PROIECT SGBD

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

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

Ligile din Europa pun la start echipe din diferite colțuri ale țărilor. Fiecare echipa are un stadion în orașul de provenienta, unde disputa meciurile de pe propriul teren. Stadioanele se clasifica după capacitate, dupa numarul de stele oferit de UEFA, depinzand de cat de modern este. O echipa este alcatuita dintr-un lot de jucători. Acestea sunt pregatiti de un antrenor principal.

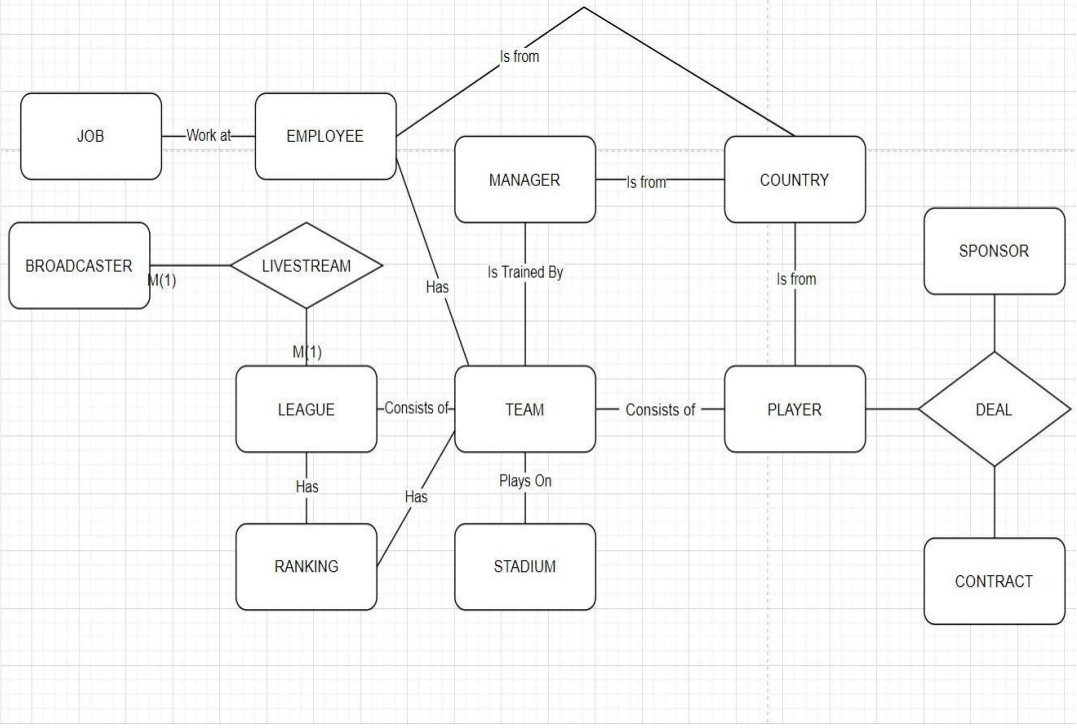
Fiecare echipa are în spate un număr mare de angajați. Aceştia primesc un salariu atractiv, în funcție de job. Pentru fiecare membru din club se cunoaște țara de proveniență.

Jucătorii sunt sponsorizati de anumite companii. Acestia au un contract bine stabilit si nu pot avea mai multe sponsorizări în același timp. Sponsorizarile constau în produse de marca, adidasi, echipament etc. Acest contract poate avea și o clauza de reziliere, in cazul in care exista, jucătorul poate renunța oricând la sponsorul sau.

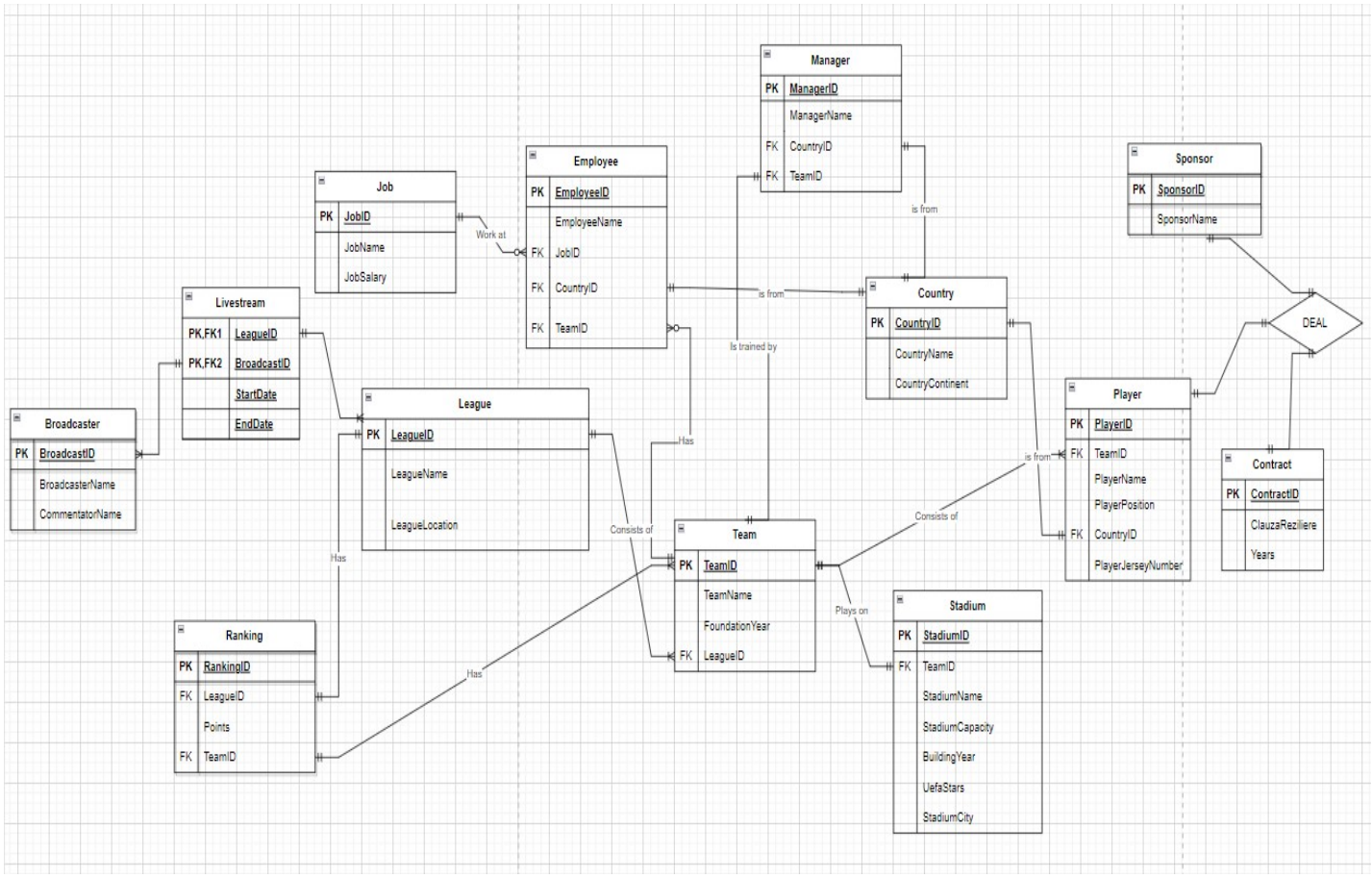
Campionatul de fotbal are mai multe contracte de livestreaming cu diferite televiziuni. Acestea sunt făcute pe o perioada bine definită.

