Proiect SGBD - Liga de baschet

Eduard-Valentin Dumitrescul

Grupa 232

# Prezentare baza de date

Prezentați pe scurt baza de date (utilitatea ei).

## Descrierea Modelului

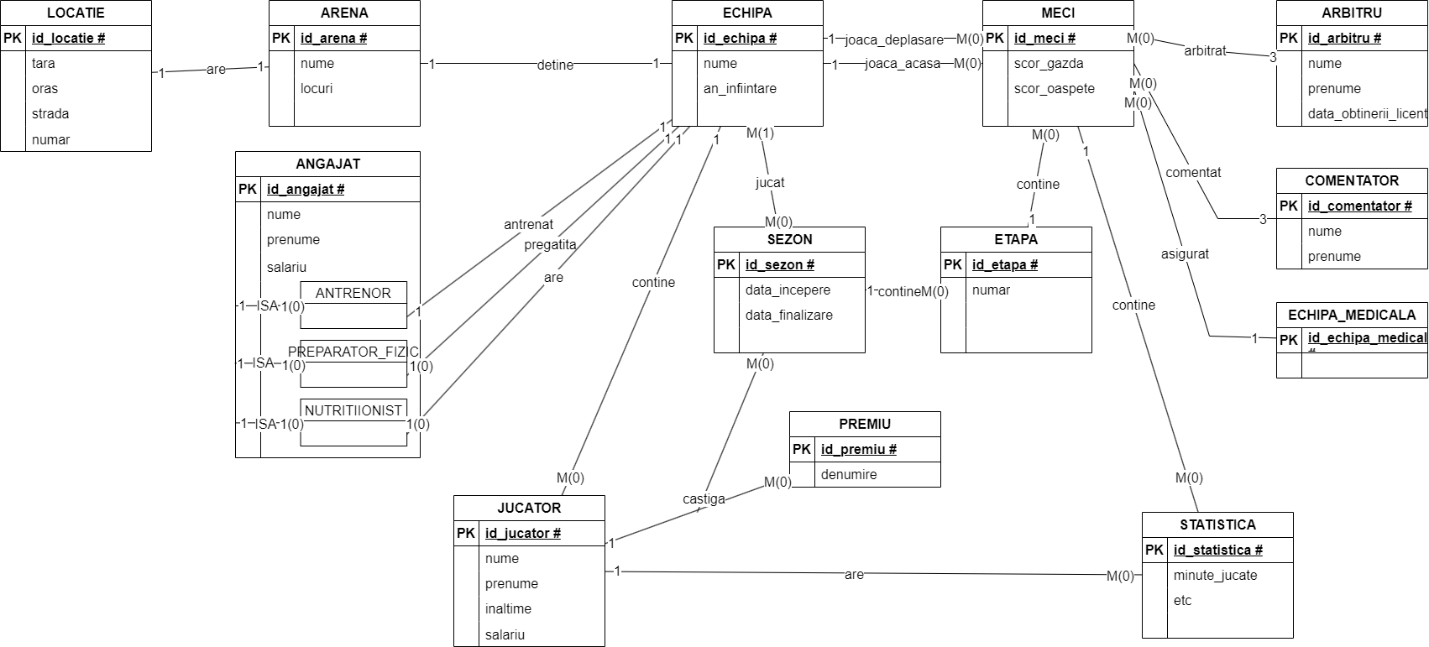
În cadrul unei ligi masculine de baschet, 16 echipe iau parte la fiecare sezon, ce se desfășoară pe parcursul unui an . Fiecare sezon este format din mai multe etape ce se joacă săptămânal, astfel încât până la final, fiecare echipa să fi jucat cu toate celelalte câte două meciuri (acasă și în deplasare). Fiecare echipa este formată din cel puțin 5 jucători, un antrenor, un preparator fizic și un nutriționist. Fiecare echipa deține arena proprie dintr-un anumit oraș, unde va juca meciurile acasă. Pe tot parcursul sezonului se rețin statisticile jucătorilor (puncte marcare, coșuri de 2 puncte, coșuri de 3 puncte, aruncări libere, recuperări, faulturi, pase decisive, apariții) aferente fiecărei etape astfel încât la final să se calculeze clasamentul echipelor și să se acorde diverse premii individuale (pentru cele mai multe puncte marcate, cele mai multe recuperări , cele mai multe pase decisive). La fiecare meci există 3 arbitrii, 3 comentatori și o echipa medicală .

## Constrângeri

* Într -un moment cel mult un sezon poate fi în curs de desfășurare
* Un sezon conține 30 de etape (pentru a permite echipelor să joace fiecare cu fiecare de exact 2 ori)
* La un sezon iau parte 16 echipe de baschet
* O etapă se desfășoară pe parcursul unei săptămâni calendaristice, iar o singur ă etapă se poate desfășura într -o săptămâna
* O etapă conține 8 meciuri (pentru că fiecare echipa să joace câte un meci).
* Fiecare echipa joacă exact un meci în fiecare etapă .
* Meciul este jucat între două echipe, arena uneia dintre ele. Echipa ce deține arena este considerată că joacă acasă , iar cealaltă în deplasare.
* Un meci nu se poate termină dacă cele două echipe au număr egal de puncte.
* La organizarea fiecărui meci iau parte 3 arbitrii, 3 comentatori și o echipa medicală .
* Fiecare echipa este formată din minim 5 jucători , un antrenor, un preparator fizic, un nutriționist . Nerespectarea minimului de 5 jucători duce la descalificare și la pierderea tuturor meciurilor.
* Fiecare echipa deține o arena aflat într -o locație , iar o arena aparține unei singure echipe.
* Un arbitru, comentator sau echipa medicală pot lua parte la un singur meci la moment dat .
* Un jucător , antrenor, preparator fizic sau nutriționist paote fi angajat de către o singur ă echipa la un moment dat .
* Statisticile sunt realizate la fiecare etapă și sunt păstrate în mod individual,pentru fiecare jucător .
* Premiile individuale sunt acordate unui singur jucător

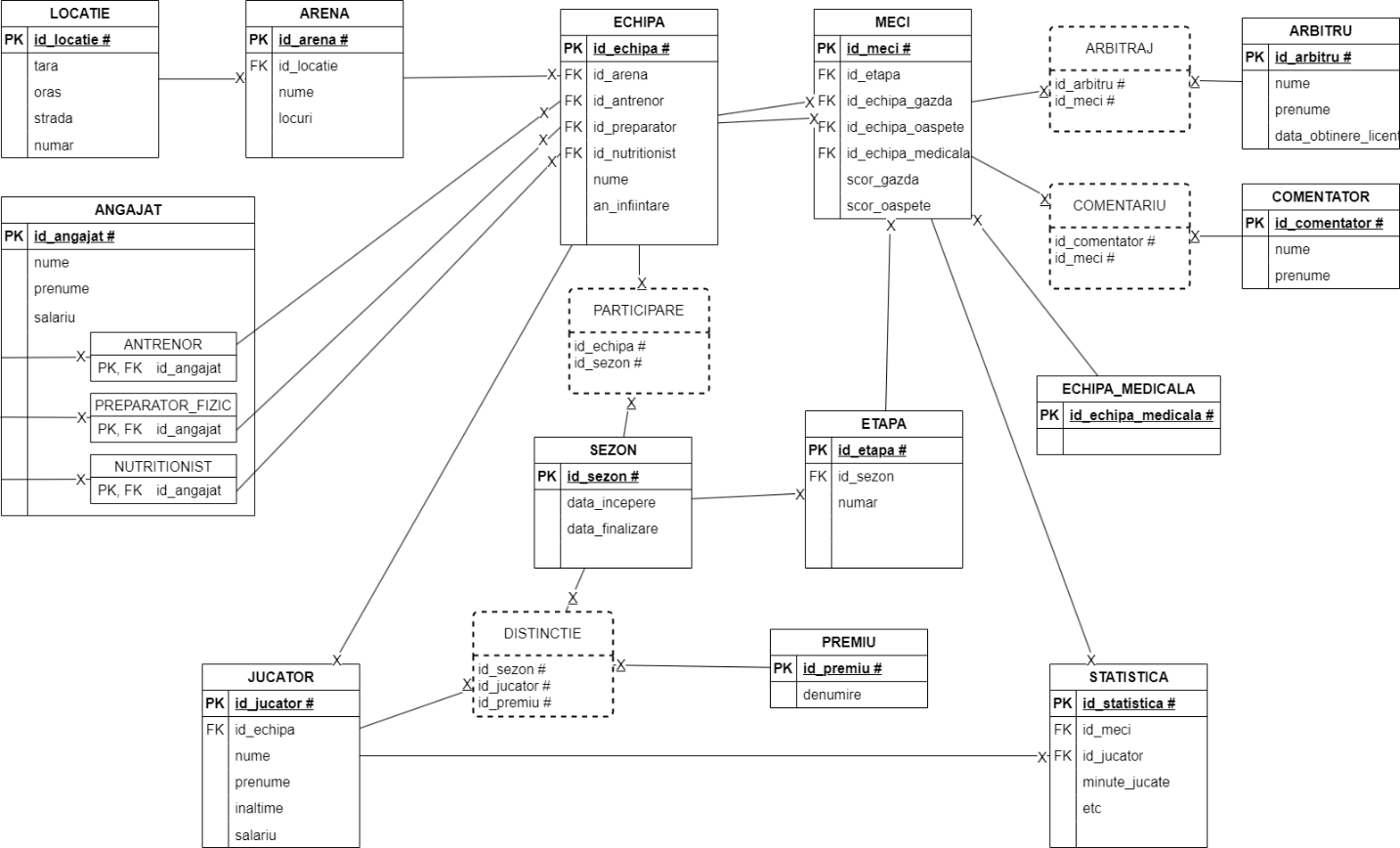
# Diagrama Entitate-Relație

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



# 3 . Diagrama Conceptuală

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



# Implementarea Bazei de Date

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

## Cod SQL

create sequence idseq  
start with 1000  
increment by 1  
nocycle  
nocache;  
  
create table sezoane (  
id\_sezon number(7) primary key,  
data\_incepere date,   
data\_finalizare date  
);  
  
create table etape(  
id\_etapa number(7) primary key,   
id\_sezon number(7) references sezoane(id\_sezon),  
numar number(2)  
);  
  
create table angajati(  
id\_angajat number(7) primary key ,   
nume varchar2(32),  
prenume varchar2(32),  
salariu number(7)  
);  
  
create table antrenori(  
id\_angajat number(7) primary key references angajati(id\_angajat)  
);  
  
create table preparatori\_fizici(  
id\_angajat number(7) primary key references angajati(id\_angajat)  
);  
  
create table nutritionisti(  
id\_angajat number(7) primary key references angajati(id\_angajat)  
);  
  
create table locatii (  
id\_locatie number(7) primary key,   
tara varchar2(32),  
oras varchar2(32),  
strada varchar2(32),  
nr number(4)  
);  
  
create table arene (  
id\_arena number(7) primary key,  
id\_locatie number(7) references locatii(id\_locatie),  
nume varchar2(32),  
locuri number(6)  
);  
  
create table echipe (  
id\_echipa number(7) primary key,   
id\_arena number(7) references arene(id\_arena),  
id\_antrenor number(7) references antrenori(id\_angajat),  
id\_preparator number(7) references preparatori\_fizici(id\_angajat),  
id\_nutritionist number(7) references nutritionisti(id\_angajat),   
nume varchar2(32),   
an\_infiintare number(4)  
);  
  
create table jucatori(  
id\_jucator number(7) primary key,   
id\_echipa number(7) references echipe(id\_echipa),  
nume varchar2(32),  
prenume varchar2(32),  
inaltime number(3, 2),  
salariu number(7)  
);  
  
create table echipe\_medicale(  
id\_echipa\_medicala number(7) primary key  
);  
  
create table meciuri (  
id\_meci number(7) primary key,  
id\_etapa number(7) references etape(id\_etapa),  
id\_echipa\_gazda number(7) references echipe(id\_echipa),  
id\_echipa\_oaspete number(7) references echipe(id\_echipa),  
id\_echipa\_medicala number(7) references echipe\_medicale(id\_echipa\_medicala),  
scor\_gazda number(3),  
scor\_oaspete number(3)  
);  
  
create table arbitrii(  
id\_arbitru number(7) primary key,  
nume varchar2(32),  
prenume varchar2(32),  
data\_obtinere\_licenta date  
);  
  
create table comentatori (  
id\_comentator number(7) primary key,  
nume varchar2(32),  
prenume varchar2(32)  
);  
  
create table statistici(  
id\_statistica number(7) primary key,  
id\_meci number(7) references meciuri(id\_meci),  
id\_jucator number(7) references jucatori(id\_jucator),  
minute\_jucate number(2),  
aruncari\_2pct number(2),  
aruncari\_2pct\_marcate number(2),  
aruncari\_3pct number(2),  
aruncari\_3pct\_marcate number(2),  
aruncari\_libere number(2),  
aruncari\_libere\_marcate number(2),  
pase\_decisive number(2),  
recuperari number(2),  
faulturi number(2)  
);  
  
create table premii(  
id\_premiu number(7) primary key,  
denumire varchar2(64)  
);  
  
create table participari (  
id\_sezon number(7) references sezoane(id\_sezon),  
id\_echipa number(7) references echipe(id\_echipa),  
constraint pk\_participari primary key(id\_sezon, id\_echipa)  
);  
  
create table arbitraje(  
id\_meci number(7) references meciuri(id\_meci),  
id\_arbitru number(7) references arbitrii(id\_arbitru),  
constraint pk\_arbitraje primary key(id\_meci, id\_arbitru)  
);  
  
create table comentarii(  
id\_meci number(7) references meciuri(id\_meci),  
id\_comentator number(7) references comentatori(id\_comentator),  
constraint pk\_comentarii primary key(id\_meci, id\_comentator)  
);  
  
create table distinctii (  
id\_sezon number(7) references sezoane(id\_sezon),  
id\_jucator number(7) references jucatori(id\_jucator),  
id\_premiu number(7) references premii(id\_premiu),  
constraint pk\_distinctii primary key(id\_sezon, id\_jucator, id\_premiu)  
);  
  
savepoint after\_create;

## Capturi de Ecran

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Popularea Bazei de Date

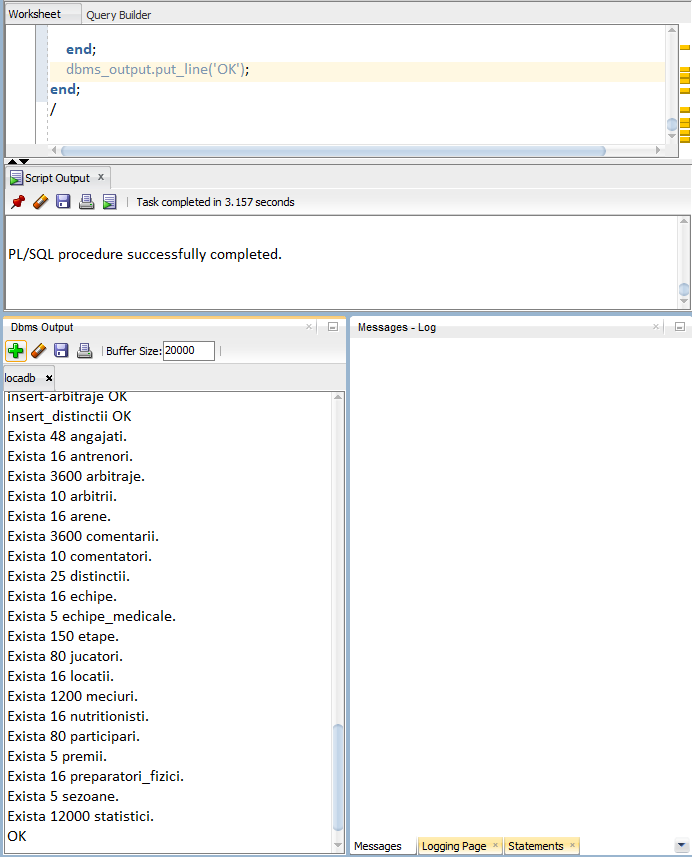
Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).

