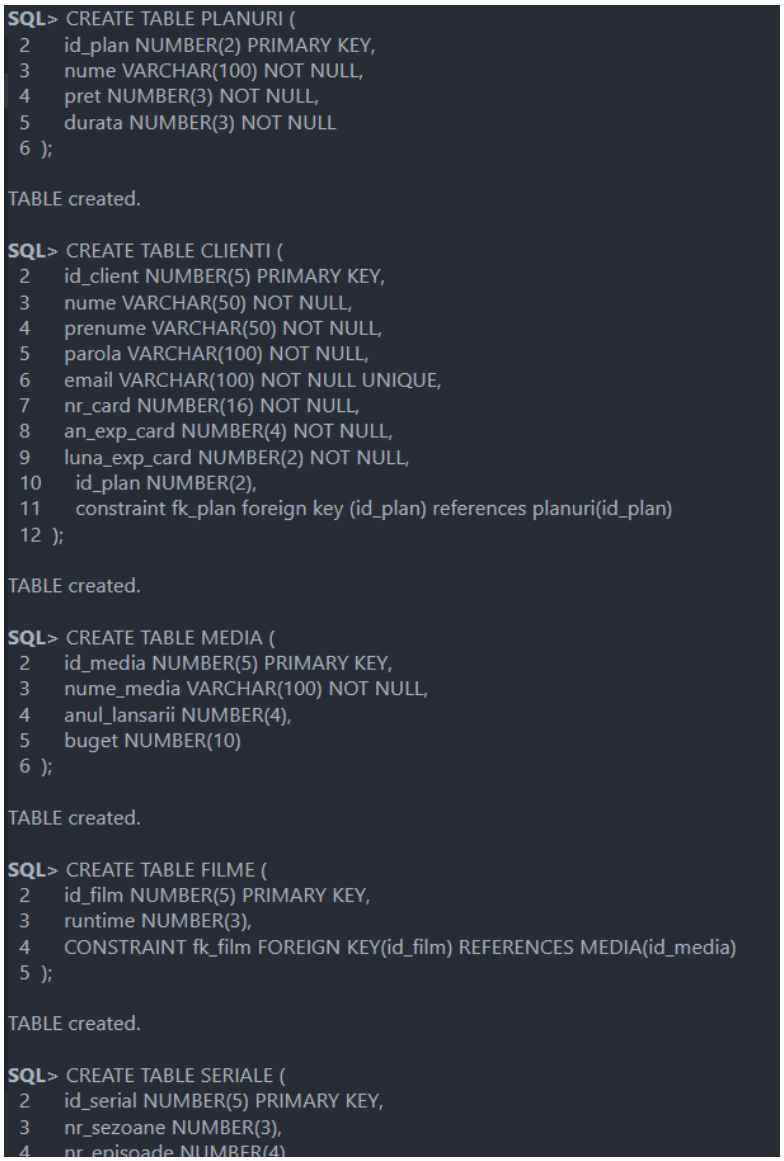
Exercitiul 3 - Pornind de la diagrama entitate-relație realizați diagram conceptuală a modelului propus, integrând toate atributele necesare.



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

```
  CREATE TABLE PLANURI (
    id_plan NUMBER(2) PRIMARY KEY,
    nume VARCHAR(100) NOT NULL,
    pret NUMBER(3) NOT NULL,
    durata NUMBER(3) NOT NULL
);

CREATE TABLE CLIENTI (
    id_client NUMBER(5) PRIMARY KEY,
    nume VARCHAR(50) NOT NULL,
    prenume VARCHAR(50) NOT NULL,
    parola VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    nr_card NUMBER(16) NOT NULL,
    an_exp_card NUMBER(4) NOT NULL,
```

```
    luna_exp_card NUMBER(2) NOT NULL,
    id_plan NUMBER(2),
    constraint fk_plan foreign key (id_plan) references planuri(id_plan)
);

CREATE TABLE MEDIA (
    id_media NUMBER(5) PRIMARY KEY,
    nume_media VARCHAR(100) NOT NULL,
    anul_lansarii NUMBER(4),
    buget NUMBER(10)
);

CREATE TABLE FILME (
    id_film NUMBER(5) PRIMARY KEY,
    runtime NUMBER(3),
    CONSTRAINT fk_film FOREIGN KEY(id_film) REFERENCES MEDIA(id_media)
);
CREATE TABLE SERIALE (
    id_serial NUMBER(5) PRIMARY KEY,
    nr_sezoane NUMBER(3),
    nr_episoade NUMBER(4),
    in_desfasurare VARCHAR(1),
    CONSTRAINT fk_serial FOREIGN KEY(id_serial) REFERENCES MEDIA(id_media)
);

CREATE TABLE GEN (
    id_gen NUMBER(3) PRIMARY KEY,
    nume_gen VARCHAR(100) NOT NULL
);

CREATE TABLE ACTORI (
    id_actor NUMBER(5) PRIMARY KEY,
    nume VARCHAR(100) NOT NULL,
    prenume VARCHAR(100) NOT NULL,
    data_nasterii DATE,
    gen VARCHAR(1)
);

CREATE TABLE DIRECTORI (
    id_director NUMBER(5) PRIMARY KEY,
    nume VARCHAR(100) NOT NULL,
    prenume VARCHAR(100) NOT NULL,
    data_nasterii DATE,
    gen VARCHAR(1)
);

CREATE TABLE VIZIONEAZA (
    id_client NUMBER(5),
    id_media NUMBER(5),
    rating NUMBER(3, 1),
    CONSTRAINT pk_vizioneaza PRIMARY KEY (id_client, id_media),
    CONSTRAINT fk_vizioneaza_client FOREIGN KEY (id_client) REFERENCES CLIENTI(id_client),
    CONSTRAINT fk_vizioneaza_media FOREIGN KEY (id_media) REFERENCES MEDIA(id_media)
);
CREATE TABLE JOACA (
    id_actor NUMBER(5),
    id_media NUMBER(5),
    CONSTRAINT pk_joaca PRIMARY KEY (id_actor, id_media),
    CONSTRAINT fk_joaca_actor FOREIGN KEY (id_actor) REFERENCES ACTORI(id_actor),
    CONSTRAINT fk_joaca_media FOREIGN KEY (id_media) REFERENCES MEDIA(id_media)
);
CREATE TABLE REGIZEAZA (
    id_director NUMBER(5),
    id_media NUMBER(5),
    CONSTRAINT pk_regizeaza PRIMARY KEY (id_director, id_media),
    CONSTRAINT fk_regizeaza_director FOREIGN KEY (id_director) REFERENCES DIRECTORI(id_director),
    CONSTRAINT fk_regizeaza_media FOREIGN KEY (id_media) REFERENCES MEDIA(id_media)
);

CREATE TABLE APARTINE
(
    id_media NUMBER(5),
    id_gen NUMBER(3),
    CONSTRAINT pk_apartine PRIMARY KEY (id_gen, id_media),
    CONSTRAINT fk_apartine_gen FOREIGN KEY (id_gen) REFERENCES GEN(id_gen),
    CONSTRAINT fk_apartine_media FOREIGN KEY (id_media) REFERENCES MEDIA(id_media)
);
```



```
SQL> CREATE TABLE PLANURI (
  2     id_plan NUMBER(2) PRIMARY KEY,
  3     nume VARCHAR(100) NOT NULL,
  4     pret NUMBER(3) NOT NULL,
  5     durata NUMBER(3) NOT NULL
  6  );

TABLE created.

SQL> CREATE TABLE CLIENTI (
  2     id_client NUMBER(5) PRIMARY KEY,
  3     nume VARCHAR(50) NOT NULL,
  4     prenume VARCHAR(50) NOT NULL,
  5     parola VARCHAR(100) NOT NULL,
  6     email VARCHAR(100) NOT NULL UNIQUE,
  7     nr_card NUMBER(16) NOT NULL,
  8     an_exp_card NUMBER(4) NOT NULL,
  9     luna_exp_card NUMBER(2) NOT NULL,
  10      id_plan NUMBER(2),
  11      constraint fk_plan foreign key (id_plan) references planuri(id_plan)
  12  );

TABLE created.

SQL> CREATE TABLE MEDIA (
  2     id_media NUMBER(5) PRIMARY KEY,
  3     nume_media VARCHAR(100) NOT NULL,
  4     anul_lansarii NUMBER(4),
  5     buget NUMBER(10)
  6  );

TABLE created.

SQL> CREATE TABLE FILME (
  2     id_film NUMBER(5) PRIMARY KEY,
  3     runtime NUMBER(3),
  4     CONSTRAINT fk_film FOREIGN KEY(id_film) REFERENCES MEDIA(id_media)
  5  );

TABLE created.

SQL> CREATE TABLE SERIALE (
  2     id_serial NUMBER(5) PRIMARY KEY,
  3     nr_sezoane NUMBER(3),
  4     nr_episoade NUMBER(4)
```