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

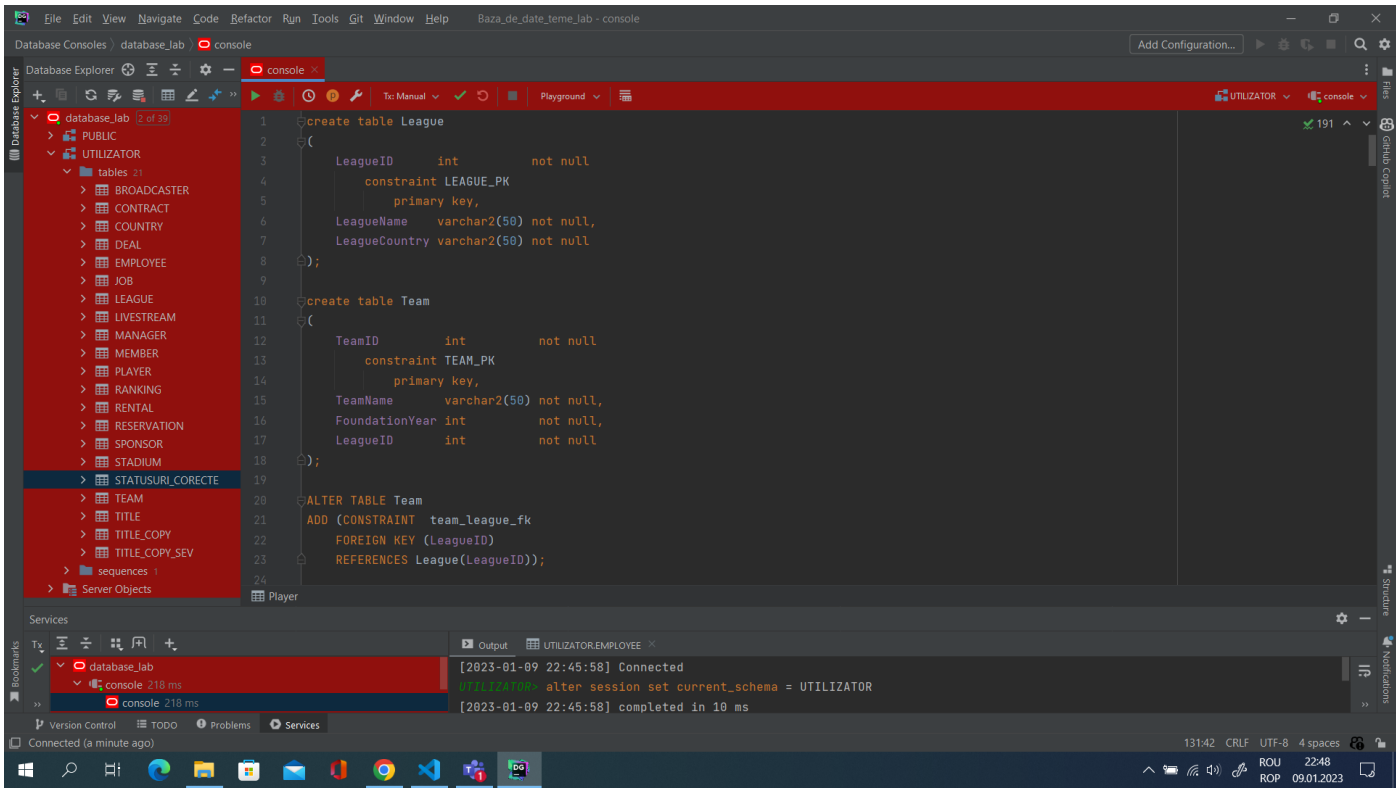
2. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.



