**GESTIUNEA ORGANIZĂRII**

**DE EVENIMENTE**

**DE TIP CONFERINȚĂ**

Armeanca Elis-Daniel

Seria 24, Grupa 242

2024-2025

Contents

[1. Prezentați pe scurt baza de date (utilitatea ei). 3](#_Toc187172118)

[2. Realizați diagrama entitate-relație (ERD): 3](#_Toc187172119)

[3.Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus 4](#_Toc187172120)

[4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, adăugând toate constrângerile de integritate necesare (chei primare, cheile externe etc). 5](#_Toc187172121)

[5. Adăugați informații coerente în tabelele create 9](#_Toc187172122)

[\*SCREENSHOTURI PENTRU EXERCIȚIILE 4 ȘI 5 15](#_Toc187172123)

[6. 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. 20](#_Toc187172124)

[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, dependent de celălalt cursor. Apelați subprogramul. 22](#_Toc187172125)

[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 create. Tratați toate excepțiile care pot apărea, incluzând excepțiile predefinite NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate. 24](#_Toc187172126)

[9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să aibă minim 2 parametri și să utilizeze într-o singură comandă SQL 5 dintre tabelele create. Definiți minim 2 excepții proprii, altele decât cele predefinite la nivel de sistem. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate. 29](#_Toc187172127)

[10.Definiți un *trigger* de tip LMD la nivel de comandă. Declanșați *trigger*-ul. 33](#_Toc187172128)

[11. Definiți un *trigger* de tip LMD la nivel de linie. Declanșați *trigger*-ul. 35](#_Toc187172129)

[12. Definiți un *trigger* de tip LDD. Declanșați *trigger*-ul. 37](#_Toc187172130)

# **Prezentați pe scurt baza de date (utilitatea ei).**

Baza de date conține informații referitoare la gestiunea organizării unor evenimente de tip conferință.

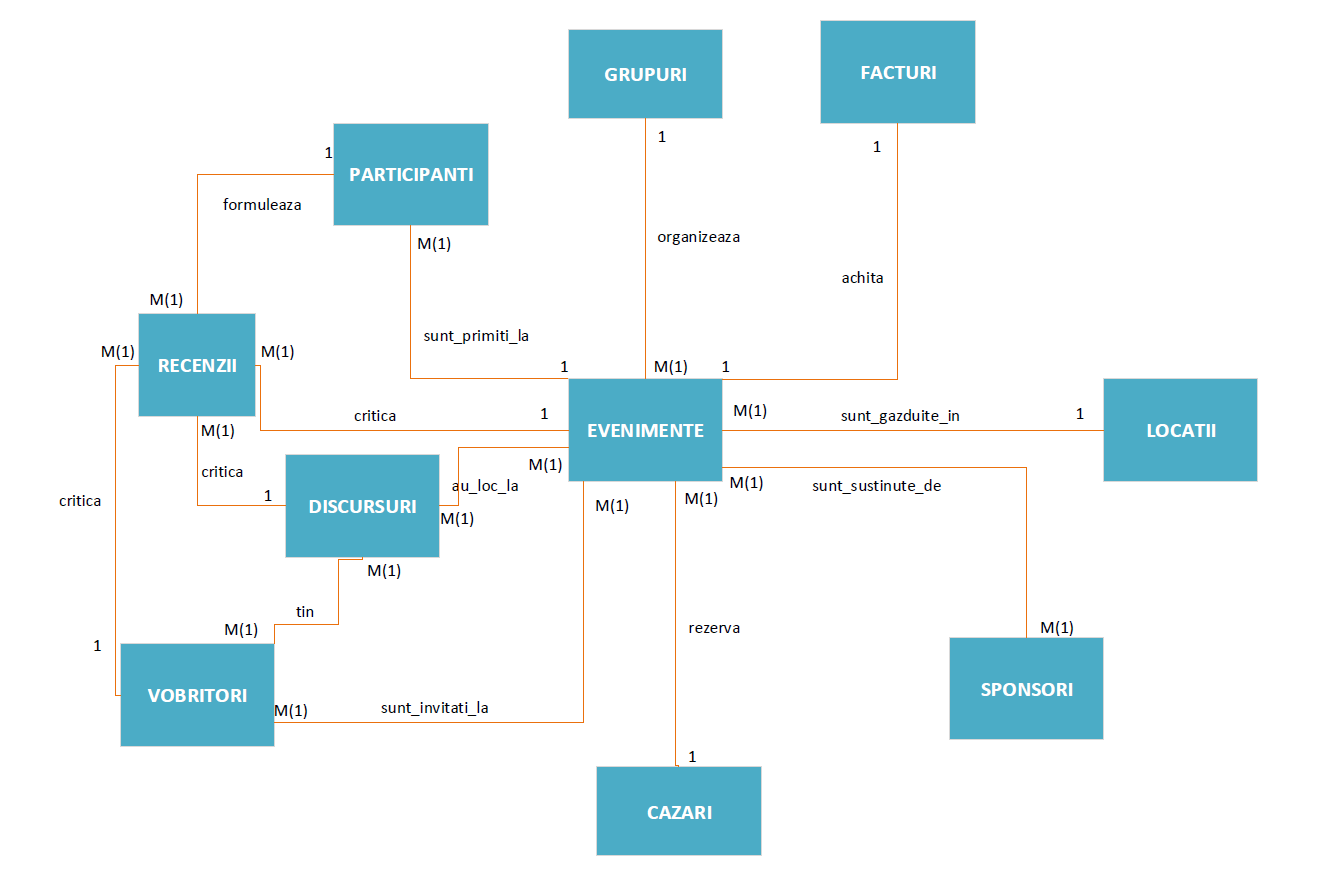
Organizarea evenimentelor se realizează astfel:

Un eveniment este găzduit de un grup, care îl planifică si coordonează. Astfel că evenimentul se desfășoară într-o locație, achită facturi, este ajutat de sponsori, rezervă cazări, primește participanți. Cam așa arată partea organizatorică a unui eveniment.

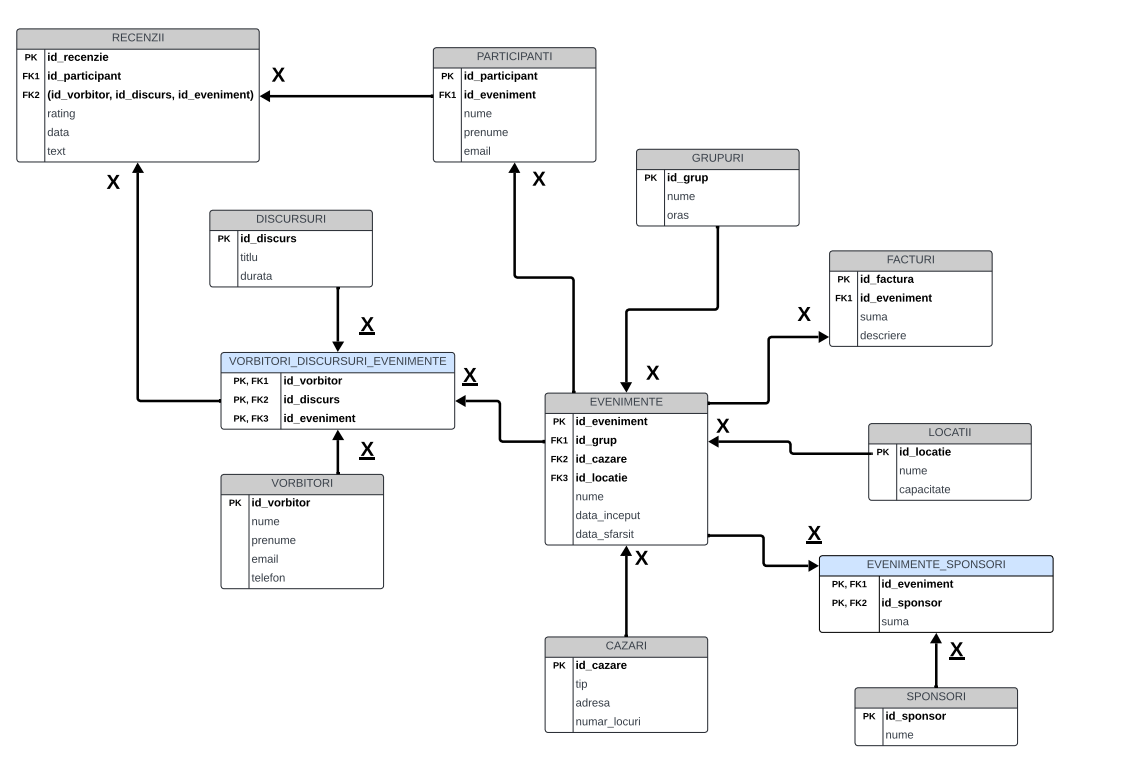
În cadrul evenimentului propriu-zis au loc mai multe discursuri, care au mai multe schițe generale centralizate, pe care fiecare vorbitor le adaptează la stilul propriu. Astfel, elementele principale ale evenimentelor sunt vorbitorii si discursurile pe care aceștia le țin. Evenimentul asigură cazarea vorbitorilor, iar participanții pot formula recenzii despre fiecare discurs în parte.

Prin urmare, baza de date îmbină atât partea „din spatele cortinei”, cât și ceea ce are loc propriu-zis „pe scenă”.

