

Proiect

Sisteme de Gestiune a Bazelor de Date pentru o platforma de jocuri video

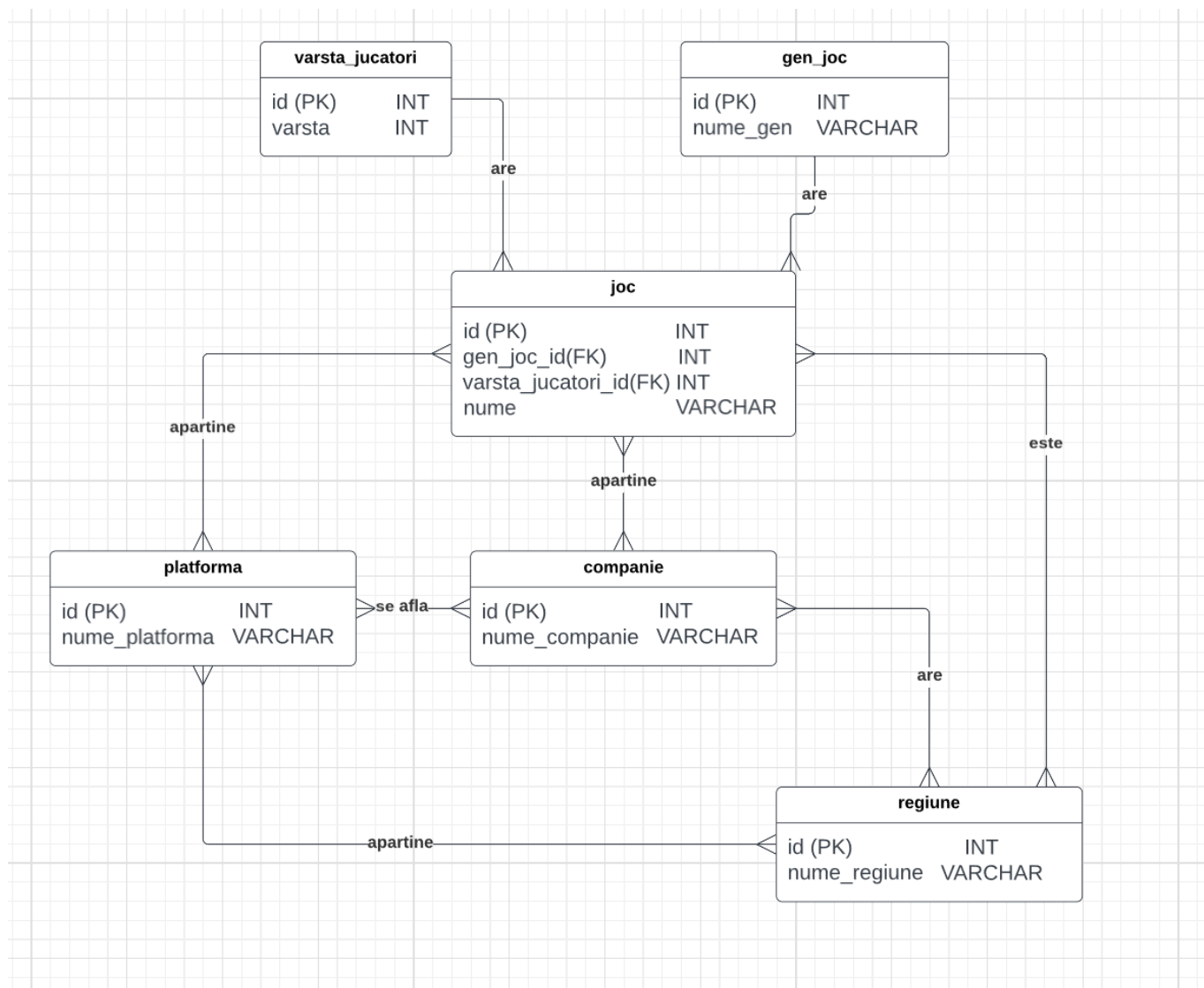
Cuprins

1. Prezentați pe scurt baza de date (utilizarea ei)
2. Realizați diagrama entitate-relație (ERD)
3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.
4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).
5. Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).
6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze două tipuri diferite de colecții studiate. Apelați subprogramul.
7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelați subprogramul.
8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.
9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO DATA FOUND și TOO MANY ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.
10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.
11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.
12. Definiți un trigger de tip LDD. Declanșați trigger-ul.
13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.
14. 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).

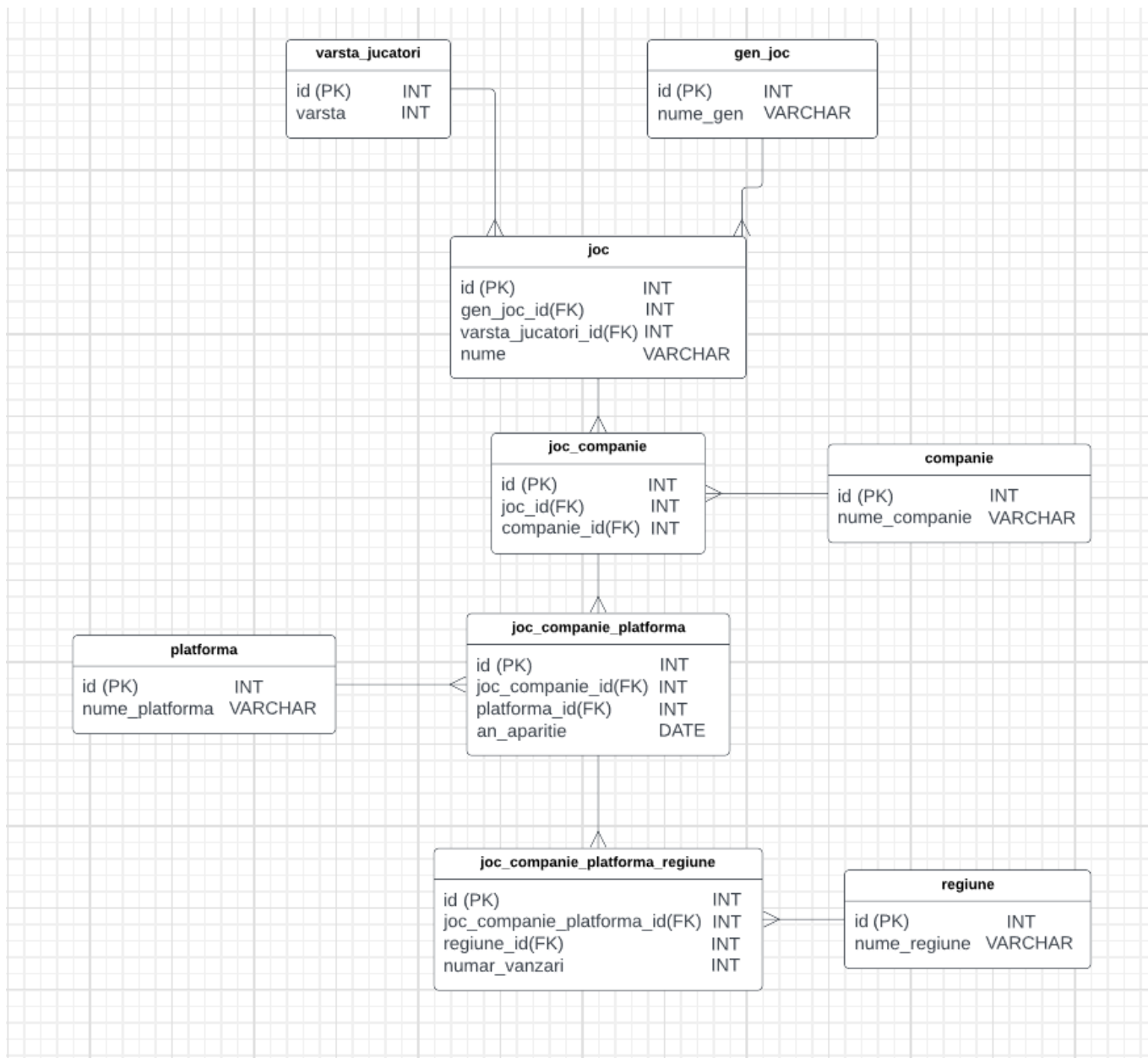
1. Prezențați pe scurt baza de date (utilizarea ei)

Modelul acestei baze de date va gestiona o platforma de jocuri video. Tabela **varsta_jucatori** va reține o lista cu vârste pentru care sunt facute jocurile. Tabela **gen_joc** va reține o lista cu toate genurile de jocuri create de către companie, cum ar fi MMORPG, MOBA, etc. Aceasta tabela ne va ajuta atunci când dorim să vedem ce gen are un anumit joc sau să vedem de exemplu care gen de joc a făcut cele mai multe încasări în funcție de regiune. Tabela **joc** va reține o lista cu toate jocurile care au fost făcute de către companie. Aceasta va conține titlul jocului și 2 chei externe, **gen_joc_id** care va face legătura cu tabela **gen_joc** și **varsta_jucatori_id** care va face legătura cu tabela **varsta_jucatori**. Tabela **companie_joc** va reține numele companiei care a publicat un anumit joc. Tabela **platforma** va reține o lista cu diferite platforme pentru care au fost făcute jocurile. Tabela **regiune** va reține o lista cu regiuni pentru care au fost făcute jocurile. Tabela **joc_companie** va reprezenta tabela asociativă dintre tabelele **joc** și **companie_joc**, deoarece avem cardinalitate M-M, pentru că un joc poate fi făcut de mai multe companii și o companie poate avea mai multe jocuri. Tabela **joc_companie_platforma** va reprezenta tabela asociativă între tabelele **joc**, **companie_joc** și **platforma**, deoarece avem cardinalitate M-M între cele 3 tabele. În acest caz o să ne folosim de tabela **joc_companie** pentru a face legătura dintre tabelele **joc**, **companie_joc** și cu **platforma**. Tabela **vanzari_regiune** va reprezenta tabela asociativă între tabelele **joc**, **companie_joc**, **platforma** și **regiune**, deoarece este cardinalitate M-M între ele. În acest caz o să ne folosim de tabela **joc_platforma** pentru a lega tabelele **joc**, **companie_joc** și **platforma** cu tabela **regiune**.

2. Realizati diagrama entitate-relatie (ERD)



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



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

a. Creare și populare tabel varsta_jucatori

```

create table varsta_jucatori(
    id_varsta number(4),
    varsta number(3) NOT NULL
);

```

```

alter table varsta_jucatori
add constraint pk_varsta_jucatori PRIMARY KEY(id_varsta);

```

```
insert into varsta_jucatori(id_varsta, varsta)
values (1, 5);
```

```
insert into varsta_jucatori(id_varsta, varsta)
values (2, 8);
```

```
insert into varsta_jucatori(id_varsta, varsta)
values (3, 15);
```

```
insert into varsta_jucatori(id_varsta, varsta)
values (4, 18);
```

```
insert into varsta_jucatori(id_varsta, varsta)
values (5, 35);
```

```
select * from varsta_jucatori;
```

```

create table varsta_jucatori(
    id_varsta number(4),
    varsta number(3) NOT NULL
);

alter table varsta_jucatori
add constraint pk_varsta_jucatori PRIMARY KEY(id_varsta);

insert into varsta_jucatori(id_varsta, varsta)
values (1, 5);

insert into varsta_jucatori(id_varsta, varsta)
values (2, 8);

insert into varsta_jucatori(id_varsta, varsta)
values (3, 15);

insert into varsta_jucatori(id_varsta, varsta)
values (4, 18);

insert into varsta_jucatori(id_varsta, varsta)
values (5, 35);

```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.003 seconds

	ID_VARSTA	VARSTA
1	1	5
2	2	8
3	3	15
4	4	18
5	5	35

- b. Creare si populare tabel gen_joc

```

create table gen_joc(
    id_gen number(4),
    nume_gen varchar2(50) NOT NULL
);

```

```

alter table gen_joc
add constraint pk_gen_joc PRIMARY KEY(id_gen);

```

```

insert into gen_joc(id_gen, nume_gen)
values (1, 'Real-time strategy');