## Cod Inserare

Am ales să fac inserările utilizând PL/SQL din cauza numărului mare de informații stocate în baza de date. De asemenea, acest mod permite și modificarea datelor ce urmează a fi introduse, prin schimbări minore ale cod. În cazul în care utilizăm doar SQL, pentru fiecare cheie străină trebuia să știm valoarea să din tabela părinte dinainte, ceea ce limita cât de folositoare ar fi fost inserările scrise astfel.

create or replace function get\_id   
return number is  
f\_id number;  
begin  
select idseq.nextval into f\_id  
from dual;  
return f\_id;  
end;  
/  
  
  
create or replace function prenume\_aleator  
return varchar2 as   
prenume varchar2(20);   
type StringArray is varray(20) of varchar2(20);  
lista\_prenume StringArray := StringArray(  
'Ethan', 'Isaac', 'Leo', 'Miles', 'Asher',  
'Maxwell', 'Oscar', 'Dylan', 'Oliver', 'Harrison',  
'Nathan', 'Gabriel', 'Jasper', 'Ezra', 'Silas',  
'Sebastian', 'Caleb', 'Gideon', 'Wyatt', 'Finn'  
);  
begin  
prenume := lista\_prenume(dbms\_random.value(1, lista\_prenume.last));  
return prenume;  
end;  
/  
  
create or replace function nume\_aleator  
return varchar2 as   
nume varchar2(20);   
type StringArray is varray(50) of varchar2(20);  
lista\_nume StringArray := StringArray('Smith', 'Johnson', 'Williams', 'Jones', 'Brown',   
'Davis', 'Miller', 'Wilson', 'Moore', 'Taylor', 'Anderson', 'Thomas', 'Jackson',  
'White', 'Harris', 'Martin', 'Thompson', 'Garcia', 'Martinez', 'Robinson', 'Clark',   
'Rodriguez', 'Lewis', 'Lee', 'Walker', 'Hall', 'Allen', 'Young', 'Hernandez', 'King',   
'Wright', 'Lopez', 'Hill', 'Scott', 'Green', 'Adams', 'Baker', 'Gonzalez', 'Nelson',  
'Carter', 'Mitchell', 'Perez', 'Roberts', 'Turner', 'Phillips', 'Campbell', 'Parker', 'Evans', 'Edwards');  
  
  
begin  
nume := lista\_nume(dbms\_random.value(1, lista\_nume.last));  
return nume;  
end;  
/  
  
begin  
<<sterge\_date>>  
begin  
delete from arbitraje;  
delete from comentarii;  
delete from distinctii;  
delete from participari;  
  
  
delete from premii;  
delete from statistici;  
  
delete from arbitrii;  
delete from comentatori;  
delete from meciuri;  
delete from echipe\_medicale;  
  
delete from jucatori;  
delete from echipe;  
delete from arene;  
delete from locatii;  
  
delete from preparatori\_fizici;  
delete from nutritionisti;  
delete from antrenori;  
delete from angajati;  
  
delete from etape;  
delete from sezoane;  
end;  
  
  
<<insert\_sezoane>>  
declare  
v\_numar\_sezoane number := 5;  
v\_format\_data varchar2(11) := 'dd-mon-yyyy';  
v\_data\_start date := to\_date('15-aug-2022',v\_format\_data);  
v\_data\_final date := to\_date('10-jun-2023', v\_format\_data);  
sezon sezoane%rowtype;  
begin  
sezon.data\_incepere := v\_data\_start;  
sezon.data\_finalizare := v\_data\_final;  
sezon.id\_sezon := get\_id();  
for cnt in 1..v\_numar\_sezoane  
loop  
insert into sezoane  
values sezon;  
sezon.data\_incepere := add\_months(sezon.data\_incepere, -12);  
sezon.data\_finalizare := add\_months(sezon.data\_finalizare, -12);  
sezon.id\_sezon := get\_id();  
end loop;  
  
dbms\_output.put\_line('insert\_sezoane OK');  
end;  
  
<<insert\_etape>>  
declare  
v\_numar\_etape number := 30;  
type id\_sezoane is table of sezoane.id\_sezon%type index by pls\_integer;  
v\_id\_sezoane id\_sezoane;   
v\_etapa etape%rowtype;  
begin  
select id\_sezon  
bulk collect into v\_id\_sezoane  
from sezoane;  
  
for cnt\_sezon in v\_id\_sezoane.first..v\_id\_sezoane.last  
loop  
for cnt\_etapa in 1..v\_numar\_etape  
loop  
v\_etapa.id\_etapa := get\_id();  
v\_etapa.id\_sezon := v\_id\_sezoane(cnt\_sezon);  
v\_etapa.numar := cnt\_etapa;  
insert into etape  
values v\_etapa;  
end loop;  
end loop;  
dbms\_output.put\_line('insert\_etape OK');  
end;  
  
<<insert\_antrenori>>  
declare  
numar\_antrenori number := 16;  
angajat angajati%rowtype;  
antrenor antrenori%rowtype;  
begin  
for i in 1..numar\_antrenori  
loop  
angajat.id\_angajat := get\_id();  
angajat.nume := nume\_aleator();  
angajat.prenume := prenume\_aleator();  
angajat.salariu := 100 \* dbms\_random.value(100, 200);  
antrenor.id\_angajat := angajat.id\_angajat;  
  
insert into angajati values angajat;  
insert into antrenori values antrenor;  
end loop;  
dbms\_output.put\_line('insert\_antrenori OK');  
end;  
  
<<insert\_preparatori>>  
declare  
numar\_preparatori number := 16;  
angajat angajati%rowtype;  
preparator preparatori\_fizici%rowtype;  
begin  
for i in 1..numar\_preparatori  
loop  
angajat.id\_angajat := get\_id();  
angajat.nume := nume\_aleator();  
angajat.prenume := prenume\_aleator();  
angajat.salariu := 100 \* dbms\_random.value(100, 200);  
preparator.id\_angajat := angajat.id\_angajat;  
  
insert into angajati values angajat;  
insert into preparatori\_fizici values preparator;  
end loop;  
dbms\_output.put\_line('insert\_preparatori OK');  
end;  
  
<<insert\_nutritionisti>>  
declare  
numar\_nutritionisti number := 16;  
angajat angajati%rowtype;  
nutritionist nutritionisti%rowtype;  
begin  
for i in 1..numar\_nutritionisti  
loop  
angajat.id\_angajat := get\_id();  
angajat.nume := nume\_aleator();  
angajat.prenume := prenume\_aleator();  
angajat.salariu := 100 \* dbms\_random.value(100, 200);  
nutritionist.id\_angajat := angajat.id\_angajat;  
  
insert into angajati values angajat;  
insert into nutritionisti values nutritionist;  
end loop;  
dbms\_output.put\_line('insert\_nutritionisti OK');  
end;  
  
<<insert\_locatii>>  
declare  
type StringArray is varray(16) of varchar2(30);  
orase StringArray := StringArray('New York City', 'Los Angeles','Las Vegas',  
'Chicago', 'San Francisco', 'Miami', 'Orlando', 'Houston','Seattle',   
'Washington D.C.', 'Boston', 'Atlanta', 'Dallas', 'Denver',   
'New Orleans', 'San Diego');  
strazi StringArray := StringArray('Fifth Avenue', 'Hollywood Boulevard',   
'Las Vegas Boulevard', 'Michigan Avenue', 'Lombard Street',   
'Ocean Drive', 'International Drive', 'NASA Road 1', 'Pike Place Market',   
'1600 Pennsylvania Avenue NW', 'Fenway Park', 'Peachtree Street',   
'Dealey Plaza', '16th Street Mall', 'Bourbon Street', 'Balboa Park');  
locatie locatii%rowtype;  
nr\_locatii number := 16;  
begin  
for i in 1..nr\_locatii  
loop  
locatie.id\_locatie := get\_id();  
locatie.tara := 'USA';  
locatie.oras := orase(i);  
locatie.strada := strazi(i);  
locatie.nr := dbms\_random.value(100, 1000);  
insert into locatii values locatie;  
end loop;  
dbms\_output.put\_line('insert\_locatii OK');  
end;  
  
<<insert\_arene>>  
declare  
type IdLocatii is table of locatii.id\_locatie%type index by pls\_integer;  
id\_locatii IdLocatii;  
numar\_arene number := 16;  
type StringArray is varray(16) of varchar2(30);  
lista\_arene StringArray := StringArray('The Thunderdome', 'The Coliseum', 'The Pit',   
'The Garden', 'The Staples Center', 'The Oracle', 'The Hoop House', 'The Den',  
'The Arena', 'The Thunderdome', 'The Dome', 'The Palace',  
'The Madhouse', 'The Pavilion', 'The Buzzer Beater', 'The Swish Center');  
arena arene%rowtype;  
begin  
select id\_locatie  
bulk collect into id\_locatii  
from locatii;  
  
for i in 1..numar\_arene  
loop  
arena.id\_arena := get\_id();  
arena.id\_locatie := id\_locatii(i);  
arena.nume := lista\_arene(i);  
arena.locuri := 1000 \* dbms\_random.value(10, 20);  
  
insert into arene values arena;  
end loop;  
dbms\_output.put\_line('insert\_arene OK');  
end;  
  
<<insert\_echipe>>  
declare  
type StringArray is varray(16) of varchar2(20);  
lista\_nume StringArray := StringArray('Lightning Bolts', 'Thunderbirds',   
'Wildcats', 'Heatwave', 'Hurricanes', 'Jaguars', 'Patriots', 'Titans',  
'Vikings', 'Dragons', 'Raptors', 'Warriors',  
'Hornets', 'Sharks', 'Lions', 'Knights');  
  
type IdTable is table of number index by pls\_integer;  
id\_arene IdTable;  
id\_antrenori IdTable;  
id\_preparatori IdTable;  
id\_nutritionisti IdTable;  
echipa echipe%rowtype;  
numar\_echipe number := 16;  
begin  
select id\_arena bulk collect into id\_arene from arene;  
select id\_angajat bulk collect into id\_antrenori from antrenori;  
select id\_angajat bulk collect into id\_preparatori from preparatori\_fizici;  
select id\_angajat bulk collect into id\_nutritionisti from nutritionisti;  
  
for i in 1..numar\_echipe  
loop  
echipa.id\_echipa := get\_id();  
echipa.id\_arena := id\_arene(i);  
echipa.id\_antrenor := id\_antrenori(i);  
echipa.id\_preparator := id\_preparatori(i);  
echipa.id\_nutritionist := id\_nutritionisti(i);  
echipa.nume := lista\_nume(i);  
echipa.an\_infiintare := 1960 + dbms\_random.value(0, 30);  
  
insert into echipe values echipa;  
end loop;  
dbms\_output.put\_line('insert\_echipe OK');  
end;   
  
<<insert\_jucatori>>  
declare  
type IdArray is table of echipe.id\_echipa%type index by pls\_integer;  
id\_echipe IdArray;  
id\_echipa echipe.id\_echipa%type;  
jucator jucatori%rowtype;  
numar\_jucatori\_per\_echipa number := 5;  
begin  
select id\_echipa bulk collect into id\_echipe from echipe;  
  
for i in id\_echipe.first..id\_echipe.last  
loop  
id\_echipa := id\_echipe(i);  
for i in 1..numar\_jucatori\_per\_echipa  
loop  
jucator.id\_jucator := get\_id();  
jucator.id\_echipa := id\_echipa;  
jucator.nume := nume\_aleator();  
jucator.prenume := prenume\_aleator();  
jucator.inaltime := dbms\_random.value(1.80, 2.25);  
jucator.salariu := 1000 \* dbms\_random.value(40, 100);  
  
insert into jucatori values jucator;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_jucatori OK');  
end;  
  
<<insert\_echipe\_medicale>>  
declare  
numar\_echipe\_medicale number := 5;  
begin  
for i in 1..numar\_echipe\_medicale  
loop   
insert into echipe\_medicale values(get\_id());  
end loop;  
  
dbms\_output.put\_line('insert\_echipe\_medicale OK');  
end;  
  
<<insert\_meciuri>>  
declare  
type IdArray is table of number index by pls\_integer;  
id\_sezoane IdArray;  
id\_echipe IdArray;  
id\_echipe\_med IdArray;  
id\_etape IdArray;  
meci meciuri%rowtype;  
type IntArray is varray(8) of number;  
x1 IntArray := IntArray(1, 2, 3, 4, 5, 6, 7, 8);  
x2 IntArray := IntArray(16, 15, 14, 13, 12, 11, 10, 9);  
rev boolean := false;  
id\_gazda number;  
id\_oaspete number;  
temp number;  
ids sezoane.id\_sezon%type;  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
select id\_echipa bulk collect into id\_echipe from echipe;  
select id\_echipa\_medicala bulk collect into id\_echipe\_med from echipe\_medicale;  
  