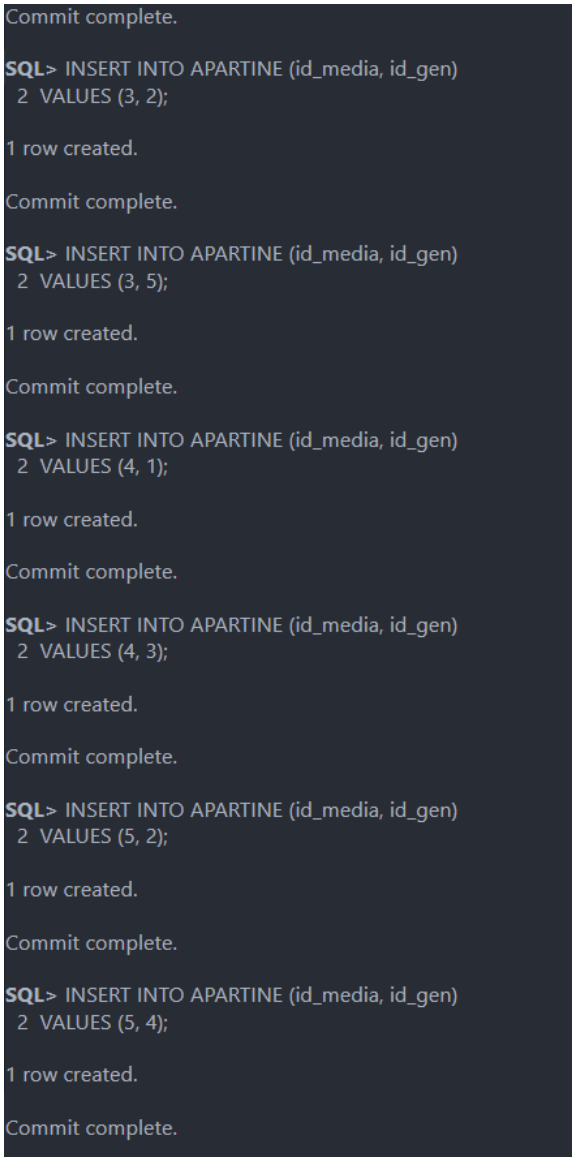
Exercitiul 5 - Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).

```
INSERT INTO PLANURI (id_plan, nume, pret, durata)
VALUES (1, 'Basic Plan', 9, 30);

INSERT INTO PLANURI (id_plan, nume, pret, durata)
VALUES (2, 'Standard Plan', 13, 30);

INSERT INTO PLANURI (id_plan, nume, pret, durata)
VALUES (3, 'Premium Plan', 16, 30);

INSERT INTO PLANURI (id_plan, nume, pret, durata)
VALUES (4, 'Mobile Plan', 6, 30);

INSERT INTO PLANURI (id_plan, nume, pret, durata)
VALUES (5, 'Ultra Plan', 20, 30);
```

```
INSERT INTO CLIENTI (id_client, nume, prenume, parola, email, nr_card, an_exp_card, luna_exp_card, id_plan)
VALUES (1, 'Smith', 'John', 'mypassword', 'johnsmith@example.com', 1234567890123456, 2025, 6, 1);

INSERT INTO CLIENTI (id_client, nume, prenume, parola, email, nr_card, an_exp_card, luna_exp_card, id_plan)
VALUES (2, 'Johnson', 'Emily', 'emilypassword', 'emilyjohnson@example.com', 9876543210987654, 2024, 9, 2);

INSERT INTO CLIENTI (id_client, nume, prenume, parola, email, nr_card, an_exp_card, luna_exp_card, id_plan)
VALUES (3, 'Davis', 'Michael', 'michaelpassword', 'michaeldavis@example.com', 5678901234567890, 2026, 1, 3);

INSERT INTO CLIENTI (id_client, nume, prenume, parola, email, nr_card, an_exp_card, luna_exp_card, id_plan)
VALUES (4, 'Miller', 'Sophia', 'sophiapassword', 'sophiamiller@example.com', 6543210987654321, 2023, 3, 2);

INSERT INTO CLIENTI (id_client, nume, prenume, parola, email, nr_card, an_exp_card, luna_exp_card, id_plan)
VALUES (5, 'Wilson', 'Oliver', 'oliverpassword', 'oliverwilson@example.com', 9876543210123456, 2024, 12, 4);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (1, 'Avengers: Endgame', 2019, 356000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (2, 'The Crown', 2016, 13000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (3, 'Stranger Things', 2016, 8000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (4, 'Inception', 2010, 160000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (5, 'Breaking Bad', 2008, 3500000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (6, 'The Avengers', 2012, 220000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (7, 'Stranger Things', 2016, 8000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (8, 'Inception', 2010, 160000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (9, 'The Crown', 2016, 13000000);

INSERT INTO MEDIA (id_media, nume_media, anul_lansarii, buget)
VALUES (10, 'The Dark Knight', 2008, 185000000);


INSERT INTO FILME (id_film, runtime)
VALUES (1, 181);

INSERT INTO FILME (id_film, runtime)
VALUES (4, 148);

INSERT INTO FILME (id_film, runtime)
VALUES (6, 143);

INSERT INTO FILME (id_film, runtime)
VALUES (8, 148);

INSERT INTO FILME (id_film, runtime)
VALUES (10, 152);


INSERT INTO SERIALE (id_serial, nr_sezoane, nr_episoade, in_desfasurare)
VALUES (2, 5, 60, 'Y');

INSERT INTO SERIALE (id_serial, nr_sezoane, nr_episoade, in_desfasurare)
VALUES (3, 3, 25, 'N');

INSERT INTO SERIALE (id_serial, nr_sezoane, nr_episoade, in_desfasurare)
VALUES (5, 6, 62, 'Y');

INSERT INTO SERIALE (id_serial, nr_sezoane, nr_episoade, in_desfasurare)
VALUES (7, 4, 32, 'Y');

INSERT INTO SERIALE (id_serial, nr_sezoane, nr_episoade, in_desfasurare)
VALUES (9, 5, 50, 'Y');


INSERT INTO GEN (id_gen, nume_gen)
VALUES (1, 'Action');
```

```
INSERT INTO GEN (id_gen, nume_gen)
VALUES (2, 'Drama');

INSERT INTO GEN (id_gen, nume_gen)
VALUES (3, 'Sci-Fi');

INSERT INTO GEN (id_gen, nume_gen)
VALUES (4, 'Comedy');

INSERT INTO GEN (id_gen, nume_gen)
VALUES (5, 'Thriller');

INSERT INTO ACTORI (id_actor, nume, prenume, data_nasterii, gen)
VALUES (1, 'Downey Jr.', 'Robert', '1965-04-04', 'M');

INSERT INTO ACTORI (id_actor, nume, prenume, data_nasterii, gen)
VALUES (2, 'Colman', 'Olivia', '1974-01-30', 'F');

INSERT INTO ACTORI (id_actor, nume, prenume, data_nasterii, gen)
VALUES (3, 'Harbour', 'David', '1975-04-10', 'M');

INSERT INTO ACTORI (id_actor, nume, prenume, data_nasterii, gen)
VALUES (4, 'DiCaprio', 'Leonardo', '1974-11-11', 'M');

INSERT INTO ACTORI (id_actor, nume, prenume, data_nasterii, gen)
VALUES (5, 'Paulson', 'Sarah', '1974-12-17', 'F');

INSERT INTO DIRECTORI (id_director, nume, prenume, data_nasterii, gen)
VALUES (1, 'Nolan', 'Christopher', '1970-07-30', 'M');

INSERT INTO DIRECTORI (id_director, nume, prenume, data_nasterii, gen)
VALUES (2, 'Favreau', 'Jon', '1966-10-19', 'M');

INSERT INTO DIRECTORI (id_director, nume, prenume, data_nasterii, gen)
VALUES (3, 'Wachowski', 'Lana', '1965-06-21', 'F');

INSERT INTO DIRECTORI (id_director, nume, prenume, data_nasterii, gen)
VALUES (4, 'Tarantino', 'Quentin', '1963-03-27', 'M');

INSERT INTO DIRECTORI (id_director, nume, prenume, data_nasterii, gen)
VALUES (5, 'Scorsese', 'Martin', '1942-11-17', 'M');

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (1, 1, 4.5);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (1, 3, 3.8);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (2, 2, 4.2);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (3, 1, 4.8);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (4, 5, 4.0);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (5, 4, 4.7);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (5, 2, 4.3);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (5, 3, 3.5);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (5, 5, 4.5);

INSERT INTO VIZIONEAZA (id_client, id_media, rating)
VALUES (5, 1, 4.6);

INSERT INTO JOACA (id_actor, id_media)
VALUES (1, 1);

INSERT INTO JOACA (id_actor, id_media)
VALUES (2, 2);

INSERT INTO JOACA (id_actor, id_media)
VALUES (3, 3);
```

```
INSERT INTO JOACA (id_actor, id_media)
VALUES (4, 4);

INSERT INTO JOACA (id_actor, id_media)
VALUES (5, 5);

INSERT INTO JOACA (id_actor, id_media)
VALUES (1, 4);

INSERT INTO JOACA (id_actor, id_media)
VALUES (2, 5);

INSERT INTO JOACA (id_actor, id_media)
VALUES (3, 2);

INSERT INTO JOACA (id_actor, id_media)
VALUES (4, 3);

INSERT INTO JOACA (id_actor, id_media)
VALUES (5, 1);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (1, 1);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (2, 2);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (3, 3);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (4, 4);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (5, 5);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (1, 4);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (2, 5);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (3, 2);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (4, 3);

INSERT INTO REGIZEAZA (id_director, id_media)
VALUES (5, 1);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (1, 1);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (1, 3);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (2, 2);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (2, 4);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (3, 2);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (3, 5);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (4, 1);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (4, 3);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (5, 2);

INSERT INTO APARTINE (id_media, id_gen)
VALUES (5, 4);
```