# Realizați diagrama entitate-relație (ERD):



# 3.Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus

****

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

**1)Tabela VORBITORI**

-------------VORBITORI---------------  
--VORBITORI: id\_vorbitori(PK), nume, prenume, email, telefon  
CREATE SEQUENCE vorbitori\_seq  
START WITH 1000;  
  
CREATE TABLE VORBITORI  
(  
 id\_vorbitor NUMBER(5) CONSTRAINT pkey\_vorbitor PRIMARY KEY,  
 nume VARCHAR(100) CONSTRAINT nume\_vorbitor NOT NULL,  
 prenume VARCHAR(100) CONSTRAINT prenume\_vorbitor NOT NULL,  
 email VARCHAR(100) CONSTRAINT email\_vorbitor NOT NULL,  
 telefon VARCHAR(20) CONSTRAINT telefon\_vorbitor NOT NULL  
);

**2)Tabela CAZARI**

-----------CAZARI----------  
--CAZARI: id\_cazare(PK), tip, adresa, numar\_locuri  
CREATE SEQUENCE cazari\_seq  
START WITH 2000;  
  
CREATE TABLE CAZARI  
(  
 id\_cazare NUMBER(5) CONSTRAINT pkey\_cazare PRIMARY KEY,  
 tip VARCHAR(100) CONSTRAINT tip\_cazare NOT NULL,  
 adresa VARCHAR(100) CONSTRAINT adresa\_cazare NOT NULL,  
 numar\_locuri NUMBER(4) CONSTRAINT numar\_locuri\_cazare NOT NULL  
);

**3)Tabela LOCATII**

---------LOCATII----------  
--LOCATII: id\_locatie(PK), nume, capacitate  
CREATE SEQUENCE locatii\_seq  
START WITH 3000;  
  
CREATE TABLE LOCATII  
(  
 id\_locatie NUMBER(5) CONSTRAINT pkey\_locatie PRIMARY KEY,  
 nume VARCHAR(100) CONSTRAINT nume\_locatie NOT NULL,  
 capacitate NUMBER(5) CONSTRAINT capacitate\_locatie NOT NULL  
);

**4)Tabela GRUPURI**

------GRUPURI--------  
--GRUPURI: id\_grup(PK), nume, oras  
CREATE SEQUENCE grupuri\_seq  
START WITH 4000;  
  
CREATE TABLE GRUPURI  
(  
 id\_grup NUMBER(5) CONSTRAINT pkey\_grup PRIMARY KEY,  
 nume VARCHAR(100) CONSTRAINT nume\_grup NOT NULL,  
 oras VARCHAR(100) CONSTRAINT oras\_grup NOT NULL  
);

**5)Tabela DISCURSURI**

---------DISCURSURI-------------  
--DISCURSURI: id\_discurs(PK), titlu, durata  
  
CREATE SEQUENCE discursuri\_seq  
START WITH 5000;  
  
CREATE TABLE DISCURSURI  
(  
 id\_discurs NUMBER(5) CONSTRAINT pkey\_discurs PRIMARY KEY,  
 titlu VARCHAR(255) CONSTRAINT titlu\_discurs NOT NULL,  
 durata NUMBER(4) CONSTRAINT durata\_discurs NOT NULL --minute  
);

**6)Tabela EVENIMENTE**

--------------EVENIMENTE---------------  
--EVENIMENTE: id\_eveniment(PK), id\_grup(FK1), id\_cazare(FK2),  
-- id\_locatie(FK3), nume, data\_inceput, data\_sfarsit  
  
CREATE SEQUENCE evenimente\_seq  
START WITH 6000;  
  
CREATE TABLE EVENIMENTE  
(  
 id\_eveniment NUMBER(5) CONSTRAINT pkey\_eveniment PRIMARY KEY,  
 nume VARCHAR(150) CONSTRAINT nume\_eveniment NOT NULL,  
 data\_inceput DATE CONSTRAINT data\_inceput\_eveniment NOT NULL,  
 data\_sfarsit DATE CONSTRAINT data\_sfarsit\_eveniment NOT NULL,  
 id\_grup NUMBER(5),  
 CONSTRAINT fk\_ev\_grup FOREIGN KEY(id\_grup) REFERENCES GRUPURI(id\_grup),  
 id\_cazare NUMBER(5),  
 CONSTRAINT fk\_ev\_cazare FOREIGN KEY(id\_cazare) REFERENCES CAZARI(id\_cazare)  
);  
  
ALTER TABLE EVENIMENTE  
ADD id\_locatie NUMBER(5);  
ALTER TABLE EVENIMENTE  
ADD CONSTRAINT fk\_ev\_locatie FOREIGN KEY (id\_locatie) REFERENCES LOCATII(id\_locatie);

**7)Tabela SPONSORI**

-------SPONSORI-----------  
--SPONSORI: id\_sponsor(PK), id\_eveniment(FK1), nume  
  
CREATE SEQUENCE sponsori\_seq  
START WITH 7000;  
  
CREATE TABLE SPONSORI  
(  
 id\_sponsor NUMBER(5) CONSTRAINT pkey\_sponsor PRIMARY KEY,  
 nume VARCHAR(150) CONSTRAINT nume\_sponsor NOT NULL  
);

**8)Tabela FACTURI**

---------------FACTURI----------------  
--FACTURI: id\_factura(PK), id\_eveniment(FK1), suma, descriere  
  
CREATE SEQUENCE facturi\_seq   
START WITH 8000;  
  
CREATE TABLE FACTURI  
(  
 id\_factura NUMBER(5) CONSTRAINT pkey\_factura PRIMARY KEY,  
 suma NUMBER(7) CONSTRAINT suma\_factura NOT NULL,  
 descriere VARCHAR(255) CONSTRAINT descriere\_factura NOT NULL,  
 id\_eveniment NUMBER(5),  
 CONSTRAINT fk\_fac\_eveniment FOREIGN KEY(id\_eveniment) REFERENCES EVENIMENTE(id\_eveniment)  
);

**9)Tabela PARTICIPANTI**

-----PARTICIPANTI------  
--PARTICIPANTI: id\_participant (PK), id\_eveniment(FK1), nume, prenume, email  
  
CREATE SEQUENCE participanti\_seq  
START WITH 9000;  
  
CREATE TABLE PARTICIPANTI  
(  
 id\_participant NUMBER(5) CONSTRAINT pkey\_participant PRIMARY KEY,  
 nume VARCHAR(150) CONSTRAINT nume\_participant NOT NULL,  
 prenume VARCHAR(150) CONSTRAINT prenume\_participant NOT NULL,  
 email VARCHAR(150) CONSTRAINT email\_participant NOT NULL,  
 id\_eveniment NUMBER(5),  
 CONSTRAINT fk\_part\_eveniment FOREIGN KEY(id\_eveniment) REFERENCES EVENIMENTE(id\_eveniment)  
);

**10)Tabela VORBITORI\_DISCURSURI\_EVENIMENTE**

-----VORBITORI\_DISCURSURI\_EVENIMENTE---------------  
-------VORBITORI\_DISCURSURI\_EVENIMENTE: id\_vorbitor(PK, FK1), id\_discurs(PK, FK2), id\_eveniment(PK, FK3)  
  
CREATE TABLE VORBITORI\_DISCURSURI\_EVENIMENTE  
(  
 id\_vorbitor NUMBER(5) CONSTRAINT pk\_c\_vorbitor REFERENCES VORBITORI(id\_vorbitor),  
 id\_discurs NUMBER(5) CONSTRAINT pk\_c\_discurs REFERENCES DISCURSURI(id\_discurs),  
 id\_eveniment NUMBER(5) CONSTRAINT pk\_c\_eveniment REFERENCES EVENIMENTE(id\_eveniment),  
 CONSTRAINT pk\_compus\_vde PRIMARY KEY (id\_vorbitor,id\_discurs,id\_eveniment )  
);

**11)Tabela RECENZII**

----------- RECENZII-----------------  
--RECENZII: id\_recenzie (PK), id\_participant(FK1),   
-- (id\_vorbitor, id\_discurs, id\_eveniment) (FK2),   
-- rating, data, text  
  
  
CREATE SEQUENCE recenzii\_seq  
START WITH 10000;  
  
  
CREATE TABLE RECENZII  
(  
 id\_recenzie NUMBER(5) CONSTRAINT pk\_recenzie PRIMARY KEY,  
 rating NUMBER (2) CONSTRAINT rating\_recenzie NOT NULL,  
 data DATE CONSTRAINT data\_recenzie NOT NULL,  
 text VARCHAR(255) CONSTRAINT text\_recenzie NOT NULL,  
 id\_vorbitor NUMBER(5),  
 id\_discurs NUMBER(5),  
 id\_eveniment NUMBER(5),  
 CONSTRAINT fk\_recenzie\_vde FOREIGN KEY (id\_vorbitor, id\_discurs, id\_eveniment)  
 REFERENCES VORBITORI\_DISCURSURI\_EVENIMENTE (id\_vorbitor, id\_discurs, id\_eveniment),  
 id\_participant NUMBER(5),  
 CONSTRAINT fk\_recenzie\_part FOREIGN KEY(id\_participant) REFERENCES PARTICIPANTI(id\_participant)  
);

**12)Tabela EVENIMENTE\_SPONSORI**

------EVENIMENTE\_SPONSORI--------  
--EVENIMENTE\_SPONSORI: id\_eveniment(PK, FK1), id\_sponsor(PK, FK2), suma  
CREATE TABLE EVENIMENTE\_SPONSORI  
(  
 id\_eveniment NUMBER(5) CONSTRAINT pk\_es\_eveniment REFERENCES EVENIMENTE(id\_eveniment),  
 id\_sponsor NUMBER(5) CONSTRAINT pk\_es\_sponsor REFERENCES SPONSORI(id\_sponsor),  
 CONSTRAINT pk\_compus\_es PRIMARY KEY (id\_eveniment,id\_sponsor ),  
 suma NUMBER(7) CONSTRAINT es\_suma NOT NULL  
);

# 5. Adăugați informații coerente în tabelele create

**1)Tabela VORBITORI**

INSERT INTO VORBITORI  
VALUES(vorbitori\_seq.nextval, 'Pana', 'Marian', 'panamarian@gmail.com', '+40734543127');  
  
INSERT INTO VORBITORI  
VALUES(vorbitori\_seq.nextval, 'Armeanca', 'Elis', 'ellisarm@gmail.com', '+40784065701');  
  
INSERT INTO VORBITORI  
VALUES(vorbitori\_seq.nextval, 'Uta', 'Catalin', 'catau@gmail.com', '+40738885577');  
  
INSERT INTO VORBITORI  
VALUES(vorbitori\_seq.nextval, 'Balea', 'Calin', 'balincalea@gmail.com', '+40766909023');  
  
INSERT INTO VORBITORI  
VALUES(vorbitori\_seq.nextval, 'Coman', 'Florinel', 'fcoman7@gmail.com', '+40734999777');  
  
commit;

**2)Tabela CAZARI**

INSERT INTO CAZARI  
VALUES (cazari\_seq.nextval, 'Hotel', 'Nicolae Balcescu 4, Bucuresti 1, Romania', 3200);  
  
INSERT INTO CAZARI  
VALUES (cazari\_seq.nextval, 'Vila', 'Intrarea Sectorului 8, Bucuresti 3, Romania', 16);  
  
INSERT INTO CAZARI  
VALUES (cazari\_seq.nextval, 'Apartament', 'Bulevardul Eroilor 15, bloc 35, ap.98, Cluj-Napoca , Romania', 8);  
  