  
  
for i in id\_sezoane.first..id\_sezoane.last  
loop  
ids := id\_sezoane(i);  
select id\_etapa bulk collect into id\_etape   
from etape  
where id\_sezon = ids;  
  
for nr\_etapa in 1..30  
loop  
for i in 1..8  
loop  
if rev = false  
then   
id\_gazda := id\_echipe(x1(i));  
id\_oaspete := id\_echipe(x2(i));  
else  
id\_gazda := id\_echipe(x2(i));  
id\_oaspete := id\_echipe(x1(i));  
end if;  
  
meci.id\_meci := get\_id();  
meci.id\_etapa := id\_etape(nr\_etapa);  
meci.id\_echipa\_gazda := id\_gazda;  
meci.id\_echipa\_oaspete := id\_oaspete;  
meci.id\_echipa\_medicala := id\_echipe\_med(dbms\_random.value(1, id\_echipe\_med.last));  
meci.scor\_gazda := dbms\_random.value(60, 100);  
meci.scor\_oaspete := meci.scor\_gazda + (dbms\_random.value(0, 94) - 47);  
  
insert into meciuri values meci;  
  
end loop;  
temp := x2(1);  
for i in 1..7  
loop  
x2(i) := x2(i+1);  
end loop;  
x2(8) := x1(8);  
for i in reverse 3..8  
loop  
x1(i) := x1(i-1);  
end loop;  
x1(2) := temp;  
  
if x1(2) = 2  
then rev := true;  
end if;  
  
end loop;  
end loop;  
end;   
  
<<insert\_arbitrii>>  
declare  
arbitru arbitrii%rowtype;  
numar\_arbitrii number := 50;  
begin  
for i in 1..numar\_arbitrii  
loop  
arbitru.nume := nume\_aleator();  
arbitru.prenume := prenume\_aleator();  
arbitru.id\_arbitru := get\_id();  
arbitru.data\_obtinere\_licenta := to\_date(trunc(  
dbms\_random.value(to\_char(date '1990-01-01','J') ,to\_char(date '2015-12-31','J') )  
),'J' );  
insert into arbitrii values arbitru;  
end loop;  
  
dbms\_output.put\_line('insert\_arbitrii OK');  
end;  
  
<<insert\_comentatori>>  
declare  
comentator comentatori%rowtype;  
numar\_comentatori number := 10;  
begin  
for i in 1..numar\_comentatori  
loop  
comentator.nume := nume\_aleator();  
comentator.prenume := prenume\_aleator();  
comentator.id\_comentator := get\_id();  
insert into comentatori values comentator;  
end loop;  
  
dbms\_output.put\_line('insert\_comentatori OK');  
end;  
  
  
<<insert\_statistici>>  
declare  
type IdArray is table of number index by pls\_integer;  
id\_meciuri IdArray;  
id\_jucatori IdArray;  
statistica statistici%rowtype;  
meci meciuri%rowtype;  
idm meciuri.id\_meci%type;  
idj jucatori.id\_jucator%type;  
begin  
select id\_meci bulk collect into id\_meciuri from meciuri;  
  
for i in id\_meciuri.first..id\_meciuri.last  
loop  
idm := id\_meciuri(i);  
select \* into meci from meciuri where id\_meci = idm;  
select id\_jucator bulk collect into id\_jucatori  
from jucatori  
where id\_echipa = meci.id\_echipa\_gazda or id\_echipa = meci.id\_echipa\_oaspete;  
  
for j in id\_jucatori.first..id\_jucatori.last  
loop  
idj := id\_jucatori(j);  
statistica.id\_statistica := get\_id();  
statistica.id\_meci := idm;  
statistica.id\_jucator := idj;  
statistica.minute\_jucate := dbms\_random.value(20, 48);  
statistica.aruncari\_2pct := dbms\_random.value(0, 30);  
statistica.aruncari\_2pct\_marcate := dbms\_random.value(0, statistica.aruncari\_2pct);  
statistica.aruncari\_3pct := dbms\_random.value(0, 20);  
statistica.aruncari\_3pct\_marcate := dbms\_random.value(0, statistica.aruncari\_3pct);  
statistica.aruncari\_libere := dbms\_random.value(0, 10);  
statistica.aruncari\_libere\_marcate := dbms\_random.value(0, statistica.aruncari\_libere);  
statistica.pase\_decisive := dbms\_random.value(0, 25);  
statistica.recuperari := dbms\_random.value(0,15);  
statistica.faulturi := dbms\_random.value(0, 5);  
  
insert into statistici values statistica;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_statistica OK');  
end;  
  
<<insert\_premii>>  
declare  
type StringArray is varray(5) of varchar2(50);  
lista\_premii StringArray := StringArray('Most Valuable Player (MVP)',   
'Team Player of the Year',  
'Defensive Player of the Year', 'Sportsmanship Award', 'Best Distance Shooter');  
premiu premii%rowtype;  
begin  
for i in lista\_premii.first..lista\_premii.last  
loop  
premiu.id\_premiu := get\_id();  
premiu.denumire := lista\_premii(i);  
insert into premii values premiu;  
end loop;  
  
dbms\_output.put\_line('inser\_premii OK');  
end;  
  
<<insert\_participari>>  
declare  
type IdArray is table of number index by pls\_integer;  
id\_sezoane IdArray;  
id\_echipe IdArray;  
participare participari%rowtype;  
ids sezoane.id\_sezon%type;  
ide echipe.id\_echipa%type;  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
select id\_echipa bulk collect into id\_echipe from echipe;  
  
for i in id\_sezoane.first..id\_sezoane.last  
loop  
ids := id\_sezoane(i);  
for j in id\_echipe.first..id\_echipe.last  
loop  
ide := id\_echipe(j);  
participare.id\_sezon := ids;  
participare.id\_echipa := ide;  
insert into participari values participare;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_participari OK');  
end;  
  
<<insert\_comentarii>>  
declare  
comentariu comentarii%rowtype;  
type IdArray is table of number index by pls\_integer;  
id\_meciuri IdArray;  
id\_comentatori IdArray;  
a number(2,0);  
b number(2,0);  
c number(2,0);  
begin  
select id\_meci bulk collect into id\_meciuri from meciuri;  
select id\_comentator bulk collect into id\_comentatori from comentatori;  
  
for i in id\_meciuri.first..id\_meciuri.last  
loop  
a := dbms\_random.value(1,id\_comentatori.last);  
b := dbms\_random.value(1,id\_comentatori.last);  
c := dbms\_random.value(1,id\_comentatori.last);  
while a = b  
loop  
b := dbms\_random.value(1,id\_comentatori.last);  
end loop;  
while a = c or b = c  
loop  
c := dbms\_random.value(1,id\_comentatori.last);  
end loop;  
comentariu.id\_meci := id\_meciuri(i);  
comentariu.id\_comentator := id\_comentatori(a);  
insert into comentarii values comentariu;  
comentariu.id\_comentator := id\_comentatori(b);  
insert into comentarii values comentariu;  
comentariu.id\_comentator := id\_comentatori(c);  
insert into comentarii values comentariu;  
end loop;  
  
dbms\_output.put\_line('insert-comentarii OK');  
end;  
  
<<insert\_arbitraje>>  
declare  
arbitraj arbitraje%rowtype;  
type IdArray is table of number index by pls\_integer;  
id\_meciuri IdArray;  
id\_arbitrii IdArray;  
a number(2,0);  
b number(2,0);  
c number(2,0);  
begin  
select id\_meci bulk collect into id\_meciuri from meciuri;  
select id\_arbitru bulk collect into id\_arbitrii from arbitrii;  
  
for i in id\_meciuri.first..id\_meciuri.last  
loop  
a := dbms\_random.value(1,id\_arbitrii.last);  
b := dbms\_random.value(1,id\_arbitrii.last);  
c := dbms\_random.value(1,id\_arbitrii.last);  
while a = b  
loop  
b := dbms\_random.value(1,id\_arbitrii.last);  
end loop;  
while a = c or b = c  
loop  
c := dbms\_random.value(1,id\_arbitrii.last);  
end loop;  
arbitraj.id\_meci := id\_meciuri(i);  
arbitraj.id\_arbitru := id\_arbitrii(a);  
insert into arbitraje values arbitraj;  
arbitraj.id\_arbitru := id\_arbitrii(b);  
insert into arbitraje values arbitraj;  
arbitraj.id\_arbitru := id\_arbitrii(c);  
insert into arbitraje values arbitraj;  
end loop;  
  
dbms\_output.put\_line('insert-arbitraje OK');  
end;  
  
<<insert\_distinctii>>  
declare  
distinctie distinctii%rowtype;  
type IdArray is table of number index by pls\_integer;  
id\_sezoane IdArray;  
id\_jucatori IdArray;  
id\_premii IdArray;  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
select id\_jucator bulk collect into id\_jucatori from jucatori;  
select id\_premiu bulk collect into id\_premii from premii;  
  
for i in id\_sezoane.first..id\_sezoane.last  
loop  
for j in id\_premii.first..id\_premii.last  
loop  
distinctie.id\_sezon := id\_sezoane(i);  
distinctie.id\_premiu := id\_premii(j);  
distinctie.id\_jucator := id\_jucatori(dbms\_random.value(1, id\_jucatori.last));  
  
insert into distinctii values distinctie;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_distinctii OK');  
end;  
  
<<verifica\_inserare>>  
declare  
cnt number;  
type StringArray is varray(20) of varchar2(20);  
tabele StringArray := StringArray('angajati', 'antrenori', 'arbitrii',   
'arene', 'comentarii', 'comentatori', 'distinctii', 'echipe',   
'echipe\_medicale', 'etape', 'jucatori', 'locatii',   
'meciuri', 'nutritionisti', 'participari', 'premii', 'preparatori\_fizici',   
'sezoane', 'statistici');  
begin  
select count(\*) into cnt from angajati;  
dbms\_output.put\_line('Exista ' || cnt || ' angajati.');  
select count(\*) into cnt from antrenori;  
dbms\_output.put\_line('Exista ' || cnt || ' antrenori.');   
select count(\*) into cnt from arbitraje;  
dbms\_output.put\_line('Exista ' || cnt || ' arbitraje.');  
select count(\*) into cnt from arbitrii;  
dbms\_output.put\_line('Exista ' || cnt || ' arbitrii.');  
select count(\*) into cnt from arene;  
dbms\_output.put\_line('Exista ' || cnt || ' arene.');  
select count(\*) into cnt from comentarii;  
dbms\_output.put\_line('Exista ' || cnt || ' comentarii.');   
select count(\*) into cnt from comentatori;  
dbms\_output.put\_line('Exista ' || cnt || ' comentatori.');   
select count(\*) into cnt from distinctii;  
dbms\_output.put\_line('Exista ' || cnt || ' distinctii.');  
select count(\*) into cnt from echipe;  
dbms\_output.put\_line('Exista ' || cnt || ' echipe.');  
select count(\*) into cnt from echipe\_medicale;  
dbms\_output.put\_line('Exista ' || cnt || ' echipe\_medicale.');  
select count(\*) into cnt from etape;  
dbms\_output.put\_line('Exista ' || cnt || ' etape.');   
select count(\*) into cnt from jucatori;  
dbms\_output.put\_line('Exista ' || cnt || ' jucatori.');  
select count(\*) into cnt from locatii;  
dbms\_output.put\_line('Exista ' || cnt || ' locatii.');  
select count(\*) into cnt from meciuri;  
dbms\_output.put\_line('Exista ' || cnt || ' meciuri.');  
select count(\*) into cnt from nutritionisti;  
dbms\_output.put\_line('Exista ' || cnt || ' nutritionisti.');   
select count(\*) into cnt from participari;  
dbms\_output.put\_line('Exista ' || cnt || ' participari.');  
select count(\*) into cnt from premii;  
dbms\_output.put\_line('Exista ' || cnt || ' premii.');  
select count(\*) into cnt from preparatori\_fizici;  
dbms\_output.put\_line('Exista ' || cnt || ' preparatori\_fizici.');   
select count(\*) into cnt from sezoane;  
dbms\_output.put\_line('Exista ' || cnt || ' sezoane.');  
select count(\*) into cnt from statistici;  
dbms\_output.put\_line('Exista ' || cnt || ' statistici.');  
  
end;  
dbms\_output.put\_line('OK');  
end;  
/

## Captură de Ecran



# Problema - Colecții de Date

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

## Cerință

În fiecare etapă se determina clasamentul celor mai buni 3 jucători (cu cele mai multe puncte în meciul din etapă respectivă) din echipele care au ieșit câștigătoare, primind titlul de "Jucătorii Etapei". După aceea, pentru fiecare sezon, se cumulează punctele fiecărui "Jucător al Etapei" (doar din etapele în care a primit acest titlu) și se realizează clasamentul final. Jucătorul cu cele mai multe puncte primește premiul "MVP (most valuable player)". Pentru fiecare sezon, să se afișeze numele, prenumele și salariul jucătorului care a primit această distincție, precum și numărul de puncte cumulate.