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

O procedura ce nu primeste parametri si va afisa pentru fiecare film fiecare recenzie primita si suma totala a acestora din punctajul maxim posibil.

```
CREATE OR REPLACE PROCEDURE procedura6_fna IS
    TYPE tip_pentru_tablou_indexat IS RECORD
    (
        nume VARCHAR(200),
        rating NUMBER(3,1)
    );

    TYPE tip_pentru_tablou_imbricat IS RECORD
    (

        id_film NUMBER(5),
```

```
        nume VARCHAR(200)
    );

    TYPE tablou_indexat IS TABLE OF tip_pentru_tablou_indexat INDEX BY PLS_INTEGER;
    TYPE tablou_imbricat IS TABLE OF tip_pentru_tablou_imbricat;

    t_filme tablou_imbricat;
    t_recenzii tablou_indexat;
    numar NUMBER(4);
    suma NUMBER(4);
BEGIN
    SELECT id_film, nume_media
    BULK COLLECT INTO t_filme
    FROM FILME JOIN MEDIA ON (FILME.id_film = MEDIA.id_media);

    FOR i in t_filme.FIRST..t_filme.LAST LOOP
        DBMS_OUTPUT.PUT('FILM: ' || t_filme(i).nume || ' ');

        numar := 0;
        suma := 0;

        SELECT C.nume, V.rating
        BULK COLLECT INTO t_recenzii
        FROM VIZIONEAZA V JOIN CLIENTI C ON (C.id_client = V.id_client)
        WHERE V.id_media = t_filme(i).id_film AND V.rating IS NOT NULL;

        SELECT COUNT(rating) * 5, SUM(rating)
        INTO numar, suma
        FROM VIZIONEAZA
        WHERE id_media = t_filme(i).id_film;

        IF numar > 0 THEN
            DBMS_OUTPUT.PUT_LINE(suma || '/' || numar);
            FOR j in t_recenzii.FIRST..t_recenzii.LAST LOOP
                DBMS_OUTPUT.PUT_LINE(t_recenzii(j).nume || ' ' || t_recenzii(j).rating);
            END LOOP;
        ELSE
            DBMS_OUTPUT.PUT_LINE('');
            DBMS_OUTPUT.PUT_LINE('Acest film nu are nicio recenzie inca');
        END IF;
    END LOOP;
END;
/
begin
    procedura6_fna;
end;
/
```

```
15    TYPE tablou_indexat IS TABLE OF tip_pentru_tablou_indexat INDEX BY PLS_INTEGER;
16    TYPE tablou_imbricat IS TABLE OF tip_pentru_tablou_imbricat;
17
18    t_filme tablou_imbricat;
19    t_recenzii tablou_indexat;
20    numar NUMBER(4);
21    suma NUMBER(4);
22  BEGIN
23    SELECT id_film, nume_media
24    BULK COLLECT INTO t_filme
25    FROM FILME JOIN MEDIA ON (FILME.id_film = MEDIA.id_media);
26
27    FOR i in t_filme.FIRST..t_filme.LAST LOOP
28       DBMS_OUTPUT.PUT('FILM: ' || t_filme(i).nume || ' ');
29
30       numar := 0;
31       suma := 0;
32
33       SELECT C.nume, V.rating
34       BULK COLLECT INTO t_recenzii
35       FROM VIZIONEAZA V JOIN CLIENTI C ON (C.id_client = V.id_client)
36       WHERE V.id_media = t_filme(i).id_film AND V.rating IS NOT NULL;
37
38       SELECT COUNT(rating) * 10, SUM(rating)
39       INTO numar, suma
40       FROM VIZIONEAZA
41       WHERE id_media = t_filme(i).id_film;
42
43       IF numar > 0 THEN
44          DBMS_OUTPUT.PUT_LINE(suma || '/' || numar);
45          FOR j in t_recenzii.FIRST..t_recenzii.LAST LOOP
46             DBMS_OUTPUT.PUT_LINE(t_recenzii(j).nume || ' ' || t_recenzii(j).rating);
47          END LOOP;
48       ELSE
49          DBMS_OUTPUT.PUT_LINE('');
50          DBMS_OUTPUT.PUT_LINE('Acest film nu are nicio recenzie inca');
51       END IF;
52    END LOOP;
53  END;
54  /

ROCEDURE created.

ommit complete.
```

```
SQL> BEGIN
  2     procedura6_fna;
  3  END;

FILM: Inception 5/5
Wilson 4.7
FILM: The Avengers
Acest film nu are nicio recenzie inca
FILM: Inception
Acest film nu are nicio recenzie inca
FILM: The Dark Knight
Acest film nu are nicio recenzie inca
FILM: Avengers: Endgame 14/15
Davis 4.8
Wilson 4.6
Smith 4.5

PL/SQL procedure successfully completed.

Commit complete.
```

Exercitiul 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. Apelați subprogramul.

O procedura ce primeste drept parametru numele unui gen de media si determina pentru toate filmele si serialele ce apartin genului respectiv numele fiecarui client ce l-a vizionat impreuna cu denumirea abonamentului sau.

```
CREATE OR REPLACE PROCEDURE procedura7_fna (p_proc GEN.nume_gen%TYPE) IS
    CURSOR c IS
        SELECT M.id_media id_media, M.nume_media nume
        FROM MEDIA M JOIN APARTINE A ON (A.id_media = M.id_media)
        JOIN GEN G ON (G.id_gen = A.id_gen)
        WHERE G.nume_gen = p_proc;

    CURSOR d(p_curs VIZIONEAZA.id_media%TYPE) IS
        SELECT C.nume nume, C.prenume prenume, P.nume nume_abonament
        FROM VIZIONEAZA V JOIN CLIENTI C ON (C.id_client = V.id_client)
        JOIN PLANURI P ON (P.id_plan = C.id_plan)
        WHERE V.id_media = p_curs;

    id_pentru_cursor MEDIA.id_media%TYPE;
    nume_media MEDIA.nume_media%TYPE;
    numar_media NUMBER :=0;
    numar_clienti NUMBER :=0;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Pentru genul ' || p_proc || ' : ');
    OPEN c;
    LOOP
        FETCH c into id_pentru_cursor, nume_media;
        EXIT WHEN c%NOTFOUND;

        numar_media := numar_media + 1;
        DBMS_OUTPUT.PUT_LINE(nume_media || ': ');
        numar_clienti :=0;
        FOR i in d(id_pentru_cursor) LOOP
            numar_clienti := numar_clienti + 1;
            DBMS_OUTPUT.PUT_LINE(numar_clienti || '. ' || i.prenume || ' ' || i.nume || ' cu abonament de tipul ' || i.nume_abonament );
        END LOOP;

        IF numar_clienti = 0 THEN
            DBMS_OUTPUT.PUT_LINE('Nimeni nu a vizionat acest film/serial.');
        END IF;
    END LOOP;
    CLOSE c;
    IF numar_media = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista filme/seriale ce apartin acestui gen.');
    END IF;
```

```
END;
/
begin
    procedura7('Action');
end;
/
```

```
3         SELECT M.id_media id_media, M.nume_media nume
4         FROM MEDIA M JOIN APARTINE A ON (A.id_media = M.id_media)
5         JOIN GEN G ON (G.id_gen = A.id_gen)
6         WHERE G.nume_gen = p_proc;
7
8     CURSOR d(p_curs VIZIONEAZA.id_media%TYPE) IS
9         SELECT C.nume nume, C.prenume prenume, P.nume nume_abonament
10          FROM VIZIONEAZA V JOIN CLIENTI C ON (C.id_client = V.id_client)
11          JOIN PLANURI P ON (P.id_plan = C.id_plan)
12          WHERE V.id_media = p_curs;
13
14    id_pentru_cursor MEDIA.id_media%TYPE;
15    nume_media MEDIA.nume_media%TYPE;
16    numar_media NUMBER :=0;
17    numar_clienti NUMBER :=0;
18  BEGIN
19    DBMS_OUTPUT.PUT_LINE('Pentru genul ' || p_proc || ' : ');
20    OPEN c;
21    LOOP
22       FETCH c into id_pentru_cursor, nume_media;
23       EXIT WHEN c%NOTFOUND;
24
25       numar_media := numar_media + 1;
26       DBMS_OUTPUT.PUT_LINE(nume_media || ': ');
27       numar_clienti :=0;
28       FOR i in d(id_pentru_cursor) LOOP
29          numar_clienti := numar_clienti + 1;
30          DBMS_OUTPUT.PUT_LINE(numar_clienti || '. ' || i.prenume || ' ' || i.nume || ' cu abonament de tipul ' || i.nume_abonament );
31       END LOOP;
32
33       IF numar_clienti = 0 THEN
34          DBMS_OUTPUT.PUT_LINE('Nimeni nu a vizionat acest film/serial.');
35       END IF;
36    END LOOP;
37    CLOSE c;
38    IF numar_media = 0 THEN
39       DBMS_OUTPUT.PUT_LINE('Nu exista filme/seriale ce apartin acestui gen.');
40    END IF;
41  END;
42  /

PROCEDURE created.

Commit complete.
```

```
SQL> begin
  2     procedura7_fna('Action');
  3  end;
  4  /

Pentru genul Action :
Avengers: Endgame:
1. Michael Davis cu abonament de tipul Premium Plan
2. Oliver Wilson cu abonament de tipul Mobile Plan
3. John Smith cu abonament de tipul Basic Plan
Inception:
1. Oliver Wilson cu abonament de tipul Mobile Plan

PL/SQL procedure successfully completed.

Commit complete.
```