INSERT INTO CAZARI  
VALUES (cazari\_seq.nextval, 'Vila', 'Strada Victoriei 23, Timisoara , Romania', 10);  
  
INSERT INTO CAZARI  
VALUES (cazari\_seq.nextval, 'Hotel', 'Strada Stadionului 1, Cluj-Napoca, Romania', 3200);  
  
commit;

**3)Tabela LOCATII**

INSERT INTO LOCATII  
VALUES (locatii\_seq.nextval, 'Romexpo, Bucuresti', 10000 );  
  
INSERT INTO LOCATII  
VALUES (locatii\_seq.nextval, 'Aula Magna UPB, Bucuresti', 1000 );  
  
INSERT INTO LOCATII  
VALUES (locatii\_seq.nextval, 'BT Arena, Cluj-Napoca', 10000 );  
  
INSERT INTO LOCATII  
VALUES (locatii\_seq.nextval, 'Auditorium Maximum, Cluj-Napoca', 800 );  
  
INSERT INTO LOCATII  
VALUES (locatii\_seq.nextval, 'Sala Polivalenta, Timisoara', 8000 );  
  
commit;

**4)Tabela GRUPURI**

INSERT INTO GRUPURI  
VALUES (grupuri\_seq.nextval, 'Bucuresti Tech Talks', 'Bucuresti' );  
  
INSERT INTO GRUPURI  
VALUES (grupuri\_seq.nextval, 'Timisoara Dev Community', 'Timisoara' );  
  
INSERT INTO GRUPURI  
VALUES (grupuri\_seq.nextval, 'Bucuresti Ted Talks', 'Bucuresti' );  
  
INSERT INTO GRUPURI  
VALUES (grupuri\_seq.nextval, 'Open-Mind Cluj', 'Cluj-Napoca' );  
  
INSERT INTO GRUPURI  
VALUES (grupuri\_seq.nextval, 'Growup Meetings Cluj-Napoca', 'Cluj-Napoca' );  
  
commit;

**5)Tabela DISCURSURI**

INSERT INTO DISCURSURI  
VALUES (discursuri\_seq.nextval, 'Cum sa imi gestionez timpul?', 45 );  
  
INSERT INTO DISCURSURI  
VALUES (discursuri\_seq.nextval, 'Cum raman up-to-date cu industria?', 30 );  
  
INSERT INTO DISCURSURI  
VALUES (discursuri\_seq.nextval, 'Gasirea satisfactiei in munca. Cum?', 50 );  
  
INSERT INTO DISCURSURI  
VALUES (discursuri\_seq.nextval, 'Tehnici de dezvoltare web', 60 );  
  
INSERT INTO DISCURSURI  
VALUES (discursuri\_seq.nextval, 'Sa ne propunem obiective personale!', 45 );  
  
commit;

**6)Tabela EVENIMENTE**

INSERT INTO EVENIMENTE  
VALUES (evenimente\_seq.nextval, 'Conferinta de TECH #1 aprilie 2025',  
 *TO\_DATE*('15-04-2025','dd-mm-yyyy'),  
 *TO\_DATE*('15-04-2025','dd-mm-yyyy'), 4000, 2000 );  
  
INSERT INTO EVENIMENTE  
VALUES (evenimente\_seq.nextval, 'Conferinta anuala DEV 2025 ',  
 *TO\_DATE*('25-04-2025','dd-mm-yyyy'),  
 *TO\_DATE*('26-04-2025','dd-mm-yyyy'), 4001, 2003 );  
  
INSERT INTO EVENIMENTE  
VALUES (evenimente\_seq.nextval, 'TED TALKS SESSION#1 2025',  
 *TO\_DATE*('01-02-2025','dd-mm-yyyy'),  
 *TO\_DATE*('01-02-2025','dd-mm-yyyy'), 4002, 2001 );  
  
INSERT INTO EVENIMENTE  
VALUES (evenimente\_seq.nextval, 'Training dezvoltare personala martie 2025',  
 *TO\_DATE*('15-03-2025','dd-mm-yyyy'),  
 *TO\_DATE*('15-03-2025','dd-mm-yyyy'), 4003, 2002 );  
  
INSERT INTO EVENIMENTE  
VALUES (evenimente\_seq.nextval, 'Seminar arta conversatiei',  
 *TO\_DATE*('02-05-2025','dd-mm-yyyy'),  
 *TO\_DATE*('04-05-2025','dd-mm-yyyy'), 4004, 2004 );  
  
UPDATE EVENIMENTE  
SET id\_locatie = 3000  
WHERE id\_eveniment =6000;  
  
UPDATE EVENIMENTE  
SET id\_locatie = 3004  
WHERE id\_eveniment =6001;  
  
  
UPDATE EVENIMENTE  
SET id\_locatie = 3001  
WHERE id\_eveniment =6002;  
  
  
UPDATE EVENIMENTE  
SET id\_locatie = 3002  
WHERE id\_eveniment =6003;  
  
  
UPDATE EVENIMENTE  
SET id\_locatie = 3003  
WHERE id\_eveniment =6004;  
commit;

**7)Tabela SPONSORI**

INSERT INTO SPONSORI  
VALUES (sponsori\_seq.nextval, 'Endava' );  
  
INSERT INTO SPONSORI  
VALUES (sponsori\_seq.nextval, 'KPMG' );  
  
INSERT INTO SPONSORI  
VALUES (sponsori\_seq.nextval, 'MIcrosoft' );  
  
INSERT INTO SPONSORI  
VALUES (sponsori\_seq.nextval, 'Google' );  
  
INSERT INTO SPONSORI  
VALUES (sponsori\_seq.nextval, 'ING Bank' );  
  
commit;

**8)Tabela FACTURI**

INSERT INTO FACTURI  
VALUES (facturi\_seq.nextval, 65000, 'Chirie locatie', 6000);  
  
INSERT INTO FACTURI  
VALUES (facturi\_seq.nextval, 6500, 'Utilitati locatie', 6001);  
  
INSERT INTO FACTURI  
VALUES (facturi\_seq.nextval, 10000, 'Plata cazare', 6002);  
  
INSERT INTO FACTURI  
VALUES (facturi\_seq.nextval, 80000, 'Chirie locatie', 6003);  
  
INSERT INTO FACTURI  
VALUES (facturi\_seq.nextval, 99990, 'Chirie echipamente audio-video', 6004);  
  
commit;

**9)Tabela PARTICIPANTI**

INSERT INTO PARTICIPANTI  
VALUES (participanti\_seq.nextval, 'Armeanca', 'Daniela', 'dana\_arm@yahoo.com', 6000);  
  
INSERT INTO PARTICIPANTI  
VALUES (participanti\_seq.nextval, 'Armeanca', 'Daniela', 'dana\_arm@yahoo.com', 6000);  
  
INSERT INTO PARTICIPANTI  
VALUES (participanti\_seq.nextval, 'Armeanca', 'Daniel', 'dani\_arm@yahoo.com', 6001);  
  
INSERT INTO PARTICIPANTI  
VALUES (participanti\_seq.nextval, 'Ion', 'Stefan', 'ionstef@yahoo.com', 6002);  
  
INSERT INTO PARTICIPANTI  
VALUES (participanti\_seq.nextval, 'Ion', 'Sofia', 'sofiii@gmail.com', 6003);  
  
INSERT INTO PARTICIPANTI  
VALUES (participanti\_seq.nextval, 'Dumitrache', 'Marius', 'dumyy80@gmail.com', 6004);  
  
commit;

**10)Tabela VORBITORI\_DISCURSURI\_EVENIMENTE**

INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1000, 5000, 6000);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1001, 5001, 6001);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1002, 5002, 6002);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1003, 5003, 6003);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1004, 5004, 6004);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1000, 5000, 6004);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1001, 5001, 6003);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1002, 5003, 6002);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1003, 5003, 6001);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1004, 5004, 6000);  
  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE  
values (1001, 5004, 6002);  
  
commit;

**11)Tabela RECENZII**

INSERT INTO RECENZII  
VALUES(recenzii\_seq.nextval, 10, *TO\_DATE*('16-04-2025','dd-mm-yyyy'),  
 'Subiectele bine abordate.', 1000, 5000, 6000, 9001 );  
  
INSERT INTO RECENZII  
VALUES(recenzii\_seq.nextval, 10, *TO\_DATE*('26-04-2025','dd-mm-yyyy'),  
 'SUPERB.', 1001, 5001, 6001, 9002 );  
   
INSERT INTO RECENZII  
VALUES(recenzii\_seq.nextval, 6, *TO\_DATE*('02-02-2025','dd-mm-yyyy'),  
 'MODEST...', 1002, 5002, 6002, 9003 );  
  
INSERT INTO RECENZII  
VALUES(recenzii\_seq.nextval, 9, *TO\_DATE*('16-03-2025','dd-mm-yyyy'),  
 'Extraordinar!', 1003, 5003, 6003, 9004 );  
  
INSERT INTO RECENZII  
VALUES(recenzii\_seq.nextval, 8, *TO\_DATE*('05-05-2025','dd-mm-yyyy'),  
 'Mi-a placut mult tema!!', 1004, 5004, 6004, 9005 );  
  
commit;

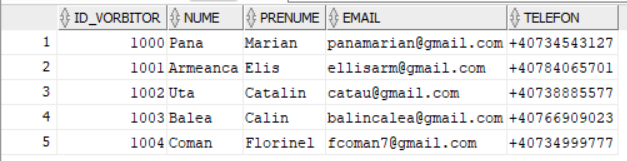
**12)Tabela EVENIMENTE\_SPONSORI**

INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6000, 7000, 12300);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6000, 7001, 120000);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6001, 7001, 12600);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6001, 7002, 62300);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6002, 7002, 1000000);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6002, 7003, 990000);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6003, 7003, 22300);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6003, 7004, 22222);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6004, 7004, 32300);  
  
INSERT INTO EVENIMENTE\_SPONSORI  
VALUES(6004, 7000, 33333);

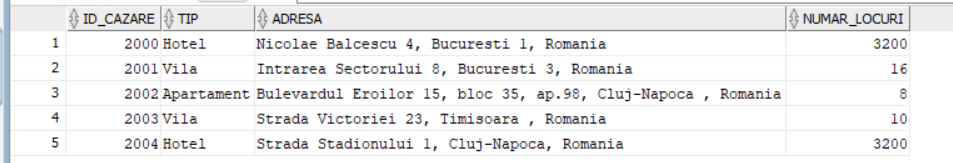
commit;