## Rezolvare

-- ex 6  
create or replace procedure ex6\_colectii   
as  
-- Am ales tabloul imbricat deoarece numarul de sezoane este necunoscut   
-- si nu fac stergeri pe parcursul programului  
type IdSezoane is table of sezoane.id\_sezon%type ;  
id\_sezoane IdSezoane;  
id\_sezon\_curent sezoane.id\_sezon%type;  
  
-- Am ales varray deoarece stiu dinainte numarul de etape (30) dintr-un sezon,  
-- respectiv numarul de meciuri dintr-o etapa (8), si nu fac stergeri in aceasta colectie  
type IdEtape is varray(30) of etape.id\_etapa%type;  
id\_etape IdEtape;  
id\_etapa\_curenta sezoane.id\_sezon%type;  
  
type RecordJucator is record  
(  
id\_jucator jucatori.id\_jucator%type,  
puncte number  
);  
type idJucatoriTop3 is varray(3) of RecordJucator;  
id\_jucatori\_3 IdJucatoriTop3;  
rec RecordJucator;  
  
-- Am ales tablou indexat deoarece indicii vor fi anumite id-uri ale jucatorilor, nu neaparat dense.  
type TabelPuncteJucatori is table of number(4) index by pls\_integer;  
puncte\_jucator TabelPuncteJucatori := TabelPuncteJucatori();  
  
nr\_etape number; -- restriciile spun ca pentru fiecare sezon exista 30 etape  
  
maxim number;  
id\_mvp jucatori.id\_jucator%type;  
jucator jucatori%rowtype;  
  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
  
for it\_sezon in id\_sezoane.first..id\_sezoane.last loop  
id\_sezon\_curent := id\_sezoane(it\_sezon);  
select count(id\_etapa) collect into nr\_etape   
from etape  
where id\_sezon = id\_sezon\_curent;  
  
if nr\_etape != 30  
then   
dbms\_output.put\_line ('In sezonul ' || id\_sezon\_curent ||   
' nu sunt 30 de etape ( ' || nr\_etape || ' )');  
continue;  
end if;  
  
select id\_etapa bulk collect into id\_etape   
from etape  
where id\_sezon = id\_sezon\_curent;  
  
puncte\_jucator := TabelPuncteJucatori();  
maxim := 0;  
for it\_etapa in id\_etape.first..id\_etape.last loop  
id\_etapa\_curenta := id\_etape(it\_etapa);  
with jucatori\_si\_puncte as  
(select j.id\_jucator, 2 \* aruncari\_2pct\_marcate + 3 \* aruncari\_3pct\_marcate + aruncari\_libere\_marcate  
from jucatori j  
join meciuri m on m.id\_etapa = id\_etapa\_curenta and  
((m.id\_echipa\_gazda = j.id\_echipa and scor\_gazda > scor\_oaspete ) or  
(m.id\_echipa\_oaspete = j.id\_echipa and scor\_gazda < scor\_oaspete ))  
join statistici s on s.id\_meci = m.id\_meci and s.id\_jucator = j.id\_jucator  
order by 2 desc)  
select \* bulk collect into id\_jucatori\_3  
from jucatori\_si\_puncte  
where rownum <= 3;  
  
for it in id\_jucatori\_3.first..id\_jucatori\_3.last loop  
rec := id\_jucatori\_3(it);  
if puncte\_jucator.exists(rec.id\_jucator) = false  
then puncte\_jucator(rec.id\_jucator) := 0;  
end if;  
  
puncte\_jucator(rec.id\_jucator) := puncte\_jucator(rec.id\_jucator) + rec.puncte;  
if maxim < puncte\_jucator(rec.id\_jucator)  
then  
maxim := puncte\_jucator(rec.id\_jucator);  
id\_mvp := rec.id\_jucator;  
end if;  
end loop;  
end loop;  
  
select \* into jucator  
from jucatori  
where id\_jucator = id\_mvp;  
  
dbms\_output.put\_line(jucator.nume || ' ' || jucator.prenume || ' id: ' || id\_mvp || ' puncte: ' || puncte\_jucator(id\_mvp));  
  
end loop;  
  
end;  
/  
  
begin  
ex6\_colectii();  
end;  
/

## Captură de Ecran

# 

# Problemă - Cursoare

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

## Cerință

Se bănuiește că există arbitrii care au influențat rezultatele meciurilor. Comisia a hotărât să găsească meciurile în care diferența de puncte este maximă și să verifice dacă arbitrajul a fost realizat de arbitrii care au legătură cu vreun jucător de pe teren (au același nume de familie). Să se găsească toate aceste perechi, afișându-se, pentru fiecare:

IdMeci: NumeArbitru PrenumeArbitru - NumeJucator PrenumeJucator - ScorMeci - Mesaj

Mesajul ne spune daca jucătorul care are legătură cu arbitrul s-a aflat in echipa câștigătoare sau cea înfrântă.

## Rezolvare

create or replace procedure ex7\_cursoare  
is  
cursor c\_id\_arbitri (idm meciuri.id\_meci%type) is  
select id\_arbitru from arbitraje where arbitraje.id\_meci = idm;  
  
  
type IdArbitri is table of arbitrii.id\_arbitru%type;  
id\_arbitri IdArbitri;  
arbitru arbitrii%rowtype;  
  
cursor meciuriVerificate is  
select \*  
from meciuri  
where abs(scor\_gazda - scor\_oaspete) =   
(select max(abs(scor\_gazda - scor\_oaspete))  
from meciuri);  
  
cursor arbitriVerificati (idm meciuri.id\_meci%type) is  
select \*  
from arbitrii  
where id\_arbitru in (select id\_arbitru from arbitraje where arbitraje.id\_meci = idm);  
  
cursor jucatoriMeci (  
id\_gazda echipe.id\_echipa%type,  
id\_oaspete echipe.id\_echipa%type,   
nume\_arbitru arbitrii.nume%type) is  
select id\_jucator , nume, prenume, id\_echipa  
from jucatori   
where (id\_echipa = id\_gazda or id\_echipa = id\_oaspete)  
and jucatori.nume = nume\_arbitru;  
  
mesaj varchar2(50);  
id\_echipa echipe.id\_Echipa%type;  
begin  
  
-- CICLU CURSOR  
for meci in meciuriVerificate loop  
-- dbms\_output.put\_line('meci: ' || meci.id\_meci);  
  
-- CURSOR CLASIC PARAMETRIZAT  
open c\_id\_arbitri(meci.id\_meci);  
fetch c\_id\_arbitri bulk collect into id\_arbitri;  
close c\_id\_arbitri;  
  
for i in id\_arbitri.first..id\_arbitri.last loop  
select \* into arbitru  
from arbitrii  
where id\_arbitru = id\_arbitri(i);  
-- dbms\_output.put\_line('arbitru: ' || arbitru.id\_arbitru);  
-- CICLU CURSOR PARAMETRIZAT  
for jucator in jucatoriMeci(meci.id\_echipa\_gazda, meci.id\_echipa\_oaspete, arbitru.nume) loop  
  
if meci.scor\_gazda > meci.scor\_oaspete  
then id\_echipa := meci.id\_echipa\_gazda;  
else id\_echipa := meci.id\_echipa\_oaspete;  
end if;  
  
if jucator.id\_echipa = id\_echipa  
then mesaj := ' (Victorie)';  
else mesaj := ' (Infrangere)';  
end if;  
  
dbms\_output.put\_line(meci.id\_meci || ': ' ||  
arbitru.id\_arbitru || ' ' || arbitru.nume || ' ' || arbitru.prenume || ' - ' ||  
jucator.id\_jucator || ' ' || jucator.nume || ' ' || jucator.prenume || ' - ' ||   
meci.scor\_gazda || '-' || meci.scor\_oaspete || mesaj);  
  
end loop;  
end loop;  
end loop;  
  
end;  
/  
  
begin  
ex7\_cursoare();  
end;  
/

## Captura de Ecran

# 

# Problema - Excepții

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 proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate

## Cerință

Se da un prenume P și un salariu minim S. Pentru fiecare echipa se calculează suma salariilor jucătorilor și angajaților cu prenumele P și salariul mai mare sau egal decât S. Să se afișeze numele echipelor care au cel puț în un angajat sau jucător care respectă condițiile și suma calculată , în ordine descrescătoare a sumei. Funcția va retuna suma tuturor salariilor ce respectă condițiile . Să se trateze cazurile în care nu există niciun jucător sau angajat cu prenumele P, respectiv cazul în care niciun jucător sau angajat cu prenumele P nu are salariul mai mare sau egal decât S.

## Rezolvare

create or replace function ex8\_exceptii  
(p\_prenume in varchar2,  
p\_salariu\_minim in number)  
return number  
is  
ex\_no\_first\_name exception;  
ex\_no\_salary exception;  
type RecordEchipa is Record (  
-- id\_echipa echipe.id\_echipa%type,  
nume varchar2(50),  
suma\_salarii number  
);  
type ListaEchipe is table of RecordEchipa;  
lista\_echipe ListaEchipe;  
  
aux number := 0;  
  
begin  
select count(\*) collect into aux  
from (select id\_jucator from jucatori where lower(prenume) like lower(p\_prenume)  
union select id\_angajat from angajati where lower(prenume) like lower(p\_prenume));  
  
if aux = 0 then  
raise ex\_no\_first\_name;  
end if;  
  
with persoane as (  
select e.id\_echipa id\_echipa, salariu   
from angajati a  
join echipe e on e.id\_antrenor = a.id\_angajat or e.id\_preparator = a.id\_angajat   
or e.id\_nutritionist = a.id\_angajat  
where lower(a.prenume) like lower(p\_prenume) and a.salariu >= p\_salariu\_minim  
union  
select id\_echipa, salariu from jucatori  
where lower(prenume) like lower(p\_prenume) and salariu >= p\_salariu\_minim  
)  
select \* bulk collect into lista\_echipe  
from (  
select e.nume, sum(p.salariu)   
from persoane p  
join echipe e on e.id\_echipa = p.id\_echipa  
group by e.nume  
)  
order by 2 desc;  
  
  
if lista\_echipe.count = 0 then  
raise ex\_no\_salary;  
end if;  
  
aux := 0;  
for i in lista\_Echipe.first..lista\_echipe.last loop  
dbms\_output.put\_line(lista\_echipe(i).nume || ' - ' || lista\_echipe(i).suma\_salarii);  
aux := aux + lista\_echipe(i).suma\_salarii;  
end loop;  
  
return aux;  
  
exception  
when ex\_no\_first\_name then  
dbms\_output.put\_line('Exceptie: nu exista prenumele ' || p\_prenume);  
return 0;  
when ex\_no\_salary then  
dbms\_output.put\_line('Exceptie: nu exista salarii mai mari decat ' || p\_salariu\_minim);  
return 0;  
when others then  
dbms\_output.put\_line('Alta Exceptie');  
return 0;  
end;  
/  
  
begin  
  
dbms\_output.put\_line(ex8\_exceptii('isaac', 50000));  
dbms\_output.put\_line(ex8\_exceptii('prenume', 1000));  
dbms\_output.put\_line(ex8\_exceptii('isaac', 100000));  
end;  
/

## Captură de Ecran

# 

# Problemă - Subprogram de Tip Procedura

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.

## Cerință

Se știe numele unui angajat. Să se afișeze numele echipei pentru care lucrează și postul acestuia. Pentru echipa în care acesta lucrează ( să se afișeze numele și id-ul), să se afle jucătorii care au câștigat primit cel puț în o distincție . Pentru fiecare dintre aceștia să se afișeze următoarele statistici: procentajul aruncărilor de 3 puncte marcate, procentajul meciurilor in care a faultat și numărul mediu de pase decisive.

## Rezolvare

create or replace procedure exceptii\_ex9  
(p\_nume in angajati.nume%type  
)  
is  
ex\_no\_player exception;  
type RecAngajat is record (  
nume angajati.nume%type,   
prenume angajati.prenume%type,  
nume\_echipa echipe.nume%type,   
post varchar2(20)  
);  
angajat RecAngajat;  
type RecEchipa is record (  
id echipe.id\_echipa%type,  
nume echipe.nume%type);  
echipa RecEchipa;  
  
type RecJucator is record(  
id jucatori.id\_jucator%type,  
nume jucatori.nume%type,   
prenume jucatori.prenume%type,   
proc\_3pct number(5, 2),   
proc\_faulturi number(5, 2),  
avg\_pase number(5, 2)  
);  
type ListaJucatori is table of RecJucator;  
lista\_jucatori ListaJucatori;  
  
begin  
  
select a.nume, a.prenume, e.nume, (case  
when a.id\_angajat = e.id\_antrenor then 'antrenor'  
when a.id\_angajat = e.id\_preparator then 'preparator fizic'  
when a.id\_angajat = e.id\_nutritionist then 'nutritionist'  
end) post  
into angajat  
from angajati a, echipe e  
where lower(a.nume) = lower(p\_nume)  
and id\_angajat in (e.id\_antrenor, e.id\_preparator, e.id\_nutritionist);  
  
dbms\_output.put\_line(angajat.nume || ' ' || angajat.prenume || ' lucreaza pentru echipa ' ||   
angajat.nume\_echipa || ' pe postul de ' || angajat.post);  
  
with info\_echipa as (  
select id\_echipa, e.nume  
from echipe e   
where (p\_nume) in  
(select nume  
from angajati  
where lower(nume) = lower(p\_nume)  
and id\_angajat in (e.id\_antrenor, e.id\_preparator, e.id\_nutritionist))  
)  
select j.id\_jucator, j.nume, j.prenume, 100 \* sum(s.aruncari\_3pct\_marcate) / sum(s.aruncari\_3pct),  
100 \* sum(case when s.faulturi > 0 then 1 else 0 end) / count(\*), avg(s.pase\_decisive)  
bulk collect into lista\_jucatori  
from jucatori j  
join statistici s on s.id\_jucator = j.id\_jucator  
where j.id\_jucator in (select distinct id\_jucator from distinctii)  
and j.id\_echipa = (select id\_echipa from info\_echipa)  
group by (j.id\_jucator, j.nume, j.prenume);  
  
if lista\_jucatori.count = 0 then  
raise ex\_no\_player;  
end if;  
  
for i in lista\_jucatori.first..lista\_jucatori.last loop  
dbms\_output.put\_line(lista\_jucatori(i).nume || ' ' || lista\_jucatori(i).prenume || ' - procentaj 3pct: '   
|| lista\_jucatori(i).proc\_3pct || '% - procentaj meciuri cu fault: ' || lista\_jucatori(i).proc\_faulturi ||   
'% - medie pase decisive: ' || lista\_jucatori(i).avg\_pase);  
end loop;  
  
exception  
when no\_data\_found then  
dbms\_output.put\_line('Nu exista niciun angajat cu numele ' || p\_nume);  
when too\_many\_rows then  
dbms\_output.put\_line('Exista mai multi angajati cu numele ' || p\_nume);  
when ex\_no\_player then  
dbms\_output.put\_line('Nu exista niciun jucator care sa fi primit vreo distinctie in echipa ' || angajat.nume\_echipa);  
end;  
/  
  
begin  
dbms\_output.put\_line('Apel cu numele: Johnson');  
exceptii\_ex9('Johnson');  
dbms\_output.put\_line('Apel cu numele: Carter');  
exceptii\_ex9('Carter');  
dbms\_output.put\_line('Apel cu numele: Clark');  
exceptii\_ex9('Clark');  
dbms\_output.put\_line('Apel cu numele: John');  
exceptii\_ex9('John');  
end;  
  
/

## Captură de Ecran

# 

# Problemă - Trigger LMD la nivel de comanda

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

## Cerință

Să se definească un trigger la fiecare ștergere sau modificare a tabelei jucători, să afișeze echipele care nu au numărul minim de jucători (5) pentru a juca meciuri.

