

PROIECT SGBD

Realizat de Stoica Elias-Valeriu
Facultatea de Matematica și Informatica Bucuresti
Grupa 251

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

Ligile din Europa pun la start echipe din diferite colțuri ale țărilor. Fiecare echipă are un stadion în orașul de proveniență, unde dispută meciurile de pe propriul teren. Stadioanele se clasifică după capacitate, după numărul de stele oferit de UEFA, depinzând de cât de modern este. O echipă este alcătuită dintr-un lot de jucători. Aceștia sunt pregătiți de un antrenor principal.

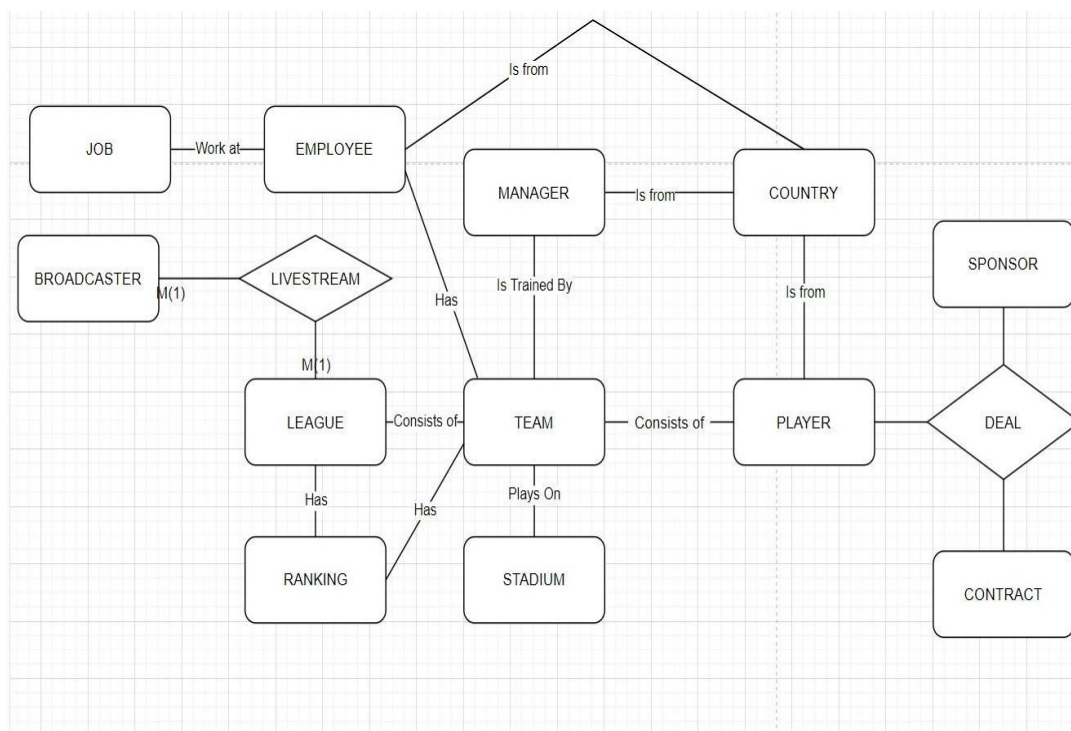
Fiecare echipă are în spate un număr mare de angajați. Aceștia primesc un salariu atractiv, în funcție de job. Pentru fiecare membru din club se cunoaște țara de proveniență.

Jucătorii sunt sponsorizați de anumite companii. Aceștia au un contract bine stabilit și nu pot avea mai multe sponsorizări în același timp. Sponsorizarile constau în produse de marca, adidas, echipament etc. Acest contract poate avea și o clauză de reziliere, în cazul în care există, jucătorul poate renunța oricând la sponsorul său.

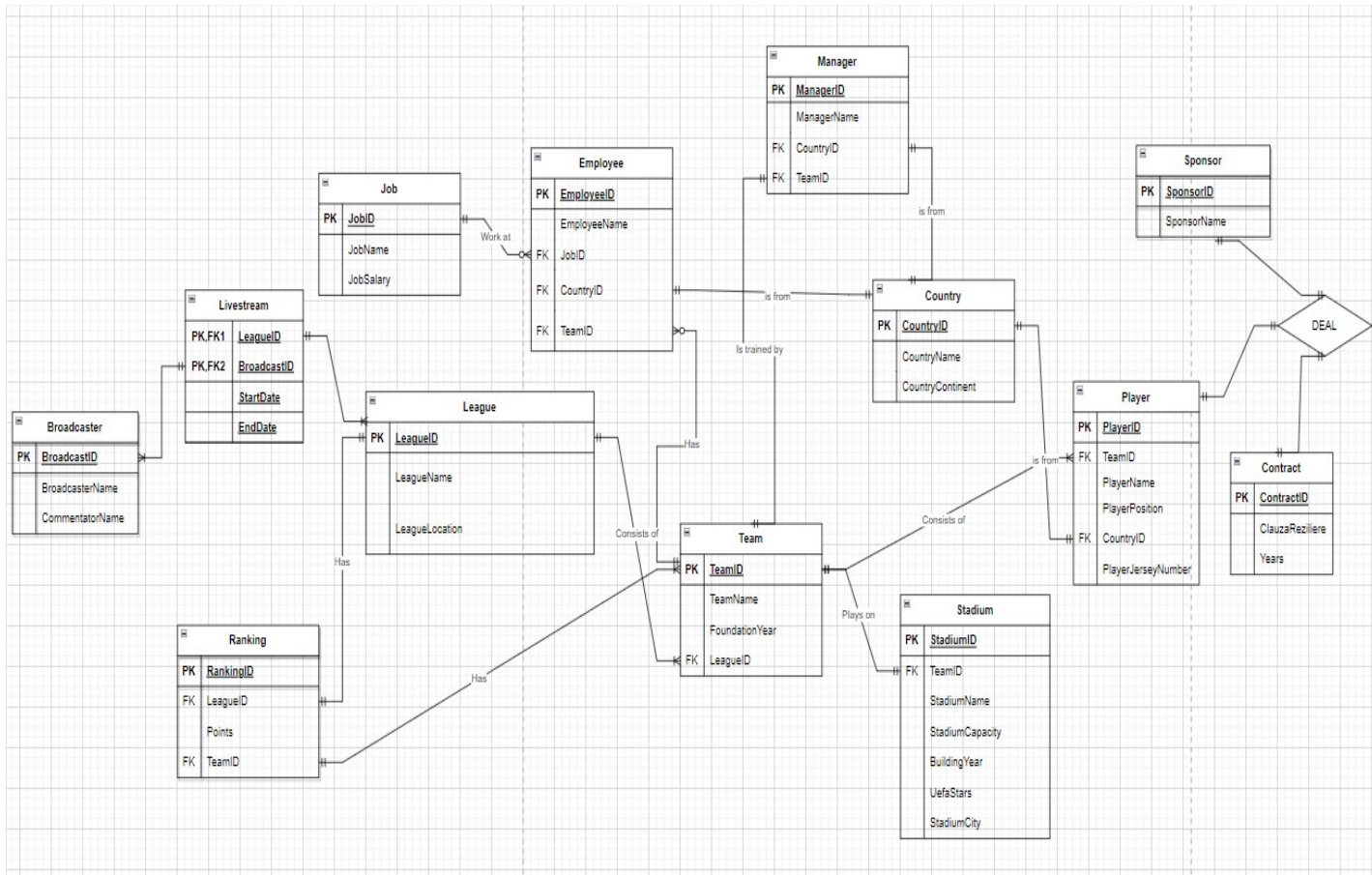
Campionatul de fotbal are mai multe contracte de livestreaming cu diferite televiziuni. Acestea sunt făcute pe o perioadă bine definită.

La finalul sezonului, echipele își află locul în clasament. În funcție de câte puncte acumulează în timpul sezonului, se pot bate pentru competițiile europene.

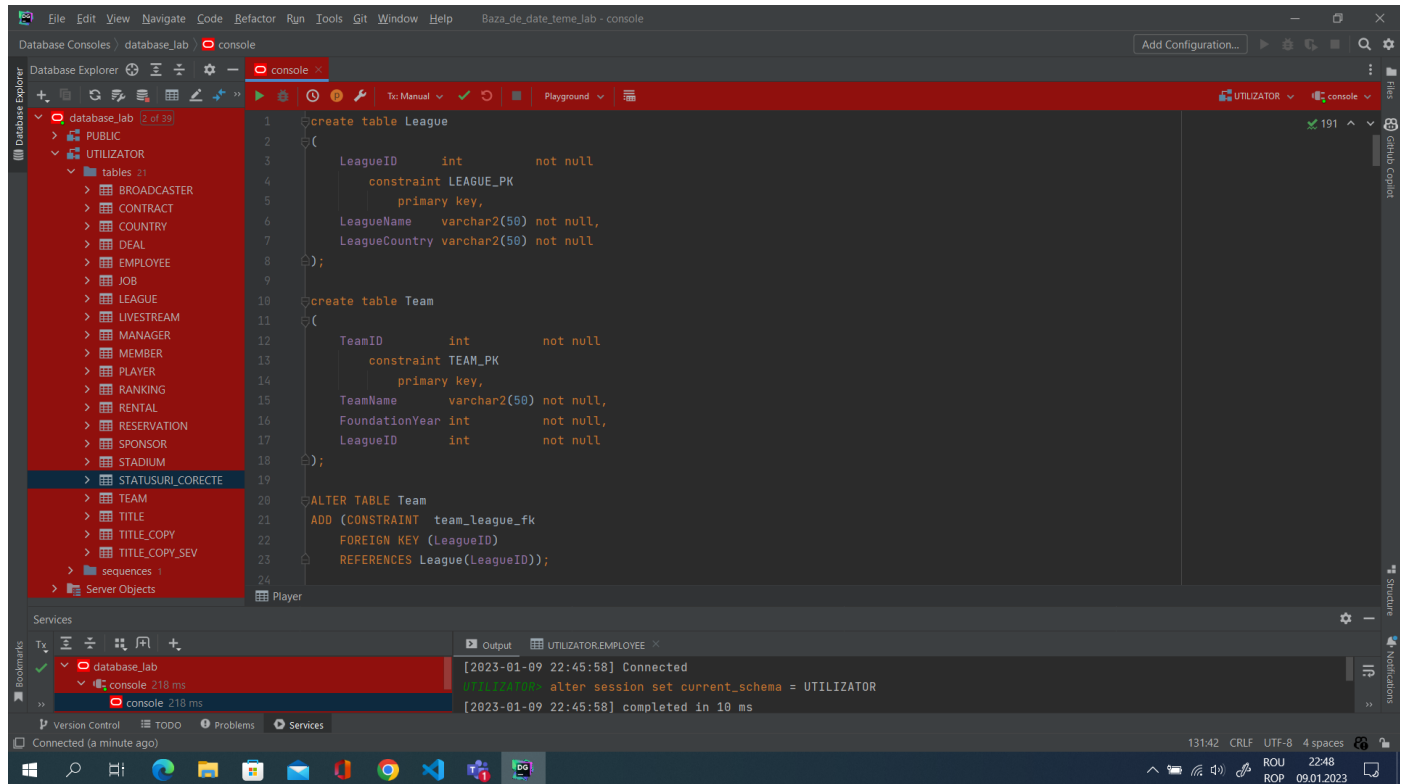
2. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.



3. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6.



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



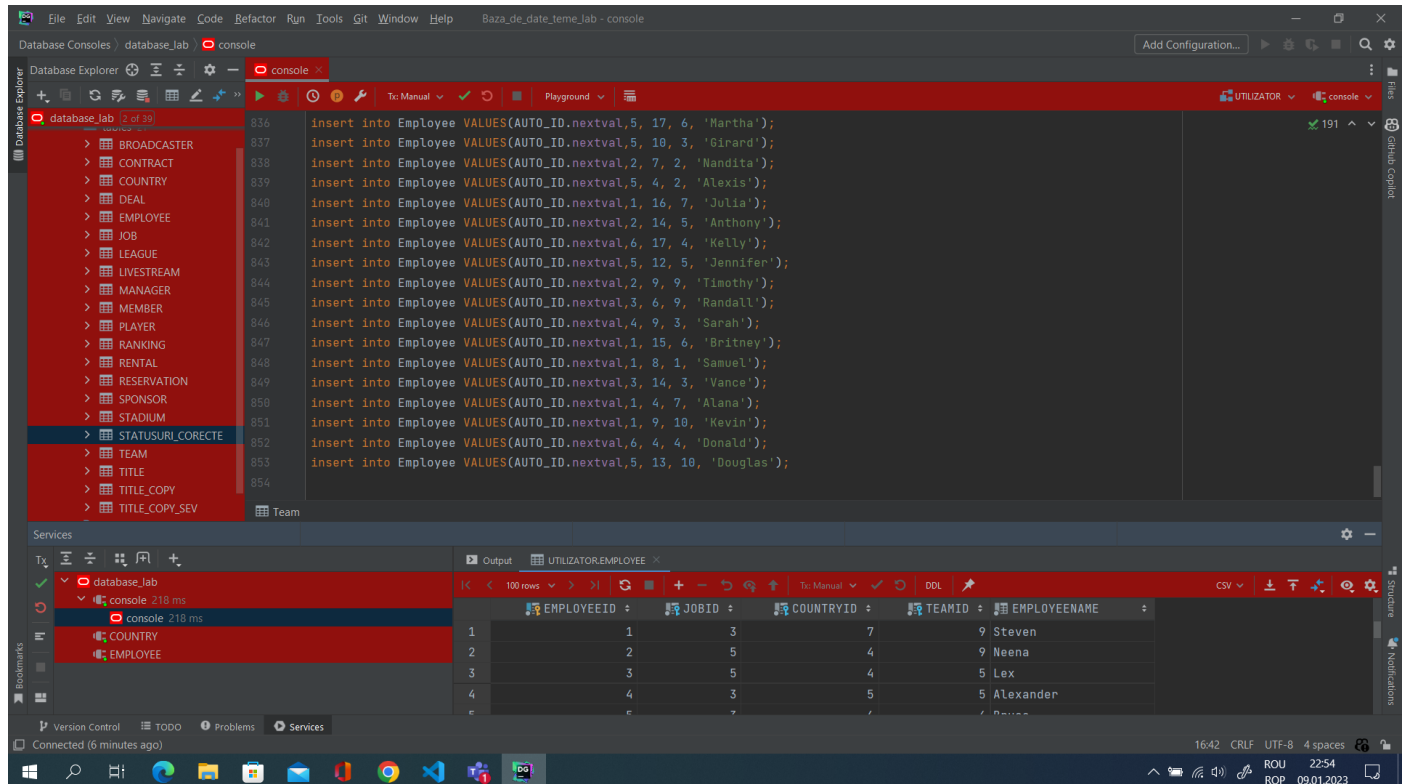
```
1 create table League
2 (
3     LeagueID      int          not null
4         constraint LEAGUE_PK
5             primary key,
6     LeagueName    varchar2(50) not null,
7     LeagueCountry varchar2(50) not null
8 );
9
10 create table Team
11 (
12     TeamID        int          not null
13         constraint TEAM_PK
14             primary key,
15     TeamName      varchar2(50) not null,
16     FoundationYear int          not null,
17     LeagueID      int          not null
18 );
19
20 ALTER TABLE Team
21 ADD (CONSTRAINT team_league_fk
22     FOREIGN KEY (LeagueID)
23     REFERENCES League(LeagueID));
24
```

The screenshot shows the Oracle SQL Developer interface. The left pane displays the Database Explorer with a tree view of the database schema, including tables like BROADCASTER, CONTRACT, COUNTRY, DEAL, EMPLOYEE, JOB, LEAGUE, LIVESTREAM, MANAGER, MEMBER, PLAYER, RANKING, RENTAL, RESERVATION, SPONSOR, STADIUM, STATUSURI_CORECTE, TEAM, TITLE, TITLE_COPY, and TITLE_COPY_SEV. The main editor window shows the SQL script for creating the League and Team tables. The League table has a primary key on LeagueID and foreign keys on LeagueName and LeagueCountry. The Team table has a primary key on TeamID and a foreign key on LeagueID. The bottom pane shows the Services window with a log of the execution, indicating that the tables were created successfully.

Fișierele de creare sunt într-un repository pe github, la adresa următoare:

<https://github.com/wood11nho/Proiect-SGBD>

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



Fișierele cu comenzile de inserare se găsesc pe următorul repository de pe github:

<https://github.com/wood11nho/Proiect-SGBD>

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

--Pentru un jucător de fotbal, căruia i se va da ID-ul de la tastatură, salvați și afișați numele tuturor coechipierilor săi, iar pentru fiecare coechipier să se salveze și sponsorul acestuia.

```
create or replace procedure afisare_coechipieri
(id_jucator_tast Player.PlayerID%type)
AS
TYPE tablou_indexat is TABLE OF Player.PlayerName%type INDEX BY PLS_INTEGER;
nume_coechipieri tablou_indexat;
TYPE tablou_imbricat is TABLE OF Sponsor.SponsorName%type;
sponsor_jucatori tablou_imbricat := tablou_imbricat();
i NUMBER;
id echipa_jucator Player.TeamID%type;
id echipa_coechipier Player.TeamID%type;
nume_jucator Player.PlayerName%type;
id_coechipier Player.PlayerID%type;
nume_coechipier Player.PlayerName%type;
nume_sponsor Sponsor.SponsorName%type;
numar_jucatori NUMBER;
BEGIN
i := 0;
select TeamID, PlayerName into id echipa_jucator, nume_jucator from Player where PlayerID
= id_jucator_tast;
select count(*) into numar_jucatori from Player;
for id_jucator in 1..numar_jucatori
loop
select TeamID into id echipa_coechipier from Player where PlayerID = id_jucator;
if id echipa_coechipier = id echipa_jucator and id_jucator_tast <> id_jucator then
i := i + 1;
select PlayerName, PlayerID into nume_coechipier, id_coechipier from Player where
PlayerID = id_jucator;
nume_coechipieri(i) := nume_coechipier;
sponsor_jucatori.extend;
select SponsorName into nume_sponsor from Sponsor where SponsorID = (select
SponsorID from Deal where PlayerID = id_coechipier);
sponsor_jucatori(i) := nume_sponsor;
```

```

        end if;
    end loop;
    DBMS_OUTPUT.PUT_LINE('Jucatorul ' || nume_jucator || ' are urmatorii coechipieri:');
    FOR j IN 1..nume_coechipieri.COUNT LOOP
        DBMS_OUTPUT.PUT_LINE(nume_coechipieri(j) || ' - ' || sponsor_jucatori(j));
    END LOOP;
END;

BEGIN
    AFISARE_COECHIPIERI(1);
end;

```

The screenshot shows an IDE with a SQL script being executed. The script defines a procedure `afisare_coechipieri` that takes a player ID as input and lists their teammates and sponsors. The execution results are shown in the bottom panel.

```

-- SQL Script
Loop
879     select TeamID into id echipa coechipier from Player where PlayerID = id_jucator;
880     if id echipa coechipier = id echipa jucator and id_jucator_tast <> id_jucator then
881         i := i + 1;
882         select PlayerName, PlayerID into nume_coechipier, id_coechipier from Player where PlayerID = id_jucator;
883         nume_coechipieri(i) := nume_coechipier;
884         sponsor_jucatori.extend;
885         select SponsorName into nume_sponsor from Sponsor where SponsorID = (select SponsorID from Deal where PlayerID = id_coechipier);
886         sponsor_jucatori(i) := nume_sponsor;
887     end if;
888 end loop;
889 DBMS_OUTPUT.PUT_LINE('Jucatorul ' || nume_jucator || ' are urmatorii coechipieri:');
890 FOR j IN 1..nume_coechipieri.COUNT LOOP
891     DBMS_OUTPUT.PUT_LINE(nume_coechipieri(j) || ' - ' || sponsor_jucatori(j));
892 END LOOP;
893 END;
894 BEGIN
895     afisare_coechipieri( id_jucator_tast );
896 END;

```

Execution Results:

```

[2023-01-10 00:47:53] completed in 33 ms
Jucatorul Dur-Bozoanca are urmatorii coechipieri:
Ursu - Nike
Yabre - Puma
Singhi - Nike
Ghiocel - Umbro
Neagu - Puma
Jardan - Nike

```

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

--Să se afișeze pentru fiecare echipă numărul de jucători. În cazul în care echipa nu are jucători înscrși, se va afișa un mesaj corespunzător. Pentru fiecare jucător cu numărul de pe tricou mai mare decât un parametru dat, să se afișeze poziția, numărul de pe tricou și țara de proveniență.

```
CREATE OR REPLACE FUNCTION filtrare_jucatori_dupa_numar
  (numar_echipa Player.TeamID%TYPE, numar_tricou Player.JerseyNumber%TYPE)
RETURN NUMBER is
  nr_jucatori_filtrati NUMBER;
BEGIN
  select count(*) into nr_jucatori_filtrati
  from Player
  where JerseyNumber > numar_tricou and TeamID = numar_echipa;
  return nr_jucatori_filtrati;
end;

CREATE OR REPLACE PROCEDURE afisare_detalii_echipe AS
  CURSOR c1 is
    SELECT t.TeamID id_echipa, TeamName nume_echipa, COUNT(PlayerID) numar_jucatori
    FROM Team t, Player p
    WHERE t.TeamID = p.TeamID(+)
    GROUP BY t.TeamID, TeamName;
  CURSOR c2(parametru NUMBER, echipa_id Team.TeamID%TYPE) is
    SELECT PlayerName nume_jucator, PlayerPosition pozitie_jucator, CountryName
    tara_jucator, JerseyNumber numar_jucator
    FROM Player p, Country c
    WHERE p.CountryID = c.CountryID(+) and JerseyNumber > parametru and TeamID =
    echipa_id;
  TYPE tip_detalii_echipa IS RECORD(
    id_echipa Team.TeamID%TYPE,
    nume_echipa Team.TeamName%TYPE,
    numar_jucatori NUMBER
  );
  detalii_echipa tip_detalii_echipa;
```