## \*SCREENSHOTURI PENTRU EXERCIȚIILE 4 ȘI 5

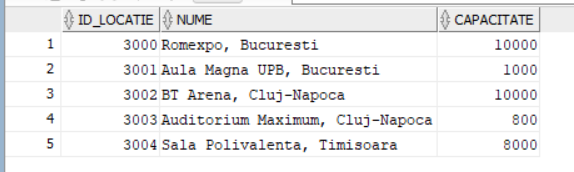
**1) SS Tabela VORBITORI**

****

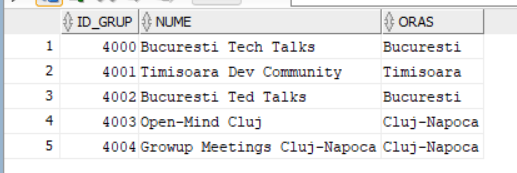
**2) SS Tabela CAZARI**

****

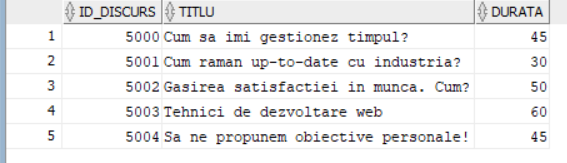
**3) SS Tabela LOCATII**



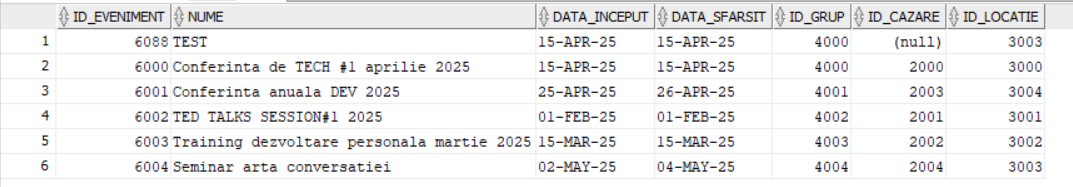
**4) SS Tabela GRUPURI**

****

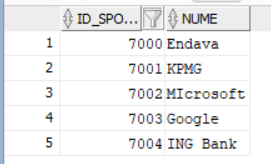
**5) SS Tabela DISCURSURI**



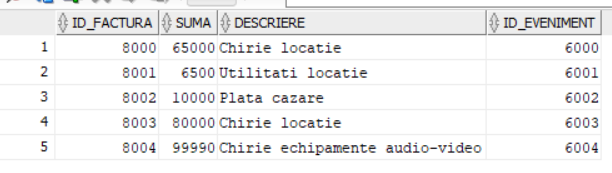
**6) SS Tabela EVENIMENTE**



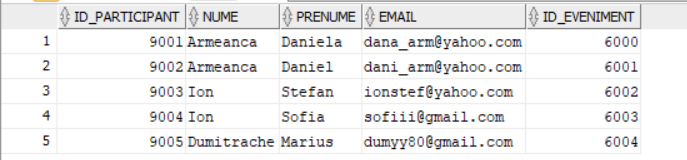
**7) SS Tabela SPONSORI**



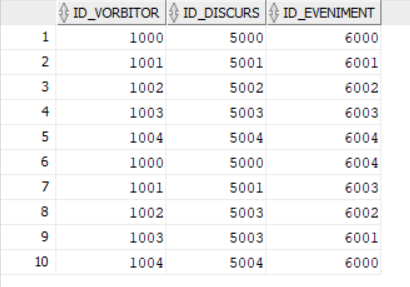
**8) SS Tabela FACTURI**



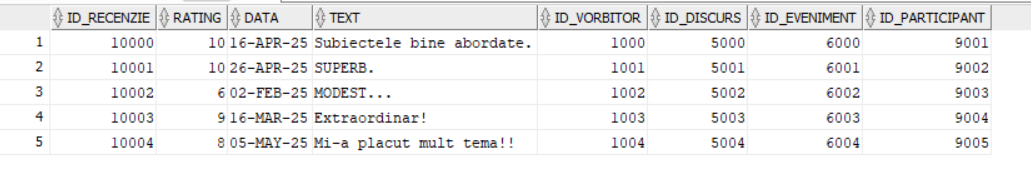
**9) SS Tabela PARTICIPANTI**



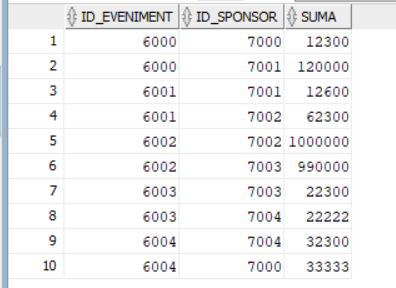
**10) SS Tabela VORBITORI\_DISCURSURI\_EVENIMENTE**



**11) SS Tabela RECENZII**



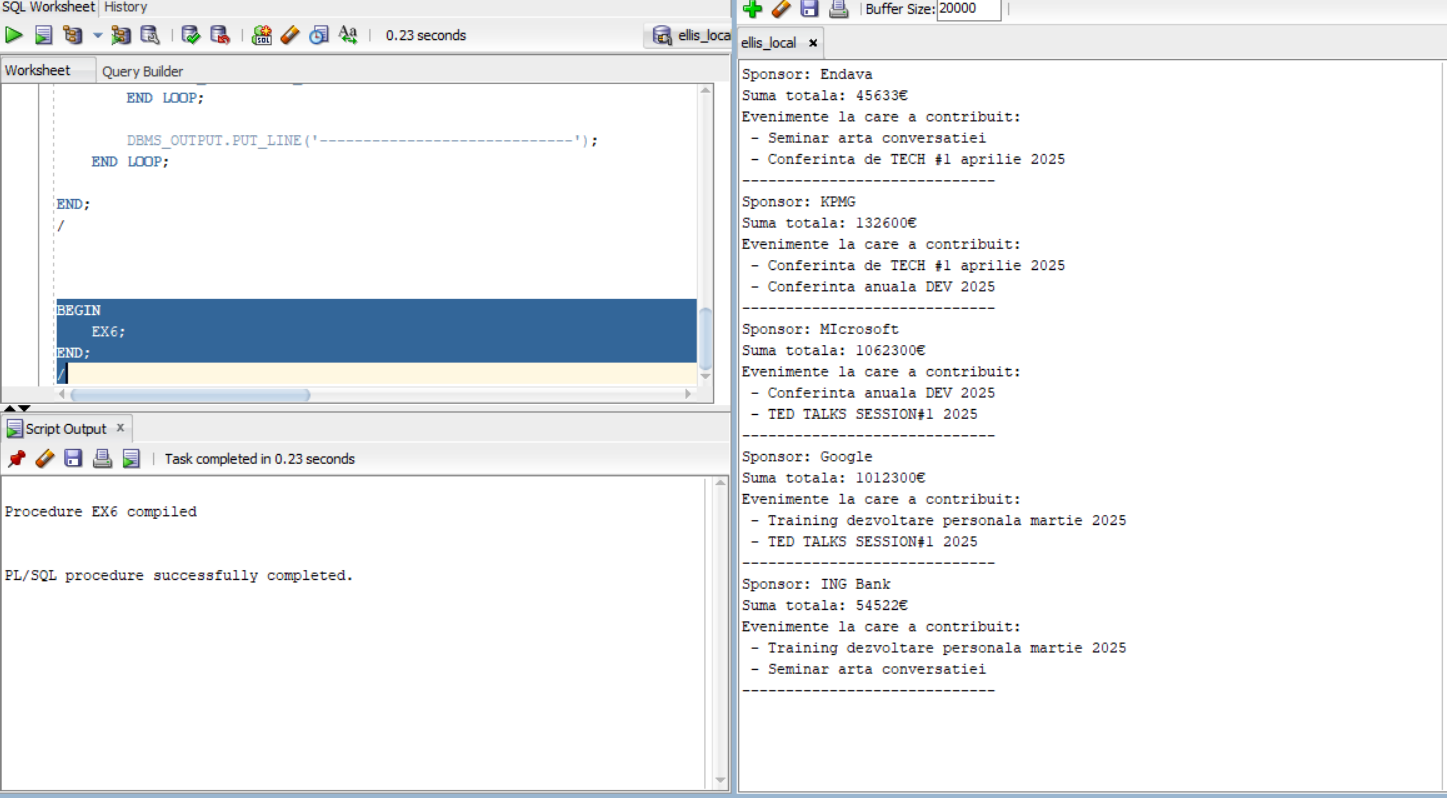
**12)SS Tabela EVENIMENTE\_SPONSORI**



# 6. 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.

**CERINȚA:** Pentru fiecare sponsor calclueaza suma totala investita in evenimente de-a lungul timpului. Dupa aceea, afiseaza fiecare sponsor, suma totala si evenimentele unde a investit (maxim 5).