```

```

insert into gen_joc(id_gen, nume_gen)
values (2, 'Shooter');

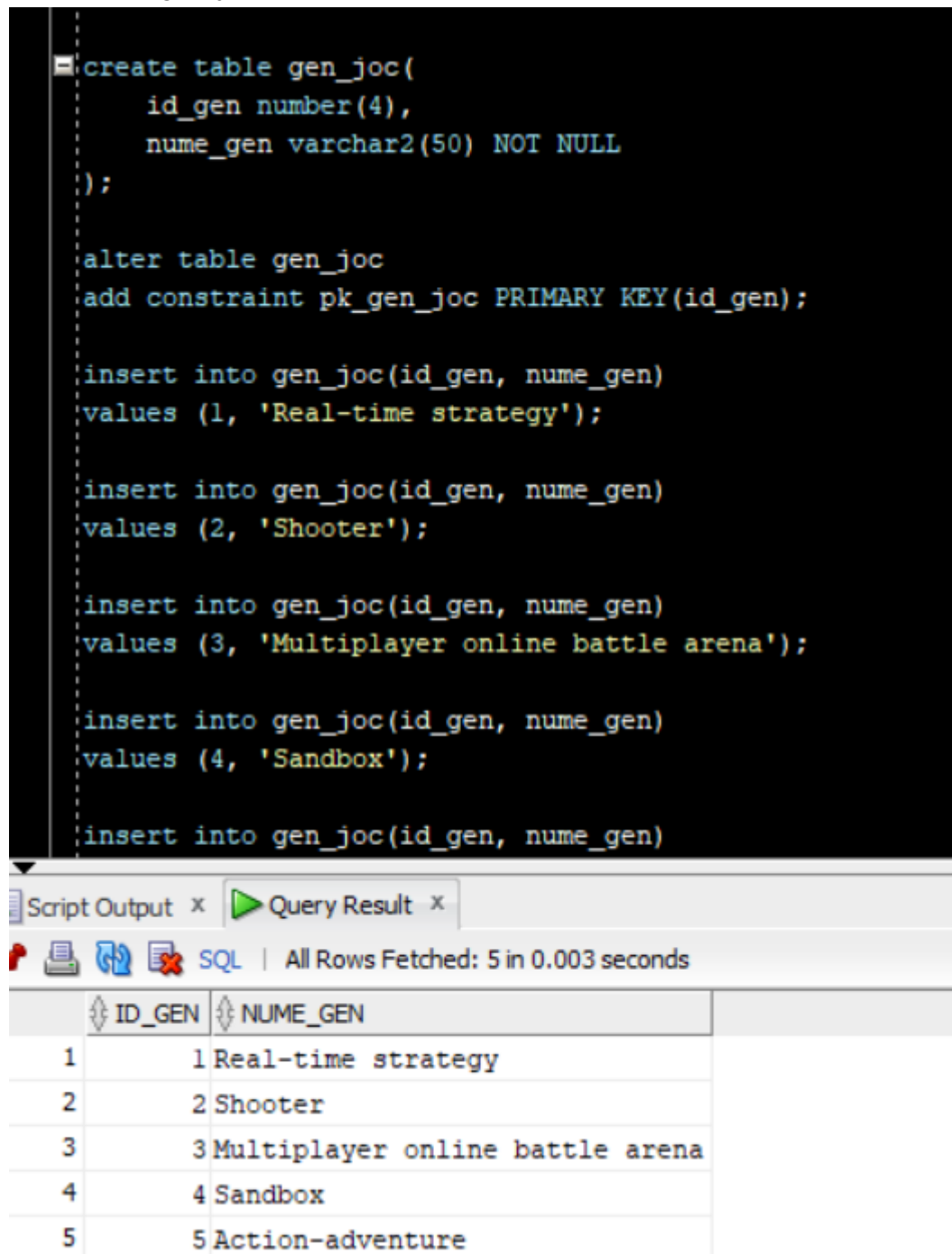
```

```
insert into gen_joc(id_gen, nume_gen)
values (3, 'Multiplayer online battle arena');
```

```
insert into gen_joc(id_gen, nume_gen)
values (4, 'Sandbox');
```

```
insert into gen_joc(id_gen, nume_gen)
values (5, 'Action-adventure');
```

```
select * from gen_joc;
```



```
create table gen_joc(
    id_gen number(4),
    nume_gen varchar2(50) NOT NULL
);

alter table gen_joc
add constraint pk_gen_joc PRIMARY KEY(id_gen);

insert into gen_joc(id_gen, nume_gen)
values (1, 'Real-time strategy');

insert into gen_joc(id_gen, nume_gen)
values (2, 'Shooter');

insert into gen_joc(id_gen, nume_gen)
values (3, 'Multiplayer online battle arena');

insert into gen_joc(id_gen, nume_gen)
values (4, 'Sandbox');

insert into gen_joc(id_gen, nume_gen)
```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.003 seconds

ID_GEN	NUME_GEN
1	Real-time strategy
2	Shooter
3	Multiplayer online battle arena
4	Sandbox
5	Action-adventure

c. Creare si populare tabel joc

```
create table joc(
    id_joc number(4) constraint pk_joc primary key,
```



```

        id_gen_joc number(4),
        id_varsta_jucatori number(4),
        nume varchar2(50),
        descriere varchar2(500)
    )

```

```

alter table joc
add constraint fk_gen_joc foreign key (id_gen_joc) references gen_joc(id_gen);

```

```

alter table joc
add constraint fk_varsta_jucatori foreign key (id_varsta_jucatori) references
varsta_jucatori(id_varsta);

```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(1, 2, 4, 'Counter Strike Global Offensive', 'The game pits two teams, Terrorists and
Counter-Terrorists, against each other in different objective-based game modes.');
```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(2, 4, 4, 'Minecraft', 'Minecraft is a video game in which players create and break
apart various kinds of blocks in three-dimensional worlds');
```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(3, 2, 2, 'Terraria', 'The game features exploration, crafting, building, painting, and
combat with a variety of creatures in a procedurally generated 2D world.');
```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(4, 1, 5, 'Warcraft', 'The game allows players to create avatar-style characters and
explore a sprawling universe while interacting with nonreal players—called nonplayer
characters and other real-world players');
```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(5, 1, 4, 'Age of Empires', 'The game focus on historical events throughout time.');
```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(6, 4, 4, 'Dota 2', 'Dota 2 is a multiplayer online battle arena video game in which
two teams of five players compete to destroy a large structure defended by the opposing
team known as the "Ancient" whilst defending their own');
```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(7, 3, 4, 'League of Legends', ' Players work with their team to break the enemy
Nexus before the enemy team breaks theirs.');
```

```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(8, 2, 4, 'Smite', 'In Smite, players control a god, goddess or other mythological
figure and take part in team-based combat, using their abilities and tactics against other
player-controlled gods and non-player-controlled minions.');
```

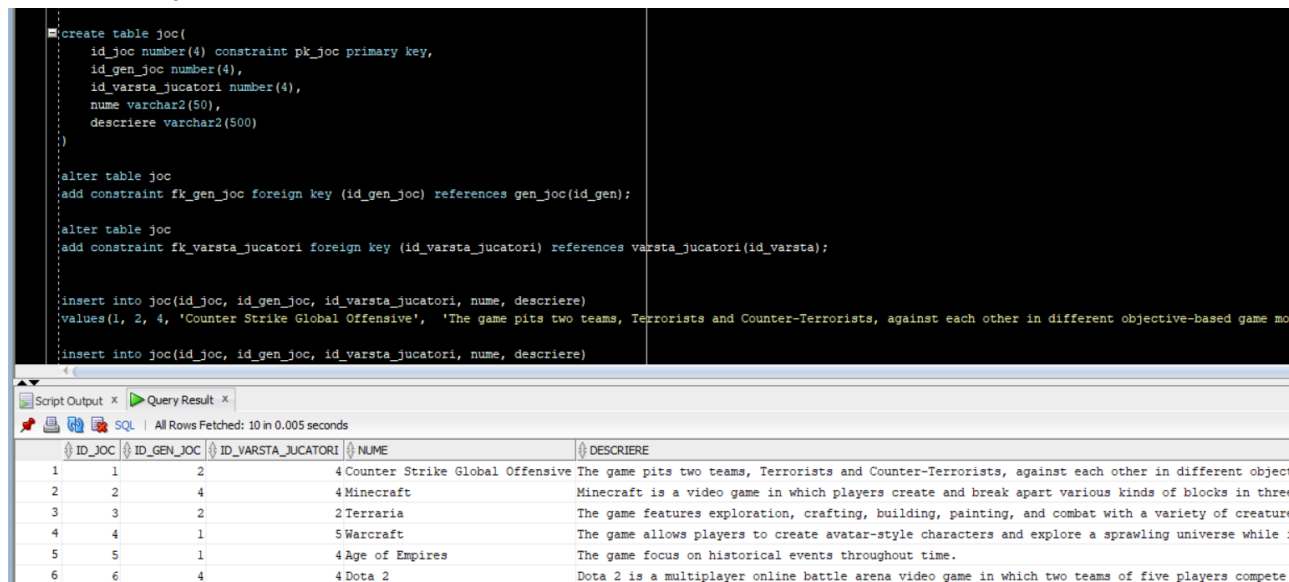
```

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(9, 5, 3, 'Assassin's Creed ', 'Assassin's Creed is an action-adventure game set in
an open-world environment and played from a third-person perspective in which the
```

player primarily assumes the role of Altair, as experienced by protagonist Desmond Miles.');

```
insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(10, 5, 4, 'Sekiro: Shadows Die Twice', 'In Sekiro™: Shadows Die Twice you are
the `one-armed wolf`, a disgraced and disfigured warrior rescued from the brink of death.
Bound to protect a young lord who is the descendant of an ancient bloodline, you become
the target of many vicious enemies, including the dangerous Ashina clan.');
```

```
select * from joc;
```



```
create table joc(
    id_joc number(4) constraint pk_joc primary key,
    id_gen_joc number(4),
    id_varsta_jucatori number(4),
    nume varchar2(50),
    descriere varchar2(500)
);

alter table joc
add constraint fk_gen_joc foreign key (id_gen_joc) references gen_joc(id_gen);

alter table joc
add constraint fk_varsta_jucatori foreign key (id_varsta_jucatori) references varsta_jucatori(id_varsta);

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(1, 2, 4, 'Counter Strike Global Offensive', 'The game pits two teams, Terrorists and Counter-Terrorists, against each other in different objective-based game modes');

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(2, 2, 4, 'Minecraft', 'Minecraft is a video game in which players create and break apart various kinds of blocks in three-dimensional worlds');

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(3, 3, 2, 'Terraria', 'The game features exploration, crafting, building, painting, and combat with a variety of creatures and bosses');

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(4, 4, 1, 'Warcraft', 'The game allows players to create avatar-style characters and explore a sprawling universe while battling against other players or AI');

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(5, 5, 1, 'Age of Empires', 'The game focus on historical events throughout time');

insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
values(6, 6, 4, 'Dota 2', 'Dota 2 is a multiplayer online battle arena video game in which two teams of five players compete to destroy the enemy team\'s base');
```

ID_JOC	ID_GEN_JOC	ID_VARSTA_JUCATORI	NUME	DESCRIERE
1	2	4	Counter Strike Global Offensive	The game pits two teams, Terrorists and Counter-Terrorists, against each other in different objective-based game modes
2	2	4	Minecraft	Minecraft is a video game in which players create and break apart various kinds of blocks in three-dimensional worlds
3	3	2	Terraria	The game features exploration, crafting, building, painting, and combat with a variety of creatures and bosses
4	4	1	Warcraft	The game allows players to create avatar-style characters and explore a sprawling universe while battling against other players or AI
5	5	1	Age of Empires	The game focus on historical events throughout time
6	6	4	Dota 2	Dota 2 is a multiplayer online battle arena video game in which two teams of five players compete to destroy the enemy team's base

d. Creare si populare tabel companie

```
create table companie(
    id_companie number(4),
    nume_companie varchar2(50)
);
```

```
alter table companie
add constraint pk_companie primary key(id_companie);
```

```
insert into companie(id_companie, nume_companie)
values(1, 'Ubisoft');
```

```
insert into companie(id_companie, nume_companie)
values(2, 'Gameloft');
```

```
insert into companie(id_companie, nume_companie)
values(3, 'Electronic Arts');
```

```
insert into companie(id_companie, nume_companie)
values(4, 'Mojang');
```

w

```
insert into companie(id_companie, nume_companie)
values(5, 'Riot Games');
```

```
select * from companie;
```

```
-- Creare si populare tabel companie

create table companie(
    id_companie number(4),
    nume_companie varchar2(50)
);

alter table companie
add constraint pk_companie primary key(id_companie);

insert into companie(id_companie, nume_companie)
values(1, 'Ubisoft');

insert into companie(id_companie, nume_companie)
values(2, 'Gameloft');

insert into companie(id_companie, nume_companie)
values(3, 'Electronic Arts');
```

ID_COMPANIE	NUME_COMPANIE
1	Ubisoft
2	Gameloft
3	Electronic Arts
4	Mojang
5	Riot Games

e. Creare si populare tabel platforma

```
create table platforma(
    id_platforma number(4),
    nume_platforma varchar2(50)
)
```

```
alter table platforma
add constraint pk_platforma primary key(id_platforma);
```

```
insert into platforma(id_platforma, nume_platforma)
values(1, 'PC');
```

```
insert into platforma(id_platforma, nume_platforma)
values(2, 'Nintendo Switch');
```

```
insert into platforma(id_platforma, nume_platforma)
values(3, 'PS4');
```

```
insert into platforma(id_platforma, nume_platforma)
values(4, 'Smartphones');
```

```
insert into platforma(id_platforma, nume_platforma)
```

```
values(5, 'XBOX');
```

```
select * from platforma;
```

```
create table platforma(
    id_platforma number(4),
    nume_platforma varchar2(50)
);

alter table platforma
add constraint pk_platforma primary key(id_platforma);

insert into platforma(id_platforma, nume_platforma)
values(1, 'PC');

insert into platforma(id_platforma, nume_platforma)
values(2, 'Nintendo Switch');

insert into platforma(id_platforma, nume_platforma)
values(3, 'PS4');

insert into platforma(id_platforma, nume_platforma)
values(4, 'Smartphones');
```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.004 seconds

	ID_PLATFORMA	NUME_PLATFORMA
1	1	PC
2	2	Nintendo Switch
3	3	PS4
4	4	Smartphones
5	5	XBOX

f. Creare si populare tabel regiune