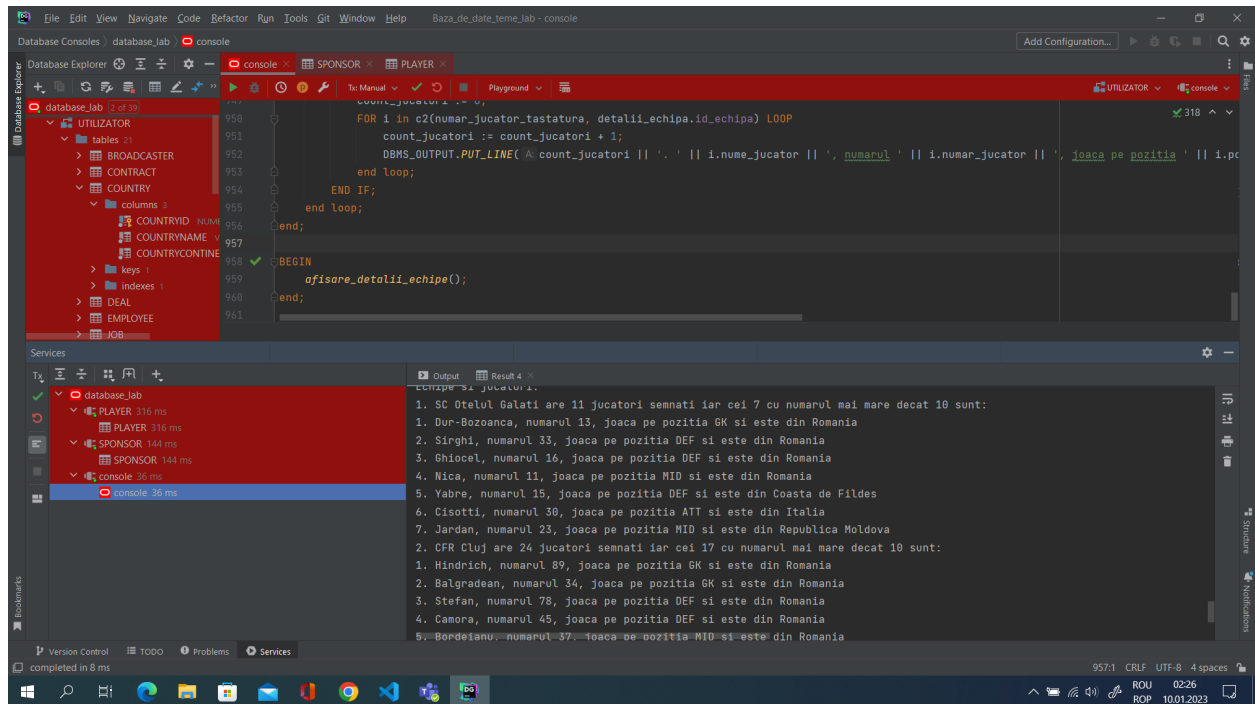


```

count_echipe NUMBER;
count_jucatori NUMBER;
numar_jucator_tastatura NUMBER := &numar_tast;
nr_jucatori_filtrati NUMBER;

BEGIN
  DBMS_OUTPUT.PUT_LINE('Echipe si jucatori: ');
  count_echipe := 0;
  OPEN c1;
  LOOP
    FETCH c1 INTO detalii echipa;
    count_echipe := count_echipe + 1;
    EXIT WHEN c1%NOTFOUND;
    IF detalii echipa.numar_jucatori = 0 THEN
      DBMS_OUTPUT.PUT_LINE(count_echipe || '. ' || detalii echipa.ume_echipa || ' nu are
niciun jucator semnat. ');
    ELSE
      nr_jucatori_filtrati:=filtrare_jucatori_dupa_numar(detalii echipa.id_echipa,numar_jucat
or_tastatura);
      DBMS_OUTPUT.PUT_LINE(count_echipe || '. ' || detalii echipa.ume_echipa || ' are ' ||
detalii echipa.numar_jucatori || ' jucatori semnati iar cei ' || nr_jucatori_filtrati || ' cu numarul mai
mare decat ' || numar_jucator_tastatura || ' sunt: ');
      count_jucatori := 0;
      FOR i in c2(numar_jucator_tastatura, detalii echipa.id_echipa) LOOP
        count_jucatori := count_jucatori + 1;
        DBMS_OUTPUT.PUT_LINE(count_jucatori || '. ' || i.ume_jucator || ', numarul ' ||
i.numar_jucator || ', joaca pe pozitia ' || i.pozitie_jucator || ' si este din ' || i.tara_jucator);
      end loop;
    END IF;
  end loop;
end;
BEGIN
  afisare_detalii_echipe();
end;

```



8. Formulati în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singura comanda SQL 3 tabele diferite. Definiți minim 2 excepții.

--Scrie o funcție care va afișa stadionul pe care evoluează echipa unui jucător, al cărui nume va fi dat de la tastatura.

```

CREATE OR REPLACE FUNCTION stadion_jucator
(nume_jucator Player.PlayerName%TYPE)
RETURN Stadium.StadiumName%TYPE IS
id_jucator Player.PlayerID%TYPE;
nume_stadion Stadium.StadiumName%TYPE;
BEGIN
    select PlayerID
    into id_jucator
    from Player
    where PlayerName = nume_jucator;
    select StadiumName
    into nume_stadion
    from Stadium s

```

```

join Team t on t.TeamID = s.TeamID
join Player p on p.TeamID = t.TeamID
where p.PlayerID = id_jucator;
return nume_stadion;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista jucator cu numele ' ||
nume_jucator);
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare');
end;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Jucatorul cu numele Sefer evolueaza pe ' ||
stadion_jucator('Sefer'));
--Raspuns:
--[2023-01-10 03:17:42] completed in 15 ms
--Jucatorul cu numele Sefer evolueaza pe stadionul Stadionul Giulesti
    DBMS_OUTPUT.PUT_LINE('Jucatorul cu numele Elias evolueaza pe ' ||
stadion_jucator('Elias'));
--Raspuns:
--[2023-01-10 03:18:44]   ORA-20000: Nu exista jucator cu numele Elias
--[2023-01-10 03:18:44]   ORA-06512: la "UTILIZATOR.STADION_JUCATOR", linia 23
--[2023-01-10 03:18:44]   ORA-06512: la linia 6
end;

```

The screenshot shows the Oracle SQL Developer interface. The main window displays the execution of a PL/SQL procedure. The code is as follows:

```

--Raspuns:
--[2023-01-10 03:17:42] completed in 15 ms
--Jucatorul cu numele Sefer evolueaza pe stadionul Stadionul Giulesti
    DBMS_OUTPUT.PUT_LINE('Jucatorul cu numele Elias evolueaza pe ' ||
stadion_jucator('Elias'));
--Raspuns:
--[2023-01-10 03:18:44]   ORA-20000: Nu exista jucator cu numele Elias
--[2023-01-10 03:18:44]   ORA-06512: la "UTILIZATOR.STADION_JUCATOR", linia 23
--[2023-01-10 03:18:44]   ORA-06512: la linia 6
end;

```

The left pane shows the database explorer with the 'TEAM' table selected. The bottom pane shows the 'Services' tab with the 'console' service selected.

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 tabele diferite. Tratați toate excepțiile care pot apărea.

--Să se afișeze numele, echipa, numărul de pe tricou și sponsorul unui jucător dat ca parametru, doar dacă acesta a terminat pe locul 1 în campionatul din România și dacă are o clauză de reziliere activă în contractul cu sponsorii săi.

```
CREATE OR REPLACE PROCEDURE afisare_jucator_exc
(nume_jucator Player.PlayerName%TYPE)
IS
CURSOR c1 is
    select t.TeamID id echipa, t.TeamName nume echipa, Points puncte
    from Team t, Ranking r
    where t.TeamID = r.TeamID;

TYPE tip_detalii echipa IS RECORD (
    id echipa Team.TeamID%TYPE,
    nume echipa Team.TeamName%TYPE,
    puncte echipa Ranking.Points%TYPE );

TYPE tip_detalii_jucator IS RECORD(
    id_jucator Player.PlayerID%TYPE,
    nume_jucator Player.PlayerName%TYPE,
    echipa_jucator Team.TeamName%TYPE,
    numar_jucator Player.JerseyNumber%TYPE,
    sponsor_jucator Sponsor.SponsorName%TYPE,
    clauza_reziliere Contract.ClauzaReziliere%TYPE);

jucator_par tip_detalii_jucator;
echipa_castigatoare tip_detalii echipa;
nr_maxim_puncte Ranking.Points%TYPE;
exceptie_clauza EXCEPTION;
exceptie echipa EXCEPTION;
BEGIN
    select MAX(Points)
    into nr_maxim_puncte
```

```

from Ranking;

open c1;
LOOP
    FETCH c1 into echipa_castigatoare;
    EXIT WHEN echipa_castigatoare.puncte_echipa = nr_maxim_puncte;
end loop;

select p.PlayerID, p.PlayerName, t.TeamName, p.JerseyNumber, s.SponsorName,
c.ClauzaReziliere
into jucator_par
from Player p
join Team t on (t.TeamID = p.TeamID)
join Deal d on (d.PlayerID = p.PlayerID)
join Contract c on (c.ContractID = d.ContractID)
join Sponsor s on (s.SponsorID = d.SponsorID)
where p.PlayerName = nume_jucator;

if jucator_par.clauza_reziliere = 0 then
    raise exceptie_clauza;
end if;
if jucator_par. echipa_jucator <> echipa_castigatoare.nume_echipa then
    raise exceptie_echipa;
end if;

DBMS_OUTPUT.PUT_LINE('Jucatorul ' || jucator_par.nume_jucator || ' evolueaza pentru
echipa ' || jucator_par. echipa_jucator || ', campioana Romaniei, poarta numarul ' ||
jucator_par.numar_jucator || ' si este sponsorizat de ' || jucator_par.sponsor_jucator);

EXCEPTION
    WHEN exceptie_clauza THEN
        RAISE_APPLICATION_ERROR(-20003, 'Jucatorul ' || nume_jucator || ' nu are o clauza
activa. ');
    WHEN exceptie_echipa THEN
        RAISE_APPLICATION_ERROR(-20003, 'Jucatorul ' || nume_jucator || ' nu joaca pentru
echipa castigatoare din Romania. ');
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista jucator cu numele ' ||
nume_jucator);
    WHEN TOO_MANY_ROWS THEN