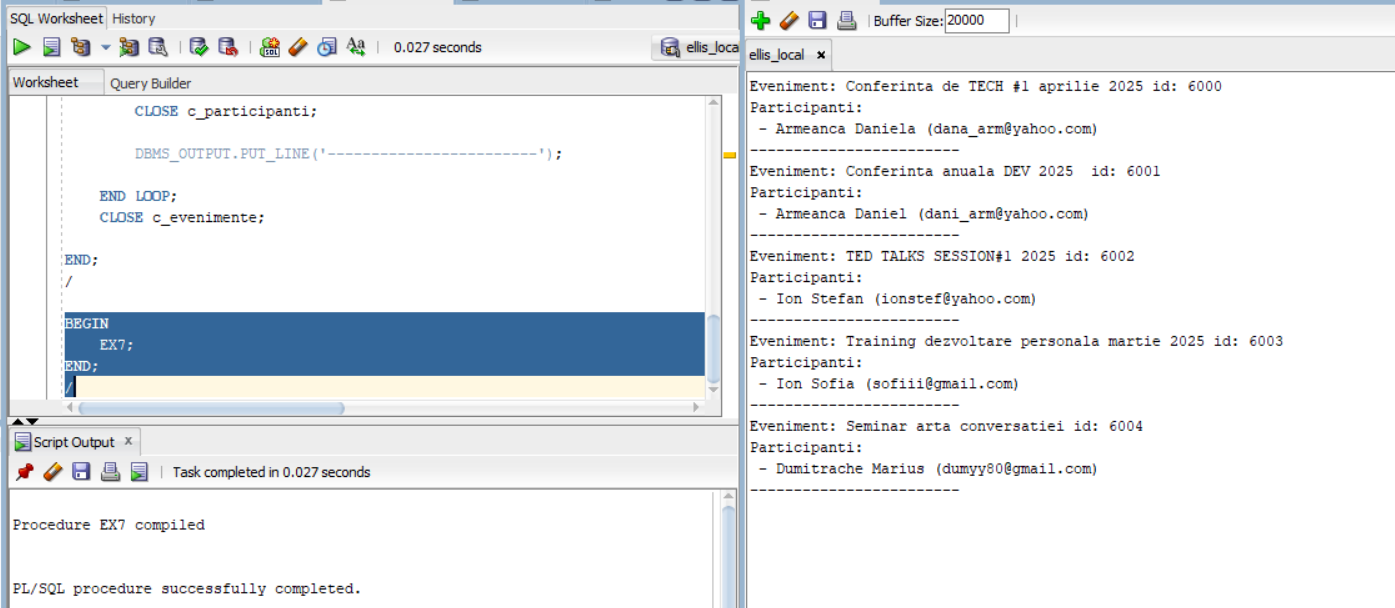
CREATE OR REPLACE PROCEDURE *EX6* IS  
   
 --definim colectiile  
 TYPE tabel\_sponsori IS TABLE OF VARCHAR2(150); -- nested table  
 TYPE sume\_sponsori IS TABLE OF NUMBER INDEX BY VARCHAR2(150); -- tabel indexat pe baza string  
 TYPE evenimente\_sponsorizate IS VARRAY(5) OF VARCHAR2(150); -- vector  
   
 sponsori tabel\_sponsori := tabel\_sponsori();  
 sumes sume\_sponsori;  
 ev evenimente\_sponsorizate :=evenimente\_sponsorizate();  
  
BEGIN  
 SELECT s.nume  
 BULK COLLECT INTO sponsori  
 FROM SPONSORI s; --SPONSORII  
   
 FOR i in 1 .. sponsori.COUNT LOOP  
 SELECT *SUM*(es.suma)  
 INTO sumes(sponsori(i))  
 FROM EVENIMENTE\_SPONSORI es  
 JOIN SPONSORI s ON es.id\_sponsor = s.id\_sponsor  
 WHERE s.nume = sponsori(i);  
 END LOOP; --sumele  
   
 FOR i in 1 .. sponsori.COUNT LOOP  
   
 ev :=evenimente\_sponsorizate();  
   
 SELECT e.nume  
 BULK COLLECT INTO ev  
 FROM EVENIMENTE e  
 JOIN EVENIMENTE\_SPONSORI es ON e.id\_eveniment = es.id\_eveniment  
 JOIN SPONSORI s ON es.id\_sponsor = s.id\_sponsor  
 WHERE s.nume = sponsori(i)  
 FETCH FIRST 5 ROWS ONLY;  
   
 DBMS\_OUTPUT.*PUT\_LINE*('Sponsor: ' || sponsori(i));  
 DBMS\_OUTPUT.*PUT\_LINE*('Suma totala: ' || sumes(sponsori(i)) || '€');  
 DBMS\_OUTPUT.*PUT\_LINE*('Evenimente la care a contribuit: ');  
   
 FOR j IN 1 .. ev.COUNT LOOP  
 DBMS\_OUTPUT.*PUT\_LINE*(' - ' || ev(j));  
 END LOOP;  
   
 DBMS\_OUTPUT.*PUT\_LINE*('-----------------------------');  
 END LOOP;  
   
END;  
/  
  
   
   
BEGIN  
 *EX6*;  
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, dependent de celălalt cursor. Apelați subprogramul.

**CERINȚA:** Pentru fiecare eveniment, afisati toti participantii care au fost prezenti.

CREATE OR REPLACE PROCEDURE *EX7* IS  
   
   
 --cursor clasic  
 CURSOR c\_evenimente IS  
 SELECT id\_eveniment, nume  
 FROM EVENIMENTE;  
   
 --cursor parametrizat  
 CURSOR c\_participanti(ev NUMBER) IS  
 SELECT nume, prenume, email  
 FROM PARTICIPANTI p  
 WHERE ev = p.id\_eveniment;  
   
 v\_id\_eveniment EVENIMENTE.id\_eveniment%TYPE;  
 v\_nume\_ev EVENIMENTE.nume%TYPE;  
   
 v\_nume PARTICIPANTI.nume%TYPE;  
 v\_prenume PARTICIPANTI.prenume%TYPE;  
 v\_email PARTICIPANTI.email%TYPE;  
   
BEGIN  
 OPEN c\_evenimente;  
 LOOP  
 FETCH c\_evenimente INTO v\_id\_eveniment, v\_nume\_ev;  
 EXIT WHEN c\_evenimente%NOTFOUND;  
   
 DBMS\_OUTPUT.*PUT\_LINE*('Eveniment: ' || v\_nume\_ev || ' id: ' || v\_id\_eveniment);  
 DBMS\_OUTPUT.*PUT\_LINE*('Participanti: ');  
   
 OPEN c\_participanti(v\_id\_eveniment);  
 LOOP  
 FETCH c\_participanti INTO v\_nume, v\_prenume, v\_email;  
 EXIT when c\_participanti %NOTFOUND;  
   
 DBMS\_OUTPUT.*PUT\_LINE*(' - ' || v\_nume || ' ' || v\_prenume || ' (' || v\_email || ')');  
 END LOOP;  
 CLOSE c\_participanti;  
   
 DBMS\_OUTPUT.*PUT\_LINE*('------------------------');  
   
 END LOOP;  
 CLOSE c\_evenimente;  
  
END;  
/  
  
BEGIN  
 *EX7*;  
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 create. Tratați toate excepțiile care pot apărea, incluzând excepțiile predefinite NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

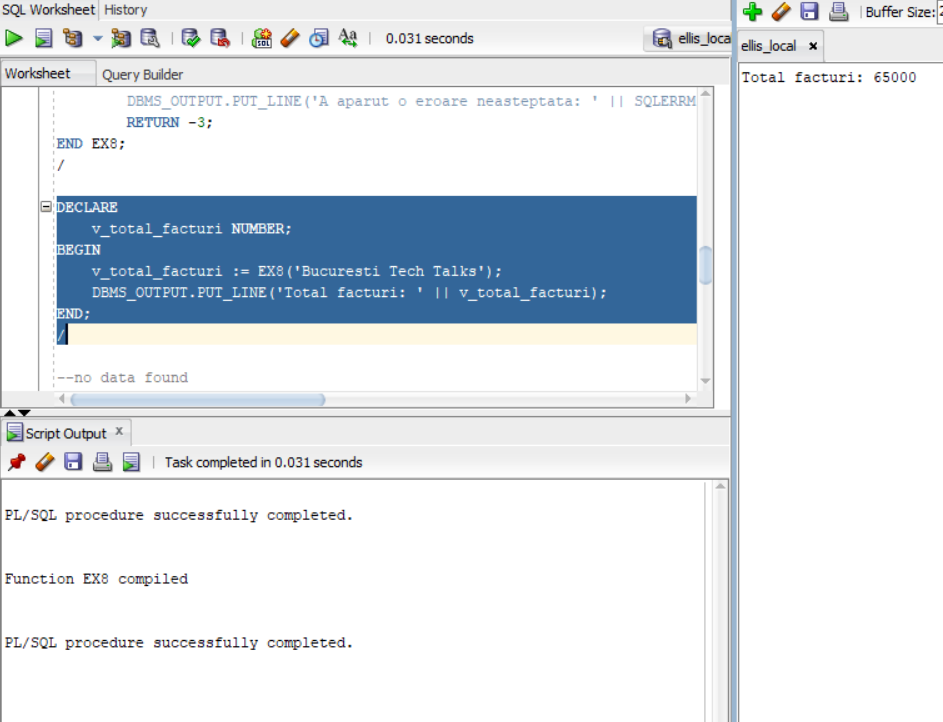
**CERINȚA:** Calculati suma totala a facturilor asociate evenimentelor organizate de un anumit grup, al carui nume este transmis ca parametru.

CREATE OR REPLACE FUNCTION *EX8*(  
 nume\_grup IN VARCHAR2  
) RETURN NUMBER IS  
 v\_test NUMBER; -- pt existenta grupului  
 suma\_facturi NUMBER;  
   
 no\_facturi\_exception EXCEPTION; -- exceptie proprie  
   
BEGIN  
   
 --verificam existenta grupului  
 SELECT g.id\_grup  
 INTO v\_test  
 FROM GRUPURI g  
 WHERE g.nume = nume\_grup;  
   
 SELECT *SUM*(f.suma)  
 INTO suma\_facturi  
 FROM FACTURI f  
 JOIN EVENIMENTE e ON f.id\_eveniment = e.id\_eveniment  
 JOIN GRUPURI g ON e.id\_grup = g.id\_grup  
 WHERE g.nume = nume\_grup;  
   
 IF suma\_facturi IS NULL THEN  
 RAISE no\_facturi\_exception;  
 END IF;  
   
 RETURN suma\_facturi;  
  
EXCEPTION  
  
 -- nu exista grupul  
 WHEN NO\_DATA\_FOUND THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('Grupul cu numele ' || nume\_grup || ' nu exista ');  
 RETURN -1;  
   
 WHEN TOO\_MANY\_ROWS THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('Exista mai multe grupuri cu numele '|| nume\_grup);  
 RETURN -2;  
   
 WHEN no\_facturi\_exception THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('Grupul ' || nume\_grup || ' nu are facturi asociate.');  
 RETURN 0;  
   
 WHEN OTHERS THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('A aparut o eroare neasteptata: ' || *SQLERRM*);  
 RETURN -3;  
END *EX8*;  
/

CAZURI RULARE:

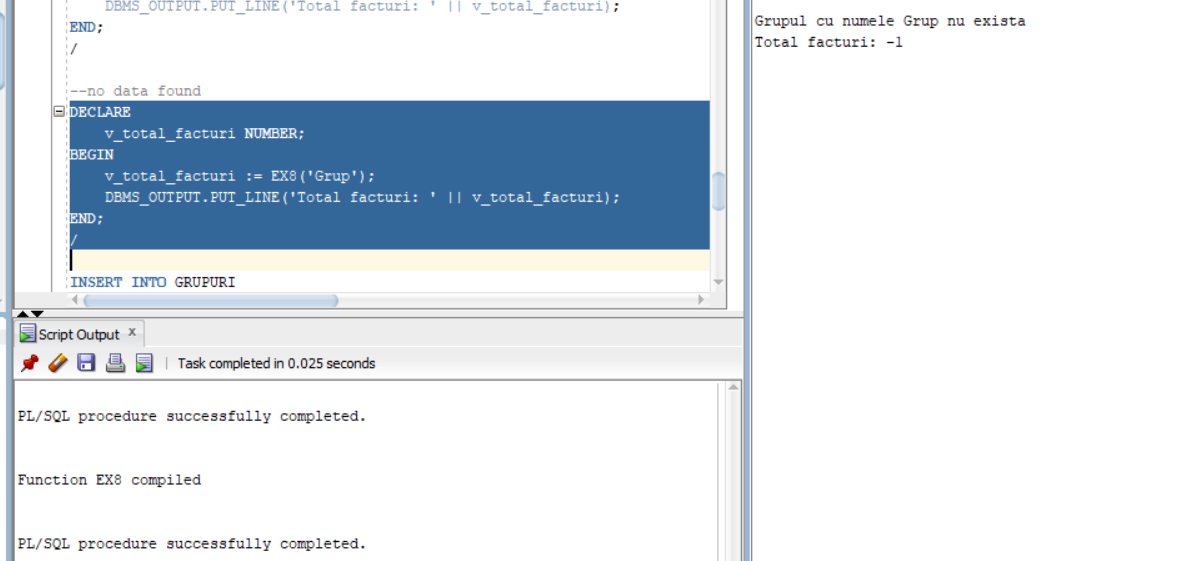
A)

DECLARE  
 v\_total\_facturi NUMBER;  
BEGIN  
 v\_total\_facturi := *EX8*('Bucuresti Tech Talks');  
 DBMS\_OUTPUT.*PUT\_LINE*('Total facturi: ' || v\_total\_facturi);  
END;  
/



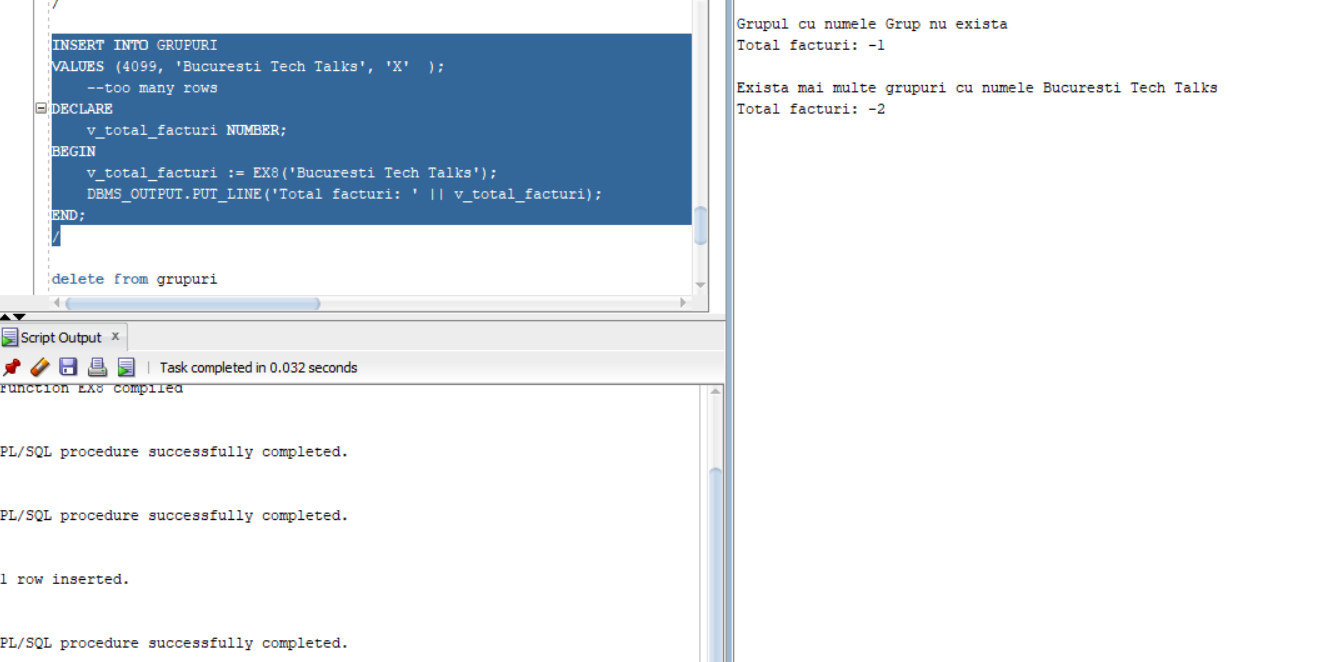
B)

DECLARE  
 v\_total\_facturi NUMBER;  
BEGIN  
 v\_total\_facturi := *EX8*('Grup');  
 DBMS\_OUTPUT.*PUT\_LINE*('Total facturi: ' || v\_total\_facturi);  
END;  
/



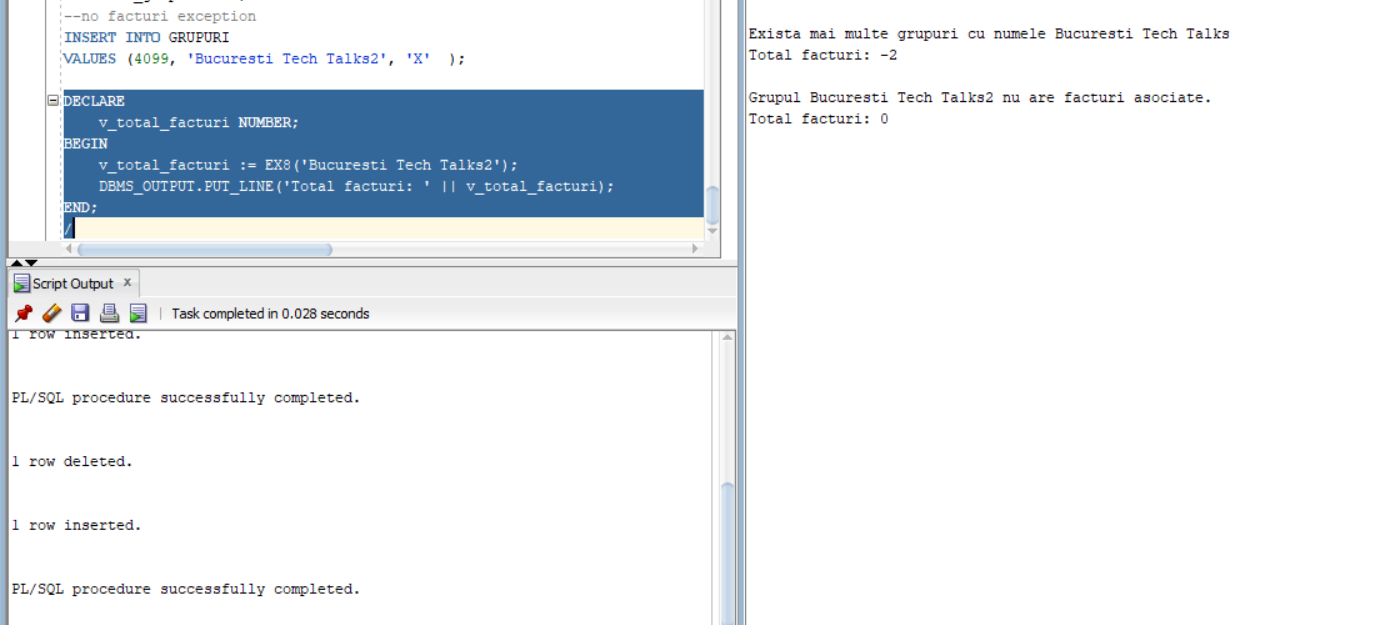
C)

INSERT INTO GRUPURI  
VALUES (4099, 'Bucuresti Tech Talks', 'X' );  
 --too many rows  
DECLARE  
 v\_total\_facturi NUMBER;  
BEGIN  
 v\_total\_facturi := *EX8*('Bucuresti Tech Talks');  
 DBMS\_OUTPUT.*PUT\_LINE*('Total facturi: ' || v\_total\_facturi);  
END;  
/



D)

--no facturi exception  
INSERT INTO GRUPURI  
VALUES (4099, 'Bucuresti Tech Talks2', 'X' );  
   
DECLARE  
 v\_total\_facturi NUMBER;  
BEGIN  
 v\_total\_facturi := *EX8*('Bucuresti Tech Talks2');  
 DBMS\_OUTPUT.*PUT\_LINE*('Total facturi: ' || v\_total\_facturi);  
END;  
/