```
create table regiune(
    id_regiune number(4),
    nume_regiune varchar2(50)
)
```

```
alter table regiune
add constraint pk_regiune primary key(id_regiune);
```

```
insert into regiune(id_regiune, nume_regiune)
values(1, 'Europe');
```

```
insert into regiune(id_regiune, nume_regiune)
```

```
values(2, 'Asia');
```

```
insert into regiune(id_regiune, nume_regiune)
values(3, 'Africa');
```

```
insert into regiune(id_regiune, nume_regiune)
values(4, 'North America');
```

```
insert into regiune(id_regiune, nume_regiune)
values(5, 'South America');
```

```
select * from regiune;
```

```

create table regiune(
    id_regiune number(4),
    nume_regiune varchar2(50)
)

alter table regiune
add constraint pk_regiune primary key(id_regiune);

insert into regiune(id_regiune, nume_regiune)
values(1, 'Europe');

insert into regiune(id_regiune, nume_regiune)
values(2, 'Asia');

insert into regiune(id_regiune, nume_regiune)
values(3, 'Africa');

insert into regiune(id_regiune, nume_regiune)
values(4, 'North America');

```

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.005 seconds

ID_REGIUNE	NUME_REGIUNE
1	Europe
2	Asia
3	Africa
4	North America
5	South America

g. Creare si populare tabel asociativ joc_companie

```
create table joc_companie(
    id_joc_companie number(4) not null,
    id_joc number(4) not null,
```

```
        id_companie number(4) not null
    );

alter table joc_companie
add constraint fk_joc_companie_id_joc foreign key (id_joc) references joc(id_joc);

alter table joc_companie
add constraint fk_joc_companie_id_companie foreign key (id_companie) references
companie(id_companie);

alter table joc_companie
add constraint pk_joc_companie1 primary key (id_joc_companie, id_joc, id_companie);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (1, 1, 2);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (2, 5, 4);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (3, 8, 1);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (4, 7, 3);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (5, 3, 3);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (6, 6, 5);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (7, 9, 5);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (8, 9, 1);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (9, 2, 4);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (10, 10, 4);

select * from joc_companie;
```

```

create table joc_companie(
    id_joc_companie number(4) not null,
    id_joc number(4) not null,
    id_companie number(4) not null
);

alter table joc_companie
add constraint fk_joc_companie_id_joc foreign key (id_joc) references joc(id_joc);

alter table joc_companie
add constraint fk_joc_companie_id_companie foreign key (id_companie) references companie(id_companie);

alter table joc_companie
add constraint pk_joc_companie primary key (id_joc_companie, id_joc, id_companie);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (1, 1, 2);

insert into joc_companie(id_joc_companie, id_joc, id_companie)
values (2, 5, 4);

```

Script Output x Query Result x

SQL | All Rows Fetched: 10 in 0.003 seconds

ID_JOC_COMPANIE	ID_JOC	ID_COMPANIE
1	1	2
2	2	4
3	3	1
4	4	3
5	5	3
6	6	5
7	7	5
8	8	1
9	9	4
10	10	4

h. Creare si populare tabel joc_companie_platforma

```

create table joc_companie_platforma(
    id_joc_companie_platforma number(4) not null,
    id_joc number(4) not null,
    id_companie number(4) not null,
    id_platforma number(4) not null,
    an_aparitiei date
)

```

```

alter table joc_companie_platforma
add constraint fk_jcp_joc foreign key(id_joc) references joc(id_joc)

```

```

alter table joc_companie_platforma
add constraint fk_jcp_companie foreign key(id_companie) references companie(id_companie);

```

```

alter table joc_companie_platforma
add constraint fk_jcp_platforma foreign key(id_platforma) references platforma(id_platforma);

```

```

alter table joc_companie_platforma

```

```
add constraint pk_jcp primary key (id_joc_companie_platforma, id_joc, id_companie,
id_platforma)
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (1, 1, 2, 1, TO_DATE('07/11/2012', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (2, 5, 4, 3, TO_DATE('17/12/2015', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (3, 8, 1, 2, TO_DATE('23/04/2002', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (4, 7, 3, 1, TO_DATE('13/10/2011', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (5, 3, 3, 5, TO_DATE('04/04/1997', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (6, 6, 5, 4, TO_DATE('13/12/2020', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (7, 9, 5, 4, TO_DATE('25/12/2022', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (8, 9, 1, 2, TO_DATE('12/01/2004', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (9, 2, 4, 1, TO_DATE('29/07/2006', 'DD/MM/YYYY'));
```

```
insert into joc_companie_platforma(id_joc_companie_platforma, id_joc, id_companie,
id_platforma, an_aparitie)
values (10, 10, 4, 2, TO_DATE('23/04/2005', 'DD/MM/YYYY'));
```

```
select * from joc_companie_platforma;
```



```

create table joc_companie_platforma(
  id_joc_companie_platforma number(4) not null,
  id_joc number(4) not null,
  id_companie number(4) not null,
  id_platforma number(4) not null,
  an_aparitiei date
)

alter table joc_companie_platforma
add constraint fk_jcp_joc foreign key(id_joc) references joc(id_joc)

alter table joc_companie_platforma
add constraint fk_jcp_companie foreign key(id_companie) references companie(id_companie);

alter table joc_companie_platforma
add constraint fk_jcp_platforma foreign key(id_platforma) references platforma(id_platforma);

alter table joc_companie_platforma
add constraint pk_jcp primary key (id_joc_companie_platforma, id_joc, id_companie, id_platforma)

```

Script Output x Query Result x

SQL | All Rows Fetched: 10 in 0.019 seconds

	ID_JOC_COMPANIE_PLATFORMA	ID_JOC	ID_COMPANIE	ID_PLATFORMA	AN_APARITIE
1		1	1	2	1 07-NOV-12
2		2	5	4	3 17-DEC-15
3		3	8	1	2 23-APR-02
4		4	7	3	1 13-OCT-11
5		5	3	3	5 04-APR-97
6		6	6	5	4 13-DEC-20
7		7	9	5	4 25-DEC-22
8		8	9	1	2 12-JAN-04
9		9	2	4	1 29-JUL-06
10		10	10	4	2 23-APR-05

- i. Creare si populare tabel joc_companie_platforma_regiune

```

create table joc_companie_platforma_regiune(
  id_jcpr number(4) not null,
  id_joc number(4) not null,
  id_companie number(4) not null,
  id_platforma number(4) not null,
  id_regiune number(4) not null,
  numar_vanzari number(4)
)

```

```

alter table joc_companie_platforma_regiune
add constraint fk_jcpr_joc foreign key(id_joc) references joc(id_joc)

```

```

alter table joc_companie_platforma_regiune
add constraint fk_jcpr_companie foreign key(id_companie) references
companie(id_companie);

```

```
alter table joc_companie_platforma_regiune
add constraint fk_jcpr_platforma foreign key(id_platforma) references
platforma(id_platforma);
```

```
alter table joc_companie_platforma_regiune
add constraint fk_jcpr_regiune foreign key(id_regiune) references regiune(id_regiune);
```

```
alter table joc_companie_platforma_regiune
add constraint pk_jcpr primary key (id_jcpr, id_joc, id_companie, id_platforma, id_regiune)
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (1, 1, 2, 1, 1, 123);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (2, 5, 4, 3, 4, 213);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (3, 8, 1, 2,2, 1000);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (4, 7, 3,1, 1, 1233);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (5, 3, 3, 5, 3, 1233);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (6, 6, 5, 4, 2, 11);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (7, 9, 5, 4, 5, 199);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (8, 9, 1, 2, 2, 333);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (9, 2, 4, 1, 5, 1222);
```

```
insert into joc_companie_platforma_regiune(id_jcpr, id_joc, id_companie, id_platforma,
id_regiune, numar_vanzari)
values (10, 10, 4, 2, 4, 120);
```

```
select * from joc_companie_platforma_regiune;
```

```

create table joc_companie_platforma_regiune(
    id_jcpr number(4) not null,
    id_joc number(4) not null,
    id_companie number(4) not null,
    id_platforma number(4) not null,
    id_regiune number(4) not null,
    numar_vanzari number(4)
)

alter table joc_companie_platforma_regiune
add constraint fk_jcpr_joc foreign key(id_joc) references joc(id_joc)

alter table joc_companie_platforma_regiune
add constraint fk_jcpr_companie foreign key(id_companie) references companie(id_companie);

alter table joc_companie_platforma_regiune
add constraint fk_jcpr_platforma foreign key(id_platforma) references platforma(id_platforma);

```

Query Result x

SQL | All Rows Fetched: 10 in 0.033 seconds

	ID_JCPR	ID_JOC	ID_COMPANIE	ID_PLATFORMA	ID_REGIUNE	NUMAR_VANZARI
1	1	1	2	1	1	123
2	2	5	4	3	4	213
3	3	8	1	2	2	1000
4	4	7	3	1	1	1233
5	5	3	3	5	3	1233
6	6	6	5	4	2	11
7	7	9	5	4	5	199
8	8	9	1	2	2	333
9	9	2	4	1	5	1222
10	10	10	4	2	4	120

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

Am adaugat tot la punctul 4.

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

Pentru fiecare joc din tabelul joc o sa determin regiunea care a generat cele mai multe vanzari.

create or replace procedure exercitiul6hrm

as

type tablou_indexat is table of joc.numa%type index by pls_integer;

numa_jocuri tablou_indexat;

type tablou_imbricat is table of regiune.numa_regiune%type;

regiuni tablou_imbricat := tablou_imbricat();

vanzare joc_companie_platforma_regiune.numa_vanzari%type;

reg regiune.numa_regiune%type;

maxim number(4);

valoare_vanzare joc_companie_platforma_regiune.numa_vanzari%type;

indice number(4);

begin

select numa

bulk collect into numa_jocuri

from joc;

for i in numa_jocuri.FIRST..numa_jocuri.LAST LOOP

select count(*)

into vanzare

from joc_companie_platforma_regiune jcpr, joc j

where upper(numa_jocuri(i)) = upper(j.numa) and j.id_joc = jcpr.id_joc;

regiuni.extend;

if vanzare = 0 then

regiuni(i) := 'Jocul nu are nicio vanzare';

elsif vanzare = 1 then

select r.numa_regiune

into reg

from regiune r, joc j, joc_companie_platforma_regiune jcpr

where upper(numa_jocuri(i)) = upper(j.numa) and j.id_joc = jcpr.id_joc and

jcpr.id_regiune = r.id_regiune;

regiuni(i) := reg;

else

select id_joc

into indice

from joc

where upper(numa_jocuri(i)) = upper(numa);

maxim := 0;

select max(numa_vanzari)

into maxim

from joc_companie_platforma_regiune

where id_joc = indice;

select r.numa_regiune

into reg

```

from joc_companie_platforma_regiune jcpr, regiune r
where jcpr.numar_vanzari = maxim and jcpr.id_regiune = r.id_regiune;

```

```

regiuni(i) := reg;

```

```

end if;
END LOOP;

```

```

for i in regiuni.FIRST..regiuni.LAST LOOP

```

```

dbms_output.put_line(ume_jocuri(i) || ' | ' || regiuni(i));

```

```

END LOOP;

```

```

end exercitiul6hrm;

```

– Apelare subprogram stocat independent
declare

```

begin

```

```

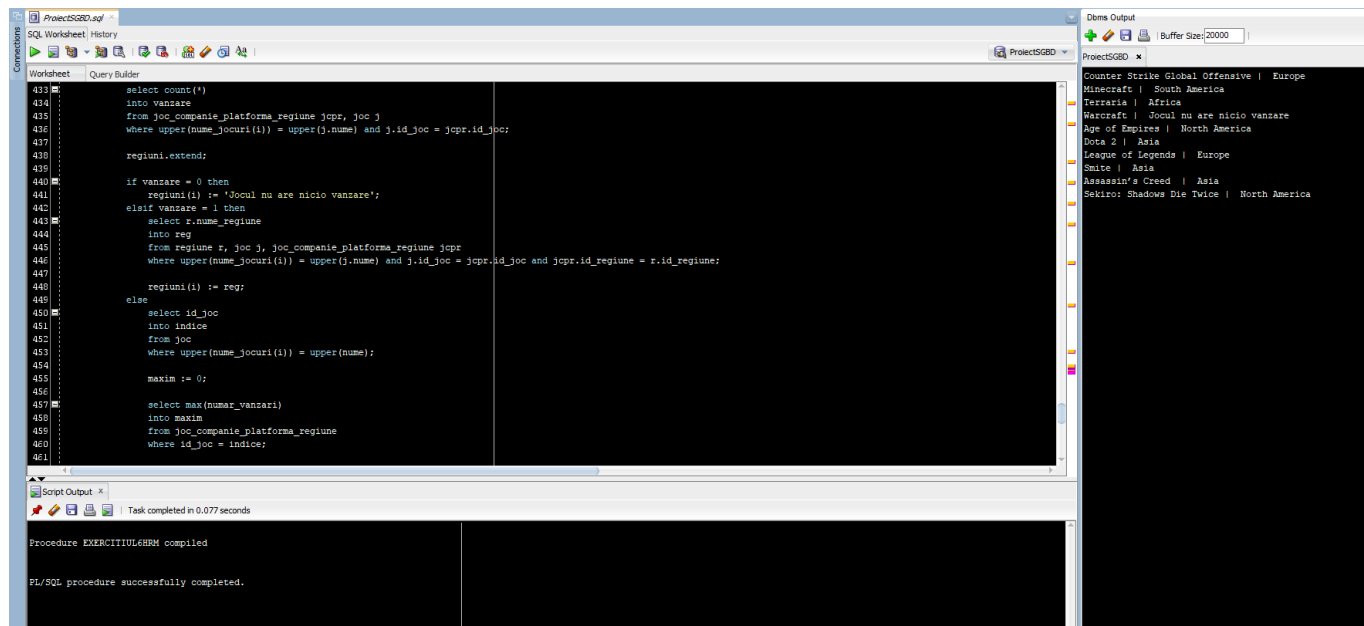
    exercitiul6hrm;

```

```

end;

```



7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelați subprogramul. Pentru fiecare joc sa se afiseze descrierea sa, varsta minima a jucatorilor si cate jocuri au aceeași limita de varsta.

create or replace procedure exercitiul7hrm

as