3. Realizarea diagramei conceptuale corespunzatoare diagramei entitate-relatie proiectate la punctul 6.

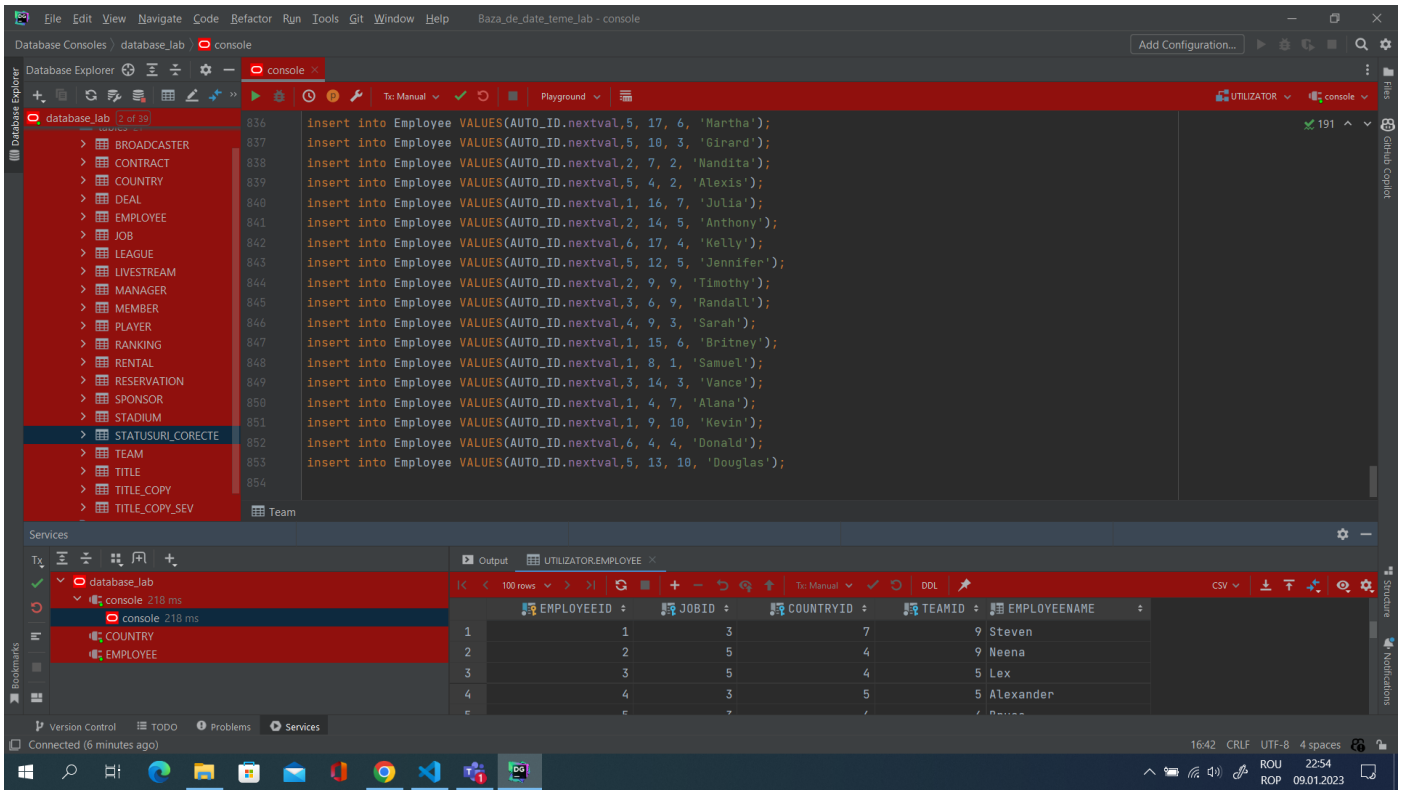


4. **Implementati in Oracle diagrama conceptuala realizata definiti toate tabelele, implementand toate constrangerile de integritate necesare (chei primare, chei externe etc.).**



Fisierele de creare sunt intr-un repository pe github, la adresa urmatoare:
https://github.com/wood11nho/Proiect-SGBD

5. Adaugati informatii coerente in tabelele create (minim 5 inregistrari pentru fiecare entitate independenta; minim 10 inregistrari pentru tabela asociativa).



Fisierele cu comenzile de inserare se gasesc pe urmatorul repository de pe github:

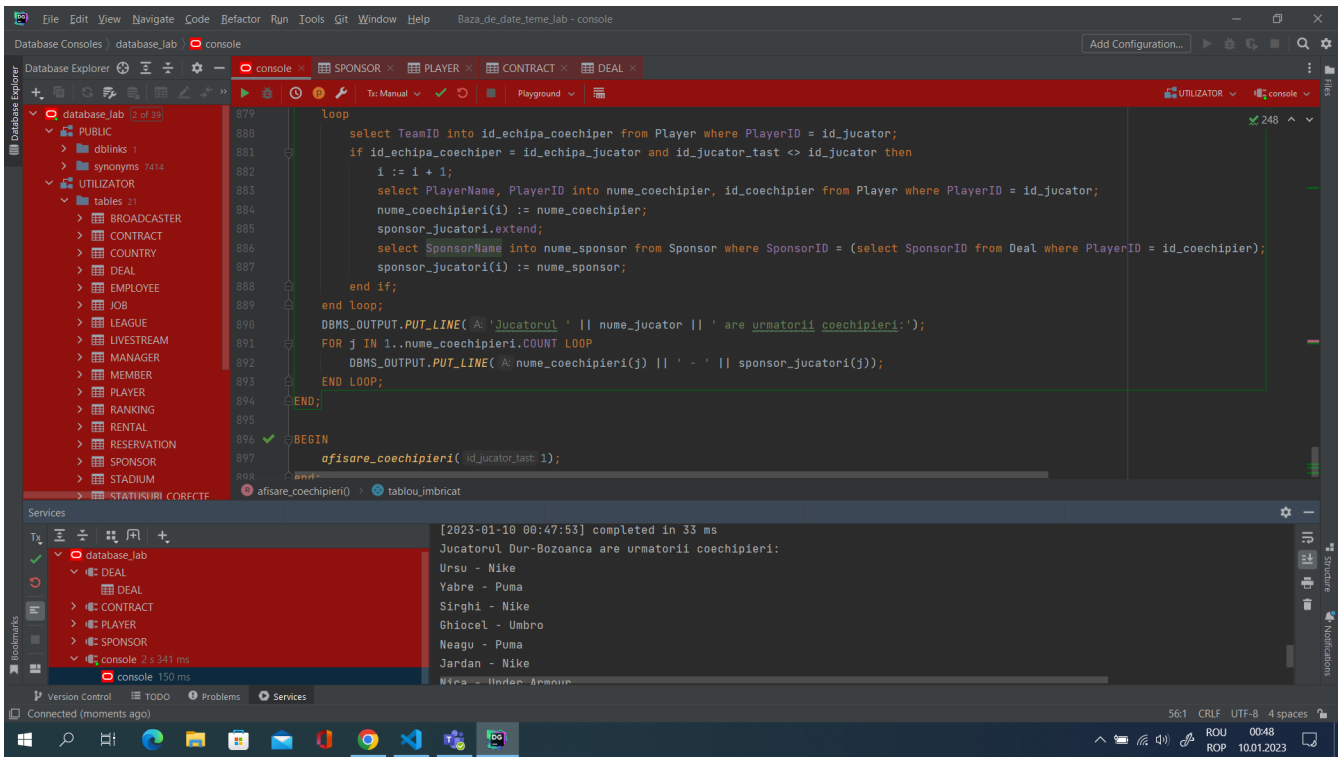
https://github.com/wood11nho/Proiect-SGBD

6. Formulati in limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent care sa utilizeze doua tipuri de colectie studiate. Apelati subprogramul.