```

```

        RAISE_APPLICATION_ERROR(-20001, 'Sunt mai multi jucatori cu numele ' ||
nume_jucator);
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare');
end;
BEGIN
    AFISARE_JUCATOR_EXC('Susic');
    --[2023-01-10 04:37:24]   ORA-20003: Jucatorul Susic nu are o clauza activa.
    --[2023-01-10 04:37:24]   ORA-06512: la "UTILIZATOR.AFISARE_JUCATOR_EXC", linia
58
    AFISARE_JUCATOR_EXC('Manea');
    --Jucatorul Manea evolueaza pentru echipa CFR Cluj, campioana Romaniei, poarta numarul 4
si este sponsorizat de Umbro
    AFISARE_JUCATOR_EXC('Elias');
    --[2023-01-10 04:37:54]   ORA-20000: Nu exista jucator cu numele Elias
    --[2023-01-10 04:37:54]   ORA-06512: la "UTILIZATOR.AFISARE_JUCATOR_EXC", linia
62
    AFISARE_JUCATOR_EXC('Carnat');
    --[2023-01-10 04:40:01]   ORA-20003: Jucatorul Carnat nu joaca pentru echipa castigatoare
din Romania.
    --[2023-01-10 04:40:01]   ORA-06512: la "UTILIZATOR.AFISARE_JUCATOR_EXC", linia
60
end;

```

The screenshot shows an IDE with a SQL script being executed. The left pane displays the Database Explorer with a tree view of the database schema, including tables like BROADCASTER, CONTRACT, DEAL, EMPLOYEE, JOB, LEAGUE, LIVESTREAM, MANAGER, MEMBER, PLAYER, and RANKING. The main editor shows the SQL script with line numbers 1074 to 1087. The output pane at the bottom shows the execution results, including error messages and the final position of the cursor.

```

--[2023-01-10 04:37:24]   ORA-20003: Jucatorul Susic nu are o clauza activa.
--[2023-01-10 04:37:24]   ORA-06512: la "UTILIZATOR.AFISARE_JUCATOR_EXC", linia 58
AFISARE_JUCATOR_EXC( nume_jucator 'Manea' );
--Jucatorul Manea evolueaza pentru echipa CFR Cluj, campioana Romaniei, poarta numarul 4 si este sponsorizat de Umbro
AFISARE_JUCATOR_EXC( nume_jucator 'Elias' );
--[2023-01-10 04:37:54]   ORA-20000: Nu exista jucator cu numele Elias
--[2023-01-10 04:37:54]   ORA-06512: la "UTILIZATOR.AFISARE_JUCATOR_EXC", linia 62
AFISARE_JUCATOR_EXC( nume_jucator 'Carnat' );
--[2023-01-10 04:40:01]   ORA-20003: Jucatorul Carnat nu joaca pentru echipa castigatoare din Romania.
--[2023-01-10 04:40:01]   ORA-06512: la "UTILIZATOR.AFISARE_JUCATOR_EXC", linia 60
end;

```

Output:

```

end;
[2023-01-10 04:40:01] [72000][20003]
[2023-01-10 04:40:01]   ORA-20003: Jucatorul Carnat nu joaca pentru echipa castigatoare din Romania.
[2023-01-10 04:40:01]   ORA-06512: la "UTILIZATOR.AFISARE_JUCATOR_EXC", linia 60
[2023-01-10 04:40:01]   ORA-06512: la linia 10
[2023-01-10 04:40:01] Position: 0

```

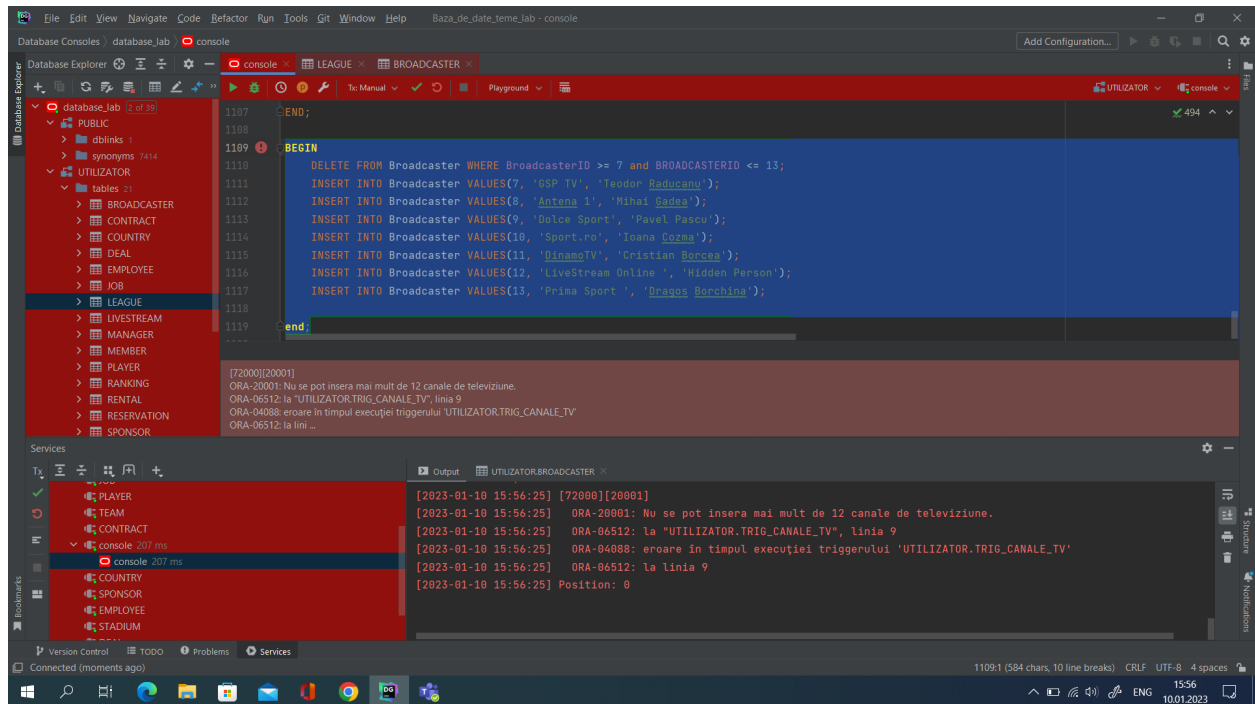
10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.

--Presupunem ca fiecare ligă poate fi transmisă pe 2 canale de televiziune diferite(concurente). Creati un trigger de tip LMD la nivel de comanda care sa nu permita inserarea a mai multe canale de televiziune decât aceasta limita permisă. Declansati trigger-ul.

```
CREATE OR REPLACE TRIGGER trig_canale_tv
  BEFORE INSERT ON Broadcaster
DECLARE
  nr NUMBER;
  nr_ligi NUMBER;
BEGIN
  SELECT COUNT(*) INTO nr FROM Broadcaster;
  SELECT COUNT(*) INTO nr_ligi from League;
  nr_ligi := nr_ligi * 2;
  IF nr >= nr_ligi THEN
    RAISE_APPLICATION_ERROR(-20001, 'Nu se pot insera mai mult de ' || nr_ligi || '
canale de televiziune.');
```

END IF;

```
END;
BEGIN
  DELETE FROM Broadcaster WHERE BroadcasterID >= 7 and BroadcasterID <= 11;
  INSERT INTO Broadcaster VALUES(7, 'GSP TV', 'Teodor Raducanu');
  INSERT INTO Broadcaster VALUES(8, 'Antena 1', 'Mihai Gadea');
  INSERT INTO Broadcaster VALUES(9, 'Dolce Sport', 'Pavel Pascu');
  INSERT INTO Broadcaster VALUES(10, 'Sport.ro', 'Ioana Cozma');
  INSERT INTO Broadcaster VALUES(11, 'DinamoTV', 'Cristian Borcea');
  INSERT INTO Broadcaster VALUES(12, 'LiveStream Online ', 'Hidden Person');
  INSERT INTO Broadcaster VALUES(13, 'Prima Sport ', 'Dragos Borchina');
  --[2023-01-10 15:37:16]    ORA-20001: Nu se pot insera mai mult de 12 canale de
televiziune.
end;
```



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

--A început perioada de transferuri. Creați un trigger de tip LMD la nivel de linie care să verifice și să actualizeze detaliile jucătorilor care au fost transferați, ținând cont de toate cazurile posibile (insert, update, delete). Declanșați trigger-ul. Pentru a evita erori de tip Mutating Table, creați o vizualizare unde veți face toate modificările.

```
CREATE OR REPLACE VIEW view_jucatori AS
```

```
  select p.PlayerID, p.TeamID, p.CountryID, p.PlayerName, p.PlayerPosition, p.JerseyNumber,
  d.SponsorID, d.ContractID
  from Player p, Deal d
  where p.PlayerID = d.PlayerID;
```

```
CREATE OR REPLACE TRIGGER transfer_jucator
```

```
  INSTEAD OF INSERT OR UPDATE OR DELETE ON view_jucatori
  FOR EACH ROW
```

```
DECLARE
```

```
  exista_deja_numar NUMBER;
  exceptie_numar EXCEPTION;
  exceptie_numar_invalid EXCEPTION;
```



```

exceptie_prea_multi_jucatori EXCEPTION;
exceptie_nu_exista EXCEPTION;
nr_jucatori NUMBER;
nume echipa_noua Team.TeamName%TYPE;
exista_jucator NUMBER;
BEGIN
  IF INSERTING or UPDATING THEN
    if :NEW.JerseyNumber > 99 or :NEW.JerseyNumber < 1 then
      raise exceptie_numar_invalid;
    end if;
    --verificare numar jucatori echipa noua
    select count(*) into nr_jucatori from view_jucatori where TeamID = :NEW.TeamID;
    select TeamName into nume echipa_noua from Team where TeamID = :NEW.TeamID;

    if nr_jucatori >= 27 then
      raise exceptie_prea_multi_jucatori;
    end if;

    exista_deja_numar := 0;
    select count(*) into exista_deja_numar
      from view_jucatori where TeamID = :NEW.TeamID and JerseyNumber =
:NEW.JerseyNumber;
    if exista_deja_numar > 0 then
      raise exceptie_numar;
    end if;
    if INSERTING THEN
      INSERT INTO Player VALUES(:NEW.PlayerID, :NEW.TeamID, :NEW.CountryID,
:NEW.PlayerName, :NEW.PlayerPosition, :NEW.JerseyNumber);
      INSERT INTO Deal VALUES(:NEW.PlayerID, :NEW.SponsorID, :NEW.ContractID);
    elsif UPDATING THEN
      UPDATE Player SET TeamID = :NEW.TeamID
        where PlayerID = :NEW.PlayerID;
    end if;
  elsif DELETING then
    select count(*) into exista_jucator from view_jucatori where PlayerID = :OLD.PlayerID;
    if exista_jucator = 0 then
      raise exceptie_nu_exista;
    end if;
    DELETE FROM Deal WHERE PlayerID = :OLD.PlayerID;
    DELETE FROM Player WHERE PlayerID = :OLD.PlayerID;
  end if;
end if;