Exercitiul 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.

Functie ce va primi numele de familie al unui client si va intoarce bugetul total al tuturor filmelor si serialelor vizionate de acesta.

```
CREATE OR REPLACE FUNCTION functie8_fna (nume_client CLIENTI.nume%TYPE)
RETURN NUMBER IS
    buget_total NUMBER;
    numar_clienti NUMBER;
    NO_CUSTOMERS EXCEPTION;
    NO_MEDIA EXCEPTION;
    TOO_MANY_CUSTOMERS EXCEPTION;
BEGIN
    SELECT SUM(M.buget)
    INTO buget_total
    FROM CLIENTI C
    JOIN VIZIONEAZA V ON C.id_client = V.id_client
    JOIN MEDIA M ON V.id_media = M.id_media
    WHERE C.nume = nume_client;

    IF buget_total IS NULL THEN
        SELECT COUNT(*)
        INTO numar_clienti
        FROM CLIENTI
        WHERE CLIENTI.nume = nume_client;

        IF numar_clienti <= 0 THEN
            RAISE NO_CUSTOMERS;
        ELSE
            RAISE NO_MEDIA;
        END IF;
    END IF;

    RETURN buget_total;

EXCEPTION
    WHEN NO_CUSTOMERS THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista client cu acest nume');
        RETURN -1;

    WHEN NO_MEDIA THEN
        DBMS_OUTPUT.PUT_LINE('Clientul acesta nu a vizionat inca niciun film/serial');
        RETURN -1;

    WHEN TOO_MANY_CUSTOMERS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multi clienti cu acest nume!');
        RETURN -1;

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE(SQLCODE);
```

```
        DBMS_OUTPUT.PUT_LINE(SQLERRM);
        RETURN -1;
END;
/

begin
    DBMS_OUTPUT.PUT_LINE(functie8_fna('Popescu'));
    DBMS_OUTPUT.PUT_LINE(functie8_fna('Smith'));
    DBMS_OUTPUT.PUT_LINE(functie8_fna('Johnson'));
end;
/
```

```
9     SELECT SUM(M.buget)
10      INTO buget_total
11      FROM CLIENTI C
12      JOIN VIZIONEAZA V ON C.id_client = V.id_client
13      JOIN MEDIA M ON V.id_media = M.id_media
14      WHERE C.nume = nume_client;
15
16      IF buget_total IS NULL THEN
17         SELECT COUNT(*)
18         INTO numar_clienti
19         FROM CLIENTI
20         WHERE CLIENTI.nume = nume_client;
21
22         IF numar_clienti <= 0 THEN
23            RAISE NO_CUSTOMERS;
24         ELSE
25            RAISE NO_MEDIA;
26         END IF;
27      END IF;
28
29      RETURN buget_total;
30
31  EXCEPTION
32      WHEN NO_CUSTOMERS THEN
33         DBMS_OUTPUT.PUT_LINE('Nu exista client cu acest nume');
34         RETURN -1;
35
36      WHEN NO_MEDIA THEN
37         DBMS_OUTPUT.PUT_LINE('Clientul acesta nu a vizionat inca niciun film/serial');
38         RETURN -1;
39
40      WHEN TOO_MANY_CUSTOMERS THEN
41         DBMS_OUTPUT.PUT_LINE('Exista mai multi clienti cu acest nume!');
42         RETURN -1;
43
44      WHEN OTHERS THEN
45         DBMS_OUTPUT.PUT_LINE(SQLCODE);
46         DBMS_OUTPUT.PUT_LINE(SQLERRM);
47         RETURN -1;
48  END;
49  /

FUNCTION created.
```

```
SQL> begin
  2     DBMS_OUTPUT.PUT_LINE(functie8_fna('Popescu'));
  3     DBMS_OUTPUT.PUT_LINE(functie8_fna('Smith'));
  4     DBMS_OUTPUT.PUT_LINE(functie8_fna('Johnson'));
  5  end;
  6  /

Nu exista client cu acest nume
-1
364000000
Clientul acesta nu a vizionat inca niciun film/serial
-1

PL/SQL procedure successfully completed.

Commit complete.
```

Exercitiul 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.