--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.

```
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_coechiper 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_coechiper from Player where PlayerID = id_jucator;
    if id_echipa_coechiper = 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;
```

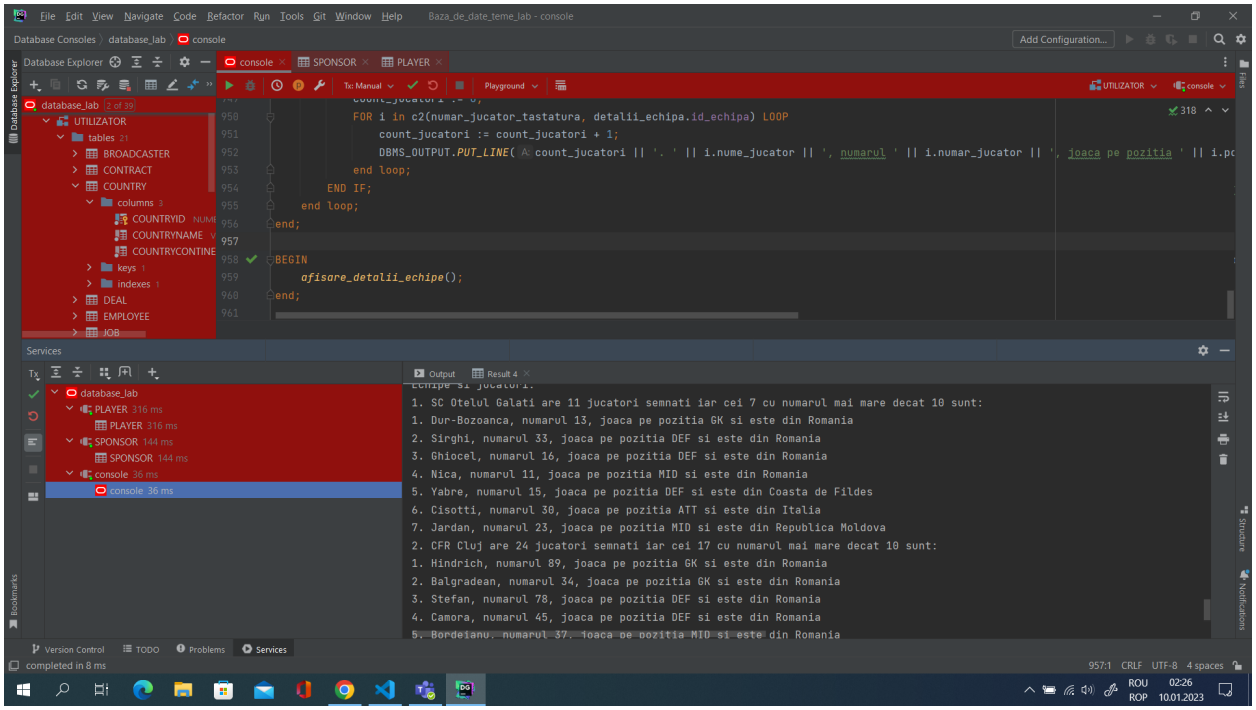


7. Formulati in limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent care sa utilizeze 2 tipuri de cursoare studiate, unul dintre acestea fiind parametrizat. Apelati subprogramul.

--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.

```
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.nume_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.nume_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.nume_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;
```

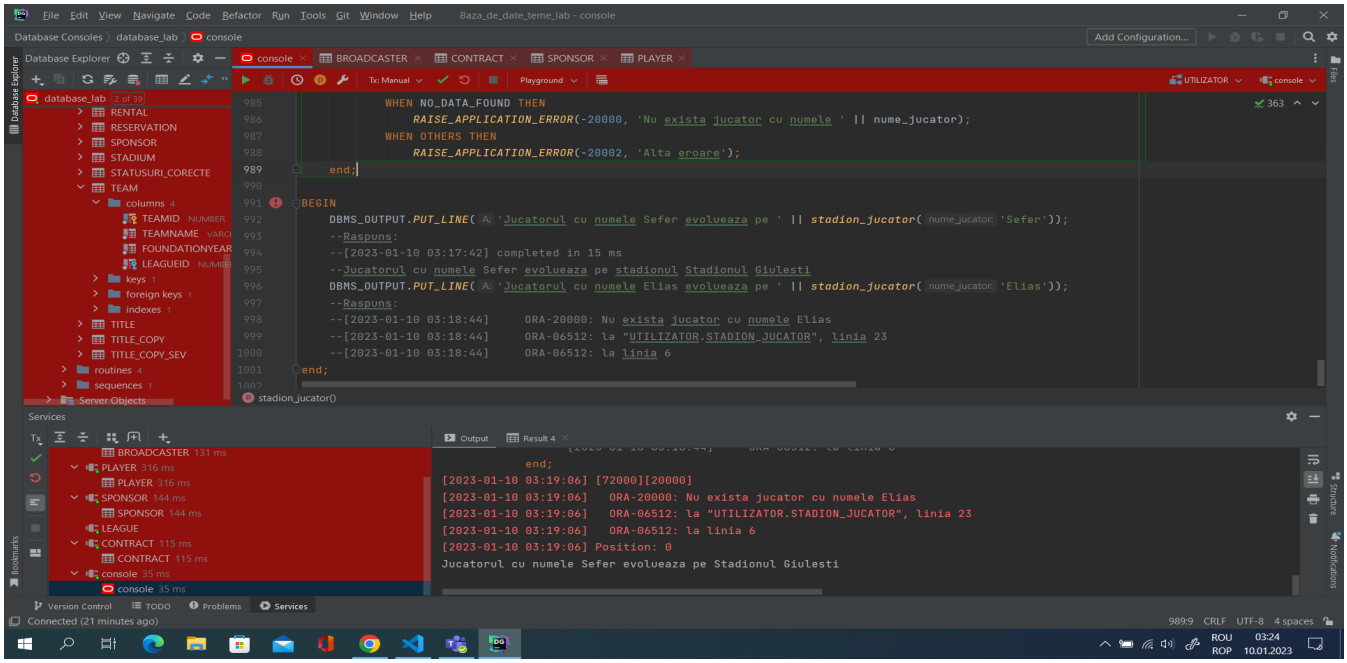


8. Formulati in limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent de tip functie care sa utilizeze intr-o singura comanda SQL 3 tabele diferite. Definiti minim 2 exceptii.