```

varstaJucator number(3);
idVarstaJucator number(3);
type tablou_indexat is table of number(3) index by pls_integer;
varste tablou_indexat;
counter_varsta number(3);

```

```

CURSOR c2 (numeJoc joc.nume%type) is

```

```

select vj.varsta varsta
from varsta_jucatori vj, joc j
where j.id_varsta_jucatori = vj.id_varsta and upper(numeJoc) = upper(j.nume)
group by vj.varsta;

```

```

type date_joc is record(
nume joc.nume%type,
descriere joc.descriere%type
);

```

```

detalii_joc date_joc;

```

```

cursor c1 return date_joc
is select nume, descriere
from joc;

```

```

begin

```

```

open c1;
loop

```

```

fetch c1 into detalii_joc;
exit when c1%notfound;

```

```

open c2(detalii_joc.nume);
loop
fetch c2 into varstaJucator;
exit when c2%notfound;
select count(*)
into counter_varsta

```

```

from varsta_jucatori vj, joc j
where varstaJucator = vj.varsta and j.id_varsta_jucatori = vj.id_varsta;
dbms_output.put_line('Nume joc: ' || detalii_joc.nume || ', descriere: ' ||
detalii_joc.descriere || ', varstaJucatori ' || varstaJucator || ', counter ' || counter_varsta);
end loop;
close c2;

```

```

end loop;
close c1;

```

```

end exercitiul7hrm;

```

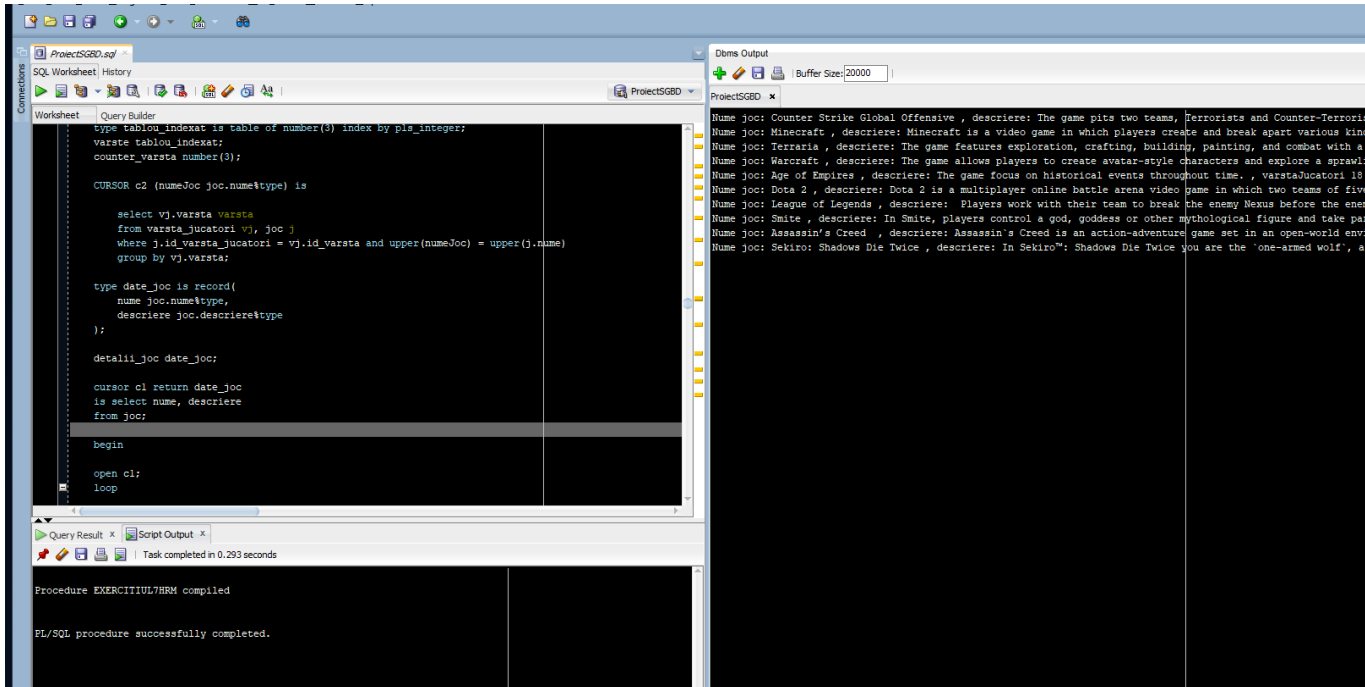
– Apelare subprogram stocat independent

declare

begin

exercitiul7hrm;

end;



8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Creați o funcție care să verifice dacă un joc a fost creat doar de către o companie

```
create or replace function exercitiul8hrm
(numeJoc joc.nume%type)
return number is
countCompanie number(3);
idJoc joc.id_joc%type;
multeCompanii exception;
begin

select id_joc
into idJoc
from joc
where upper(numeJoc) = upper(nume);

select count(jc.id_companie)
into countCompanie
from joc j, joc_companie jc, companie c
```

```

    where upper(umeJoc) = upper(j.ume) and j.id_joc = jc.id_joc and jc.id_comanie =
c.id_comanie;

    if countComanie > 1 then
    raise multeComanii;
    end if;

    return countComanie;

    exception
    when NO_DATA_FOUND
    then RAISE_APPLICATION_ERROR(-20000, 'Nu exista jocul ' || umeJoc || ' in baza de date');
    when multeComanii
    then RAISE_APPLICATION_ERROR(-20000, 'Jocul ' || umeJoc || ' este creat de catre mai multe
comanii');
    when OTHERS
    then RAISE_APPLICATION_ERROR(-20002, 'Alta eroare');

end exercitiul8hrm;

/

select * from joc_comanie;
-- Apel cu exceptie predefinita
declare

    rezultat number(3);
begin
    rezultat := exercitiul8hrm('Assassin's Creed ');
    if rezultat = 1 then
    dbms_output.put_line('Jocul Assassin's Creed a fost creat de catre o companie');
    end if;
end;

-- Apel valid

declare

    rezultat number(3);
begin
    rezultat := exercitiul8hrm('Counter Strike Global Offensive');
    if rezultat = 1 then
    dbms_output.put_line('Jocul Counter Strike Global Offensive a fost creat de catre o companie');
    end if;
end;

-- Apel cu exceptie NO_DATA_FOUND

declare

    rezultat number(3);

```

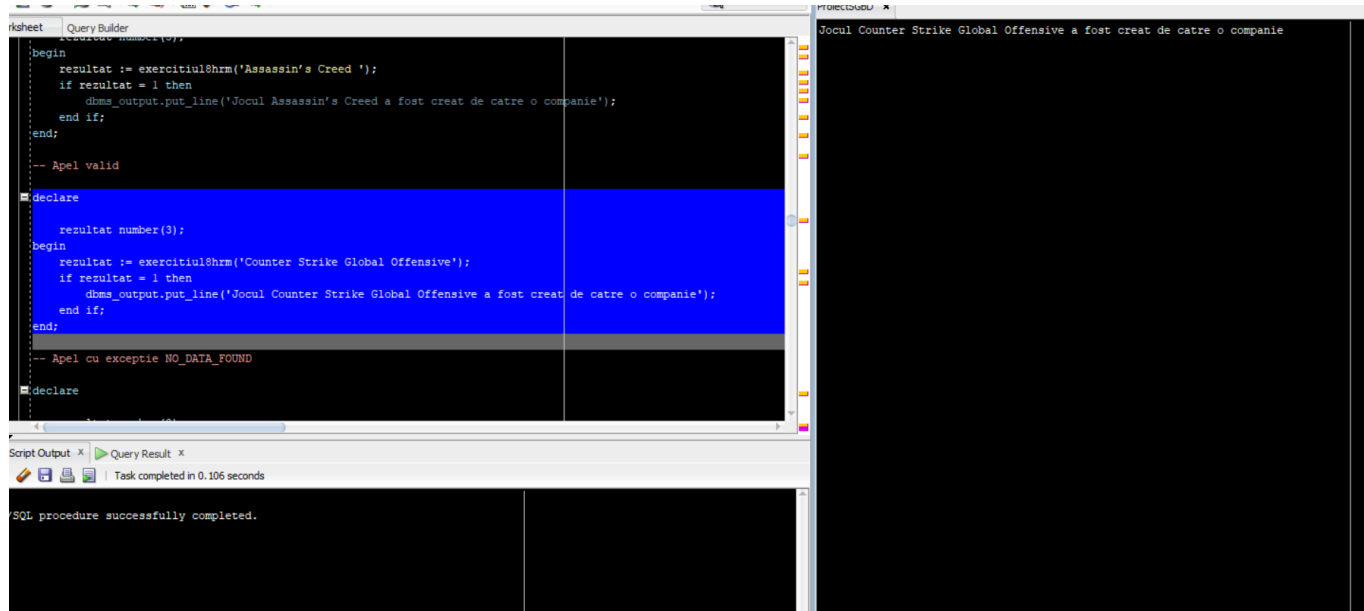


```

begin
    rezultat := exercitiul8hrm('Adopt Me');
    if rezultat = 1 then
        dbms_output.put_line('Jocul Adopt Me a fost creat de catre o companie');
    end if;
end;

```

Apel corect



Apel cu exceptie NO_DATA_FOUND

```

-- Apel cu exceptie NO_DATA_FOUND

declare
    rezultat number(3);
begin
    rezultat := exercitiul8hrm('Adopt Me');
    if rezultat = 1 then
        dbms_output.put_line('Jocul Adopt Me a fost creat de catre o companie');
    end if;
end;

/

-- Exerciitiul 9
-- Sa se afiseze regiunea in care se joaca un anumit joc

```

Script Output x Query Result x

Task completed in 0.059 seconds

```

Error starting at line : 566 in command -
declare

    rezultat number(3);
begin
    rezultat := exercitiul8hrm('Adopt Me');
    if rezultat = 1 then
        dbms_output.put_line('Jocul Adopt Me a fost creat de catre o companie');
    end if;
end;
Error report -
ORA-20000: Nu exista jocul Adopt Me in baza de date
ORA-06512: at "ALIEN14.EXERCITIUL8HRM", line 27
ORA-06512: at line 5
20000. 00000 - "%s"
*Cause:      The stored procedure 'raise_application_error'

```

Apel cu exceptie predefinita

```

select * from joc_companie;
-- Apel cu exceptie predefinita
declare
    rezultat number(3);
begin
    rezultat := exercitiul8hrm('Assassin's Creed ');
    if rezultat = 1 then
        dbms_output.put_line('Jocul Assassin's Creed a fost creat de catre o companie');
    end if;
end;

-- Apel valid
declare
    rezultat number(3);
begin
    rezultat := exercitiul8hrm('Counter Strike Global Offensive');

```

Script Output x Query Result x

Task completed in 0.076 seconds

Error starting at line : 542 in command -

```

declare

    rezultat number(3);
begin
    rezultat := exercitiul8hrm('Assassin's Creed ');
    if rezultat = 1 then
        dbms_output.put_line('Jocul Assassin's Creed a fost creat de catre o companie');
    end if;
end;
Error report -
ORA-20000: Jocul Assassin's Creed este creat de catre mai multe companii
ORA-06512: at "ALIEN14.EXERCITIUL8HRM", line 29
ORA-06512: at line 5

```

9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Sa se afiseze regiunea in care se joaca un anumit joc

```

create or replace procedure exercitiul9hrm
    (numeJoc joc.nume%TYPE)
as

    numeRegiune regiune.nume_regiune%type;

begin

    select r.nume_regiune
    into numeRegiune

```

```

        from joc j, joc_companie jc, joc_companie_platforma jcp,
joc_companie_platforma_regiune jcpr, regiune r
        where upper(umeJoc) = upper(j.ume) and j.id_joc = jc.id_joc and jc.id_joc = jcp.id_joc
and jcp.id_joc = jcpr.id_joc and jcpr.id_regiune = r.id_regiune;

```

```

        dbms_output.put_line('Jocul ' || umeJoc || ' se afla in ' || umeRegiune);

```

```

exception

```

```

    when no_data_found
    then dbms_output.put_line('Jocul ' || umeJoc || ' nu se afla in nicio regiune');
    when too_many_rows
    then dbms_output.put_line('Jocul ' || umeJoc || ' se afla in mai multe regiuni');
    when others
    then dbms_output.put_line('Alt tip de eroare');

```

```

end exercitiul9hrm;

```

```

-- Apel corect

```

```

declare

```

```

begin

```

```

    exercitiul9hrm('Minecraft');

```

```

end;

```

```

-- Apel NO_DATA_FOUND

```

```

declare

```

```

begin

```

```

    exercitiul9hrm('Warcraft');

```

```

end;

```

```

-- Apel TOO_MANY_ROWS

```

```

declare

```

```

begin

```

```

    exercitiul9hrm('Assassin's Creed ');

```

```

End;

```

```

Apel corect

```

The screenshot shows the Oracle SQL Developer interface. The 'Worksheet' tab is active, displaying a PL/SQL procedure named `exercitiul9hrm`. The procedure includes an exception block with three cases: `no_data_found`, `too_many_rows`, and `others`. The `no_data_found` case outputs 'Jocul nu se afla in nicio regiune', the `too_many_rows` case outputs 'Jocul se afla in mai multe regiuni', and the `others` case outputs 'Alt tip de eroare'. The procedure is called with the argument 'Minecraft'. The 'Query Result' tab shows the output: 'Jocul Minecraft se afla in South America'. The status bar indicates 'Task completed in 0.047 seconds' and 'PL/SQL procedure successfully completed.'