O procedura ce primeste drept parametru prenumele unui actor si afiseaza numele sau complet numarul de filme si seriale in care a jucat rating-ul mediu si total al filmelor si serialelor in care a jucat precum si numarul de spectatori al filmelor si serialelor in care a jucat

```
CREATE OR REPLACE PROCEDURE procedura9_fna (prenume_actor ACTORI.prenume%TYPE) IS
    nume ACTORI.nume%TYPE;
    nr_media NUMBER;
    buget_media NUMBER;
    numar_rating NUMBER;
    suma_rating NUMBER;
    avg_rating NUMBER;
    numar_clienti NUMBER;
BEGIN
    SELECT A.nume, COUNT(J.id_media), SUM(M.buget), COUNT(V.rating)*5, SUM(V.rating), ROUND(AVG(V.rating), 2), COUNT(C.id_client)
    INTO nume, nr_media, buget_media, numar_rating, suma_rating, avg_rating, numar_clienti
    FROM ACTORI A LEFT JOIN JOACA J ON (J.id_actor = A.id_actor)
    LEFT JOIN MEDIA M ON (M.id_media = J.id_media)
    LEFT JOIN VIZIONEAZA V ON (V.id_media = M.id_media)
    LEFT JOIN CLIENTI C ON (C.id_client = V.id_client)
    WHERE A.prenume = prenume_actor
    GROUP BY A.nume;

    DBMS_OUTPUT.PUT_LINE('Actorul ' || prenume_actor || ' ' || nume || ': ');
    DBMS_OUTPUT.PUT_LINE('Numar filme/seriale in care a jucat:' || nr_media);
    DBMS_OUTPUT.PUT_LINE('Rating mediu: '|| avg_rating || '/ 5');
    DBMS_OUTPUT.PUT_LINE('Rating total: ' || suma_rating || '/' || numar_rating);
    DBMS_OUTPUT.PUT_LINE('Numar spectatori:' || numar_clienti);

EXCEPTION
    WHEN TOO_MANY_ROWS
    THEN RAISE_APPLICATION_ERROR(-20007, 'Exista mai multi actori cu acest prenume');
    WHEN NO_DATA_FOUND
    THEN RAISE_APPLICATION_ERROR(-20006, 'Nu exista actori cu acest prenume');
END;
/
begin
    procedura9_fna('Olivia');
    procedura9_fna('David');
    --procedura9_fna('Brad');
end;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE procedura9_fna (prenume_actor ACTORI.prenume%TYPE) IS
  2     nume ACTORI.nume%TYPE;
  3     nr_media NUMBER;
  4     buget_media NUMBER;
  5     numar_rating NUMBER;
  6     suma_rating NUMBER;
  7     avg_rating NUMBER;
  8     numar_clienti NUMBER;
  9  BEGIN
 10     SELECT A.nume, COUNT(J.id_media), SUM(M.buget), COUNT(V.rating)*10, SUM(V.rating), ROUND(AVG(V.rating), 2), COUNT(C.id_client)
 11     INTO nume, nr_media, buget_media, numar_rating, suma_rating, avg_rating, numar_clienti
 12     FROM ACTORI A LEFT JOIN JOACA J ON (J.id_actor = A.id_actor)
 13     LEFT JOIN MEDIA M ON (M.id_media = J.id_media)
 14     LEFT JOIN VIZIONEAZA V ON (V.id_media = M.id_media)
 15     LEFT JOIN CLIENTI C ON (C.id_client = V.id_client)
 16     WHERE A.prenume = prenume_actor
 17     GROUP BY A.nume;
 18
 19     DBMS_OUTPUT.PUT_LINE('Actorul ' || prenume_actor || ' ' || nume || ': ');
 20     DBMS_OUTPUT.PUT_LINE('Numar filme/seriale in care a jucat:' || nr_media);
 21     DBMS_OUTPUT.PUT_LINE('Rating mediu: '|| avg_rating || '/ 10');
 22     DBMS_OUTPUT.PUT_LINE('Rating total: ' || suma_rating || '/' || numar_rating);
 23     DBMS_OUTPUT.PUT_LINE('Numar spectatori:' || numar_clienti);
 24
 25  EXCEPTION
 26     WHEN TOO_MANY_ROWS
 27     THEN RAISE_APPLICATION_ERROR(-20007, 'Exista mai multi actori cu acest prenume');
 28     WHEN NO_DATA_FOUND
 29     THEN RAISE_APPLICATION_ERROR(-20006, 'Nu exista actori cu acest prenume');
 30  END;
 31  /

PROCEDURE created.

Commit complete.
```

```
SQL> begin
  2     procedura9_fna('Olivia');
  3     procedura9_fna('David');
  4     --procedura9_fna('Brad');
  5  end;
  6  /

Actorul Olivia Colman:
Numar filme/seriale in care a jucat:3
Rating mediu: 4.27/ 5
Rating total: 12.8/15
Numar spectatori:3
Actorul David Harbour:
Numar filme/seriale in care a jucat:3
Rating mediu: 3.87/ 5
Rating total: 11.6/15
Numar spectatori:3

PL/SQL procedure successfully completed.

Commit complete.
```

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

Trigger te tip LMD la nivel de comanda ce impiedica stergerea filmelor si serialelor din lista de vizionate.

```
CREATE OR REPLACE TRIGGER trigger10_fna
    BEFORE DELETE ON VIZIONEAZA
BEGIN
    RAISE_APPLICATION_ERROR(-20011, 'Nu puteti elimina un film/serial din lista de vizionate!');
END;
/
DELETE FROM VIZIONEAZA
WHERE id_client = 4;
/
```

```
Commit complete.

SQL> CREATE OR REPLACE TRIGGER trigger10_fna
  2    BEFORE DELETE ON VIZIONEAZA
  3  BEGIN
  4    RAISE_APPLICATION_ERROR(-20011, 'Nu puteti elimina un film/serial din lista de vizionate!');
  5  END;
  6  /

TRIGGER created.

Commit complete.

SQL> DELETE FROM VIZIONEAZA
  2  WHERE id_client = 4;

DELETE FROM VIZIONEAZA
            *
ERROR at line 7:
ORA-20011: Nu puteti elimina un film/serial din lista de vizionate!
ORA-06512: at "GRUPA244.TRIGGER10_FNA", line 2
ORA-04088: error during execution of trigger 'GRUPA244.TRIGGER10_FNA'
```

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

Trigger de tip LMD la nivel de linie ce impiedica crearea sau modificarea unui rating oferit de un utilizator astfel incat acesta sa fie mai mare de 5.

```
CREATE OR REPLACE TRIGGER trigger_11_fna
    BEFORE INSERT OR UPDATE ON VIZIONEAZA
    FOR EACH ROW
BEGIN
    IF(:NEW.rating > 5) THEN
        RAISE_APPLICATION_ERROR(-20010, 'Nota accordata unui film/serial nu poate fi mai mare de 10!');
    END IF;
END;
/
INSERT INTO VIZIONEAZA
VALUES(1, 12, 7.5);

UPDATE VIZIONEAZA
SET rating = 7
WHERE id_client = 1 AND id_media = 1;
/
```

```
SQL> CREATE OR REPLACE TRIGGER trigger_11_fna
  2     BEFORE INSERT OR UPDATE ON VIZIONEAZA
  3     FOR EACH ROW
  4  BEGIN
  5     IF(:NEW.rating > 5) THEN
  6        RAISE_APPLICATION_ERROR(-20010, 'Nota accordata unui film/serial nu poate fi mai mare de 10!');
  7     END IF;
  8  END;
  9  /

TRIGGER created.

Commit complete.

SQL> INSERT INTO VIZIONEAZA
  2  VALUES(1, 12, 7.5);

INSERT INTO VIZIONEAZA
            *
ERROR at line 10:
ORA-20010: Nota accordata unui film/serial nu poate fi mai mare de 10!
ORA-06512: at "GRUPA244.TRIGGER_11_FNA", line 3
ORA-04088: error during execution of trigger 'GRUPA244.TRIGGER_11_FNA'
SQL> UPDATE VIZIONEAZA
  2  SET rating = 7
  3  WHERE id_client = 1 AND id_media = 1;

UPDATE VIZIONEAZA
       *
ERROR at line 13:
ORA-20010: Nota accordata unui film/serial nu poate fi mai mare de 10!
ORA-06512: at "GRUPA244.TRIGGER_11_FNA", line 3
ORA-04088: error during execution of trigger 'GRUPA244.TRIGGER_11_FNA'
```

Exercitiul 12 - Definiți un triggerde tip LDD. Declanșați trigger-ul.