## Rezolvare

create or replace trigger min5\_jucatori\_echipa  
after delete or update on jucatori  
declare  
type ListaNumeEchipe is table of echipe.nume%type;  
nume\_echipe ListaNumeEchipe;  
begin  
select e.nume bulk collect into nume\_echipe  
from echipe e   
where 5 > (select count(\*) from jucatori j where j.id\_echipa = e.id\_echipa);  
  
if nume\_echipe.count > 0 then  
dbms\_output.put\_line('Urmatoarele Echipe nu au suficienti jucatori pentru a juca meciuri');  
for i in nume\_echipe.first..nume\_echipe.last loop  
dbms\_output.put\_line(nume\_echipe(i));  
end loop;  
end if;  
end;  
/  
  
begin  
update jucatori  
set id\_echipa = 1236  
where nume = 'Green';  
end;  
/

## Captură de Ecran

# 

# Problemă - Trigger LMD la nivel de linie

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

## Cerință

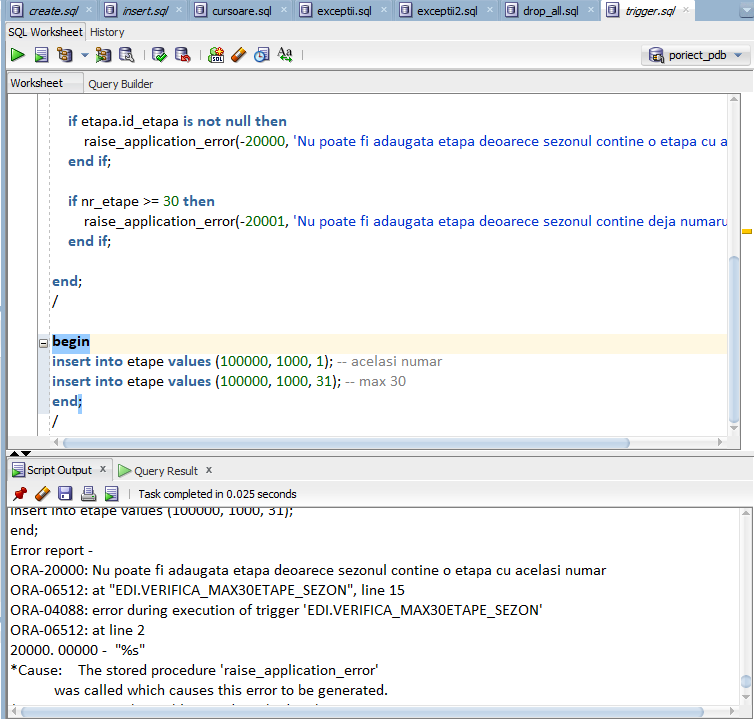
Să se definească un trigger pentru următoarea constrângere:

* Să nu existe mai mult de 30 de etape într-un sezon, și să nu existe două etape într-un sezon cu același număr.

## Rezolvare

create or replace trigger verifica\_max30etape\_sezon  
before insert on etape  
for each row  
declare  
nr\_etape number := 0;  
  
etapa etape%rowtype;  
begin  
select count(\*) into nr\_etape  
from etape   
where id\_sezon = :new.id\_sezon;  
  
if nr\_etape >= 30 then  
raise\_application\_error(-20001, 'Nu poate fi adaugata etapa deoarece sezonul contine deja numarul maxim (30)');  
end if;  
  
select \* into etapa  
from etape  
where id\_sezon = :new.id\_sezon and numar = :new.numar;  
  
if etapa.id\_etapa is not null then  
raise\_application\_error(-20000, 'Nu poate fi adaugata etapa deoarece sezonul contine o etapa cu acelasi numar');  
end if;  
  
exception  
when no\_data\_found then  
nr\_etape := 0;  
end;  
/  
  
-- max 30  
insert into etape values (1, 1000, 1);   
  
-- etapa cu acelasi numar  
insert into sezoane values (10000, '', '');  
insert into etape values(10016, 10000, 6);  
insert into etape values(10015, 10000, 6);

## Captură de Ecran



# Problema - Trigger LDD

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

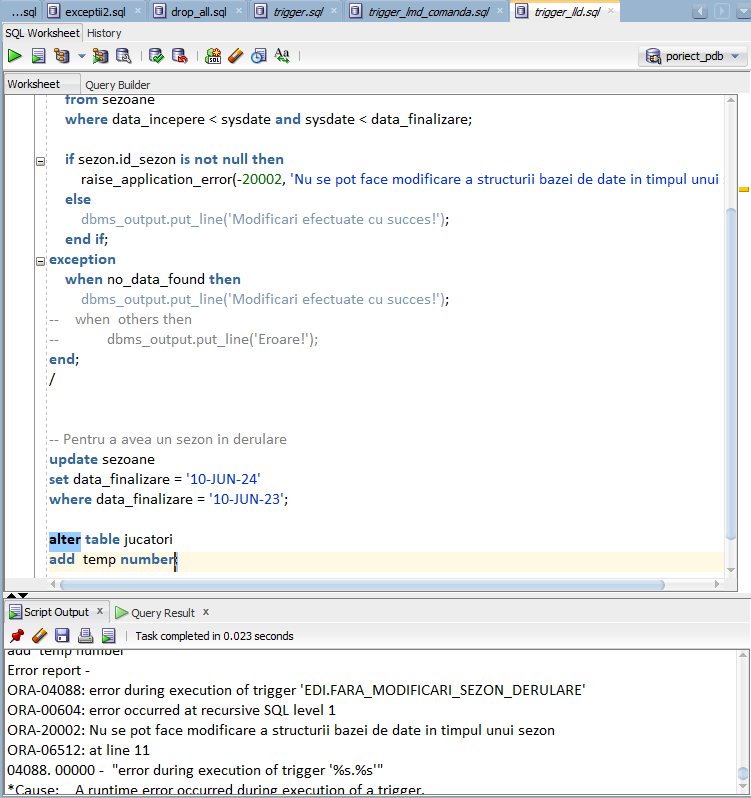
## Cerință

Să se definească un trigger care să împiedice adăugarea, ștergerea sau modificarea tabelelor dacă există un sezon în derulare în acel moment.

## Rezolvare

create or replace trigger fara\_modificari\_sezon\_derulare  
before ddl  
on schema  
declare   
sezon sezoane%rowtype;  
begin  
select \* into sezon  
from sezoane  
where data\_incepere < sysdate and sysdate < data\_finalizare;  
  
if sezon.id\_sezon is not null then  
raise\_application\_error(-20002, 'Nu se pot face modificare a structurii bazei de date in timpul unui sezon');  
else  
dbms\_output.put\_line('Modificari efectuate cu succes!');  
end if;  
exception  
when no\_data\_found then  
dbms\_output.put\_line('Modificari efectuate cu succes!');  
-- when others then  
-- dbms\_output.put\_line('Eroare!');  
end;  
/  
  
  
-- Pentru a avea un sezon in derulare  
update sezoane  
set data\_finalizare = '10-JUN-24'   
where data\_finalizare = '10-JUN-23';  
  
alter table jucatori   
add temp number;

## Captură de Ecran



# Opțional - Definire Pachet 1

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

## Cod SQL

create or replace package pachet as   
function get\_id return number;  
function prenume\_aleator return varchar2;  
function nume\_aleator return varchar2;  
  
procedure sterge\_date;  
procedure insert\_sezoane;  
procedure insert\_etape;  
procedure insert\_antrenori;  
procedure insert\_preparatori;  
procedure insert\_nutritionisti;  
procedure insert\_locatii;  
procedure insert\_arene;  
procedure insert\_echipe;  
procedure insert\_jucatori;  
procedure insert\_echipe\_medicale;  
procedure insert\_meciuri;  
procedure insert\_arbitri;  
procedure insert\_comentatori;  
procedure insert\_statistici;  
procedure insert\_premii;  
procedure insert\_participari;  
procedure insert\_comentarii;  
procedure insert\_arbitraje;  
procedure insert\_distinctii;  
procedure verifica\_inserare;  
  
procedure efectueaza\_inserare;  
procedure ex6\_colectii;  
procedure ex7\_cursoare;  
  
function ex8\_exceptii(p\_prenume in varchar2, p\_salariu\_minim in number) return number;  
procedure ex9\_exceptii (p\_nume in angajati.nume%type);  
end pachet;  
/  
  
create or replace package body pachet as  
function get\_id   
return number is  
f\_id number;  
begin  
select idseq.nextval into f\_id  
from dual;  
return f\_id;  
end get\_id;  
  
function prenume\_aleator  
return varchar2 as   
prenume varchar2(20);   
type StringArray is varray(20) of varchar2(20);  
lista\_prenume StringArray := StringArray(  
'Ethan', 'Isaac', 'Leo', 'Miles', 'Asher',  
'Maxwell', 'Oscar', 'Dylan', 'Oliver', 'Harrison',  
'Nathan', 'Gabriel', 'Jasper', 'Ezra', 'Silas',  
'Sebastian', 'Caleb', 'Gideon', 'Wyatt', 'Finn'  
);  
begin  
prenume := lista\_prenume(dbms\_random.value(1, lista\_prenume.last));  
return prenume;  
end prenume\_aleator;  
  
function nume\_aleator  
return varchar2 as   
nume varchar2(20);   
type StringArray is varray(50) of varchar2(20);  
lista\_nume StringArray := StringArray('Smith', 'Johnson', 'Williams', 'Jones', 'Brown',   
'Davis', 'Miller', 'Wilson', 'Moore', 'Taylor', 'Anderson', 'Thomas', 'Jackson',  
'White', 'Harris', 'Martin', 'Thompson', 'Garcia', 'Martinez', 'Robinson', 'Clark',   
'Rodriguez', 'Lewis', 'Lee', 'Walker', 'Hall', 'Allen', 'Young', 'Hernandez', 'King',   
'Wright', 'Lopez', 'Hill', 'Scott', 'Green', 'Adams', 'Baker', 'Gonzalez', 'Nelson',  
'Carter', 'Mitchell', 'Perez', 'Roberts', 'Turner', 'Phillips', 'Campbell', 'Parker', 'Evans', 'Edwards');  
  
  
begin  
nume := lista\_nume(dbms\_random.value(1, lista\_nume.last));  
return nume;  
end nume\_aleator;  
  
procedure sterge\_date is   
begin  
delete from arbitraje;  
delete from comentarii;  
delete from distinctii;  
delete from participari;  
  
  
delete from premii;  
delete from statistici;  
  
delete from arbitrii;  
delete from comentatori;  
delete from meciuri;  
delete from echipe\_medicale;  
  
delete from jucatori;  
delete from echipe;  
delete from arene;  
delete from locatii;  
  
delete from preparatori\_fizici;  
delete from nutritionisti;  
delete from antrenori;  
delete from angajati;  
  
delete from etape;  
delete from sezoane;  
end sterge\_date;  
  
procedure insert\_sezoane is   
v\_numar\_sezoane number := 5;  
v\_format\_data varchar2(11) := 'dd-mon-yyyy';  
v\_data\_start date := to\_date('15-aug-2022',v\_format\_data);  
v\_data\_final date := to\_date('10-jun-2023', v\_format\_data);  
sezon sezoane%rowtype;  
begin  
sezon.data\_incepere := v\_data\_start;  
sezon.data\_finalizare := v\_data\_final;  
sezon.id\_sezon := get\_id();  
for cnt in 1..v\_numar\_sezoane  
loop  
insert into sezoane  
values sezon;  
sezon.data\_incepere := add\_months(sezon.data\_incepere, -12);  
sezon.data\_finalizare := add\_months(sezon.data\_finalizare, -12);  
sezon.id\_sezon := get\_id();  
end loop;  
  
dbms\_output.put\_line('insert\_sezoane OK');  
end insert\_sezoane;  
  
procedure insert\_etape is  
v\_numar\_etape number := 30;  
type id\_sezoane is table of sezoane.id\_sezon%type index by pls\_integer;  
v\_id\_sezoane id\_sezoane;   
v\_etapa etape%rowtype;  
begin  
select id\_sezon  
bulk collect into v\_id\_sezoane  
from sezoane;  
  
for cnt\_sezon in v\_id\_sezoane.first..v\_id\_sezoane.last  
loop  
for cnt\_etapa in 1..v\_numar\_etape  
loop  
v\_etapa.id\_etapa := get\_id();  
v\_etapa.id\_sezon := v\_id\_sezoane(cnt\_sezon);  
v\_etapa.numar := cnt\_etapa;  
insert into etape  
values v\_etapa;  
end loop;  
end loop;  
dbms\_output.put\_line('insert\_etape OK');  
end insert\_etape;  
  
procedure insert\_antrenori is  
numar\_antrenori number := 16;  
angajat angajati%rowtype;  
antrenor antrenori%rowtype;  
begin  
for i in 1..numar\_antrenori  
loop  
angajat.id\_angajat := get\_id();  
angajat.nume := nume\_aleator();  
angajat.prenume := prenume\_aleator();  
angajat.salariu := 100 \* dbms\_random.value(100, 200);  
antrenor.id\_angajat := angajat.id\_angajat;  
  
insert into angajati values angajat;  
insert into antrenori values antrenor;  
end loop;  
dbms\_output.put\_line('insert\_antrenori OK');  
end insert\_antrenori;  
  
procedure insert\_preparatori is  
numar\_preparatori number := 16;  
angajat angajati%rowtype;  
preparator preparatori\_fizici%rowtype;  
begin  
for i in 1..numar\_preparatori  
loop  
angajat.id\_angajat := get\_id();  
angajat.nume := nume\_aleator();  
angajat.prenume := prenume\_aleator();  
angajat.salariu := 100 \* dbms\_random.value(100, 200);  
preparator.id\_angajat := angajat.id\_angajat;  
  
insert into angajati values angajat;  
insert into preparatori\_fizici values preparator;  
end loop;  
dbms\_output.put\_line('insert\_preparatori OK');  
end insert\_preparatori;  
  
procedure insert\_nutritionisti is  
numar\_nutritionisti number := 16;  
angajat angajati%rowtype;  
nutritionist nutritionisti%rowtype;  
begin  
for i in 1..numar\_nutritionisti  
loop  
angajat.id\_angajat := get\_id();  
angajat.nume := nume\_aleator();  
angajat.prenume := prenume\_aleator();  
angajat.salariu := 100 \* dbms\_random.value(100, 200);  
nutritionist.id\_angajat := angajat.id\_angajat;  
  
insert into angajati values angajat;  
insert into nutritionisti values nutritionist;  
end loop;  
dbms\_output.put\_line('insert\_nutritionisti OK');  
end insert\_nutritionisti;  
  
procedure insert\_locatii is  
type StringArray is varray(16) of varchar2(30);  
orase StringArray := StringArray('New York City', 'Los Angeles','Las Vegas',  
'Chicago', 'San Francisco', 'Miami', 'Orlando', 'Houston','Seattle',   
'Washington D.C.', 'Boston', 'Atlanta', 'Dallas', 'Denver',   
'New Orleans', 'San Diego');  
strazi StringArray := StringArray('Fifth Avenue', 'Hollywood Boulevard',   
'Las Vegas Boulevard', 'Michigan Avenue', 'Lombard Street',   
'Ocean Drive', 'International Drive', 'NASA Road 1', 'Pike Place Market',   
'1600 Pennsylvania Avenue NW', 'Fenway Park', 'Peachtree Street',   
'Dealey Plaza', '16th Street Mall', 'Bourbon Street', 'Balboa Park');  
locatie locatii%rowtype;  
nr\_locatii number := 16;  
begin  
for i in 1..nr\_locatii  
loop  
locatie.id\_locatie := get\_id();  
locatie.tara := 'USA';  
locatie.oras := orase(i);  
locatie.strada := strazi(i);  
locatie.nr := dbms\_random.value(100, 1000);  
insert into locatii values locatie;  
end loop;  
dbms\_output.put\_line('insert\_locatii OK');  
end insert\_locatii;  
  