```

```

end if;
EXCEPTION
    WHEN exceptie_numar THEN
        RAISE_APPLICATION_ERROR(-20003, 'Jucatorul ' || :NEW.PlayerName || ' nu poate
        purta numarul ' || :NEW.JerseyNumber || ' pentru ca echipa ' || nume echipa_noua || ' are deja un
        jucator cu acest numar.');
```

```

    WHEN exceptie_numar_invalid THEN
        RAISE_APPLICATION_ERROR(-20003, 'Jucatorul ' || :NEW.PlayerName || ' nu are un
        numar de tricou valid.');
```

```

    WHEN exceptie_prea_multi_jucatori THEN
        RAISE_APPLICATION_ERROR(-20003, 'Echipa ' || nume echipa_noua || ' are deja 27 de
        jucatori.');
```

```

    WHEN exceptie_nu_exista THEN
        RAISE_APPLICATION_ERROR(-20003, 'Jucatorul cu ID-ul ' || :OLD.PlayerID || ' nu
        exista.');
```

```

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE(SQLERRM);
end;
```

--Declansare trigger:

--Se transfera un jucator care vrea sa poarte un numar deja existent la CFR CLUJ.

```
insert into view_jucatori values (139, 1, 20, 'Jefte', 'ATT', 9, 1, 1);
```

```
--[2023-01-10 16:32:30]   ORA-20003: Jucatorul Jefte nu poate purta numarul 9 pentru ca
echipa CFR Cluj are deja un jucator cu acest numar.
```

--Se transfera un jucator cu un numar invalid

```
insert into view_jucatori values (139, 1, 20, 'Jefte', 'ATT', 122, 1, 1);
```

```
--[2023-01-10 16:33:11]   ORA-20003: Jucatorul Jefte nu are un numar de tricou valid.
```

--Se transfera un jucator, dar echipa are deja 27 de jucatori

```
insert into view_jucatori values (139, 4, 20, 'Iglesias', 'MID', 88, 1, 1);
```

```
--[2023-01-10 16:37:21]   ORA-20003: Echipa FC Dinamo Bucuresti are deja 27 de jucatori.
```

--Se transfera un jucator nou

```
insert into view_jucatori values (139, 3, 20, 'Iglesias', 'MID', 88, 1, 1);
```

--Stergere jucator

```
delete from view_jucatori where PlayerID = 139;
```

```
--UTILIZATOR> delete from view_jucatori where PlayerID = 139
```

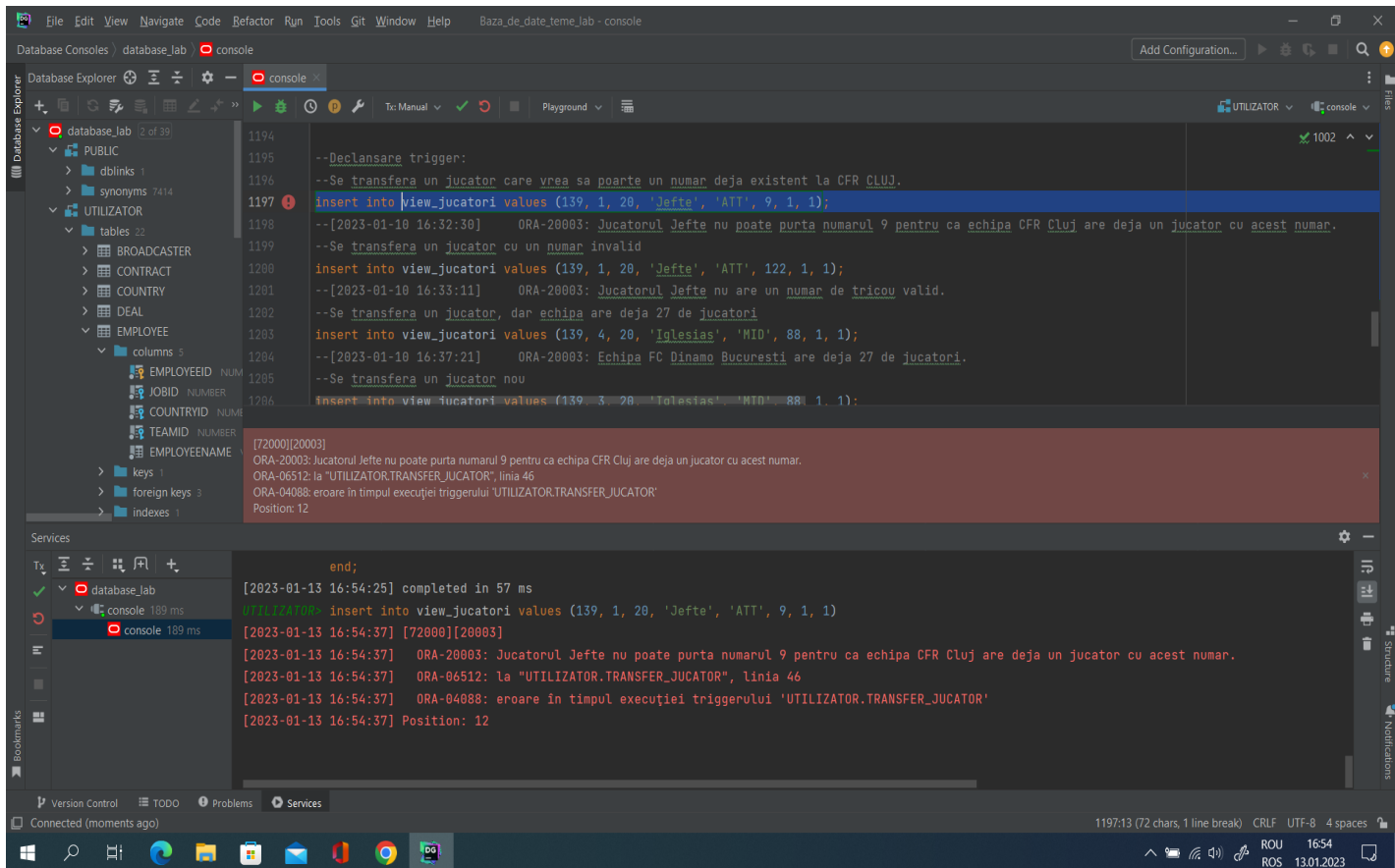
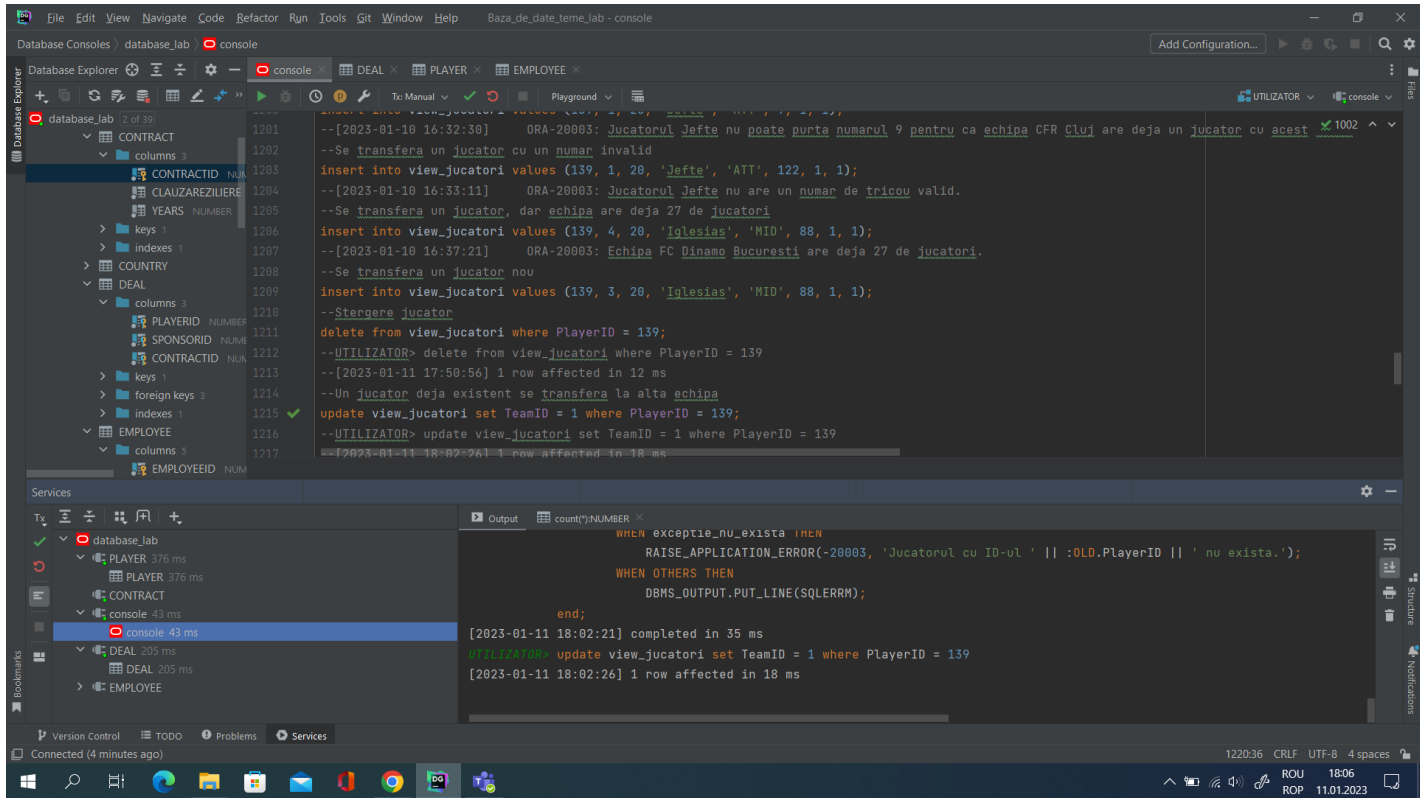
```
--[2023-01-11 17:50:56] 1 row affected in 12 ms
```