Trigger de tip LDD ce afiseaza faptul ca a fost executata o comanda LDD de fiecare data cand este cazul

```
CREATE OR REPLACE TRIGGER trigger12_fna
    AFTER CREATE OR ALTER OR DROP ON SCHEMA
BEGIN
    DBMS_OUTPUT.PUT_LINE('A fost efectuata o comanda LDD');
END;
/

ALTER TABLE CLIENTI
ADD data_nasterii DATE;

ALTER TABLE CLIENTI
DROP COLUMN data_nasterii;
/
```

```
SQL> CREATE OR REPLACE TRIGGER trigger12_fna
  2     AFTER CREATE OR ALTER OR DROP ON SCHEMA
  3  BEGIN
  4     DBMS_OUTPUT.PUT_LINE('A fost efectuata o comanda LDD');
  5  END;
  6  /

TRIGGER created.

Commit complete.
```

```
SQL> CREATE OR REPLACE PACKAGE pachet_proiect_fna
  2 IS
  3     PROCEDURE procedura6_fna;
  4     PROCEDURE procedura7_fna(p_proc GEN.nume_gen%TYPE);
  5     FUNCTION functie8_fna(nume_client CLIENTI.nume%TYPE) RETURN NUMBER;
  6     PROCEDURE procedura9_fna(prenume_actor ACTORI.prenume%TYPE);
  7  END;

A fost efectuata o comanda LDD
```