--Scrie o functie care va afisa stadionul pe care evolueaza echipa unui jucator, al carui 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;
```



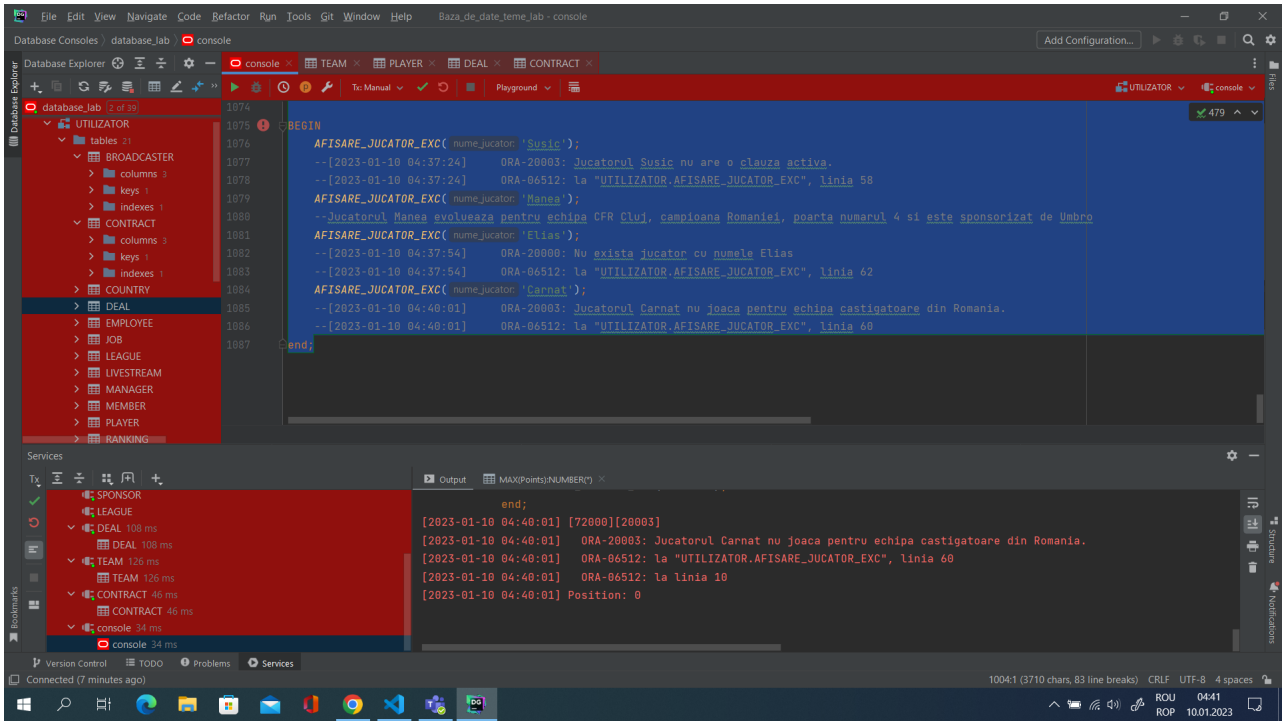
9. Formulati in limbaj natural o problema pe care sa o rezolvati folosind un subprogram stocat independent de tip procedura care sa utilizeze intr-o singura comanda SQL 5 tabele diferite. Tratati toate exceptiile caare pot aparea.

--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.