--Un jucator deja existent se transfera la alta echipa

```
update view_jucatori set TeamID = 1 where PlayerID = 139;
```

```
--UTILIZATOR> update view_jucatori set TeamID = 1 where PlayerID = 139
```

```
--[2023-01-11 18:02:26] 1 row affected in 18 ms
```



The screenshot shows an IDE window titled "Baza_de_date_teme_lab - console". The "Database Explorer" on the left shows a schema with tables like PUBLIC, dblinks, synonyms, UTILIZATOR, BROADCASTER, CONTRACT, COUNTRY, DEAL, and EMPLOYEE. The main console area displays SQL queries and their execution results. The queries include inserting data into a view and transferring players. The results show error messages from the database, such as "ORA-20003: Jucatorul Jefe nu poate purta numarul 9 pentru ca echipa CFR Cluj are deja un jucator cu acest numar." and "ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'". The Services panel at the bottom shows the execution of the queries and the resulting error messages.

```
--Declansare trigger:
--Se transfera un jucator care vrea sa poarte un numar deja existent la CFR CLUJ.
insert into view_jucatori values (139, 1, 20, 'Jefe', 'ATT', 9, 1, 1);
--[2023-01-10 16:32:30] ORA-20003: Jucatorul Jefe nu poate purta numarul 9 pentru ca echipa CFR Cluj are deja un jucator cu acest numar.
--Se transfera un jucator cu un numar invalid
insert into view_jucatori values (139, 1, 20, 'Jefe', 'ATT', 122, 1, 1);
--[2023-01-10 16:33:11] ORA-20003: Jucatorul Jefe nu are un numar de tricou valid.
--Se transfera un jucator, dar echipa are deja 27 de jucatori
insert into view_jucatori values (139, 4, 20, 'Iglesias', 'MID', 88, 1, 1);
--[2023-01-10 16:37:21] ORA-20003: Echipa FC Dinamo Bucuresti are deja 27 de jucatori.
--Se transfera un jucator nou
insert into view_jucatori values (139, 3, 20, 'Iglesias', 'MID', 88, 1, 1);

[72000][20003]
ORA-20003: Jucatorul Jefe nu are un numar de tricou valid.
ORA-06512: la "UTILIZATOR.TRANSFER_JUCATOR", linia 48
ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'
Position: 12

[2023-01-13 16:54:59] ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'
[2023-01-13 16:54:59] Position: 12
[72000][20003] insert into view_jucatori values (139, 1, 20, 'Jefe', 'ATT', 122, 1, 1)
[2023-01-13 16:55:49] [72000][20003]
[2023-01-13 16:55:49] ORA-20003: Jucatorul Jefe nu are un numar de tricou valid.
[2023-01-13 16:55:49] ORA-06512: la "UTILIZATOR.TRANSFER_JUCATOR", linia 48
[2023-01-13 16:55:49] ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'
[2023-01-13 16:55:49] Position: 12
```

The screenshot shows the same IDE window as above, but with different SQL queries and results. The queries include inserting data into a view and transferring players. The results show error messages from the database, such as "ORA-20003: Echipa FC Dinamo Bucuresti are deja 27 de jucatori." and "ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'". The Services panel at the bottom shows the execution of the queries and the resulting error messages.

```
--Declansare trigger:
--Se transfera un jucator care vrea sa poarte un numar deja existent la CFR CLUJ.
insert into view_jucatori values (139, 1, 20, 'Jefe', 'ATT', 9, 1, 1);
--[2023-01-10 16:32:30] ORA-20003: Jucatorul Jefe nu poate purta numarul 9 pentru ca echipa CFR Cluj are deja un jucator cu acest numar.
--Se transfera un jucator cu un numar invalid
insert into view_jucatori values (139, 1, 20, 'Jefe', 'ATT', 122, 1, 1);
--[2023-01-10 16:33:11] ORA-20003: Jucatorul Jefe nu are un numar de tricou valid.
--Se transfera un jucator, dar echipa are deja 27 de jucatori
insert into view_jucatori values (139, 4, 20, 'Iglesias', 'MID', 88, 1, 1);
--[2023-01-10 16:37:21] ORA-20003: Echipa FC Dinamo Bucuresti are deja 27 de jucatori.
--Se transfera un jucator nou
insert into view_jucatori values (139, 3, 20, 'Iglesias', 'MID', 88, 1, 1);

[72000][20003]
ORA-20003: Echipa FC Dinamo Bucuresti are deja 27 de jucatori.
ORA-06512: la "UTILIZATOR.TRANSFER_JUCATOR", linia 50
ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'
Position: 12

[2023-01-13 16:55:49] ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'
[2023-01-13 16:55:49] Position: 12
[72000][20003] insert into view_jucatori values (139, 4, 20, 'Iglesias', 'MID', 88, 1, 1)
[2023-01-13 16:55:55] [72000][20003]
[2023-01-13 16:55:55] ORA-20003: Echipa FC Dinamo Bucuresti are deja 27 de jucatori.
[2023-01-13 16:55:55] ORA-06512: la "UTILIZATOR.TRANSFER_JUCATOR", linia 50
[2023-01-13 16:55:55] ORA-04088: eroare in timpul executiei triggerului 'UTILIZATOR.TRANSFER_JUCATOR'
[2023-01-13 16:55:55] Position: 12
```

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

--Definiți un trigger de tip LDD care să permită modificarea bazei de date doar dacă utilizatorul curent este administratorul bazei de date(UTILIZATOR). Salvați modificările făcute într-o tabelă nouă.

```
CREATE TABLE info_admin
```

```
(  
    username VARCHAR2(30),  
    nume_bd VARCHAR2(50),  
    data_modificare TIMESTAMP(6),  
    comanda VARCHAR2(1000),  
    eroare VARCHAR2(1000)  
);
```

```
CREATE OR REPLACE TRIGGER trigger_admin
```

```
    BEFORE CREATE OR ALTER OR DROP ON SCHEMA  
DECLARE
```

```
    comanda_aux VARCHAR2(1000);
```

```
BEGIN
```

```
--in comanda_aux vreau sa am comanda care a declansat trigger-ul  
comanda_aux := 'alter table Player drop column Salary;';  
if USER != 'UTILIZATOR' then  
    insert into info_admin values (USER, SYS_CONTEXT('USERENV',  
'DB_NAME'), SYSDATE, comanda_aux, 'Nu aveti drepturi de administrator.');
```

```
        raise_application_error(-20000, 'Nu aveti drepturi de administrator.');
```

```
    end if;  
    insert into info_admin values (USER, SYS_CONTEXT('USERENV',  
'DB_NAME'), SYSDATE, comanda_aux, 'OK');
```

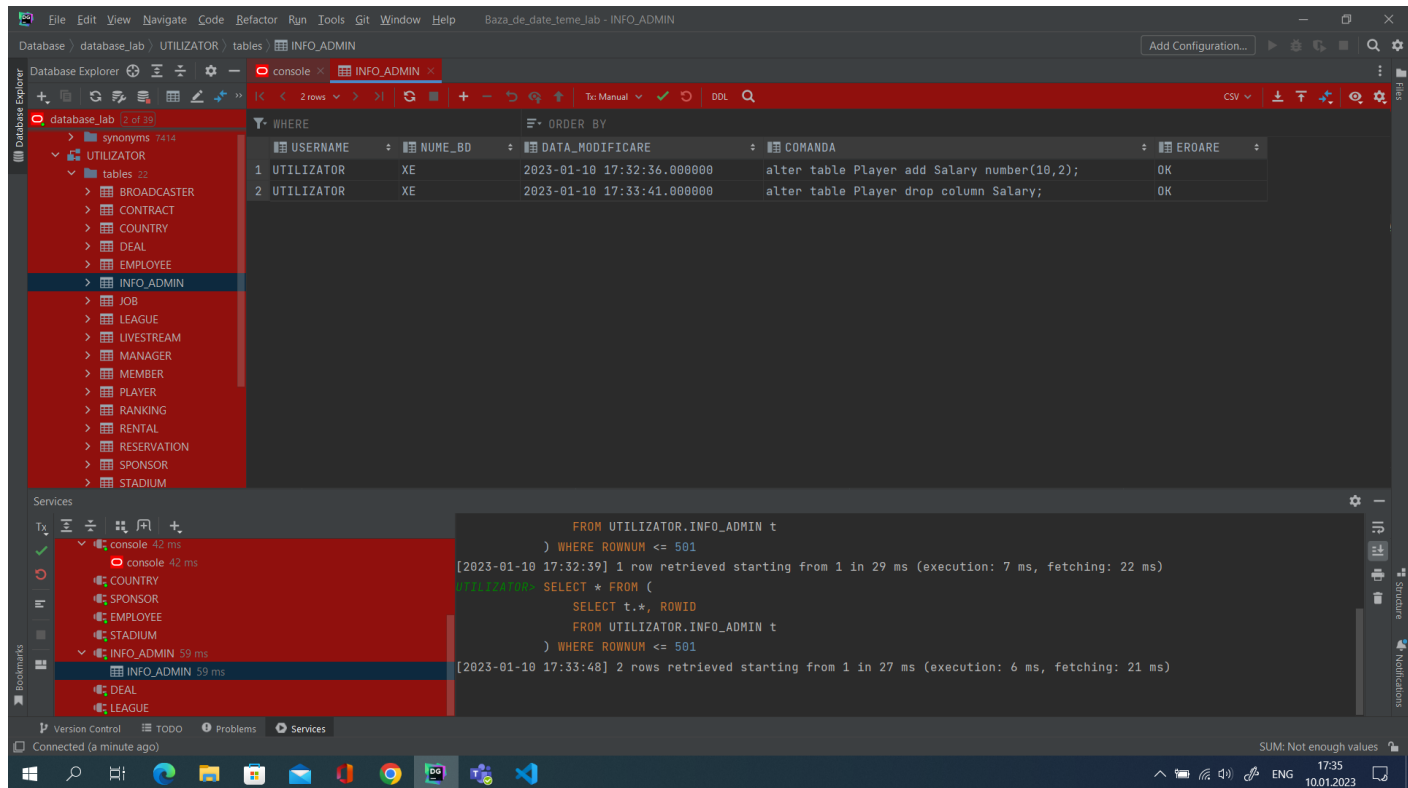
```
end;
```

```
--Declansare trigger:
```

```
alter table Player add Salary number(10,2);
```

```
--Acum vreau sa sterg coloana asta adaugata
```

```
alter table Player drop column Salary;
```



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

```

CREATE OR REPLACE PACKAGE proiect_sgbd_evs AS
  PROCEDURE afisare_coechipieri(id_jucator_tast Player.PlayerID%TYPE);
  FUNCTION filtrare_jucatori_dupa_numar(numar_echipa
    Player.TeamID%TYPE, numar_tricou Player.JerseyNumber%TYPE) RETURN
    NUMBER;
  PROCEDURE afisare_detalii_echipe;
  FUNCTION stadion_jucator(numa_jucator Player.PlayerName%TYPE)
    RETURN Stadium.StadiumName%TYPE;
  PROCEDURE afisare_jucator_exc(numa_jucator Player.PlayerName%TYPE);
END proiect_sgbd_evs;