```

686 exception
687     when no_data_found
688     then dbms_output.put_line('Jocul nu se afla in nicio regiune');
689     when too_many_rows
690     then dbms_output.put_line('Jocul se afla in mai multe regiuni');
691     when others
692     then dbms_output.put_line('Alt tip de eroare');
693
694
695 end exercitiul9hrm;
696
697 declare
698
699 begin
700     exercitiul9hrm('Minecraft');
701 end;
702
703

```

Script Output x Query Result x

Task completed in 0.047 seconds

PL/SQL procedure successfully completed.

ProjectSGBD x

Jocul Minecraft se afla in South America

Apel NO_DATA_FOUND

The screenshot shows the Oracle SQL Developer interface. The 'Worksheet' tab is active, displaying the same PL/SQL procedure `exercitiul9hrm` as in the previous image. The procedure is called with the argument 'Warcraft'. The 'Query Result' tab shows the output: 'Jocul nu se afla in nicio regiune'. The status bar indicates 'Task completed in 0.057 seconds' and 'PL/SQL procedure successfully completed.'

```

686 exception
687     when no_data_found
688     then dbms_output.put_line('Jocul nu se afla in nicio regiune');
689     when too_many_rows
690     then dbms_output.put_line('Jocul se afla in mai multe regiuni');
691     when others
692     then dbms_output.put_line('Alt tip de eroare');
693
694
695 end exercitiul9hrm;
696
697 declare
698
699 begin
700     exercitiul9hrm('Warcraft');
701 end;
702
703

```

Script Output x Query Result x

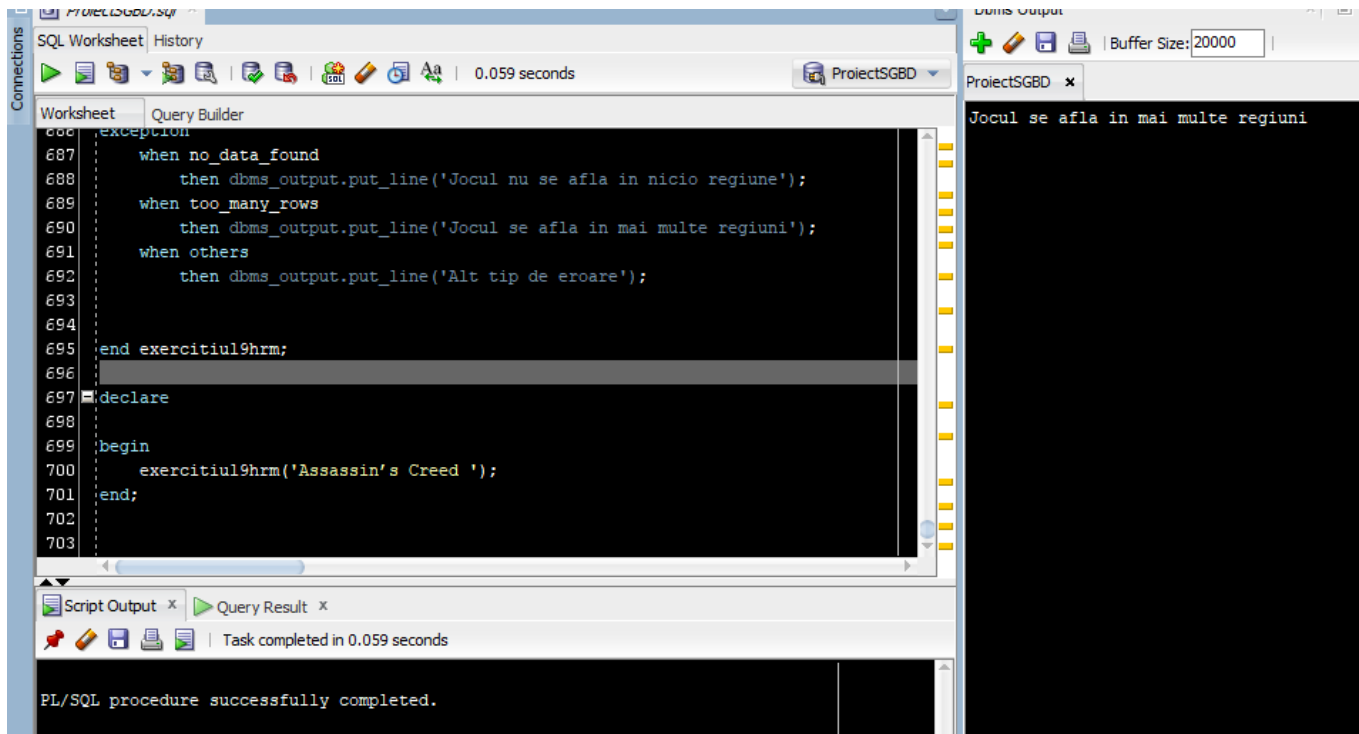
Task completed in 0.057 seconds

PL/SQL procedure successfully completed.

ProjectSGBD x

Jocul nu se afla in nicio regiune

Apel TOO_MANY_ROWS



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

Sa se creeze un trigger de tip LMD la nivel de instructiune care sa nu permita inserarea in tabelului joc

```
create or replace trigger exercitiul10hrm
  before insert or delete on joc
declare
  counterIdJoc joc.id_joc%type;
begin
  select count(id_joc)
  into counterIdJoc
  from joc;

  if counterIdJoc > 10 then
    raise_application_error(-20000,'Nu puteti insera in acest tabel');
  end if;

end;

/

declare

begin

  for i in 1..5 loop
    insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
    values(10 + i, 1, 4, 'Minecraft 2.0', 'A new version of Minecraft');
  end loop;
```

```
--      update joc
--      set nume = 'Minecraft 2.0'
--      where id_gen_joc = 4;
```

End;

Declansare trigger

```

declare
begin
  for i in 1..5 loop
    insert into joc(id_joc, id_gen_joc, id_varsta_jucatori, nume, descriere)
      values(10 + i, 1, 4, 'Minecraft 2.0', 'A new version of Minecraft');
  end loop;

  --      update joc
  --      set nume = 'Minecraft 2.0'
  --      where id_gen_joc = 4;

end;

```

Script Output x Query Result x

Task completed in 0.077 seconds

```

values(10 + i, 1, 4, 'Minecraft 2.0', 'A new version of Minecraft');
end loop;

--      update joc
--      set nume = 'Minecraft 2.0'
--      where id_gen_joc = 4;

end;
Error report -
ORA-20000: Nu puteti insera in acest tabel
ORA-06512: at "ALIEN14.EXERCITIUL10HRM", line 9
ORA-04088: error during execution of trigger 'ALIEN14.EXERCITIUL10HRM'
ORA-06512: at line 6
20000. 00000 - "%s"
Cause:      The stored procedure 'raise_application_error'
            was called which causes this error to be generated.
Action:     Correct the problem as described in the error message or contact
            the application administrator or DBA for more information.

```

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

Sa se creeze un trigger la nivel de linie care nu permite micșorarea numărului de vanzari ale unui joc

create or replace trigger exercitiul11hrm

before update of numar_vanzari on joc_companie_platforma_regiune

for each row

begin

if(:NEW.numar_vanzari < :OLD.numar_vanzari) then

raise_application_error(-20002, 'Numarul de vanzari nu poate fi micșorat');

end if;

end;

/

```
-- Rulare valida
update joc_companie_platforma_regiune
set numar_vanzari = 124
where id_jcpr = 1;
```

```
rollback;
```

```
-- Declansare trigger
update joc_companie_platforma_regiune
set numar_vanzari = 50
where id_jcpr = 1;
```

Rulare valida

```

Worksheet  Query Builder
-- create or replace trigger exercitiu11hrm
  before update of numar_vanzari on joc_companie_platforma_regiune
  for each row
-- begin
  if(:NEW.numar_vanzari < :OLD.numar_vanzari) then
    raise_application_error(-20002, 'Numarul de vanzari nu poate fi micsorat');
  end if;
end;

/
-- Rulare valida
update joc_companie_platforma_regiune
set numar_vanzari = 124
where id_jcpr = 1;

-- Declansare trigger
update joc_companie_platforma_regiune

Query Result x  Script Output x
Task completed in 0.096 seconds

Trigger EXERCITIUL11HRM compiled

1 row updated.
```

Declansare trigger


```
-- Declansare trigger
update joc_companie_platforma_regiune
set numar_vanzari = 50
where id_jcpr = 1;

select user from dual;

select * from joc_companie_platforma_regiune;
```

Query Result x Script Output x

Task completed in 0.077 seconds

```
Error starting at line : 627 in command -
update joc_companie_platforma_regiune
set numar_vanzari = 50
where id_jcpr = 1
Error report -
ORA-20002: Numarul de vanzari nu poate fi micorat
ORA-06512: at "ALIEN14.EXERCITIUL11HRM", line 4
ORA-04088: error during execution of trigger 'ALIEN14.EXERCITIUL11HRM'
```

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

Sa se creeze un trigger care permite modificarea tabelelor din baza de date doar de catre user-ul ALIEN14 si toate modificarile sa fie salvate intr-un tabel

– Acum creez tabelul care va retine modificarile facute in baza de date

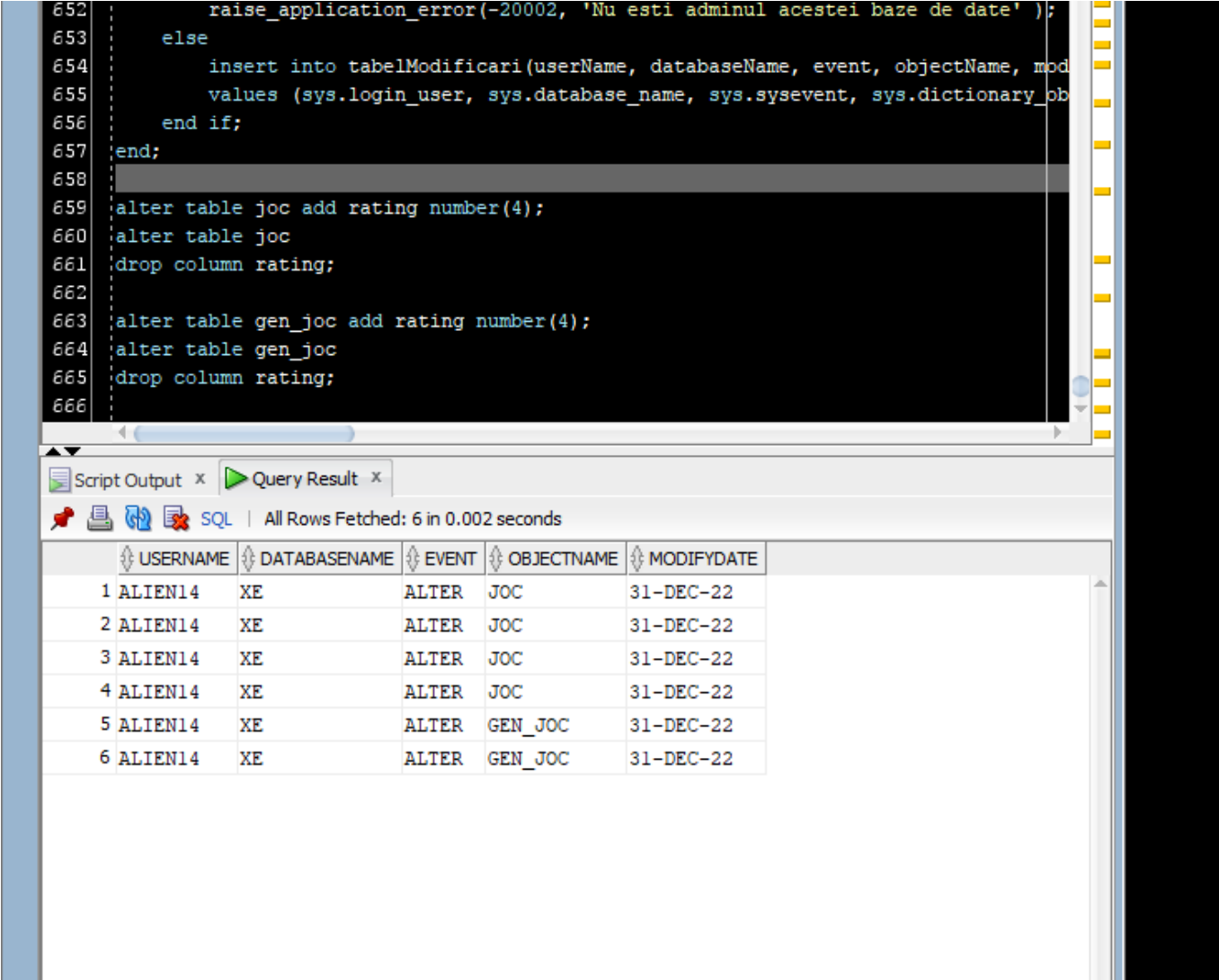
```
create table tabelModificari(
    userName varchar2(50),
    databaseName varchar2(50),
    event varchar2(50),
    objectName varchar2(50),
    modifyDate date
);
```