```
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
```

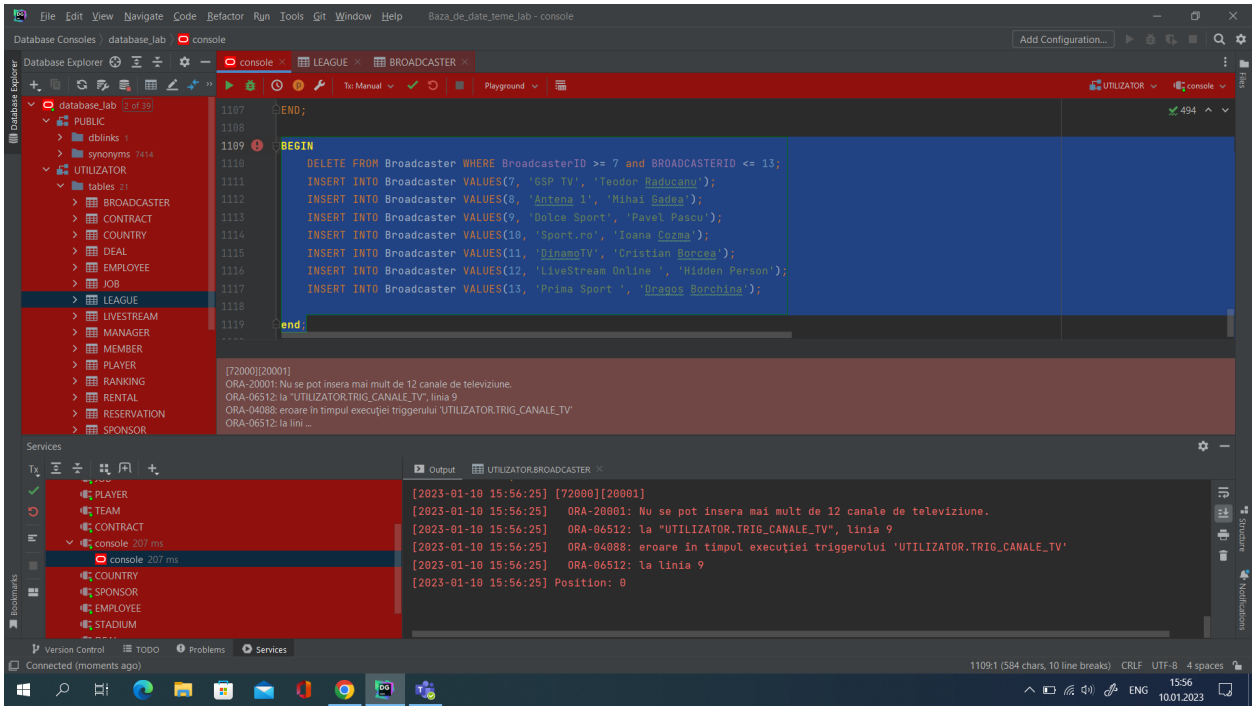
```
        RAISE_APPLICATION_ERROR(-20001, 'Sunt mai multi jucatori cu numele ' ||
nume_jucator);
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare');
end;
BEGIN
  AFISARE_JUCATOR_EXC('Susic');
  --[2023-01-10 04:37:24]  ORA-20003: Jucatorul Susic nu are o clauza activa.
  --[2023-01-10 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;
```



10. Definiti un trigger de tip LMD la nivel de comanda. Declansati trigger-ul.

--Presupunem ca fiecare liga poate fi transmisa 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 decat aceasta limita permisa. 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. Definiti un trigger de tip LMD la nivel de linie. Declansati trigger-ul.

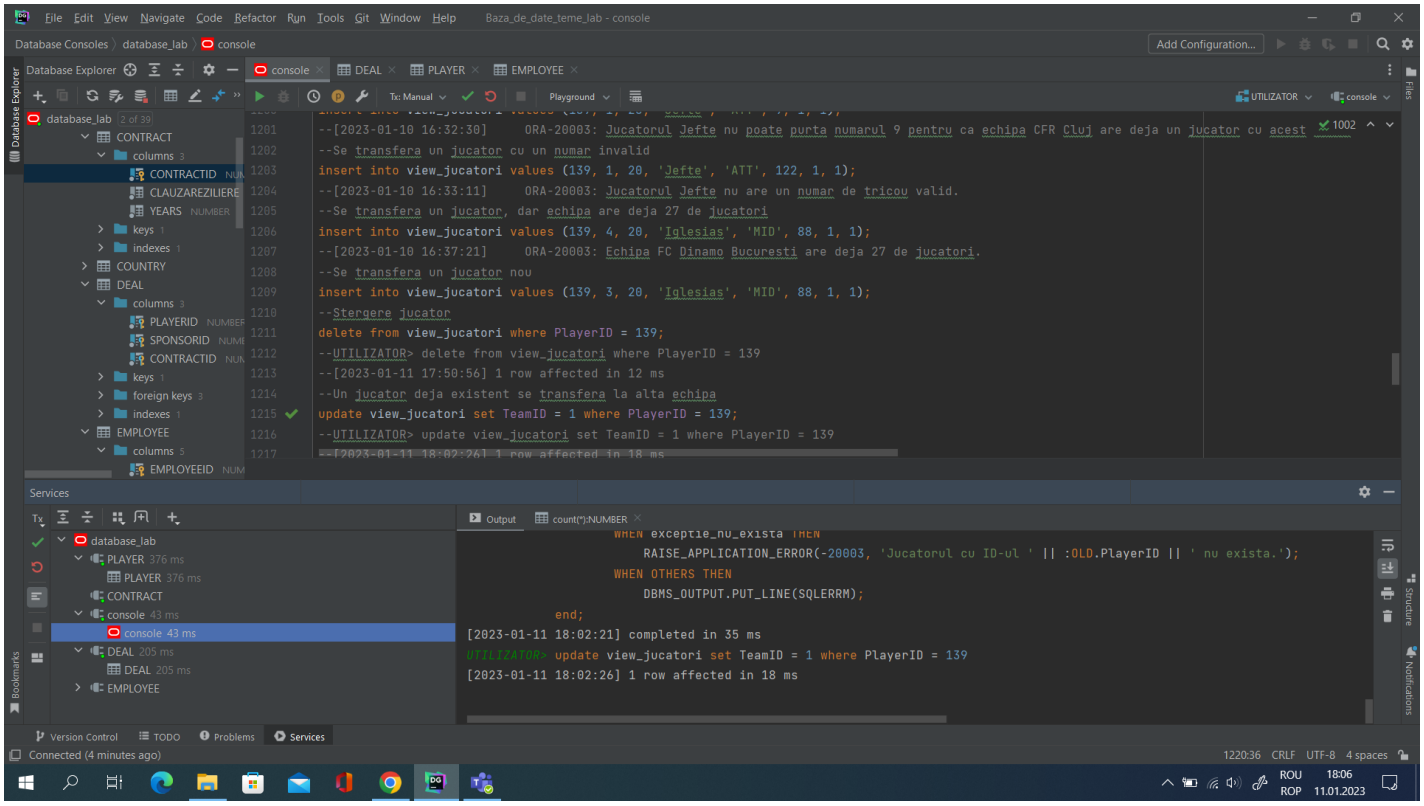
--A inceput perioada de transferuri. Creati un trigger de tip LMD la nivel de linie care sa verifice si sa actualizeze detaliile jucatorilor care au fost transferati, tinand cont de toate cazurile posibile (insert, update, delete). Declansati trigger-ul. Pentru a evita erori de tip Mutating Table, creati o vizualizare unde veti face toate modificarile

```
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 DELETING THEN
    DBMS_OUTPUT.PUT_LINE('TEST DELETE');
  end if;
  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;
  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