```

CREATE OR REPLACE PACKAGE BODY proiect_sgbd_evs AS

--Exercitiul 6

--Pentru un jucator de fotbal, caruia i se va da ID-ul de la tastatura, salvati si afisati

--numele tuturor coechipierilor sai, iar pentru fiecare coechipier sa se salveze si sponsorul acestuia.

```
procedure afisare_coechipieri
(id_jucator_tast Player.PlayerID%type)
AS
    TYPE tablou_indexat is TABLE OF Player.PlayerName%type INDEX BY
    PLS_INTEGER;
    nume_coechipieri tablou_indexat;
    TYPE tablou_imbricat is TABLE OF Sponsor.SponsorName%type;
    sponsor_jucatori tablou_imbricat := tablou_imbricat();
    i NUMBER;
    id echipa_jucator Player.TeamID%type;
    id echipa_coechipier Player.TeamID%type;
    nume_jucator Player.PlayerName%type;
    id_coechipier Player.PlayerID%type;
    nume_coechipier Player.PlayerName%type;
    nume_sponsor Sponsor.SponsorName%type;
    numar_jucatori NUMBER;
    id_jucator Player.PlayerID%type;
BEGIN
    i := 0;
    select TeamID, PlayerName into id echipa_jucator, nume_jucator from Player
    where PlayerID = id_jucator_tast;
    select count(*) into numar_jucatori from Player;
    id_jucator := 1;
    while id_jucator <= numar_jucatori
    loop
        select TeamID into id echipa_coechipier from Player where PlayerID =
        id_jucator;
        if id echipa_coechipier = id echipa_jucator and id_jucator_tast <> id_jucator
        then
            i := i + 1;
            select PlayerName, PlayerID into nume_coechipier, id_coechipier from
            Player where PlayerID = id_jucator;
```

```

        nume_coechipieri(i) := nume_coechipier;
        sponsor_jucatori.extend;
        select SponsorName into nume_sponsor from Sponsor where SponsorID =
(select SponsorID from Deal where PlayerID = id_coechipier);
        sponsor_jucatori(i) := nume_sponsor;
    end if;
    id_jucator := id_jucator + 1;
end loop;
DBMS_OUTPUT.PUT_LINE('Jucatorul ' || nume_jucator || ' are urmatorii
coechipieri:');
FOR j IN 1..nume_coechipieri.COUNT LOOP
    DBMS_OUTPUT.PUT_LINE(nume_coechipieri(j) || ' - ' || sponsor_jucatori(j));
END LOOP;
END;

```

--Exercitiul 7

--Sa se afiseze pentru fiecare echipa numarul de jucatori. In cazul in
--care echipa nu are jucatori inscrisi, se va afisa un mesaj corespunzator.
--Pentru fiecare jucator cu numarul de pe tricou mai mare decat un parametru dat,
--sa se afiseze pozitia, numarul de pe tricou si tara de provenienta.

```

FUNCTION filtrare_jucatori_dupa_numar
    (numar_echipa Player.TeamID%TYPE, numar_tricou
Player.JerseyNumber%TYPE)
RETURN NUMBER is
    nr_jucatori_filtrati NUMBER;
BEGIN
    select count(*) into nr_jucatori_filtrati
    from Player
    where JerseyNumber > numar_tricou and TeamID = numar_echipa;
    return nr_jucatori_filtrati;
end;

```

PROCEDURE *afisare_detalii_echipe* AS


```

CURSOR c1 is
    SELECT t.TeamID id_echipa, TeamName nume_echipa, COUNT(PlayerID)
numar_jucatori
    FROM Team t, Player p
    WHERE t.TeamID = p.TeamID(+)
    GROUP BY t.TeamID, TeamName;

```

```

CURSOR c2(parametru NUMBER, echipa_id Team.TeamID%TYPE) is
    SELECT PlayerName nume_jucator, PlayerPosition pozitie_jucator,
CountryName tara_jucator, JerseyNumber numar_jucator
    FROM Player p, Country c
    WHERE p.CountryID = c.CountryID(+) and JerseyNumber > parametru and
TeamID = echipa_id;

```

```

TYPE tip_detalii_echipa IS RECORD(
    id_echipa Team.TeamID%TYPE,
    nume_echipa Team.TeamName%TYPE,
    numar_jucatori NUMBER
);
detalii_echipa tip_detalii_echipa;
count_echipe NUMBER;
count_jucatori NUMBER;
numar_jucator_tastatura NUMBER := &numar_tricou;
nr_jucatori_filtrati NUMBER;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Echipe si jucatori: ');
    count_echipe := 0;
    OPEN c1;
    LOOP
        FETCH c1 INTO detalii_echipa;
        count_echipe := count_echipe + 1;
        EXIT WHEN c1%NOTFOUND;
        IF detalii_echipa.numar_jucatori = 0 THEN
            DBMS_OUTPUT.PUT_LINE(count_echipe || '. ' ||
detalii_echipa.nume_echipa || ' nu are niciun jucator semnat.');
```

```

ELSE
    nr_jucatori_filtrati :=
    filtrare_jucatori_dupa_numar(detalii echipa.id echipa, numar_jucator_tastatura);
    DBMS_OUTPUT.PUT_LINE(count_echipe || '. ' ||
    detalii echipa.ume echipa || ' are ' || detalii echipa.numar_jucatori || ' jucatori
    semnati iar cei ' || nr_jucatori_filtrati || ' cu numarul mai mare decat ' ||
    numar_jucator_tastatura || ' sunt: ');
    count_jucatori := 0;
    FOR i in c2(numar_jucator_tastatura, detalii echipa.id echipa) LOOP
        count_jucatori := count_jucatori + 1;
        DBMS_OUTPUT.PUT_LINE(count_jucatori || '. ' || i.ume_jucator || ',
    numarul ' || i.numar_jucator || ', joaca pe pozitia ' || i.pozitie_jucator || ' si este din ' ||
    i.tara_jucator);
    end loop;
    END IF;
    end loop;
end;

```

--Exercitiul 8

--Scrie o functie care va afisa stadionul pe care evolueaza echipa unui jucator, al
carui nume va fi dat de la tastatura.

```

FUNCTION stadion_jucator
    (nume_jucator Player.PlayerName%TYPE)
RETURN Stadium.StadiumName%TYPE IS
    id_jucator Player.PlayerID%TYPE;
    nume_stadion Stadium.StadiumName%TYPE;
BEGIN
    select PlayerID
    into id_jucator
    from Player
    where PlayerName = nume_jucator;

    select StadiumName
    into nume_stadion
    from Stadium s

```

```

join Team t on t.TeamID = s.TeamID
join Player p on p.TeamID = t.TeamID
where p.PlayerID = id_jucator;

return nume_stadion;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista jucator cu numele ' ||
nume_jucator);
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare');
end;
```

--Exercitiul 9

--Sa se afiseze numele, echipa, numarul de pe tricou si sponsorul unui jucator dat
ca parametru, doar daca
--acesta a terminat pe locul 1 in campionatul din Romania si daca are o clauza de
reziliere activa in contractul cu sponsorii sai.

```

PROCEDURE afisare_jucator_exc
(nume_jucator Player.PlayerName%TYPE)
IS
```

```

    CURSOR c1 is
        select t.TeamID id echipa, t.TeamName nume echipa, Points puncte
        from Team t, Ranking r
        where t.TeamID = r.TeamID;
```

```

TYPE tip_detalii echipa IS RECORD (
    id echipa Team.TeamID%TYPE,
    nume echipa Team.TeamName%TYPE,
    puncte echipa Ranking.Points%TYPE );
```

```

TYPE tip_detalii jucator IS RECORD(
    id_jucator Player.PlayerID%TYPE,
```

```

    nume_jucator Player.PlayerName%TYPE,
    echipa_jucator Team.TeamName%TYPE,
    numar_jucator Player.JerseyNumber%TYPE,
    sponsor_jucator Sponsor.SponsorName%TYPE,
    clauza_reziliere Contract.ClauzaReziliere%TYPE);

jucator_par tip_detalii_jucator;
echipa_castigatoare tip_detalii_echipa;
nr_maxim_puncte Ranking.Points%TYPE;
exceptie_clauza EXCEPTION;
exceptie_echipa EXCEPTION;
BEGIN
    select MAX(Points)
    into nr_maxim_puncte
    from Ranking;

    open c1;
    LOOP
        FETCH c1 into echipa_castigatoare;
        EXIT WHEN echipa_castigatoare.puncte_echipa = nr_maxim_puncte;
    end loop;

    select p.PlayerID, p.PlayerName, t.TeamName, p.JerseyNumber,
    s.SponsorName, c.ClauzaReziliere
    into jucator_par
    from Player p
    join Team t on (t.TeamID = p.TeamID)
    join Deal d on (d.PlayerID = p.PlayerID)
    join Contract c on (c.ContractID = d.ContractID)
    join Sponsor s on (s.SponsorID = d.SponsorID)
    where p.PlayerName = nume_jucator;

    if jucator_par.clauza_reziliere = 0 then
        raise exceptie_clauza;
    end if;

```

```

if jucator_par.echipa_jucator <> echipa_castigatoare.ume_echipa then
    raise exceptie_echipa;
end if;

DBMS_OUTPUT.PUT_LINE('Jucatorul ' || jucator_par.ume_jucator || '
evolueaza pentru echipa ' || jucator_par.echipa_jucator || ', campioana Romaniei,
poarta numarul ' || jucator_par.numar_jucator || ' si este sponsorizat de ' ||
jucator_par.sponsor_jucator);

EXCEPTION
    WHEN exceptie_clauza THEN
        RAISE_APPLICATION_ERROR(-20003, 'Jucatorul ' || ume_jucator || '
nu are o clauza activa.');
```

```

    WHEN exceptie_echipa THEN
        RAISE_APPLICATION_ERROR(-20003, 'Jucatorul ' || ume_jucator || '
nu joaca pentru echipa castigatoare din Romania.');
```

```

    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista jucator cu numele ' ||
ume_jucator);
```

```

    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Sunt mai multi jucatori cu
numele ' || ume_jucator);
```

```

    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare');
```

```

end;
END proiect_sgbd_evs;
```

--Testare pachet

begin

--Exercitiul 6

proiect_sgbd_evs.afisare_coechipieri(1);