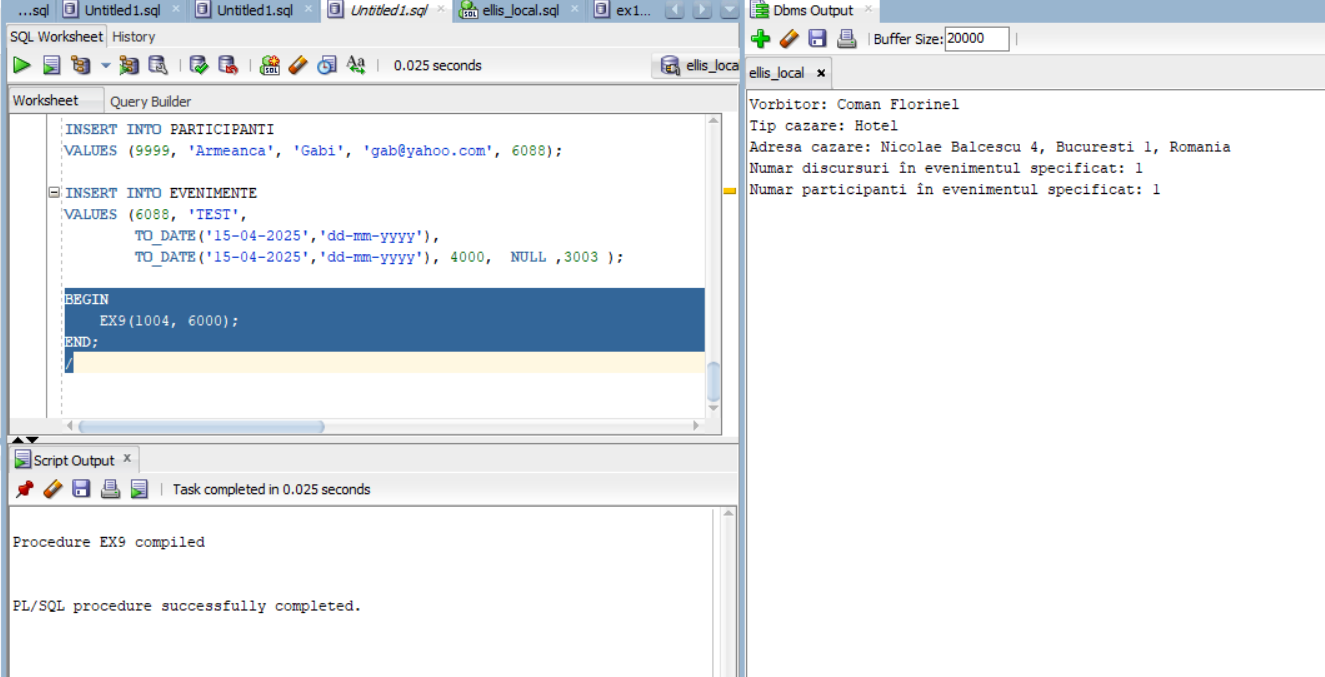
# 9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să aibă minim 2 parametri și să utilizeze într-o singură comandă SQL 5 dintre tabelele create. Definiți minim 2 excepții proprii, altele decât cele predefinite la nivel de sistem. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.

**CERINȚA:** Primind ca parametri un *id\_vorbitor* si un *id\_eveniment* afisati numarul de discursuri prestate, participantii si locul unde a fost cazat vorbitorul in cadrul evenimentului respectiv.

CREATE OR REPLACE PROCEDURE *EX9*(  
 p\_id\_vorbitor IN NUMBER,  
 p\_id\_eveniment IN NUMBER  
) IS  
 v\_nume\_vorbitor VARCHAR2(100);  
 v\_prenume\_vorbitor VARCHAR2(100);  
 v\_tip\_cazare VARCHAR2(100);  
 v\_adresa\_cazare VARCHAR2(100);  
 v\_numar\_discursuri NUMBER;  
 v\_numar\_participanti NUMBER;  
  
 -- Exceptii proprii  
 fara\_discursuri EXCEPTION;  
 fara\_participanti EXCEPTION;  
 fara\_cazare EXCEPTION;  
BEGIN  
 -- Interogare pentru a obtine detalii despre vorbitor, cazare, discursuri si numarul de participan?i  
 SELECT v.nume,   
 v.prenume,   
 c.tip,   
 c.adresa,  
 *COUNT*(DISTINCT vde.id\_discurs) AS numar\_discursuri,  
 *COUNT*(DISTINCT p.id\_participant) AS numar\_participanti  
 INTO v\_nume\_vorbitor,   
 v\_prenume\_vorbitor,   
 v\_tip\_cazare,   
 v\_adresa\_cazare,  
 v\_numar\_discursuri,  
 v\_numar\_participanti  
 FROM VORBITORI v  
 JOIN VORBITORI\_DISCURSURI\_EVENIMENTE vde ON v.id\_vorbitor = vde.id\_vorbitor  
 JOIN EVENIMENTE e ON vde.id\_eveniment = e.id\_eveniment  
 LEFT JOIN CAZARI c ON e.id\_cazare = c.id\_cazare  
 LEFT JOIN PARTICIPANTI p ON e.id\_eveniment = p.id\_eveniment  
 WHERE v.id\_vorbitor = p\_id\_vorbitor  
 AND vde.id\_eveniment = p\_id\_eveniment  
 GROUP BY v.nume, v.prenume, c.tip, c.adresa;  
  
   
 IF v\_numar\_discursuri = 0 THEN  
 RAISE fara\_discursuri;  
 END IF;  
  
 IF v\_numar\_participanti = 0 THEN  
 RAISE fara\_participanti;  
 END IF;  
  
 IF v\_tip\_cazare IS NULL THEN  
 RAISE fara\_cazare;  
 END IF;  
  
 DBMS\_OUTPUT.*PUT\_LINE*('Vorbitor: ' || v\_nume\_vorbitor || ' ' || v\_prenume\_vorbitor);  
 DBMS\_OUTPUT.*PUT\_LINE*('Tip cazare: ' || v\_tip\_cazare);  
 DBMS\_OUTPUT.*PUT\_LINE*('Adresa cazare: ' || v\_adresa\_cazare);  
 DBMS\_OUTPUT.*PUT\_LINE*('Numar discursuri în evenimentul specificat: ' || v\_numar\_discursuri);  
 DBMS\_OUTPUT.*PUT\_LINE*('Numar participanti în evenimentul specificat: ' || v\_numar\_participanti);  
  
EXCEPTION  
 WHEN NO\_DATA\_FOUND THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('Nu exista date pentru vorbitorul cu ID-ul ' || p\_id\_vorbitor || ' la evenimentul cu ID-ul ' || p\_id\_eveniment || '.');  
  
 WHEN fara\_participanti THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('Evenimentul cu ID-ul ' || p\_id\_eveniment || ' nu are participanti.');  
 WHEN fara\_cazare THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('Evenimentul cu ID-ul ' || p\_id\_eveniment || ' nu are cazare asociata.');  
 WHEN OTHERS THEN  
 DBMS\_OUTPUT.*PUT\_LINE*('A aparut o eroare neasteptata: ' || *SQLERRM*);  
END *EX9*;  
/

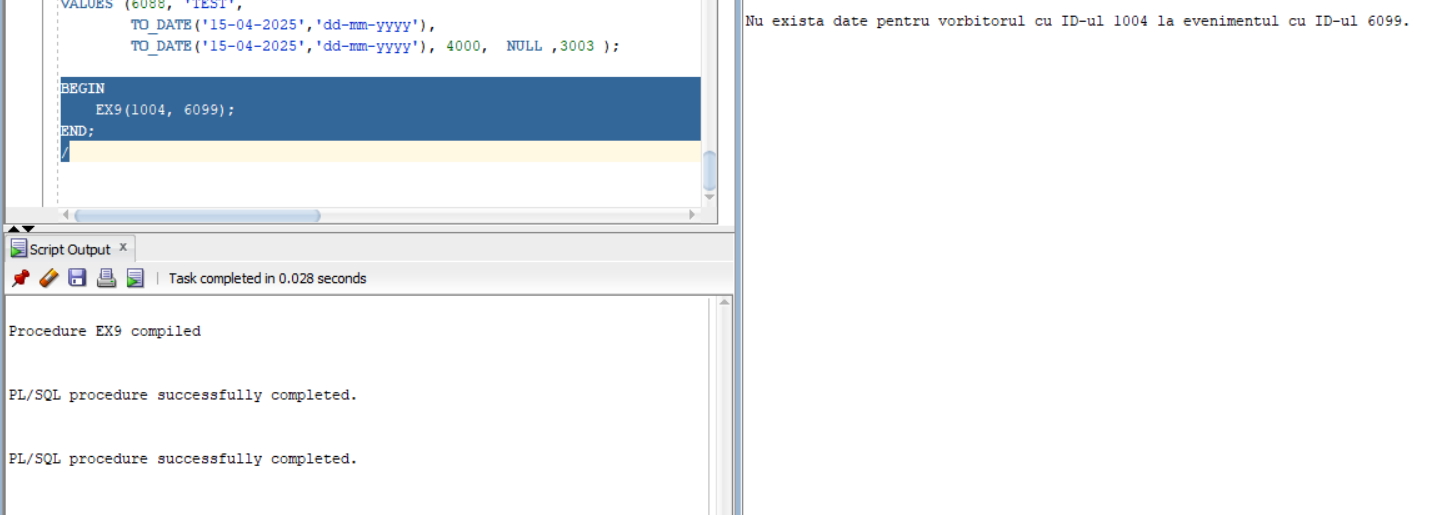
CAZURI RULARE:  
A)

BEGIN  
 *EX9*(1004, 6000);  
END;  
/



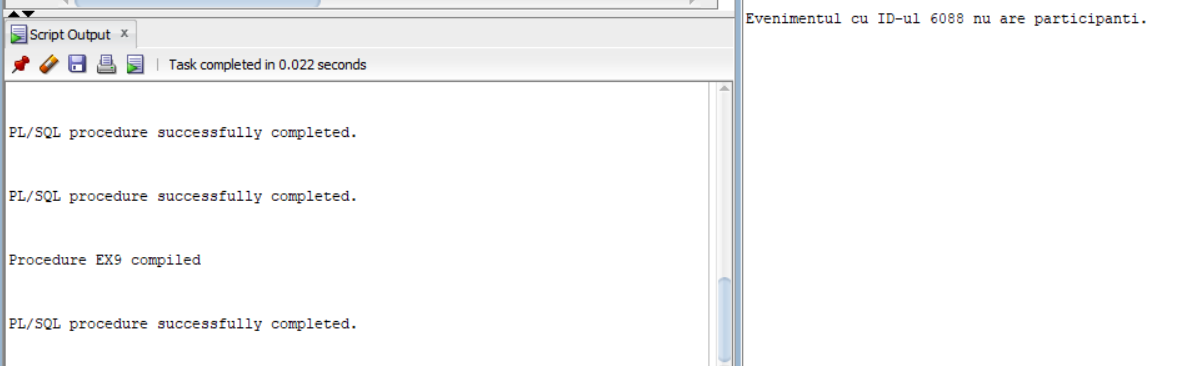
B)