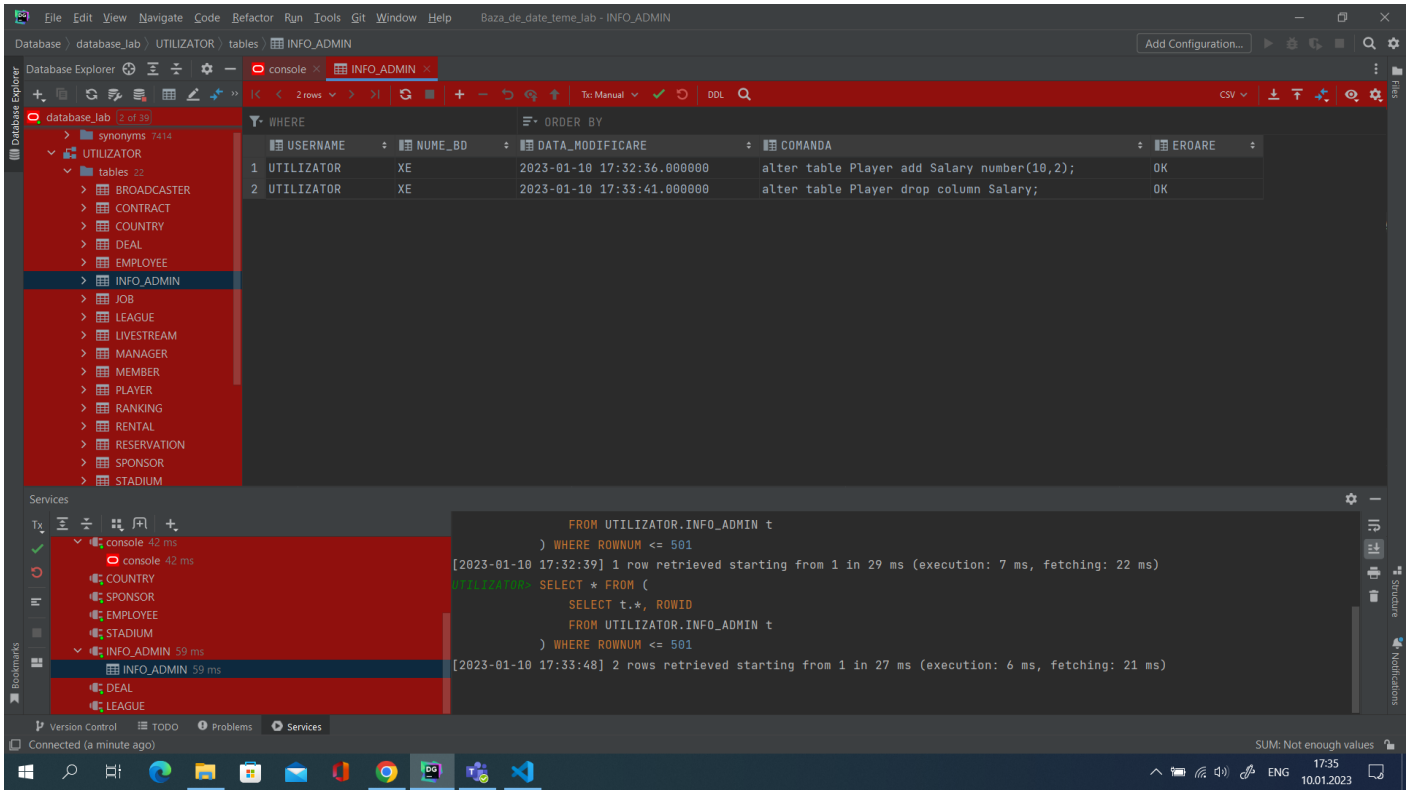


12. Definiti un trigger de tip LDD. Declansati trigger-ul.

--Definiti un trigger de tip LDD care sa permita modificarea bazei de date doar daca utilizatorul curent este administratorul bazei de date(UTILIZATOR). Salvati modificarile facute intr-o tabela noua.

```
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. Definiti un pachet care sa contina toate obiectele definite in 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(nume_jucator Player.PlayerName%TYPE)
RETURN Stadium.StadiumName%TYPE;
  PROCEDURE afisare_jucator_exc(nume_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_coechiper 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_coechiper from Player where PlayerID =
id_jucator;
     if id_echipa_coechiper = 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.nume_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.nume_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,
```