procedure insert\_arene is  
type IdLocatii is table of locatii.id\_locatie%type index by pls\_integer;  
id\_locatii IdLocatii;  
numar\_arene number := 16;  
type StringArray is varray(16) of varchar2(30);  
lista\_arene StringArray := StringArray('The Thunderdome', 'The Coliseum', 'The Pit',   
'The Garden', 'The Staples Center', 'The Oracle', 'The Hoop House', 'The Den',  
'The Arena', 'The Thunderdome', 'The Dome', 'The Palace',  
'The Madhouse', 'The Pavilion', 'The Buzzer Beater', 'The Swish Center');  
arena arene%rowtype;  
begin  
select id\_locatie  
bulk collect into id\_locatii  
from locatii;  
  
for i in 1..numar\_arene  
loop  
arena.id\_arena := get\_id();  
arena.id\_locatie := id\_locatii(i);  
arena.nume := lista\_arene(i);  
arena.locuri := 1000 \* dbms\_random.value(10, 20);  
  
insert into arene values arena;  
end loop;  
dbms\_output.put\_line('insert\_arene OK');  
end insert\_arene;  
  
procedure insert\_echipe is  
type StringArray is varray(16) of varchar2(20);  
lista\_nume StringArray := StringArray('Lightning Bolts', 'Thunderbirds',   
'Wildcats', 'Heatwave', 'Hurricanes', 'Jaguars', 'Patriots', 'Titans',  
'Vikings', 'Dragons', 'Raptors', 'Warriors',  
'Hornets', 'Sharks', 'Lions', 'Knights');  
  
type IdTable is table of number index by pls\_integer;  
id\_arene IdTable;  
id\_antrenori IdTable;  
id\_preparatori IdTable;  
id\_nutritionisti IdTable;  
echipa echipe%rowtype;  
numar\_echipe number := 16;  
begin  
select id\_arena bulk collect into id\_arene from arene;  
select id\_angajat bulk collect into id\_antrenori from antrenori;  
select id\_angajat bulk collect into id\_preparatori from preparatori\_fizici;  
select id\_angajat bulk collect into id\_nutritionisti from nutritionisti;  
  
for i in 1..numar\_echipe  
loop  
echipa.id\_echipa := get\_id();  
echipa.id\_arena := id\_arene(i);  
echipa.id\_antrenor := id\_antrenori(i);  
echipa.id\_preparator := id\_preparatori(i);  
echipa.id\_nutritionist := id\_nutritionisti(i);  
echipa.nume := lista\_nume(i);  
echipa.an\_infiintare := 1960 + dbms\_random.value(0, 30);  
  
insert into echipe values echipa;  
end loop;  
dbms\_output.put\_line('insert\_echipe OK');  
end insert\_echipe;   
  
procedure insert\_jucatori is   
type IdArray is table of echipe.id\_echipa%type index by pls\_integer;  
id\_echipe IdArray;  
id\_echipa echipe.id\_echipa%type;  
jucator jucatori%rowtype;  
numar\_jucatori\_per\_echipa number := 5;  
begin  
select id\_echipa bulk collect into id\_echipe from echipe;  
  
for i in id\_echipe.first..id\_echipe.last  
loop  
id\_echipa := id\_echipe(i);  
for i in 1..numar\_jucatori\_per\_echipa  
loop  
jucator.id\_jucator := get\_id();  
jucator.id\_echipa := id\_echipa;  
jucator.nume := nume\_aleator();  
jucator.prenume := prenume\_aleator();  
jucator.inaltime := dbms\_random.value(1.80, 2.25);  
jucator.salariu := 1000 \* dbms\_random.value(40, 100);  
  
insert into jucatori values jucator;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_jucatori OK');  
end insert\_jucatori;  
  
procedure insert\_echipe\_medicale is  
numar\_echipe\_medicale number := 5;  
begin  
for i in 1..numar\_echipe\_medicale  
loop   
insert into echipe\_medicale values(get\_id());  
end loop;  
  
dbms\_output.put\_line('insert\_echipe\_medicale OK');  
end insert\_echipe\_medicale;  
  
procedure insert\_meciuri is  
type IdArray is table of number index by pls\_integer;  
id\_sezoane IdArray;  
id\_echipe IdArray;  
id\_echipe\_med IdArray;  
id\_etape IdArray;  
meci meciuri%rowtype;  
type IntArray is varray(8) of number;  
x1 IntArray := IntArray(1, 2, 3, 4, 5, 6, 7, 8);  
x2 IntArray := IntArray(16, 15, 14, 13, 12, 11, 10, 9);  
rev boolean := false;  
id\_gazda number;  
id\_oaspete number;  
temp number;  
ids sezoane.id\_sezon%type;  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
select id\_echipa bulk collect into id\_echipe from echipe;  
select id\_echipa\_medicala bulk collect into id\_echipe\_med from echipe\_medicale;  
  
  
  
for i in id\_sezoane.first..id\_sezoane.last  
loop  
ids := id\_sezoane(i);  
select id\_etapa bulk collect into id\_etape   
from etape  
where id\_sezon = ids;  
  
for nr\_etapa in 1..30  
loop  
for i in 1..8  
loop  
if rev = false  
then   
id\_gazda := id\_echipe(x1(i));  
id\_oaspete := id\_echipe(x2(i));  
else  
id\_gazda := id\_echipe(x2(i));  
id\_oaspete := id\_echipe(x1(i));  
end if;  
  
meci.id\_meci := get\_id();  
meci.id\_etapa := id\_etape(nr\_etapa);  
meci.id\_echipa\_gazda := id\_gazda;  
meci.id\_echipa\_oaspete := id\_oaspete;  
meci.id\_echipa\_medicala := id\_echipe\_med(dbms\_random.value(1, id\_echipe\_med.last));  
meci.scor\_gazda := dbms\_random.value(60, 100);  
meci.scor\_oaspete := meci.scor\_gazda + (dbms\_random.value(0, 94) - 47);  
  
insert into meciuri values meci;  
  
end loop;  
temp := x2(1);  
for i in 1..7  
loop  
x2(i) := x2(i+1);  
end loop;  
x2(8) := x1(8);  
for i in reverse 3..8  
loop  
x1(i) := x1(i-1);  
end loop;  
x1(2) := temp;  
  
if x1(2) = 2  
then rev := true;  
end if;  
  
end loop;  
end loop;  
end insert\_meciuri;   
  
procedure insert\_arbitri is  
arbitru arbitrii%rowtype;  
numar\_arbitrii number := 50;  
begin  
for i in 1..numar\_arbitrii  
loop  
arbitru.nume := nume\_aleator();  
arbitru.prenume := prenume\_aleator();  
arbitru.id\_arbitru := get\_id();  
arbitru.data\_obtinere\_licenta := to\_date(trunc(  
dbms\_random.value(to\_char(date '1990-01-01','J') ,to\_char(date '2015-12-31','J') )  
),'J' );  
insert into arbitrii values arbitru;  
end loop;  
  
dbms\_output.put\_line('insert\_arbitrii OK');  
end insert\_arbitri;  
  
procedure insert\_comentatori is  
comentator comentatori%rowtype;  
numar\_comentatori number := 10;  
begin  
for i in 1..numar\_comentatori  
loop  
comentator.nume := nume\_aleator();  
comentator.prenume := prenume\_aleator();  
comentator.id\_comentator := get\_id();  
insert into comentatori values comentator;  
end loop;  
  
dbms\_output.put\_line('insert\_comentatori OK');  
end insert\_comentatori;  
  
procedure insert\_statistici is  
type IdArray is table of number index by pls\_integer;  
id\_meciuri IdArray;  
id\_jucatori IdArray;  
statistica statistici%rowtype;  
meci meciuri%rowtype;  
idm meciuri.id\_meci%type;  
idj jucatori.id\_jucator%type;  
begin  
select id\_meci bulk collect into id\_meciuri from meciuri;  
  
for i in id\_meciuri.first..id\_meciuri.last  
loop  
idm := id\_meciuri(i);  
select \* into meci from meciuri where id\_meci = idm;  
select id\_jucator bulk collect into id\_jucatori  
from jucatori  
where id\_echipa = meci.id\_echipa\_gazda or id\_echipa = meci.id\_echipa\_oaspete;  
  
for j in id\_jucatori.first..id\_jucatori.last  
loop  
idj := id\_jucatori(j);  
statistica.id\_statistica := get\_id();  
statistica.id\_meci := idm;  
statistica.id\_jucator := idj;  
statistica.minute\_jucate := dbms\_random.value(20, 48);  
statistica.aruncari\_2pct := dbms\_random.value(0, 30);  
statistica.aruncari\_2pct\_marcate := dbms\_random.value(0, statistica.aruncari\_2pct);  
statistica.aruncari\_3pct := dbms\_random.value(0, 20);  
statistica.aruncari\_3pct\_marcate := dbms\_random.value(0, statistica.aruncari\_3pct);  
statistica.aruncari\_libere := dbms\_random.value(0, 10);  
statistica.aruncari\_libere\_marcate := dbms\_random.value(0, statistica.aruncari\_libere);  
statistica.pase\_decisive := dbms\_random.value(0, 25);  
statistica.recuperari := dbms\_random.value(0,15);  
statistica.faulturi := dbms\_random.value(0, 5);  
  
insert into statistici values statistica;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_statistica OK');  
end insert\_statistici;  
  
procedure insert\_premii is  
type StringArray is varray(5) of varchar2(50);  
lista\_premii StringArray := StringArray('Most Valuable Player (MVP)',   
'Team Player of the Year',  
'Defensive Player of the Year', 'Sportsmanship Award', 'Best Distance Shooter');  
premiu premii%rowtype;  
begin  
for i in lista\_premii.first..lista\_premii.last  
loop  
premiu.id\_premiu := get\_id();  
premiu.denumire := lista\_premii(i);  
insert into premii values premiu;  
end loop;  
  
dbms\_output.put\_line('inser\_premii OK');  
end insert\_premii;  
  
procedure insert\_participari is  
type IdArray is table of number index by pls\_integer;  
id\_sezoane IdArray;  
id\_echipe IdArray;  
participare participari%rowtype;  
ids sezoane.id\_sezon%type;  
ide echipe.id\_echipa%type;  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
select id\_echipa bulk collect into id\_echipe from echipe;  
  
for i in id\_sezoane.first..id\_sezoane.last  
loop  
ids := id\_sezoane(i);  
for j in id\_echipe.first..id\_echipe.last  
loop  
ide := id\_echipe(j);  
participare.id\_sezon := ids;  
participare.id\_echipa := ide;  
insert into participari values participare;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_participari OK');  
end insert\_participari;  
  
procedure insert\_comentarii is  
comentariu comentarii%rowtype;  
type IdArray is table of number index by pls\_integer;  
id\_meciuri IdArray;  
id\_comentatori IdArray;  
a number(2,0);  
b number(2,0);  
c number(2,0);  
begin  
select id\_meci bulk collect into id\_meciuri from meciuri;  
select id\_comentator bulk collect into id\_comentatori from comentatori;  
  
for i in id\_meciuri.first..id\_meciuri.last  
loop  
a := dbms\_random.value(1,id\_comentatori.last);  
b := dbms\_random.value(1,id\_comentatori.last);  
c := dbms\_random.value(1,id\_comentatori.last);  
while a = b  
loop  
b := dbms\_random.value(1,id\_comentatori.last);  
end loop;  
while a = c or b = c  
loop  
c := dbms\_random.value(1,id\_comentatori.last);  
end loop;  
comentariu.id\_meci := id\_meciuri(i);  
comentariu.id\_comentator := id\_comentatori(a);  
insert into comentarii values comentariu;  
comentariu.id\_comentator := id\_comentatori(b);  
insert into comentarii values comentariu;  
comentariu.id\_comentator := id\_comentatori(c);  
insert into comentarii values comentariu;  
end loop;  
  
dbms\_output.put\_line('insert-comentarii OK');  
end insert\_comentarii;  
  
procedure insert\_arbitraje is  
arbitraj arbitraje%rowtype;  
type IdArray is table of number index by pls\_integer;  
id\_meciuri IdArray;  
id\_arbitrii IdArray;  
a number(2,0);  
b number(2,0);  
c number(2,0);  
begin  
select id\_meci bulk collect into id\_meciuri from meciuri;  
select id\_arbitru bulk collect into id\_arbitrii from arbitrii;  
  
for i in id\_meciuri.first..id\_meciuri.last  
loop  
a := dbms\_random.value(1,id\_arbitrii.last);  
b := dbms\_random.value(1,id\_arbitrii.last);  
c := dbms\_random.value(1,id\_arbitrii.last);  
while a = b  
loop  
b := dbms\_random.value(1,id\_arbitrii.last);  
end loop;  
while a = c or b = c  
loop  
c := dbms\_random.value(1,id\_arbitrii.last);  
end loop;  
arbitraj.id\_meci := id\_meciuri(i);  
arbitraj.id\_arbitru := id\_arbitrii(a);  
insert into arbitraje values arbitraj;  
arbitraj.id\_arbitru := id\_arbitrii(b);  
insert into arbitraje values arbitraj;  
arbitraj.id\_arbitru := id\_arbitrii(c);  
insert into arbitraje values arbitraj;  
end loop;  
  
dbms\_output.put\_line('insert-arbitraje OK');  
end insert\_arbitraje;  
  
procedure insert\_distinctii is  
distinctie distinctii%rowtype;  
type IdArray is table of number index by pls\_integer;  
id\_sezoane IdArray;  
id\_jucatori IdArray;  
id\_premii IdArray;  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
select id\_jucator bulk collect into id\_jucatori from jucatori;  
select id\_premiu bulk collect into id\_premii from premii;  
  
for i in id\_sezoane.first..id\_sezoane.last  
loop  
for j in id\_premii.first..id\_premii.last  
loop  
distinctie.id\_sezon := id\_sezoane(i);  
distinctie.id\_premiu := id\_premii(j);  
distinctie.id\_jucator := id\_jucatori(dbms\_random.value(1, id\_jucatori.last));  
  
insert into distinctii values distinctie;  
end loop;  
end loop;  
  
dbms\_output.put\_line('insert\_distinctii OK');  
end insert\_distinctii;  
  