BEGIN  
 *EX9*(1004, 6099);  
END;  
/



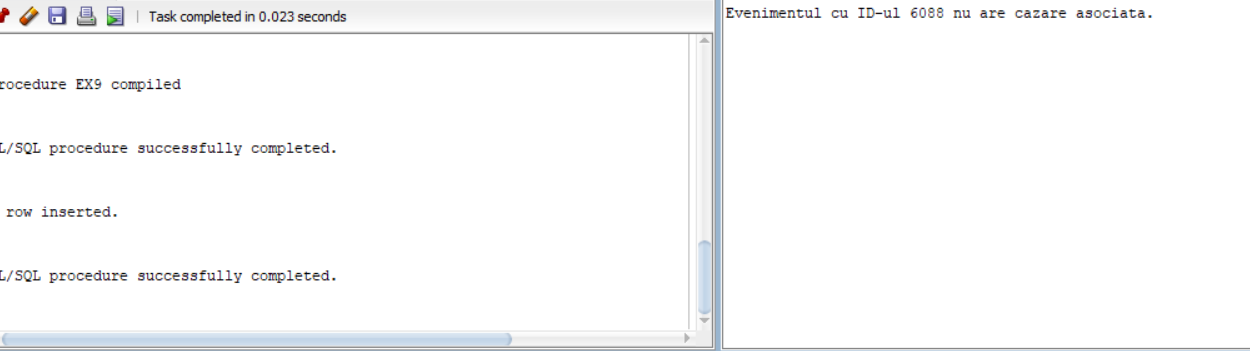
C)

INSERT INTO EVENIMENTE  
VALUES (6088, 'TEST',   
 *TO\_DATE*('15-04-2025','dd-mm-yyyy'),   
 *TO\_DATE*('15-04-2025','dd-mm-yyyy'), 4000, NULL ,3003 );  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE VALUES(1004, 5000, 6088);  
  
BEGIN  
 *EX9*(1004, 6088);  
END;  
/



D)

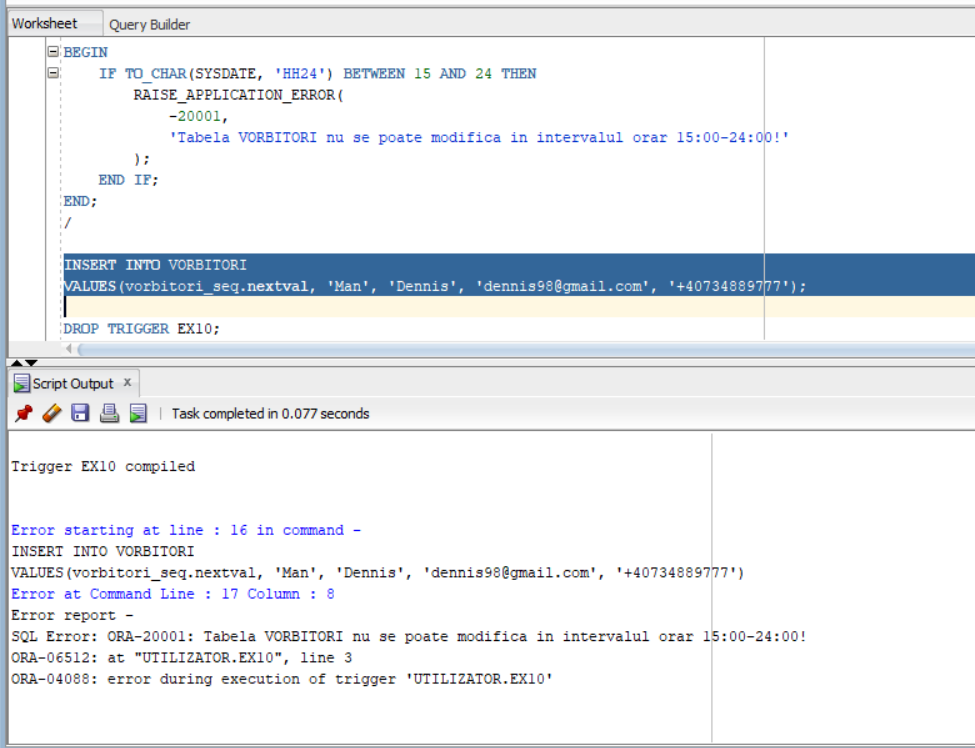
INSERT INTO EVENIMENTE  
VALUES (6088, 'TEST',  
 *TO\_DATE*('15-04-2025','dd-mm-yyyy'),  
 *TO\_DATE*('15-04-2025','dd-mm-yyyy'), 4000, NULL ,3003 );  
INSERT INTO VORBITORI\_DISCURSURI\_EVENIMENTE VALUES(1004, 5000, 6088);  
INSERT INTO PARTICIPANTI  
VALUES (9999, 'Armeanca', 'Gabi', 'gab@yahoo.com', 6088);  
BEGIN  
 *EX9*(1004, 6088);  
END;  
/



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

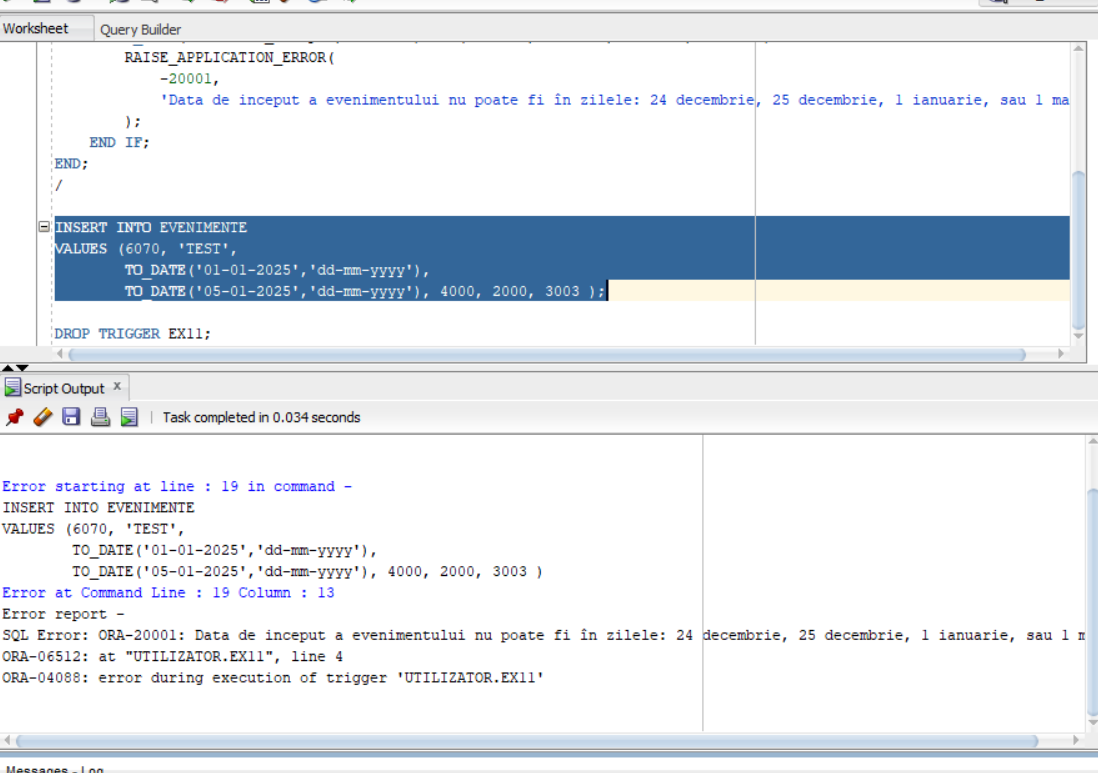
* **voi crea un trigger care se va declansa atunci cand vreau sa modific tabela VORBITORI in intervalul orar 15:00 - 24:00**

CREATE OR REPLACE TRIGGER EX10  
BEFORE INSERT OR DELETE OR UPDATE ON VORBITORI  
BEGIN  
 IF *TO\_CHAR*(*SYSDATE*, 'HH24') BETWEEN 15 AND 24 THEN  
 *RAISE\_APPLICATION\_ERROR*(  
 -20001,  
 'Tabela VORBITORI nu se poate modifica in intervalul orar 15:00-24:00!'  
 );  
 END IF;  
END;  
/  
  
INSERT INTO VORBITORI  
VALUES(vorbitori\_seq.nextval, 'Man', 'Dennis', 'dennis98@gmail.com', '+40734889777');  
  
DROP TRIGGER EX10;



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

* voi crea un trigger care se va declansa atunci cand vreau sa adaug o linie in tabela EVENIMENT a carei data de inceput sa fie 24, 25 decembrie, 1 ianuarie sau 1 mai



CREATE OR REPLACE TRIGGER EX11  
BEFORE INSERT OR UPDATE ON EVENIMENTE  
FOR EACH ROW  
DECLARE  
BEGIN  
 IF *TO\_CHAR*(:NEW.data\_inceput, 'DD-MM') IN ('24-12', '25-12', '01-01', '01-05') THEN  
 *RAISE\_APPLICATION\_ERROR*(  
 -20001,  
 'Data de inceput a evenimentului nu poate fi în zilele: 24 decembrie, 25 decembrie, 1 ianuarie, sau 1 mai.'  
 );  
 END IF;  
END;  
/  
  
INSERT INTO EVENIMENTE  
VALUES (6070, 'TEST',   
 *TO\_DATE*('01-01-2025','dd-mm-yyyy'),   
 *TO\_DATE*('05-01-2025','dd-mm-yyyy'), 4000, 2000, 3003 );  
   
DROP TRIGGER EX11;

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

CREATE TABLE audit\_events (  
 nume\_bd VARCHAR2(50),  
 user\_logat VARCHAR2(30),  
 eveniment VARCHAR2(20),  
 tip\_obiect\_referit VARCHAR2(30),  
 nume\_obiect\_referit VARCHAR2(30),  
 data TIMESTAMP(3)  
);  
  
  
CREATE OR REPLACE TRIGGER audit\_modificari  
AFTER CREATE OR DROP OR ALTER ON SCHEMA  
BEGIN  
 INSERT INTO audit\_events  
 VALUES (  
 SYS.*DATABASE\_NAME*, -- Numele bazei de date  
 SYS.*LOGIN\_USER*, -- Utilizatorul conectat  
 SYS.*SYSEVENT*, -- Evenimentul (CREATE, DROP, ALTER)  
 SYS.*DICTIONARY\_OBJ\_TYPE*, -- Tipul obiectului (TABLE, INDEX etc.)  
 SYS.*DICTIONARY\_OBJ\_NAME*, -- Numele obiectului  
 *SYSTIMESTAMP*(3) -- Momentul exact al evenimentului  
 );  
END;  
/  
  
CREATE TABLE exemplu (coloana1 NUMBER);  
  
ALTER TABLE exemplu ADD (coloana2 VARCHAR2(50));  
  
DROP TABLE exemplu;  
  
SELECT \* FROM audit\_events;  
  
DROP TABLE audit\_events;  
  
  
DROP TRIGGER audit\_modificari;