```
create or replace trigger exercitiul12hrm
before drop or alter or create on schema
begin
    if upper('Alien14') != USER then
        raise_application_error(-20002, 'Nu esti adminul acestei baze de date' );
    else
        insert into tabelModificari(userName, databaseName, event, objectName, modifyDate)
        values (sys.login_user, sys.database_name, sys.sysevent, sys.dictionary_obj_name,
        sysdate);
    end if;
end;
```

```
alter table joc add rating number(4);
alter table joc
drop column rating;
```

```
alter table gen_joc add rating number(4);
alter table gen_joc
drop column rating;
```

```
select * from tabelModificari;
```



The screenshot shows a SQL IDE interface. The top pane displays a script with the following SQL statements:

```

652      raise_application_error(-20002, 'Nu esti adminul acestei baze de date' );
653      else
654          insert into tabelModificari(userName, databaseName, event, objectName, mod
655          values (sys.login_user, sys.database_name, sys.sysevent, sys.dictionary_ob
656      end if;
657  end;
658
659  alter table joc add rating number(4);
660  alter table joc
661  drop column rating;
662
663  alter table gen_joc add rating number(4);
664  alter table gen_joc
665  drop column rating;
666

```

The bottom pane shows the 'Query Result' tab with the following data:

	USERNAME	DATABASENAME	EVENT	OBJECTNAME	MODIFYDATE
1	ALIEN14	XE	ALTER	JOC	31-DEC-22
2	ALIEN14	XE	ALTER	JOC	31-DEC-22
3	ALIEN14	XE	ALTER	JOC	31-DEC-22
4	ALIEN14	XE	ALTER	JOC	31-DEC-22
5	ALIEN14	XE	ALTER	GEN_JOC	31-DEC-22
6	ALIEN14	XE	ALTER	GEN_JOC	31-DEC-22

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

```

create or replace package exercitiul13hrm as
    procedure exercitiul6hrm;
    procedure exercitiul7hrm;
    function exercitiul8hrm (numeJoc joc.nume%type) return number;
    procedure exercitiul9hrm (numeJOc joc.nume%type);
end exercitiul13hrm;
```

```
create or replace package body exercitiul13hrm as
```

```
-- Exercitiul 6
```

```
-- Pentru fiecare joc din tabelul joc o sa determin regiunea care a generat cele mai multe vanzari.
```

```
-- În cazul în care jocul nu a fost vândut o sa adaug mesajul 'Jocul nu are nicio vanzare',
```

```
-- în cazul în care are o singura regiune în care s-a vandut o sa adaug regiunea respectiv? ?i
```

-- în cazul în care jocul are mai multe regiuni în care s-a vandut o sa calculez valoarea maxim? a vanzarilor

-- si dup? o sa determin regiunea careia îi corespunde maximul vanzarilor.

-- La final o sa afisez numele jocului urmat? de regiunea care a generat cele mai multe vanzari.

```

procedure exercitiul6hrm
as
type tablou_indexat is table of joc.nume%type index by pls_integer;
nume_jocuri tablou_indexat;
type tablou_imbricat is table of regiune.nume_regiune%type;
regiuni tablou_imbricat := tablou_imbricat();
vanzare joc_companie_platforma_regiune.numar_vanzari%type;
reg regiune.nume_regiune%type;
maxim number(4);
valoare_vanzare joc_companie_platforma_regiune.numar_vanzari%type;
indice number(4);

begin
select nume
bulk collect into nume_jocuri
from joc;

for i in nume_jocuri.FIRST..nume_jocuri.LAST LOOP

    select count(*)
    into vanzare
    from joc_companie_platforma_regiune jcpr, joc j
    where upper(nume_jocuri(i)) = upper(j.nume) and j.id_joc = jcpr.id_joc;

    regiuni.extend;

    if vanzare = 0 then
        regiuni(i) := 'Jocul nu are nicio vanzare';
    elsif vanzare = 1 then
        select r.nume_regiune
        into reg
        from regiune r, joc j, joc_companie_platforma_regiune jcpr
        where upper(nume_jocuri(i)) = upper(j.nume) and j.id_joc = jcpr.id_joc and
        jcpr.id_regiune = r.id_regiune;

        regiuni(i) := reg;
    else
        select id_joc
        into indice
        from joc
        where upper(nume_jocuri(i)) = upper(nume);

        maxim := 0;

        select max(numar_vanzari)
        into maxim

```

```

from joc_companie_platforma_regiune
where id_joc = indice;

select r.ume_regiune
into reg
from joc_companie_platforma_regiune jcpr, regiune r
where jcpr.numar_vanzari = maxim and jcpr.id_regiune = r.id_regiune;

regiuni(i) := reg;

end if;
END LOOP;

for i in regiuni.FIRST..regiuni.LAST LOOP

    dbms_output.put_line(ume_jocuri(i) || ' | ' || regiuni(i));

END LOOP;

end exercitiul6hrm;

```

-- Exercițiul 7

-- Pentru fiecare joc sa se afiseze descrierea sa, varsta minima a jucatorilor si cate jocuri au aceea?i limita de varsta.

```

procedure exercitiul7hrm

as

varstaJucator number(3);
idVarstaJucator number(3);
type tablou_indexat is table of number(3) index by pls_integer;
varste tablou_indexat;
counter_varsta number(3);

CURSOR c2 (numeJoc joc.ume%type) is

    select vj.varsta varsta
    from varsta_jucatori vj, joc j
    where j.id_varsta_jucatori = vj.id_varsta and upper(umeJoc) = upper(j.ume)
    group by vj.varsta;

type date_joc is record(
    ume joc.ume%type,
    descriere joc.descriere%type
);

detalii_joc date_joc;

cursor c1 return date_joc

```

```

is select nume, descriere
from joc;

begin

open c1;
loop

    fetch c1 into detalii_joc;
    exit when c1%notfound;

    open c2(detalii_joc.nume);
    loop
    fetch c2 into varstaJucator;
    exit when c2%notfound;
    select count(*)
    into counter_varsta
    from varsta_jucatori vj, joc j
    where varstaJucator = vj.varsta and j.id_varsta_jucatori = vj.id_varsta;
    dbms_output.put_line('Nume joc: ' || detalii_joc.nume || ', descriere: ' ||
detalii_joc.descriere || ', varstaJucatori ' || varstaJucator || ', counter ' || counter_varsta);
    end loop;
    close c2;

end loop;
close c1;

end exercitiul7hrm;

-- Exerciitiul 8
-- Creati o functie care sa verifice daca un joc a fost creat doar de catre o companie

function exercitiul8hrm
(numeJoc joc.nume%type)
return number is
countCompanie number(3);
idJoc joc.id_joc%type;
multeCompanii exception;
begin

select id_joc
into idJoc
from joc
where upper(numeJoc) = upper(nume);

select count(jc.id_companie)
into countCompanie
from joc j, joc_companie jc, companie c
where upper(numeJoc) = upper(j.nume) and j.id_joc = jc.id_joc and jc.id_companie =
c.id_companie;

```

```

    if countCompanie > 1 then
        raise multeCompanii;
    end if;

    return countCompanie;

    exception
    when NO_DATA_FOUND
    then RAISE_APPLICATION_ERROR(-20000, 'Nu exista jocul ' || numeJoc || ' in baza de
date');
    when multeCompanii
    then RAISE_APPLICATION_ERROR(-20001, 'Jocul ' || numeJoc || ' este creat de catre
mai multe companii');
    when OTHERS
    then RAISE_APPLICATION_ERROR(-20002, 'Alta eroare');

end exercitiul8hrm;

-- Exercițiul 9
-- Sa se afișeze regiunea in care se joaca un anumit joc

procedure exercitiul9hrm
(numeJoc joc.nume%TYPE)
as
    numeRegiune regiune.nume_regiune%type;
begin

    select r.nume_regiune
    into numeRegiune
    from joc j, joc_companie jc, joc_companie_platforma jcp,
joc_companie_platforma_regiune jcpr, regiune r
    where upper(numeJoc) = upper(j.nume) and j.id_joc = jc.id_joc and jc.id_joc = jcp.id_joc
and jcp.id_joc = jcpr.id_joc and jcpr.id_regiune = r.id_regiune;

    dbms_output.put_line('Jocul ' || numeJoc || ' se afla in ' || numeRegiune);

    exception
    when no_data_found
    then dbms_output.put_line('Jocul ' || numeJoc || ' nu se afla in nicio regiune');
    when too_many_rows
    then dbms_output.put_line('Jocul ' || numeJoc || ' se afla in mai multe regiuni');
    when others
    then dbms_output.put_line('Alt tip de eroare');

end exercitiul9hrm;
end exercitiul13hrm;

-- Apel pentru exercitiul 6
execute exercitiul13hrm.exercitiul6hrm;

```

```
-- Apel pentru exercitiul 7
execute exercitiul13hrm.exercitiul7hrm;
```

```
-- Apel pentru exercitiul 8
```

```
-- Apel cu exceptie predefinita
declare
```

```
rezultat number(3);
begin
rezultat := exercitiul13hrm.exercitiul8hrm('Assassin's Creed ');
if rezultat = 1 then
dbms_output.put_line('Jocul Assassin's Creed a fost creat de catre o companie');
end if;
end;
```

```
-- Apel valid
```

```
declare
```

```
rezultat number(3);
begin
rezultat := exercitiul13hrm.exercitiul8hrm('Counter Strike Global Offensive');
if rezultat = 1 then
dbms_output.put_line('Jocul Counter Strike Global Offensive a fost creat de catre o
companie');
end if;
end;
```

```
-- Apel cu exceptie NO_DATA_FOUND
```

```
declare
```

```
rezultat number(3);
begin
rezultat := exercitiul13hrm.exercitiul8hrm('Adopt Me');
if rezultat = 1 then
dbms_output.put_line('Jocul Adopt Me a fost creat de catre o companie');
end if;
end;
```

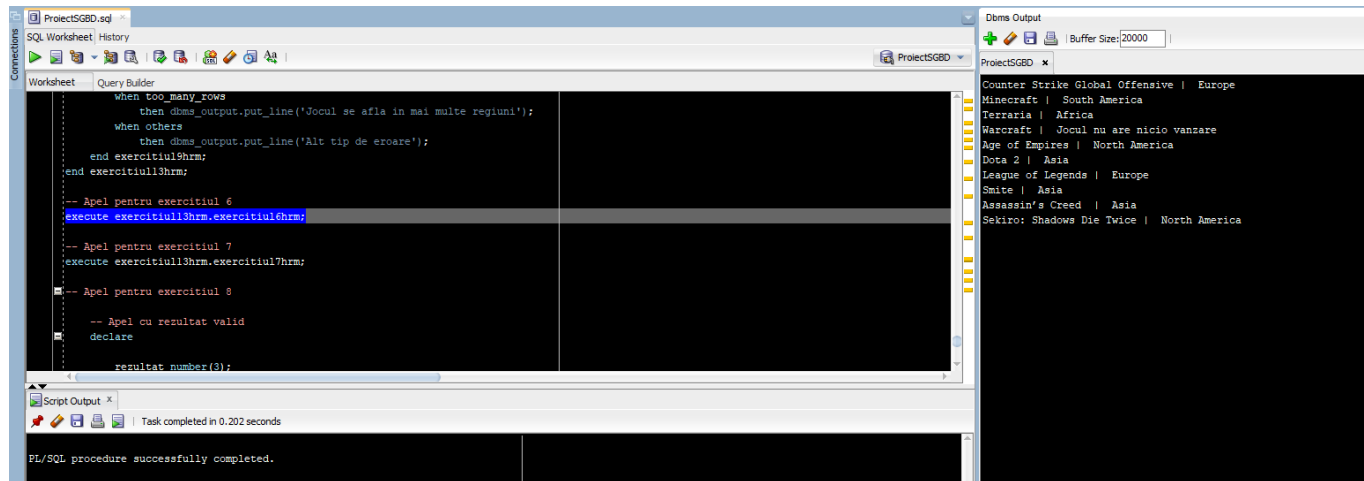
```
-- Apel pentru exercitiul 9
```