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

```
CREATE OR REPLACE PACKAGE pachet_proiect_fna
IS
    PROCEDURE procedura6_fna;
    PROCEDURE procedura7_fna(p_proc GEN.nume_gen%TYPE);
    FUNCTION functie8_fna(nume_client CLIENTI.nume%TYPE) RETURN NUMBER;
    PROCEDURE procedura9_fna(prenume_actor ACTORI.prenume%TYPE);
END;
/
CREATE OR REPLACE PACKAGE BODY pachet_proiect_fna
IS
    PROCEDURE procedura6_fna IS
        TYPE tip_pentru_tablou_indexat IS RECORD
        (
            nume VARCHAR(200),
            rating NUMBER(3,1)
        );

        TYPE tip_pentru_tablou_imbricat IS RECORD
        (

            id_film NUMBER(5),
            nume VARCHAR(200)
        );

        TYPE tablou_indexat IS TABLE OF tip_pentru_tablou_indexat INDEX BY PLS_INTEGER;
        TYPE tablou_imbricat IS TABLE OF tip_pentru_tablou_imbricat;

        t_filme tablou_imbricat;
        t_recenzii tablou_indexat;
        numar NUMBER(4);
        suma NUMBER(4);

    BEGIN
        SELECT id_film, nume_media
        BULK COLLECT INTO t_filme
        FROM FILME JOIN MEDIA ON (FILME.id_film = MEDIA.id_media);

        FOR i in t_filme.FIRST..t_filme.LAST LOOP
            DBMS_OUTPUT.PUT('FILM: ' || t_filme(i).nume || ' ');

            numar := 0;
            suma := 0;
            --t_recenzii.DELETE(1);

            SELECT C.nume, V.rating
            BULK COLLECT INTO t_recenzii
            FROM VIZIONEAZA V JOIN CLIENTI C ON (C.id_client = V.id_client)
            WHERE V.id_media = t_filme(i).id_film AND V.rating IS NOT NULL;

            SELECT COUNT(rating) * 5, SUM(rating)
            INTO numar, suma
            FROM VIZIONEAZA
            WHERE id_media = t_filme(i).id_film;

            IF numar > 0 THEN
                DBMS_OUTPUT.PUT_LINE(suma || '/' || numar);
                FOR j in t_recenzii.FIRST..t_recenzii.LAST LOOP
                    DBMS_OUTPUT.PUT_LINE(t_recenzii(j).nume || ' ' || t_recenzii(j).rating);
                END LOOP;
            ELSE
                DBMS_OUTPUT.PUT_LINE('');
                DBMS_OUTPUT.PUT_LINE('Acest film nu are nicio recenzie inca');
```

```
        END IF;
    END LOOP;
END procedura6_fna;

PROCEDURE procedura7_fna (p_proc GEN.nume_gen%TYPE) IS
    CURSOR c IS
    SELECT M.id_media id_media, M.nume_media nume
    FROM MEDIA M JOIN APARTINE A ON (A.id_media = M.id_media)
    JOIN GEN G ON (G.id_gen = A.id_gen)
    WHERE G.nume_gen = p_proc;

    CURSOR d(p_curs VIZIONEAZA.id_media%TYPE) IS
    SELECT C.nume nume, C.prenume prenume, P.nume nume_abonament
    FROM VIZIONEAZA V JOIN CLIENTI C ON (C.id_client = V.id_client)
    JOIN PLANURI P ON (P.id_plan = C.id_plan)
    WHERE V.id_media = p_curs;

    id_pentru_cursor MEDIA.id_media%TYPE;
    nume_media MEDIA.nume_media%TYPE;
    numar_media NUMBER :=0;
    numar_clienti NUMBER :=0;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Pentru genul ' || p_proc || ' : ');
    OPEN c;
    LOOP
        FETCH c into id_pentru_cursor, nume_media;
        EXIT WHEN c%NOTFOUND;

        numar_media := numar_media + 1;
        DBMS_OUTPUT.PUT_LINE(nume_media || ': ');
        numar_clienti :=0;
        FOR i in d(id_pentru_cursor) LOOP
            numar_clienti := numar_clienti + 1;
            DBMS_OUTPUT.PUT_LINE(numar_clienti || '. ' || i.prenume || ' ' || i.nume || ' cu abonament de tipul ' || i.nume_abonament )
        END LOOP;

        IF numar_clienti = 0 THEN
            DBMS_OUTPUT.PUT_LINE('Nimeni nu a vizionat acest film/serial.');
        END IF;
    END LOOP;
    CLOSE c;
    IF numar_media = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista filme/seriale ce apartin acestui gen.');
    END IF;
END procedura7_fna;


FUNCTION functie8_fna (nume_client CLIENTI.nume%TYPE) RETURN NUMBER IS
    buget_total NUMBER;
    numar_clienti NUMBER;
    NO_CUSTOMERS EXCEPTION;
    NO_MEDIA EXCEPTION;
    TOO_MANY_CUSTOMERS EXCEPTION;
BEGIN
    SELECT SUM(M.buget)
    INTO buget_total
    FROM CLIENTI C
    JOIN VIZIONEAZA V ON C.id_client = V.id_client
    JOIN MEDIA M ON V.id_media = M.id_media
    WHERE C.nume = nume_client;

    IF buget_total IS NULL THEN
        SELECT COUNT(*)
        INTO numar_clienti
        FROM CLIENTI
        WHERE CLIENTI.nume = nume_client;

        IF numar_clienti <= 0 THEN
            RAISE NO_CUSTOMERS;
        ELSE
            RAISE NO_MEDIA;
        END IF;
    END IF;

    RETURN buget_total;

EXCEPTION
    WHEN NO_CUSTOMERS THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista client cu acest nume');
        RETURN -1;
```

```
        WHEN NO_MEDIA THEN
            DBMS_OUTPUT.PUT_LINE('Clientul acesta nu a vizionat inca niciun film/serial');
            RETURN -1;

        WHEN TOO_MANY_CUSTOMERS THEN