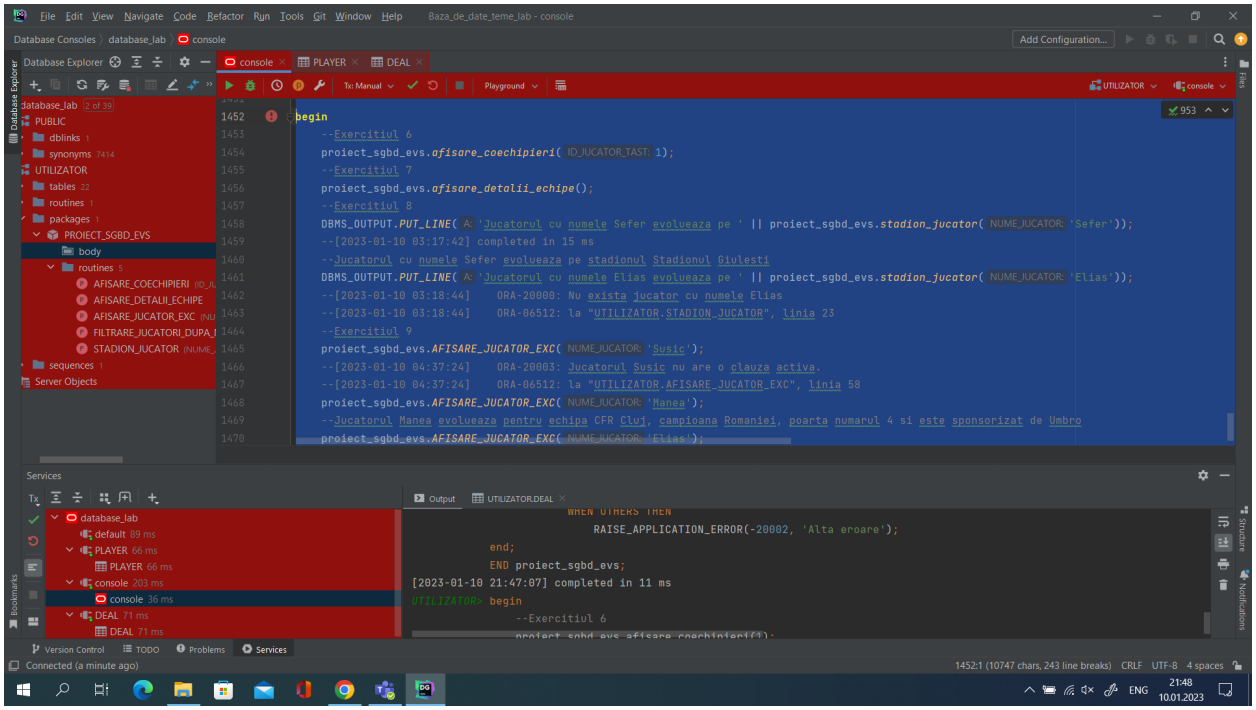
```
    nume_jucator Player.PlayerName%TYPE,
    echipa_jucator Team.TeamName%TYPE,
    numar_jucator Player.JerseyNumber%TYPE,
    sponsor_jucator Sponsor.SponsorName%TYPE,
    clauza_reziliere Contract.ClauzaReziliere%TYPE);

  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.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;
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. Definiti un pachet care sa includa tipuri de date complexe si obiecte necesare unui flux de actiuni integrate, specifice bazei de date definite.

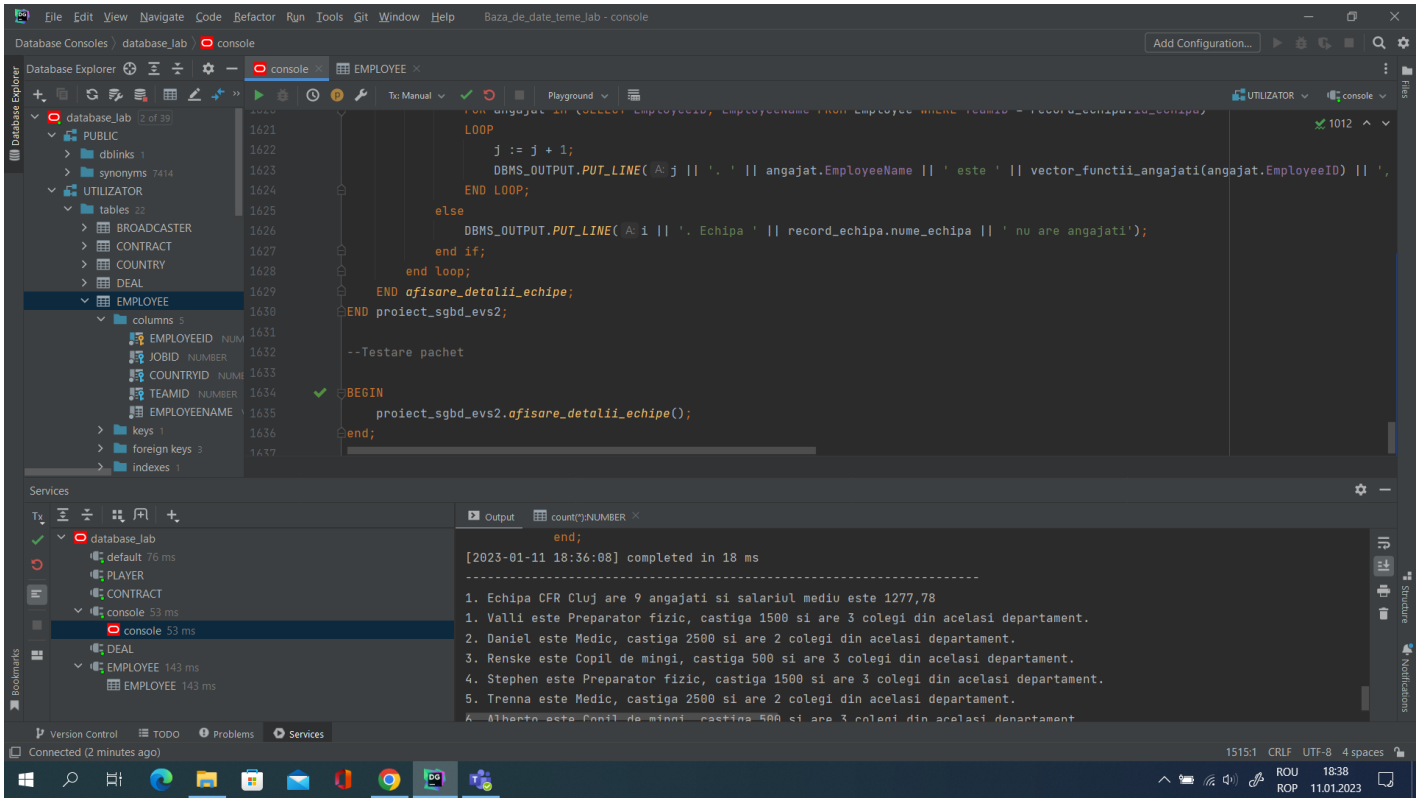
--Pentru fiecare echipa, obtineti numele precum si lista numelor angajatilor care isi desfasoara activitatea in cadrul acestora, si calculati salariul mediu pentru fiecare echipa. Pentru fiecare angajat se va afisa si functia acestuia si cati colegi din acelasi departament are si la aceeasi echipa are.

```
CREATE OR REPLACE PACKAGE proiect_sgbd_evs2 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(nume_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_mediu_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;
```


File Edit View Navigate Code Refactor Run Tools Git Window Help
Baza_de_date_teme_lab - console
Database Consoles database_lab console
Add Configuration...
Database Explorer
console
EMPLOYEE
Tx: Manual
Playground
UTILIZATOR
database_lab
PUBLIC
dblinks 1
synonyms 7414
UTILIZATOR
tables 22
BROADCASTER
CONTRACT
COUNTRY
DEAL
EMPLOYEE
columns 5
EMPLOYEEID NUM
JOBID NUMBER
COUNTRYID NUME
TEAMID NUMBER
EMPLOYEENAME
keys 1
foreign keys 3
indexes 1
1012
LOOP
j := j + 1;
DBMS_OUTPUT.PUT_LINE(j || '. ' || angajat.EmployeeName || ' este ' || vector_functii_angajati(angajat.EmployeeID) || ',
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;
Services
database_lab
default 76 ms
PLAYER
CONTRACT
console 53 ms
console 53 ms
DEAL
EMPLOYEE 143 ms
EMPLOYEE 143 ms
Output
count(*):NUMBER
end;
[2023-01-11 18:36:08] completed in 18 ms
1. Echipa CFR Cluj are 9 angajati si salariul mediu este 1277,78
1. Valli este Preparator fizic, castiga 1500 si are 3 colegi din acelasi departament.
2. Daniel este Medic, castiga 2500 si are 2 colegi din acelasi departament.
3. Renske este Copil de mingi, castiga 500 si are 3 colegi din acelasi departament.
4. Stephen este Preparator fizic, castiga 1500 si are 3 colegi din acelasi departament.
5. Trenna este Medic, castiga 2500 si are 2 colegi din acelasi departament.
Version Control
TODO
Problems
Services
Connected (2 minutes ago)
1515:1 CRLF UTF-8 4 spaces
ROU
ROP
18:38
11.01.2023