```
-- Apel corect
execute exercitiul13hrm.exercitiul9hrm('Minecraft');
```

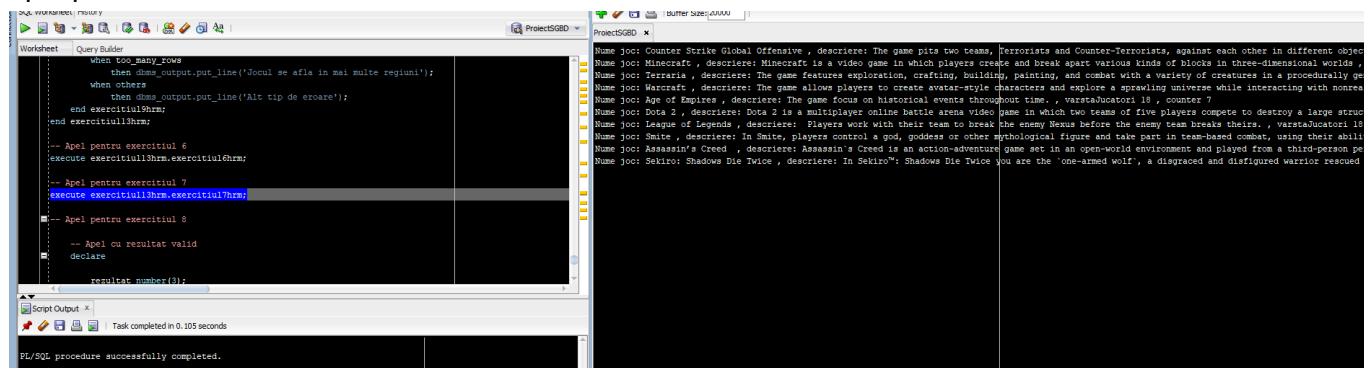
```
-- Apel NO_DATA_FOUND
execute exercitiul13hrm.exercitiul9hrm('Warcraft');
```

```
-- Apel TOO_MANY_ROWS
execute exercitiul13hrm.exercitiul9hrm('Assassin's Creed ');
```

Apel pentru exercitiul 6

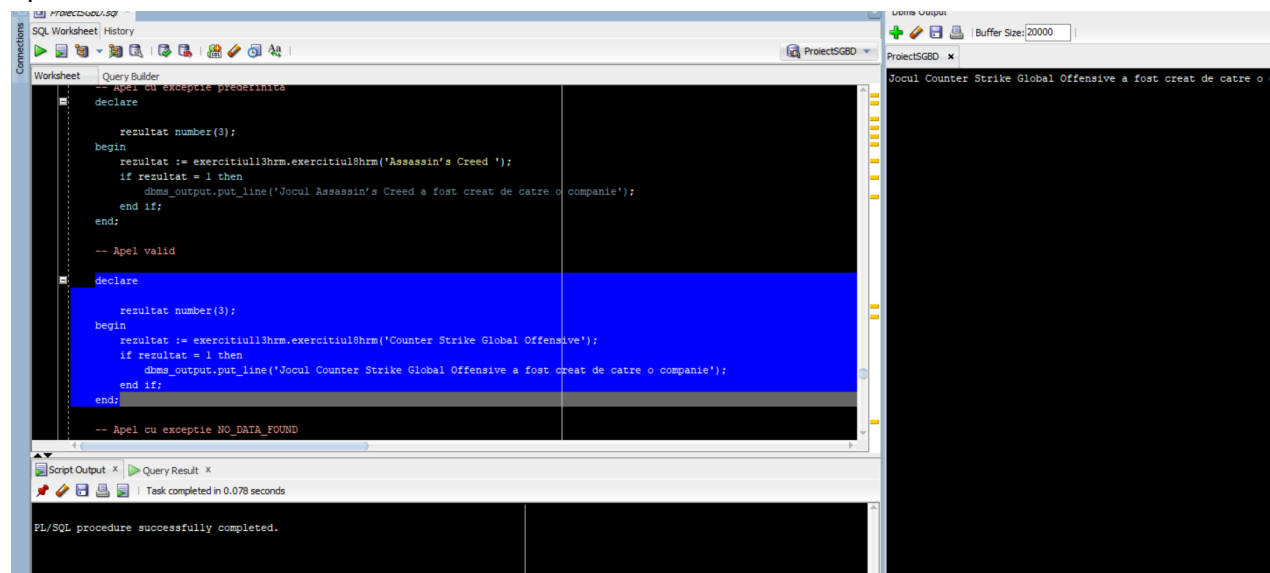


Apel pentru exercitiul 7



Apel pentru exercitiul 8

- Apel cu rezultat valid



- Apel cu exceptie NO_DATA_FOUND


```

-- Apel pentru exercitiul 9

declare
    rezultat number(3);
begin
    rezultat := exercitiul13hrm.exercitiul8hrm('Adopt Me');
    if rezultat = 0 then
        dbms_output.put_line('Jocul Adopt Me nu a fost creat inca de nicio companie.');
```

```

    elsif rezultat = 1 then
        dbms_output.put_line('Jocul Adopt Me a fost creat de catre o companie.');
```

```

    else
        dbms_output.put_line('Jocul Adopt Me a fost creat de catre mai multe companii.');
```

```

    end if;
end;
```

```

-- Apel pentru exercitiul 9
```

Script Output x

Task completed in 0.098 seconds

```

rezultat := exercitiul13hrm.exercitiul8hrm('Adopt Me');
if rezultat = 0 then
    dbms_output.put_line('Jocul Adopt Me nu a fost creat inca de nicio companie.');
```

```

elsif rezultat = 1 then
    dbms_output.put_line('Jocul Adopt Me a fost creat de catre o companie.');
```

```

else
    dbms_output.put_line('Jocul Adopt Me a fost creat de catre mai multe companii.');
```

```

end if;
end;
```

Error report -

ORA-20000: Nu exista jocul Adopt Me in baza de date

ORA-06512: at "ALIEN14.EXERCITIUL13HRM", line 141

ORA-06512: at line 5

- Apel exceptie predefinita

```

-- Apel cu exceptie predefinita
declare
    rezultat number(3);
begin
    rezultat := exercitiul13hrm.exercitiul8hrm('Assassin's Creed ');
    if rezultat = 1 then
        dbms_output.put_line('Jocul Assassin's Creed a fost creat de catre o companie');
    end if;
end;

-- Apel valid

```

Script Output x Query Result x

Task completed in 0.084 seconds

PL/SQL procedure successfully completed.

Error starting at line : 947 in command -

```

declare
    rezultat number(3);
begin
    rezultat := exercitiul13hrm.exercitiul8hrm('Assassin's Creed ');
    if rezultat = 1 then
        dbms_output.put_line('Jocul Assassin's Creed a fost creat de catre o companie');
    end if;
end;

```

Error report -

ORA-20000: Jocul Assassin's Creed este creat de catre mai multe companii

ORA-06512: at "ALIEN14.EXERCITIUL13HRM", line 164

ORA-06512: at line 5

20000. 00000 - "%s"

*Cause: The stored procedure 'raise_application_error' was called which causes this error to be generated.

*Action: Correct the problem as described in the error message or contact

Apel pentru exercitiul 9

- Apel corect

SQL Worksheet History

Worksheet Query Builder

```

dbms_output.put_line('Jocul Adopt Me a fost creat de catre mai multe companii');
end if;
end;

-- Apel pentru exercitiul 9

-- Apel corect
execute exercitiul13hrm.exercitiul9hrm('Minecraft');

-- Apel NO_DATA_FOUND
execute exercitiul13hrm.exercitiul9hrm('Warcraft');

-- Apel TOO_MANY_ROWS
execute exercitiul13hrm.exercitiul9hrm('Assassin's Creed ');

```

Script Output x

Task completed in 0.075 seconds

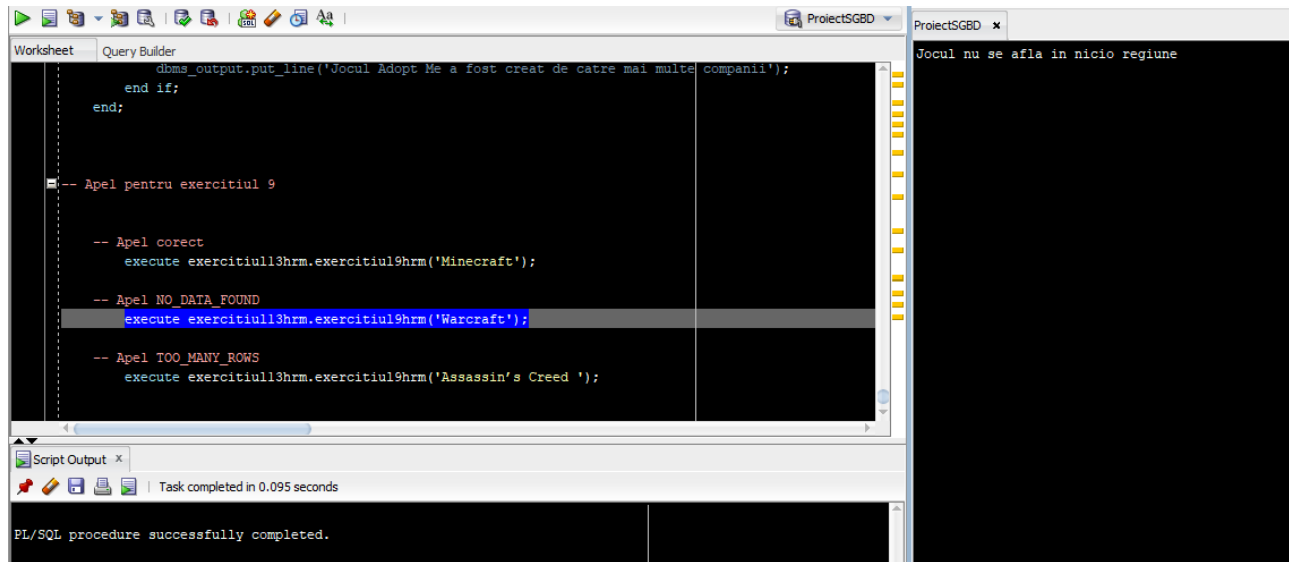
PL/SQL procedure successfully completed.

ProjectSGBD x

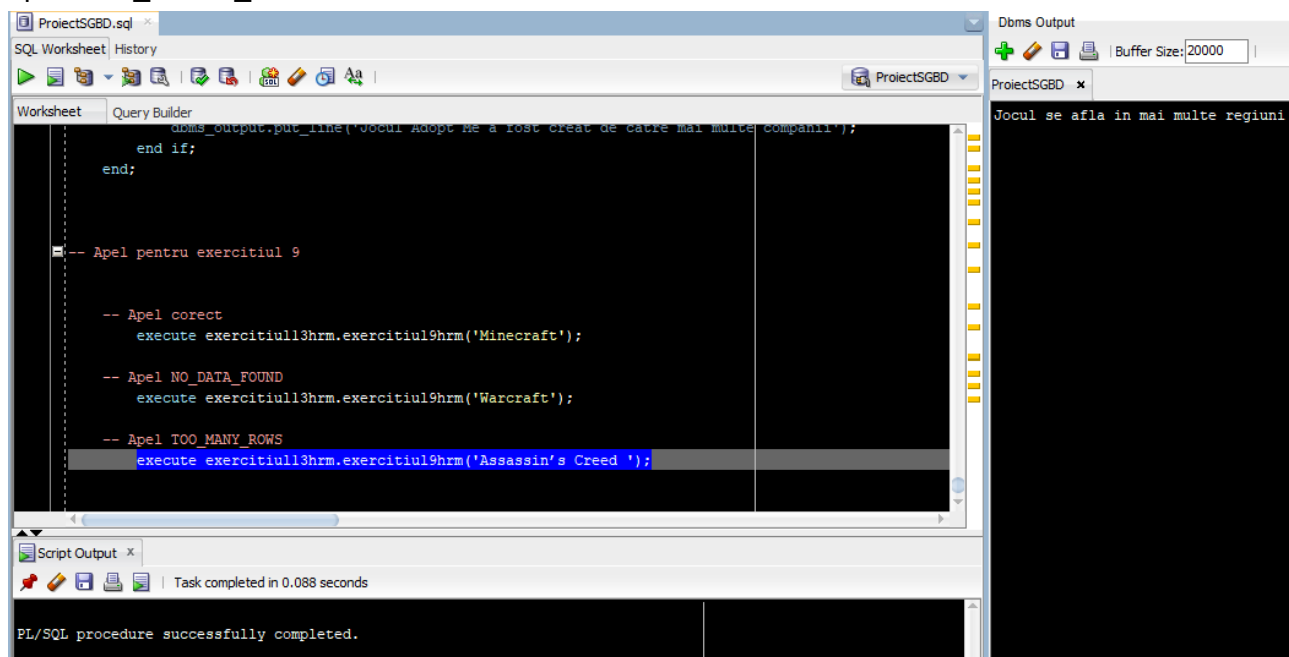
ProjectSGBD x

Jocul Minecraft se afla in South America

- Apel NO_DATA_FOUND



- Apel TOO_MANY_ROWS



14. 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).