            DBMS_OUTPUT.PUT_LINE('Exista mai multi clienti cu acest nume!');
            RETURN -1;

        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE(SQLCODE);
            DBMS_OUTPUT.PUT_LINE(SQLERRM);
            RETURN -1;
    END functie8_fna;


    PROCEDURE procedura9_fna (prenume_actor ACTORI.prenume%TYPE) IS
        nume ACTORI.nume%TYPE;
        nr_media NUMBER;
        buget_media NUMBER;
        numar_rating NUMBER;
        suma_rating NUMBER;
        avg_rating NUMBER;
        numar_clienti NUMBER;
    BEGIN
        SELECT A.nume, COUNT(J.id_media), SUM(M.buget), COUNT(V.rating)*5, SUM(V.rating), ROUND(AVG(V.rating), 2), COUNT(C.id_client)
        INTO nume, nr_media, buget_media, numar_rating, suma_rating, avg_rating, numar_clienti
        FROM ACTORI A LEFT JOIN JOACA J ON (J.id_actor = A.id_actor)
        LEFT JOIN MEDIA M ON (M.id_media = J.id_media)
        LEFT JOIN VIZIONEAZA V ON (V.id_media = M.id_media)
        LEFT JOIN CLIENTI C ON (C.id_client = V.id_client)
        WHERE A.prenume = prenume_actor
        GROUP BY A.nume;

        DBMS_OUTPUT.PUT_LINE('Actorul ' || prenume_actor || ' ' || nume || ': ');
        DBMS_OUTPUT.PUT_LINE('Numar filme/seriale in care a jucat:' || nr_media);
        DBMS_OUTPUT.PUT_LINE('Rating mediu: '|| avg_rating || '/ 5');
        DBMS_OUTPUT.PUT_LINE('Rating total: ' || suma_rating || '/' || numar_rating);
        DBMS_OUTPUT.PUT_LINE('Numar spectatori:' || numar_clienti);

    EXCEPTION
        WHEN TOO_MANY_ROWS
        THEN RAISE_APPLICATION_ERROR(-20007, 'Exista mai multi actori cu acest prenume');
        WHEN NO_DATA_FOUND
        THEN RAISE_APPLICATION_ERROR(-20006, 'Nu exista actori cu acest prenume');
    END procedura9_fna;

END pachet_proiect_fna;
/
begin
    pachet_proiect_fna.procedura6_fna;
    pachet_proiect_fna.procedura7_fna('Action');
end;
/
```

```
SQL> CREATE OR REPLACE PACKAGE pachet_proiect_fna
  2  IS
  3     PROCEDURE procedura6_fna;
  4     PROCEDURE procedura7_fna(p_proc GEN.nume_gen%TYPE);
  5     FUNCTION functie8_fna(nume_client CLIENTI.nume%TYPE) RETURN NUMBER;
  6     PROCEDURE procedura9_fna(prenume_actor ACTORI.prenume%TYPE);
  7  END;
  8  /

A fost efectuata o comanda LDD

PACKAGE created.

Commit complete.
```

```
146             DBMS_OUTPUT.PUT_LINE(SQLERRM);
147             RETURN -1;
148     END functie8_fna;
149
150
151     PROCEDURE procedura9_fna (prenume_actor ACTORI.prenume%TYPE) IS
152         nume ACTORI.nume%TYPE;
153         nr_media NUMBER;
154         buget_media NUMBER;
155         numar_rating NUMBER;
156         suma_rating NUMBER;
157         avg_rating NUMBER;
158         numar_clienti NUMBER;
159     BEGIN
160         SELECT A.nume, COUNT(J.id_media), SUM(M.buget), COUNT(V.rating)*5, SUM(V.rating), ROUND(AVG(V.rating), 2), COUNT(C.id_cli
161         INTO nume, nr_media, buget_media, numar_rating, suma_rating, avg_rating, numar_clienti
162         FROM ACTORI A LEFT JOIN JOACA J ON (J.id_actor = A.id_actor)
163         LEFT JOIN MEDIA M ON (M.id_media = J.id_media)
164         LEFT JOIN VIZIONEAZA V ON (V.id_media = M.id_media)
165         LEFT JOIN CLIENTI C ON (C.id_client = V.id_client)
166         WHERE A.prenume = prenume_actor
167         GROUP BY A.nume;
168
169         DBMS_OUTPUT.PUT_LINE('Actorul ' || prenume_actor || ' ' || nume || ': ');
170         DBMS_OUTPUT.PUT_LINE('Numar filme/seriale in care a jucat:' || nr_media);
171         DBMS_OUTPUT.PUT_LINE('Rating mediu: '|| avg_rating || '/ 5');
172         DBMS_OUTPUT.PUT_LINE('Rating total: ' || suma_rating || '/' || numar_rating);
173         DBMS_OUTPUT.PUT_LINE('Numar spectatori:' || numar_clienti);
174
175     EXCEPTION
176         WHEN TOO_MANY_ROWS
177         THEN RAISE_APPLICATION_ERROR(-20007, 'Exista mai multi actori cu acest prenume');
178         WHEN NO_DATA_FOUND
179         THEN RAISE_APPLICATION_ERROR(-20006, 'Nu exista actori cu acest prenume');
180     END procedura9_fna;
181
182 END pachet_proiect_fna;
183 /

A fost efectuata o comanda LDD

PACKAGE BODY created.

Commit complete.
```

```
Commit complete.

SQL> begin
  2      pachet_proiect_fna.procedura6_fna;
  3      pachet_proiect_fna.procedura7_fna('Action');
  4  end;
  5  /

FILM: Inception 5/5
Wilson 4.7
FILM: The Avengers
Acest film nu are nicio recenzie inca
FILM: Inception
Acest film nu are nicio recenzie inca
FILM: The Dark Knight
Acest film nu are nicio recenzie inca
FILM: Avengers: Endgame 14/15
Smith 5
Davis 4.8
Wilson 4.6
Pentru genul Action :
Inception:
1. Oliver Wilson cu abonament de tipul Mobile Plan
Avengers: Endgame:
1. Michael Davis cu abonament de tipul Premium Plan
2. Oliver Wilson cu abonament de tipul Mobile Plan
3. John Smith cu abonament de tipul Basic Plan

PL/SQL procedure successfully completed.

Commit complete.
```