procedure verifica\_inserare is  
cnt number;  
type StringArray is varray(20) of varchar2(20);  
tabele StringArray := StringArray('angajati', 'antrenori', 'arbitrii',   
'arene', 'comentarii', 'comentatori', 'distinctii', 'echipe',   
'echipe\_medicale', 'etape', 'jucatori', 'locatii',   
'meciuri', 'nutritionisti', 'participari', 'premii', 'preparatori\_fizici',   
'sezoane', 'statistici');  
begin  
select count(\*) into cnt from angajati;  
dbms\_output.put\_line('Exista ' || cnt || ' angajati.');  
select count(\*) into cnt from antrenori;  
dbms\_output.put\_line('Exista ' || cnt || ' antrenori.');   
select count(\*) into cnt from arbitraje;  
dbms\_output.put\_line('Exista ' || cnt || ' arbitraje.');  
select count(\*) into cnt from arbitrii;  
dbms\_output.put\_line('Exista ' || cnt || ' arbitrii.');  
select count(\*) into cnt from arene;  
dbms\_output.put\_line('Exista ' || cnt || ' arene.');  
select count(\*) into cnt from comentarii;  
dbms\_output.put\_line('Exista ' || cnt || ' comentarii.');   
select count(\*) into cnt from comentatori;  
dbms\_output.put\_line('Exista ' || cnt || ' comentatori.');   
select count(\*) into cnt from distinctii;  
dbms\_output.put\_line('Exista ' || cnt || ' distinctii.');  
select count(\*) into cnt from echipe;  
dbms\_output.put\_line('Exista ' || cnt || ' echipe.');  
select count(\*) into cnt from echipe\_medicale;  
dbms\_output.put\_line('Exista ' || cnt || ' echipe\_medicale.');  
select count(\*) into cnt from etape;  
dbms\_output.put\_line('Exista ' || cnt || ' etape.');   
select count(\*) into cnt from jucatori;  
dbms\_output.put\_line('Exista ' || cnt || ' jucatori.');  
select count(\*) into cnt from locatii;  
dbms\_output.put\_line('Exista ' || cnt || ' locatii.');  
select count(\*) into cnt from meciuri;  
dbms\_output.put\_line('Exista ' || cnt || ' meciuri.');  
select count(\*) into cnt from nutritionisti;  
dbms\_output.put\_line('Exista ' || cnt || ' nutritionisti.');   
select count(\*) into cnt from participari;  
dbms\_output.put\_line('Exista ' || cnt || ' participari.');  
select count(\*) into cnt from premii;  
dbms\_output.put\_line('Exista ' || cnt || ' premii.');  
select count(\*) into cnt from preparatori\_fizici;  
dbms\_output.put\_line('Exista ' || cnt || ' preparatori\_fizici.');   
select count(\*) into cnt from sezoane;  
dbms\_output.put\_line('Exista ' || cnt || ' sezoane.');  
select count(\*) into cnt from statistici;  
dbms\_output.put\_line('Exista ' || cnt || ' statistici.');  
  
end verifica\_inserare;  
  
procedure efectueaza\_inserare is  
begin  
sterge\_date();  
insert\_sezoane();  
insert\_etape();  
insert\_antrenori();  
insert\_preparatori();  
insert\_nutritionisti();  
insert\_locatii();  
insert\_arene();  
insert\_echipe();  
insert\_jucatori();  
insert\_echipe\_medicale();  
insert\_meciuri();  
insert\_arbitri();  
insert\_comentatori();  
insert\_statistici();  
insert\_premii();  
insert\_participari();  
insert\_comentarii();  
insert\_arbitraje();  
insert\_distinctii();  
verifica\_inserare();  
end efectueaza\_inserare;  
  
procedure ex6\_colectii is  
-- Am ales tabloul imbricat deoarece numarul de sezoane este necunoscut   
-- si nu fac stergeri pe parcursul programului  
type IdSezoane is table of sezoane.id\_sezon%type ;  
id\_sezoane IdSezoane;  
id\_sezon\_curent sezoane.id\_sezon%type;  
  
-- Am ales varray deoarece stiu dinainte numarul de etape (30) dintr-un sezon,  
-- respectiv numarul de meciuri dintr-o etapa (8), si nu fac stergeri in aceasta colectie  
type IdEtape is varray(30) of etape.id\_etapa%type;  
id\_etape IdEtape;  
id\_etapa\_curenta sezoane.id\_sezon%type;  
  
type RecordJucator is record  
(  
id\_jucator jucatori.id\_jucator%type,  
puncte number  
);  
type idJucatoriTop3 is varray(3) of RecordJucator;  
id\_jucatori\_3 IdJucatoriTop3;  
rec RecordJucator;  
  
-- Am ales tablou indexat deoarece indicii vor fi anumite id-uri ale jucatorilor, nu neaparat dense.  
type TabelPuncteJucatori is table of number(4) index by pls\_integer;  
puncte\_jucator TabelPuncteJucatori := TabelPuncteJucatori();  
  
nr\_etape number; -- restriciile spun ca pentru fiecare sezon exista 30 etape  
  
maxim number;  
id\_mvp jucatori.id\_jucator%type;  
jucator jucatori%rowtype;  
  
begin  
select id\_sezon bulk collect into id\_sezoane from sezoane;  
  
for it\_sezon in id\_sezoane.first..id\_sezoane.last loop  
id\_sezon\_curent := id\_sezoane(it\_sezon);  
select count(id\_etapa) collect into nr\_etape   
from etape  
where id\_sezon = id\_sezon\_curent;  
  
if nr\_etape != 30  
then   
dbms\_output.put\_line ('In sezonul ' || id\_sezon\_curent ||   
' nu sunt 30 de etape ( ' || nr\_etape || ' )');  
continue;  
end if;  
  
select id\_etapa bulk collect into id\_etape   
from etape  
where id\_sezon = id\_sezon\_curent;  
  
puncte\_jucator := TabelPuncteJucatori();  
maxim := 0;  
for it\_etapa in id\_etape.first..id\_etape.last loop  
id\_etapa\_curenta := id\_etape(it\_etapa);  
with jucatori\_si\_puncte as  
(select j.id\_jucator, 2 \* aruncari\_2pct\_marcate + 3 \* aruncari\_3pct\_marcate + aruncari\_libere\_marcate  
from jucatori j  
join meciuri m on m.id\_etapa = id\_etapa\_curenta and  
((m.id\_echipa\_gazda = j.id\_echipa and scor\_gazda > scor\_oaspete ) or  
(m.id\_echipa\_oaspete = j.id\_echipa and scor\_gazda < scor\_oaspete ))  
join statistici s on s.id\_meci = m.id\_meci and s.id\_jucator = j.id\_jucator  
order by 2 desc)  
select \* bulk collect into id\_jucatori\_3  
from jucatori\_si\_puncte  
where rownum <= 3;  
  
for it in id\_jucatori\_3.first..id\_jucatori\_3.last loop  
rec := id\_jucatori\_3(it);  
if puncte\_jucator.exists(rec.id\_jucator) = false  
then puncte\_jucator(rec.id\_jucator) := 0;  
end if;  
  
puncte\_jucator(rec.id\_jucator) := puncte\_jucator(rec.id\_jucator) + rec.puncte;  
if maxim < puncte\_jucator(rec.id\_jucator)  
then  
maxim := puncte\_jucator(rec.id\_jucator);  
id\_mvp := rec.id\_jucator;  
end if;  
end loop;  
end loop;  
  
select \* into jucator  
from jucatori  
where id\_jucator = id\_mvp;  
  
dbms\_output.put\_line(jucator.nume || ' ' || jucator.prenume || ' id: ' || id\_mvp || ' puncte: ' || puncte\_jucator(id\_mvp));  
  
end loop;  
  
end ex6\_colectii;  
  
procedure ex7\_cursoare is  
cursor c\_id\_arbitri (idm meciuri.id\_meci%type) is  
select id\_arbitru from arbitraje where arbitraje.id\_meci = idm;  
  
  
type IdArbitri is table of arbitrii.id\_arbitru%type;  
id\_arbitri IdArbitri;  
arbitru arbitrii%rowtype;  
  
cursor meciuriVerificate is  
select \*  
from meciuri  
where abs(scor\_gazda - scor\_oaspete) =   
(select max(abs(scor\_gazda - scor\_oaspete))  
from meciuri);  
  
cursor arbitriVerificati (idm meciuri.id\_meci%type) is  
select \*  
from arbitrii  
where id\_arbitru in (select id\_arbitru from arbitraje where arbitraje.id\_meci = idm);  
  
cursor jucatoriMeci (  
id\_gazda echipe.id\_echipa%type,  
id\_oaspete echipe.id\_echipa%type,   
nume\_arbitru arbitrii.nume%type) is  
select id\_jucator , nume, prenume, id\_echipa  
from jucatori   
where (id\_echipa = id\_gazda or id\_echipa = id\_oaspete)  
and jucatori.nume = nume\_arbitru;  
  
mesaj varchar2(50);  
id\_echipa echipe.id\_Echipa%type;  
begin  
  
-- CICLU CURSOR  
for meci in meciuriVerificate loop  
-- dbms\_output.put\_line('meci: ' || meci.id\_meci);  
  
-- CURSOR CLASIC PARAMETRIZAT  
open c\_id\_arbitri(meci.id\_meci);  
fetch c\_id\_arbitri bulk collect into id\_arbitri;  
close c\_id\_arbitri;  
  
for i in id\_arbitri.first..id\_arbitri.last loop  
select \* into arbitru  
from arbitrii  
where id\_arbitru = id\_arbitri(i);  
-- dbms\_output.put\_line('arbitru: ' || arbitru.id\_arbitru);  
-- CICLU CURSOR PARAMETRIZAT  
for jucator in jucatoriMeci(meci.id\_echipa\_gazda, meci.id\_echipa\_oaspete, arbitru.nume) loop  
  
if meci.scor\_gazda > meci.scor\_oaspete  
then id\_echipa := meci.id\_echipa\_gazda;  
else id\_echipa := meci.id\_echipa\_oaspete;  
end if;  
  
if jucator.id\_echipa = id\_echipa  
then mesaj := ' (Victorie)';  
else mesaj := ' (Infrangere)';  
end if;  
  
dbms\_output.put\_line(meci.id\_meci || ': ' ||  
arbitru.id\_arbitru || ' ' || arbitru.nume || ' ' || arbitru.prenume || ' - ' ||  
jucator.id\_jucator || ' ' || jucator.nume || ' ' || jucator.prenume || ' - ' ||   
meci.scor\_gazda || '-' || meci.scor\_oaspete || mesaj);  
  
end loop;  
end loop;  
end loop;  
  
end;  
  
function ex8\_exceptii  
(p\_prenume in varchar2,  
p\_salariu\_minim in number)  
return number  
is  
ex\_no\_first\_name exception;  
ex\_no\_salary exception;  
type RecordEchipa is Record (  
-- id\_echipa echipe.id\_echipa%type,  
nume varchar2(50),  
suma\_salarii number  
);  
type ListaEchipe is table of RecordEchipa;  
lista\_echipe ListaEchipe;  
  
aux number := 0;  
  
begin  
select count(\*) collect into aux  
from (select id\_jucator from jucatori where lower(prenume) like lower(p\_prenume)  
union select id\_angajat from angajati where lower(prenume) like lower(p\_prenume));  
  
if aux = 0 then  
raise ex\_no\_first\_name;  
end if;  
  
with persoane as (  
select e.id\_echipa id\_echipa, salariu   
from angajati a  
join echipe e on e.id\_antrenor = a.id\_angajat or e.id\_preparator = a.id\_angajat   
or e.id\_nutritionist = a.id\_angajat  
where lower(a.prenume) like lower(p\_prenume) and a.salariu >= p\_salariu\_minim  
union  
select id\_echipa, salariu from jucatori  
where lower(prenume) like lower(p\_prenume) and salariu >= p\_salariu\_minim  
)  
select \* bulk collect into lista\_echipe  
from (  
select e.nume, sum(p.salariu)   
from persoane p  
join echipe e on e.id\_echipa = p.id\_echipa  
group by e.nume  
)  
order by 2 desc;  
  
  
if lista\_echipe.count = 0 then  
raise ex\_no\_salary;  
end if;  
  
aux := 0;  
for i in lista\_Echipe.first..lista\_echipe.last loop  
dbms\_output.put\_line(lista\_echipe(i).nume || ' - ' || lista\_echipe(i).suma\_salarii);  
aux := aux + lista\_echipe(i).suma\_salarii;  
end loop;  
  
return aux;  
  
exception  
when ex\_no\_first\_name then  
dbms\_output.put\_line('Exceptie: nu exista prenumele ' || p\_prenume);  
return 0;  
when ex\_no\_salary then  
dbms\_output.put\_line('Exceptie: nu exista salarii mai mari decat ' || p\_salariu\_minim);  
return 0;  
when others then  
dbms\_output.put\_line('Alta Exceptie');  
return 0;  
end ex8\_exceptii;  
  
procedure ex9\_exceptii  
(p\_nume in angajati.nume%type  
)  
is  
ex\_no\_player exception;  
type RecAngajat is record (  
nume angajati.nume%type,   
prenume angajati.prenume%type,  
nume\_echipa echipe.nume%type,   
post varchar2(20)  
);  
angajat RecAngajat;  
type RecEchipa is record (  
id echipe.id\_echipa%type,  
nume echipe.nume%type);  
echipa RecEchipa;  
  
type RecJucator is record(  
id jucatori.id\_jucator%type,  
nume jucatori.nume%type,   
prenume jucatori.prenume%type,   
proc\_3pct number(5, 2),   
proc\_faulturi number(5, 2),  
avg\_pase number(5, 2)  
);  
type ListaJucatori is table of RecJucator;  
lista\_jucatori ListaJucatori;  
  
begin  
  
select a.nume, a.prenume, e.nume, (case  
when a.id\_angajat = e.id\_antrenor then 'antrenor'  
when a.id\_angajat = e.id\_preparator then 'preparator fizic'  
when a.id\_angajat = e.id\_nutritionist then 'nutritionist'  
end) post  
into angajat  
from angajati a, echipe e  
where lower(a.nume) = lower(p\_nume)  
and id\_angajat in (e.id\_antrenor, e.id\_preparator, e.id\_nutritionist);  
  
dbms\_output.put\_line(angajat.nume || ' ' || angajat.prenume || ' lucreaza pentru echipa ' ||   
angajat.nume\_echipa || ' pe postul de ' || angajat.post);  
  
with info\_echipa as (  
select id\_echipa, e.nume  
from echipe e   
where (p\_nume) in  
(select nume  
from angajati  
where lower(nume) = lower(p\_nume)  
and id\_angajat in (e.id\_antrenor, e.id\_preparator, e.id\_nutritionist))  
)  
select j.id\_jucator, j.nume, j.prenume, 100 \* sum(s.aruncari\_3pct\_marcate) / sum(s.aruncari\_3pct),  
100 \* sum(case when s.faulturi > 0 then 1 else 0 end) / count(\*), avg(s.pase\_decisive)  
bulk collect into lista\_jucatori  
from jucatori j  
join statistici s on s.id\_jucator = j.id\_jucator  
where j.id\_jucator in (select distinct id\_jucator from distinctii)  
and j.id\_echipa = (select id\_echipa from info\_echipa)  
group by (j.id\_jucator, j.nume, j.prenume);  
  
if lista\_jucatori.count = 0 then  
raise ex\_no\_player;  
end if;  
  
for i in lista\_jucatori.first..lista\_jucatori.last loop  
dbms\_output.put\_line(lista\_jucatori(i).nume || ' ' || lista\_jucatori(i).prenume || ' - procentaj 3pct: '   
|| lista\_jucatori(i).proc\_3pct || '% - procentaj meciuri cu fault: ' || lista\_jucatori(i).proc\_faulturi ||   
'% - medie pase decisive: ' || lista\_jucatori(i).avg\_pase);  
end loop;  
  
exception  
when no\_data\_found then  
dbms\_output.put\_line('Nu exista niciun angajat cu numele ' || p\_nume);  
when too\_many\_rows then  
dbms\_output.put\_line('Exista mai multi angajati cu numele ' || p\_nume);  
when ex\_no\_player then  
dbms\_output.put\_line('Nu exista niciun jucator care sa fi primit vreo distinctie in echipa ' || angajat.nume\_echipa);  
end ex9\_exceptii;  
  