--Exercitiul 7

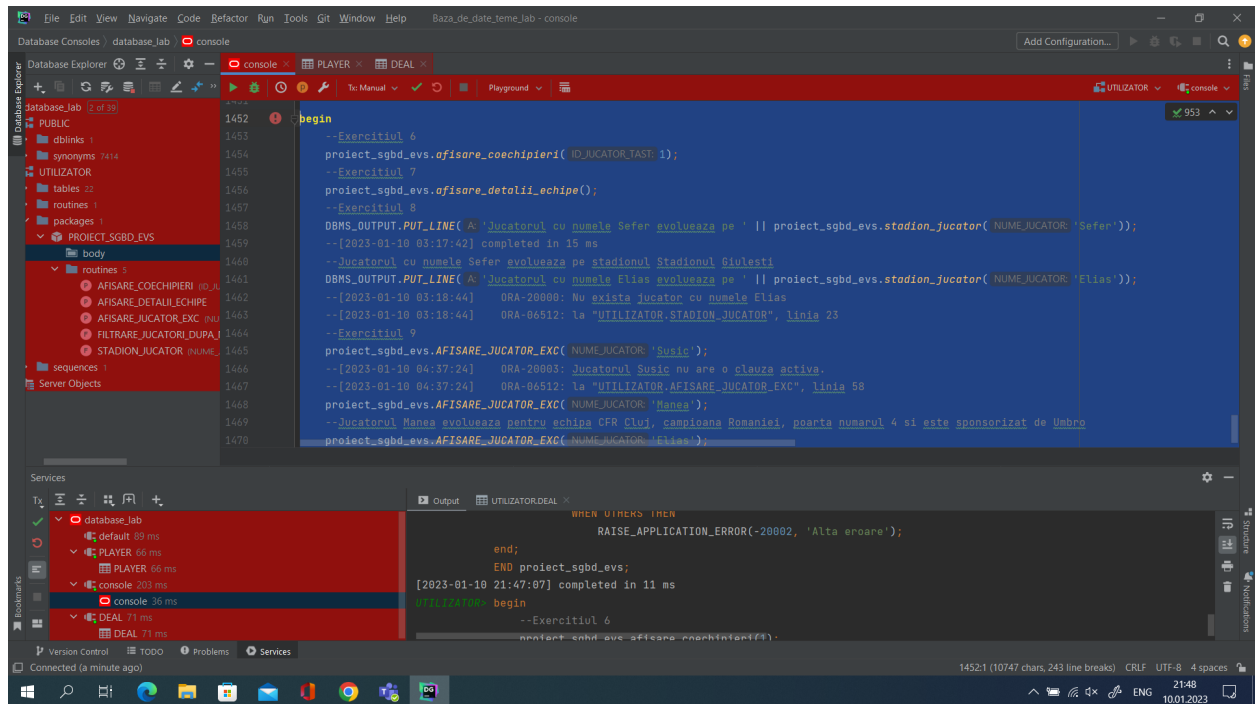
proiect_sgbd_evs.afisare_detalii_echipe();

--Exercitiul 8

```

    DBMS_OUTPUT.PUT_LINE('Jucatorul cu numele Sefer evolueaza pe ' ||
proiect_sgbd_evs.stadion_jucator('Sefer'));
--[2023-01-10 03:17:42] completed in 15 ms
--Jucatorul cu numele Sefer evolueaza pe stadionul Stadionul Giulesti
    DBMS_OUTPUT.PUT_LINE('Jucatorul cu numele Elias evolueaza pe ' ||
proiect_sgbd_evs.stadion_jucator('Elias'));
--[2023-01-10 03:18:44]   ORA-20000: Nu exista jucator cu numele Elias
--[2023-01-10 03:18:44]   ORA-06512: la
"UTILIZATOR.STADION_JUCATOR", linia 23
--Exercitiul 9
proiect_sgbd_evs.AFISARE_JUCATOR_EXC('Susic');
--[2023-01-10 04:37:24]   ORA-20003: Jucatorul Susic nu are o clauza
activa.
--[2023-01-10 04:37:24]   ORA-06512: la
"UTILIZATOR.AFISARE_JUCATOR_EXC", linia 58
proiect_sgbd_evs.AFISARE_JUCATOR_EXC('Manea');
--Jucatorul Manea evolueaza pentru echipa CFR Cluj, campioana
Romaniei, poarta numarul 4 si este sponsorizat de Umbro
proiect_sgbd_evs.AFISARE_JUCATOR_EXC('Elias');
--[2023-01-10 04:37:54]   ORA-20000: Nu exista jucator cu numele Elias
--[2023-01-10 04:37:54]   ORA-06512: la
"UTILIZATOR.AFISARE_JUCATOR_EXC", linia 62
proiect_sgbd_evs.AFISARE_JUCATOR_EXC('Carnat');
--[2023-01-10 04:40:01]   ORA-20003: Jucatorul Carnat nu joaca pentru
echipa castigatoare din Romania.
--[2023-01-10 04:40:01]   ORA-06512: la
"UTILIZATOR.AFISARE_JUCATOR_EXC", linia 60
end;

```



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

--Pentru fiecare echipă, obțineți numele precum și lista numelor angajaților care își desfășoară activitatea în cadrul acesteia, și calculați salariul mediu pentru fiecare echipă. Pentru fiecare angajat se va afișa și funcția acestuia și cati colegi din același departament și aceeași echipă are.

```
CREATE OR REPLACE PACKAGE proiect_sgbd_efs2 AS
  TYPE tablou_imbricat IS TABLE OF VARCHAR2(100);
  TYPE echipa_record IS RECORD (
    id echipa Team.TeamID%TYPE,
    nume echipa Team.TeamName%TYPE);
  CURSOR c_echipa RETURN echipa_record;
  FUNCTION get_salariu(id_angajat Employee.EmployeeID%TYPE) RETURN NUMBER;
  FUNCTION functii_angajati RETURN tablou_imbricat;
  FUNCTION salariu_mediu_echipa(nume_echipa Team.TeamName%TYPE) RETURN
  NUMBER;
  FUNCTION afisare_nr_colegi_angajat(id_angajat Employee.EmployeeID%TYPE) RETURN
  NUMBER;
  PROCEDURE afisare_detalii_echipe;
```

```
END proiect_sgbd_evs2;
```

```
CREATE OR REPLACE PACKAGE BODY proiect_sgbd_evs2 AS
```

```
  CURSOR c_echipa RETURN echipa_record IS
```

```
    SELECT Team.TeamID, Team.TeamName
```

```
    FROM Team;
```

```
  FUNCTION get_salariu(id_angajat Employee.EmployeeID%TYPE) RETURN NUMBER IS  
    salariu NUMBER;
```

```
  BEGIN
```

```
    select JobSalary
```

```
    into salariu
```

```
    from Job
```

```
    where JobID = (select JobID
```

```
                    from Employee
```

```
                    where EmployeeID = id_angajat);
```

```
    RETURN salariu;
```

```
  END get_salariu;
```

```
  FUNCTION functii_angajati RETURN tablou_imbricat IS
```

```
    v_tablou tablou_imbricat := tablou_imbricat();
```

```
    nr_angajati NUMBER;
```

```
  BEGIN
```

```
    SELECT COUNT(*)
```

```
    INTO nr_angajati
```

```
    FROM Employee;
```

```
    FOR id_angajat IN 1..nr_angajati
```

```
    LOOP
```

```
      v_tablou.extend;
```

```
      SELECT Job.JobName
```

```
      INTO v_tablou(id_angajat)
```

```
      FROM Employee, Job
```

```
      WHERE Employee.JobID = Job.JobID AND Employee.EmployeeID = id_angajat;
```

```
    END LOOP;
```

```
    RETURN v_tablou;
```

```
  END functii_angajati;
```

```
  FUNCTION salariu_mediu_echipa(ume_echipa Team.TeamName%TYPE)
```

```
    RETURN NUMBER IS
```

```
    salariu_mediu NUMBER;
```



```

BEGIN
    SELECT Round(AVG(JobSalary), 2)
    INTO salariu_mediu
    FROM Employee, Job, Team
    WHERE Employee.JobID = Job.JobID AND Employee.TeamID = Team.TeamID AND
TeamName = nume_echipa;
    RETURN salariu_mediu;
END salariu_mediu_echipa;

FUNCTION afisare_nr_colegi_angajat(id_angajat Employee.EmployeeID%TYPE)
RETURN NUMBER IS
nr_colegi NUMBER;
TYPE angajat_record is RECORD(
    id_angajat Employee.EmployeeName%TYPE,
    id_job_angajat Job.JobID%TYPE,
    id_echipa_angajat Team.TeamID%TYPE
);
angajatul_nostru angajat_record;
BEGIN
    SELECT Employee.EmployeeName, Employee.JobID, Employee.TeamID
    INTO angajatul_nostru
    FROM Employee
    WHERE Employee.EmployeeID = id_angajat;

    SELECT COUNT(*)
    INTO nr_colegi
    FROM Employee
    WHERE Employee.JobID = angajatul_nostru.id_job_angajat AND Employee.TeamID =
angajatul_nostru.id_echipa_angajat;

    RETURN nr_colegi;
END afisare_nr_colegi_angajat;

PROCEDURE afisare_detalii_echipe IS
nr_angajati NUMBER;
i NUMBER;
j NUMBER;
vector_functii_angajati tablou_imbricat := tablou_imbricat();
record_echipa echipa_record;
BEGIN

```

```

vector_functii_angajati := functii_angajati;
OPEN c echipa;
i := 0;
LOOP
    i := i + 1;
    FETCH c echipa INTO record_echipa;
    EXIT WHEN c echipa%NOTFOUND;
    SELECT COUNT(*) INTO nr_angajati FROM Employee WHERE TeamID =
record_echipa.id_echipa;

DBMS_OUTPUT.PUT_LINE('-----');
    if nr_angajati > 0 then
        DBMS_OUTPUT.PUT_LINE(i || '. Echipa ' || record_echipa.nume_echipa || ' are ' ||
nr_angajati || ' angajati si salariul mediu este ' ||
salariu_meniu_echipa(record_echipa.nume_echipa));
        --AFISARE ANGAJATI
        j := 0;
        FOR angajat in (SELECT EmployeeID, EmployeeName FROM Employee WHERE
TeamID = record_echipa.id_echipa)
            LOOP
                j := j + 1;
                DBMS_OUTPUT.PUT_LINE(j || '. ' || angajat.EmployeeName || ' este ' ||
vector_functii_angajati(angajat.EmployeeID) || ', castiga ' || get_salariu(angajat.EmployeeID) || '
si are ' || afisare_nr_colegi_angajat(angajat.EmployeeID) || ' colegi din acelasi departament. ');
                END LOOP;
            else
                DBMS_OUTPUT.PUT_LINE(i || '. Echipa ' || record_echipa.nume_echipa || ' nu are
angajati');
            end if;
        end loop;
    END afisare_detalii_echipe;
END proiect_sgbd_evs2;

--Testare pachet
BEGIN
    proiect_sgbd_evs2.afisare_detalii_echipe();
end;

```