```
create or replace package exercitiul14hrm as
```

```
    function numarJocuriIncasari(suma
    joc_companie_platforma_regiune.numar_vanzari%type)
    return number;
```

```
    function numarJocuri(loc regiune.nume_regiune%type,suma
    joc_companie_platforma_regiune.numar_vanzari%type)
    return number;
```

```
    procedure listaCompanii(numeRegiune regiune.nume_regiune%type);
```

```

    procedure platformaVanzari;
end exercitiul14hrm;

```

create or replace package body exercitiul14hrm as

-- O functie care pentru un numar dat returneaza cate jocuri au produs incasari mai mari decat acel numar

```

    function numarJocuriIncasari(
        suma joc_companie_platforma_regiune.numar_vanzari%type
    )
    return number
    is
    contor number;
    begin

```

```

        select count(numar_vanzari)
        into contor
        from joc_companie_platforma_regiune
        where numar_vanzari > suma;

```

```

    return contor;

```

```

    exception
    WHEN NO_DATA_FOUND
    THEN return 0;

```

```

    end numarJocuriIncasari;

```

-- O functie care returneaza cate jocuri au incasari mai mari decat un numar dat si fac parte dintr-o anumita regiune

```

    function numarJocuri(loc regiune.ume_regiune%type,suma
joc_companie_platforma_regiune.numar_vanzari%type)
    return number
    as
    contor number;
    begin

```

```

        select count(numar_vanzari)
        into contor
        from joc_companie_platforma_regiune jcpr, regiune r
        where upper(r.ume_regiune) = upper(loc) and jcpr.numar_vanzari > suma and
r.id_regiune = jcpr.id_regiune;

```

```

    return contor;

```

```

    exception
    WHEN NO_DATA_FOUND
    THEN return 0;

```

```

    end;

```

-- O procedura care afiseaza companiile cu cele mai mari incasari pentru o anumita regiune

```

procedure listaCompanii(umeRegiune regiune.ume_regiune%type)
IS

cursor listaTop
is
select c.ume_comanie, sum(jcpr.numar_vanzari)
from joc_comanie_platforma_regiune jcpr, regiune r, companie c
where r.id_regiune = jcpr.id_regiune and upper(r.ume_regiune) = upper(umeRegiune)
and c.id_comanie = jcpr.id_comanie
group by c.ume_comanie
order by sum(jcpr.numar_vanzari) desc;

ume companie.ume_comanie%type;
vanzari joc_comanie_platforma_regiune.numar_vanzari%type;

begin

open listaTop;
loop

fetch listaTop into ume, vanzari;
exit when listaTop%NOTFOUND;

dbms_output.put_line(ume || ' ' || vanzari);

end loop;
close listaTop;

end;

-- O procedura care afiseaza pentru fiecare platforma toate regiunile in care sunt jocuri si
pentru fiecare regiune toate companiile ordonate descrescator dupa numarul de vanzari
procedure platformaVanzari
is
cursor platforme
is
select ume_platforma
from platforma;
type umePlatforma is table of platforma.ume_platforma%type;
var _platforma umePlatforma;
-- ume_platforma platforma.ume_platforma%type;

CURSOR c2 (umePlatforma platforma.ume_platforma%type) is

select r.id_regiune from regiune r, joc_comanie_platforma_regiune jcpr, platforma p
where upper(p.ume_platforma) = upper(umePlatforma) and p.id_platforma =
jcpr.id_platforma and jcpr.id_regiune = r.id_regiune
group by r.id_regiune;

```

```

regiuneId regiune.id_regiune%type;
numeRegiune regiune.nume_regiune%type;

begin

open platforme;
fetch platforme bulk collect into var_platforma;
close platforme;

for i in var_platforma.FIRST..var_platforma.LAST LOOP
dbms_output.put_line('Platforma: ' || var_platforma(i));
open c2(var_platforma(i));
loop
    fetch c2 into regiuneId;
    exit when c2%notfound;

    select nume_regiune
    into numeRegiune
    from regiune
    where id_regiune = regiuneId;

    dbms_output.put_line('Regiune: ' || numeRegiune);

    listaCompanii(numeRegiune);
    dbms_output.new_line();

end loop;
close c2;

dbms_output.new_line();

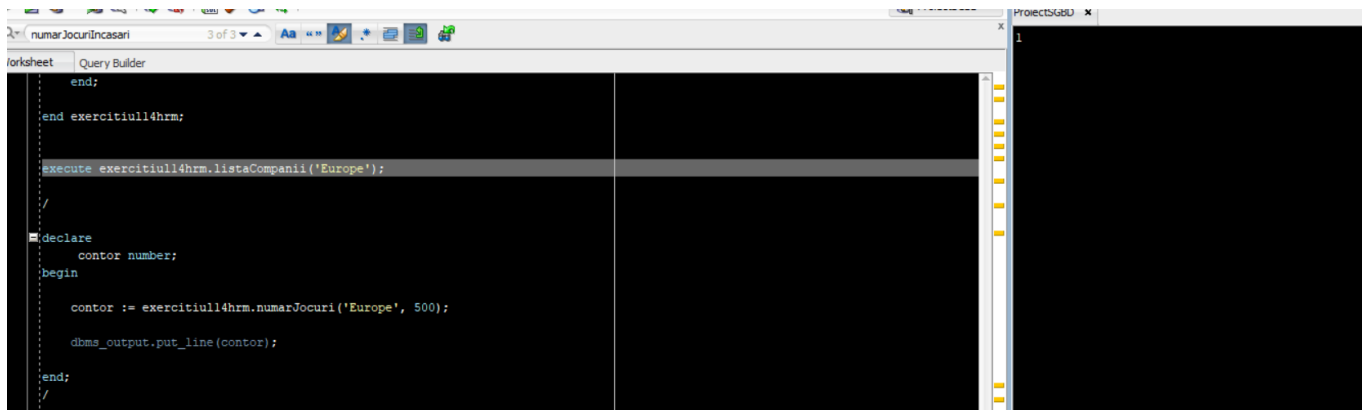
END LOOP;

end;
end exercitiul14hrm;

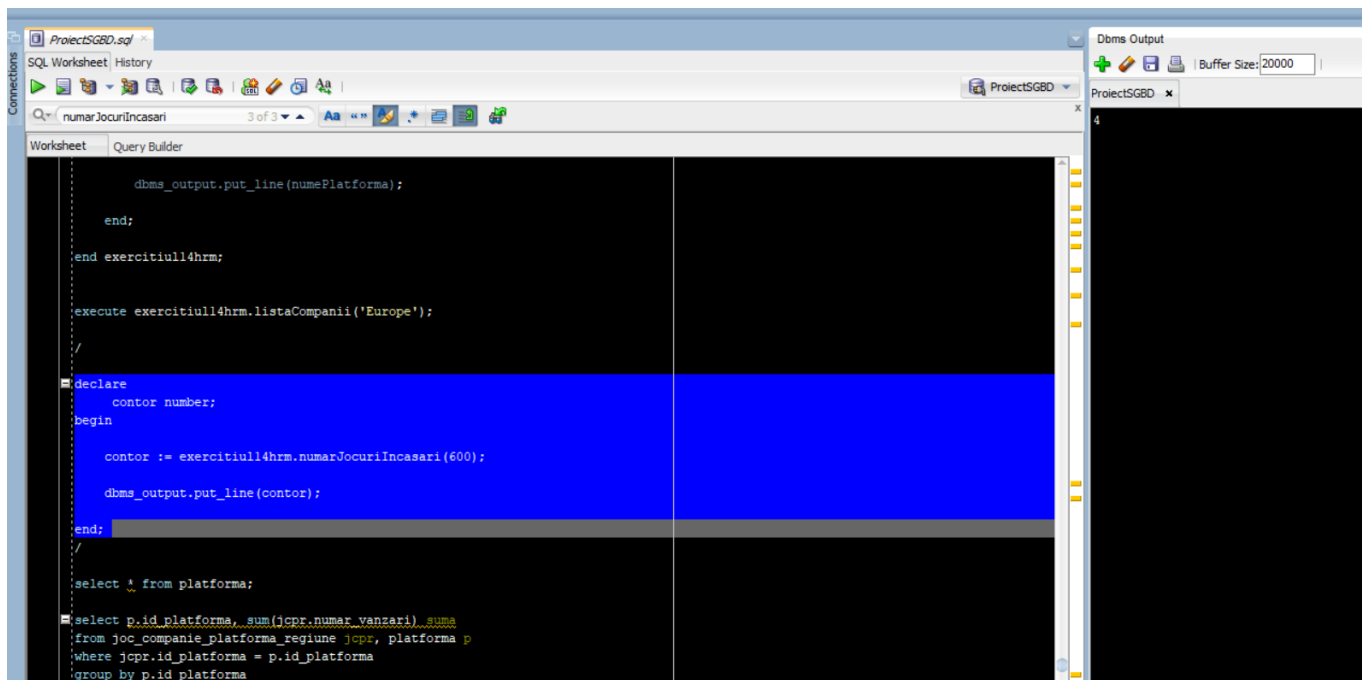
execute exercitiul14hrm.platformaVanzari;

Apel numarJocuri

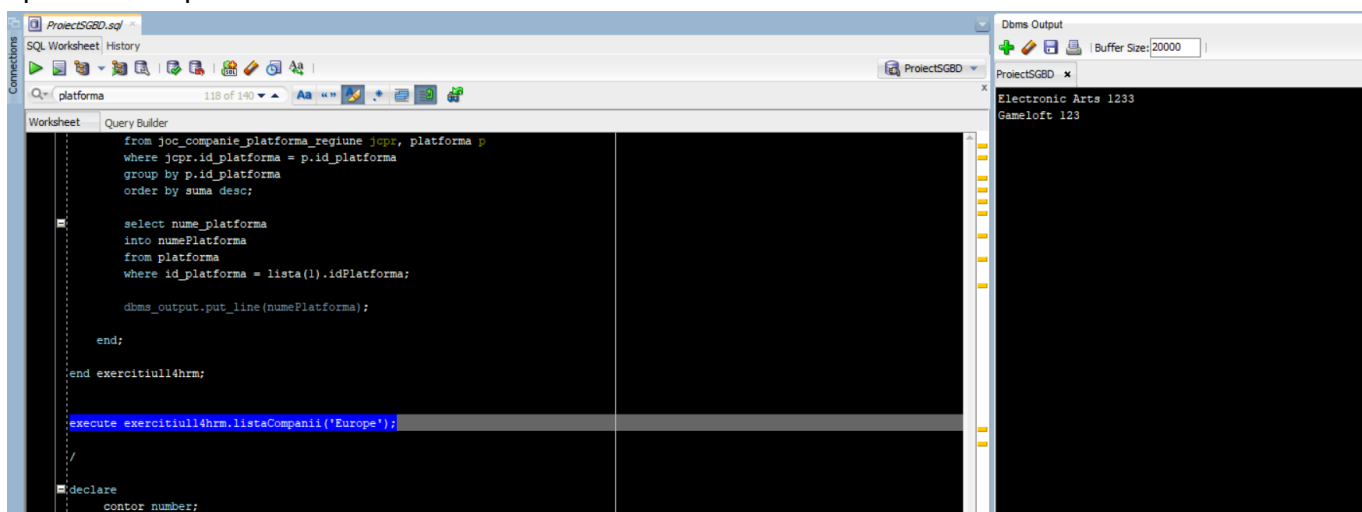
```



Apel numarJocuriIncasari



Apel listaCompanii



Apel platformaVanzari

SQL Worksheet History

ProjectSGBD

Worksheet Query Builder

```

        dbms_output.put_line('Regiune: ' || numeRegiune);

        listaCompanii(numeRegiune);
        dbms_output.new_line();

    end loop;
    close c2;

    dbms_output.new_line();

END LOOP;

end;
end exercitiul14hrm;

/

execute exercitiul14hrm.platformaVanzari;

```

Script Output x Query Result x

Task completed in 0.078 seconds

PL/SQL procedure successfully completed.

ProjectSGBD x

Platforma: PC
Regiune: Europe
Electronic Arts 1233
Gameloft 123

Regiune: South America
Mojang 1222
Riot Games 199

Platforma: Nintendo Switch
Regiune: Asia
Ubisoft 1333
Riot Games 11

Regiune: North America
Mojang 333

Platforma: PS4
Regiune: North America
Mojang 333

Platforma: Smartphones
Regiune: Asia
Ubisoft 1333
Riot Games 11

Regiune: South America
Mojang 1222
Riot Games 199

Platforma: XBOX
Regiune: Africa
Electronic Arts 1233