  
  
end pachet;  
/

## Exemplu Execuție

-- Din cauza faptului ca majoritatea datelor din baza de date sunt generate aleator,   
-- este posibil ca unele apeluri sa obtina rezultate diferite decat in capturile de ecran  
  
begin  
pachet.efectueaza\_inserare();  
pachet.ex6\_colectii();  
pachet.ex7\_cursoare();  
  
-- EX 8  
dbms\_output.put\_line(pachet.ex8\_exceptii('isaac', 50000));  
dbms\_output.put\_line(pachet.ex8\_exceptii('prenume', 1000));  
dbms\_output.put\_line(pachet.ex8\_exceptii('isaac', 100000));  
  
-- EX 9  
dbms\_output.put\_line('Apel cu numele: Johnson');  
pachet.ex9\_exceptii('Johnson');  
dbms\_output.put\_line('Apel cu numele: Carter');  
pachet.ex9\_exceptii('Carter');  
dbms\_output.put\_line('Apel cu numele: Clark');  
pachet.ex9\_exceptii('Clark');  
dbms\_output.put\_line('Apel cu numele: John');  
pachet.ex9\_exceptii('John');  
end;  
/

## Captură de Ecran

# 

# Opțional - Definire Pachet 2

Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).

## Cerinta

La finalul unui sezon, asociatia de baschet doreste sa verifice daca toate datele corespunzatoare meciurilor din sezonul curent se afla in baza de date (toate etapele exista, iar fiecare echipa a jucat 30 de meciuri). In caz afirmativ, trebuie sa acorde cele 5 premii:

* "Most Valuable Player (MVP)" - pentru jucatorul cu cele mai multe puncte inscrise
* "Team Player of the Year" - pentru jucatorul cu cele mai multe pase decisive
* "Defensive Player of the Year" - pentru jucatorul cu cele mai multe recuperari
* "Sportsmanship Award" - pentru jucatorul cu cele mai putine faulturi
* "Best Distance Shooter" - pentru jucatorul cu cele mai multe cosuri de 3 puncte inscrise

In caz de egalitate, comisia trebuie sa fie instiintata printr-un mesaj pentru a decide cine este premiat (nu se va adauga premiul in baza de date).

## Rezolvare

select \* from premii;  
  
create or replace package acordare\_premii as  
type StatisticiJucator is record (puncte number, pase number, recuperari number, faulturi number, cosuri3pct number);  
type TabelStatistici is table of StatisticiJucator index by pls\_integer;  
statisticiSezon TabelStatistici := TabelStatistici();  
  
ultimul\_sezon\_calculat number := -1; -- Id-ul sezonului pentru care s-au calculat statisticile ultima data.  
  
function numeSiEchipaJucator(idj in number) return varchar2;  
  
function verifica\_sezon(an\_incepere in number) return boolean;  
  
procedure acorda\_premii(an\_incepere in number);  
  
function obtine\_statistici\_jucator(idj in number, ids in number) return StatisticiJucator;  
  
procedure calculeaza\_statistici(ids number);  
  
  
end acordare\_premii;  
/  
  
create or replace package body acordare\_premii as  
function verifica\_sezon(an\_incepere in number)  
return boolean as  
ids number := 0;  
nr\_etape number := 0;  
nr\_meciuri number := 0;  
type TabelIdEchipe is table of echipe.id\_echipa%type;  
id\_echipe TabelIdEchipe;  
begin  
select id\_sezon into ids  
from sezoane s  
where to\_char(s.data\_incepere, 'yyyy') = to\_char(an\_incepere);  
  
if ids = 0 then  
dbms\_output.put\_line('Sezonul nu exista');  
return false;  
end if;  
  
select count(\*) into nr\_etape  
from etape e  
where e.id\_sezon = ids;  
  
if nr\_etape != 30 then  
dbms\_output.put\_line('Sezonul nu contine ' || nr\_etape || ' etape in loc de 30');  
return false;  
end if;  
  
select e.id\_echipa bulk collect into id\_echipe  
from echipe e  
where exists (select \* from participari p where p.id\_sezon = ids and p.id\_echipa = e.id\_echipa);  
  
if id\_echipe.count() != 16 then  
dbms\_output.put\_line('Sezonul nu contine 16 echipe');  
return false;  
end if;  
  
for i in id\_echipe.first..id\_echipe.last loop  
select count(\*) collect into nr\_meciuri  
from meciuri m   
where (m.id\_echipa\_oaspete = id\_echipe(i) or m.id\_echipa\_gazda = id\_echipe(i))  
and ids = (select e.id\_sezon from etape e join sezoane s on s.id\_sezon = e.id\_sezon where e.id\_etapa = m.id\_etapa);  
  
if nr\_meciuri != 30 then  
dbms\_output.put\_line('Echipa cu id ' || id\_echipe(i) || 'a jucat ' || nr\_meciuri || ' in loc de 30');  
return false;  
end if;  
end loop;  
  
return true;  
exception  
when no\_data\_found then dbms\_output.put\_line('EX: NO DATA FOUND');  
return false;  
end verifica\_sezon;  
  
procedure acorda\_premii(an\_incepere in number)  
as  
aux number := 0;  
maxim StatisticiJucator := StatisticiJucator(0, 0, 0, 100000, 0);  
id\_castigator StatisticiJucator := StatisticiJucator();  
cnt StatisticiJucator := StatisticiJucator(0, 0, 0, 0, 0);  
ids number := 0;  
begin  
dbms\_output.put\_line('Se incearca acordarea premiilor');  
if verifica\_sezon(an\_incepere) = false then  
dbms\_output.put\_line('Nu au fost acordate premiile');  
return;  
end if;  
  
select id\_sezon into ids  
from sezoane s  
where to\_char(s.data\_incepere, 'yyyy') = to\_char(an\_incepere);  
  
if ids != ultimul\_sezon\_calculat then  
calculeaza\_statistici(ids);  
end if;  
  
  
for i in statisticiSezon.first..statisticiSezon.last loop  
if maxim.puncte < statisticiSezon(i).puncte then   
maxim.puncte := statisticiSezon(i).puncte;  
cnt.puncte := 1;  
id\_castigator.puncte := i;  
elsif maxim.puncte = statisticiSezon(i).puncte then   
cnt.puncte := cnt.puncte + 1;  
end if;  
  
if maxim.pase < statisticiSezon(i).pase then   
maxim.pase := statisticiSezon(i).pase;   
cnt.pase := 1;  
id\_castigator.pase := i;  
elsif maxim.pase = statisticiSezon(i).pase then   
cnt.pase := cnt.pase + 1;  
end if;  
  
if maxim.recuperari < statisticiSezon(i).recuperari then   
maxim.recuperari := statisticiSezon(i).recuperari;   
cnt.recuperari := 1;  
id\_castigator.recuperari := i;  
elsif maxim.recuperari = statisticiSezon(i).recuperari then   
cnt.recuperari := cnt.recuperari + 1;  
end if;  
  
if maxim.faulturi > statisticiSezon(i).faulturi then   
maxim.faulturi := statisticiSezon(i).faulturi;   
cnt.faulturi := 1;  
id\_castigator.faulturi := i;  
elsif maxim.faulturi = statisticiSezon(i).faulturi then   
cnt.faulturi := cnt.faulturi + 1;  
end if;  
  
if maxim.cosuri3pct < statisticiSezon(i).cosuri3pct then   
maxim.cosuri3pct := statisticiSezon(i).cosuri3pct;   
cnt.cosuri3pct := 1;  
id\_castigator.cosuri3pct := i;  
elsif maxim.cosuri3pct = statisticiSezon(i).cosuri3pct then   
cnt.cosuri3pct := cnt.cosuri3pct + 1;  
end if;  
  
end loop;  
  
delete from distinctii where id\_sezon = ids;  
  
if cnt.puncte = 1 then  
dbms\_output.put\_line(numeSiEchipaJucator(id\_castigator.puncte) || ' --- ' || 'Most Valuable Player (MVP)');  
select id\_premiu into aux from premii where denumire like 'Most Valuable Player (MVP)';  
insert into distinctii values (ids, id\_castigator.puncte, aux);  
else  
dbms\_output.put\_line('Sunt mai multi jucatori care pot castiga ' || 'Most Valuable Player (MVP)' );  
end if;  
  
  
if cnt.pase = 1 then  
dbms\_output.put\_line(numeSiEchipaJucator(id\_castigator.pase) || ' --- ' || 'Team Player of the Year');  
select id\_premiu into aux from premii where denumire like 'Team Player of the Year';  
insert into distinctii values (ids, id\_castigator.pase, aux);  
else  
dbms\_output.put\_line('Sunt mai multi jucatori care pot castiga ' || 'Team Player of the Year');  
end if;   
  
if cnt.recuperari = 1 then  
dbms\_output.put\_line(numeSiEchipaJucator(id\_castigator.recuperari) || ' --- ' || 'Defensive Player of the Year');  
select id\_premiu into aux from premii where denumire like 'Defensive Player of the Year';  
insert into distinctii values (ids, id\_castigator.recuperari, aux);  
else  
dbms\_output.put\_line('Sunt mai multi jucatori care pot castiga ' || 'Defensive Player of the Year');  
end if;  
  
if cnt.faulturi = 1 then  
dbms\_output.put\_line(numeSiEchipaJucator(id\_castigator.faulturi) || ' --- ' || 'Sportsmanship Award');  
select id\_premiu into aux from premii where denumire like 'Sportsmanship Award';  
insert into distinctii values (ids, id\_castigator.faulturi, aux);  
else  
dbms\_output.put\_line('Sunt mai multi jucatori care pot castiga ' || 'Sportsmanship Award');  
end if;  
  
if cnt.cosuri3pct = 1 then  
dbms\_output.put\_line(numeSiEchipaJucator(id\_castigator.cosuri3pct) || ' --- ' || 'Best Distance Shooter');  
select id\_premiu into aux from premii where denumire like 'Best Distance Shooter';  
insert into distinctii values (ids, id\_castigator.cosuri3pct, aux);  
else  
dbms\_output.put\_line('Sunt mai multi jucatori care pot castiga ' || 'Best Distance Shooter');  
end if;  
-- dbms\_output.put\_line(maxim.puncte || ' ' || maxim.pase || ' ' || maxim.recuperari || ' ' || maxim.faulturi || ' ' || maxim.cosuri3pct);  
-- dbms\_output.put\_line(cnt.puncte || ' ' || cnt.pase || ' ' || cnt.recuperari || ' ' || cnt.faulturi || ' ' || cnt.cosuri3pct);  
  
dbms\_output.put\_line('S-au acordat premiile cu succes');  
end acorda\_premii;  
  
function numeSiEchipaJucator(idj in number) return varchar2  
is  
str varchar2(50) := '';  
begin  
select j.nume || ' ' || j.prenume || ' - ' || e.nume into str  
from jucatori j  
join echipe e on e.id\_echipa = j.id\_echipa  
where j.id\_jucator = idj;  
  
return str;  
exception  
when others then dbms\_output.put\_line('Eroare la obtinerea numelui si echipei unui jucator');  
return '';  
end numeSiEchipaJucator;  
  
function obtine\_statistici\_jucator(idj in number, ids in number) return StatisticiJucator  
as  
statistica StatisticiJucator;  
begin  
select 2 \* sum(s.aruncari\_2pct\_marcate) + 3 \* sum(s.aruncari\_3pct\_marcate) + sum(s.aruncari\_libere\_marcate),   
sum(s.pase\_decisive),   
sum(s.recuperari),   
sum(s.faulturi),   
sum(s.aruncari\_3pct\_marcate)  
into statistica  
from statistici s   
where s.id\_jucator = idj and ids = (select e.id\_sezon from etape e join meciuri m on m.id\_etapa = e.id\_etapa where s.id\_meci = m.id\_meci);  
  
return statistica;  
end obtine\_statistici\_jucator;  
  
procedure calculeaza\_statistici(ids number)  
is  
type TabelIdJucatori is table of jucatori.id\_jucator%type;  
id\_jucatori TabelIdJucatori;  
idj number;  
begin  
dbms\_output.put\_line('Se calculeaza statisticile');  
  
select j.id\_jucator bulk collect into id\_jucatori  
from jucatori j  
join echipe e on e.id\_echipa = j.id\_echipa  
join participari p on p.id\_echipa = e.id\_echipa and p.id\_sezon = ids;  
  
statisticiSezon := TabelStatistici();  
ultimul\_sezon\_calculat := ids;  
  
for i in id\_jucatori.first..id\_jucatori.last loop  
idj := id\_jucatori(i);  
statisticiSezon(idj) := obtine\_statistici\_jucator(idj, ids);  
  
-- dbms\_output.put\_line(statistici(idj).puncte);  
end loop;  
  
  
exception  
when others then dbms\_output.put\_line('Exceptie: Nu au putut fi calculate statisticile');  
end calculeaza\_statistici;  
  
end acordare\_premii;  
/  
  
select \* from distinctii;  
  
begin   
acordare\_premii.acorda\_premii(2022);  
-- acordare\_premii.calculeaza\_statistici(1001);  
  
-- dbms\_output.put\_line(acordare\_premii.ultimul\_sezon\_calculat);S  
end;  
/

## Captura de Ecran

