

## **Model al bazei de date - Galerie de artă**

Silaghi Mara

Grupa 241

1. Prezentați pe scurt baza de date (utilitatea ei).

2.Diagrama ERD

3.Diagrama conceptuală

4 și 5. Implementarea bazei de date în Oracle

6.Problemă cu 3 tipuri de colecții

7.Problemă cu cursoare

8.Problemă rezolvată cu subprogram de tip funcție cu 3 dintre tabelele definite într-o singură cerere SQL

9.Problemă rezolvată cu subprogram de tip procedură cu 5 dintre tabelele definite într-o singură cerere SQL

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.

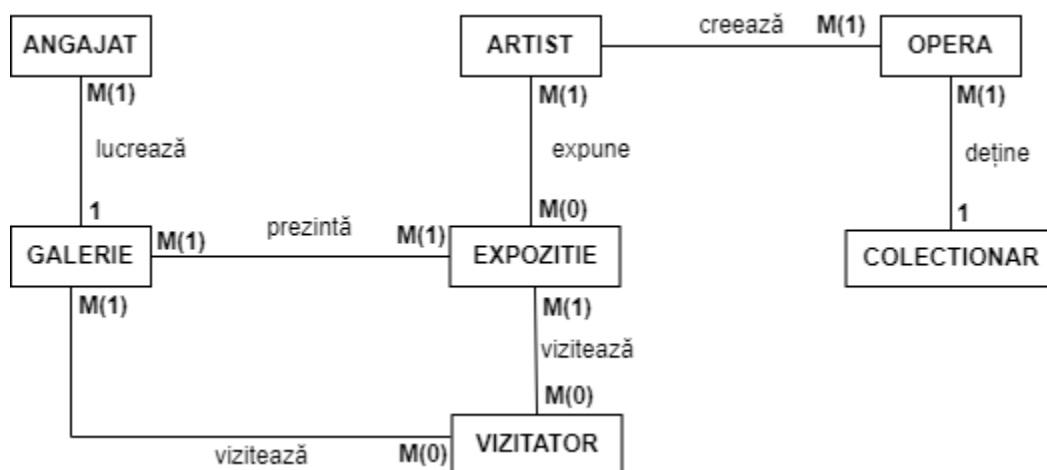
## 1. Prezențați pe scurt baza de date (utilitatea ei).

Baza de date a unei galerii de artă cuprinde informații despre diferite entități și relații importante pentru gestionarea operei de artă, a artiștilor și a clienților. Printre entități se numără operele de artă, artiștii, colecționarii, angajații și vizitatorii. Pentru fiecare dintre aceste entități se pot colecta informații specifice, cum ar fi numele, adresa, prețul.

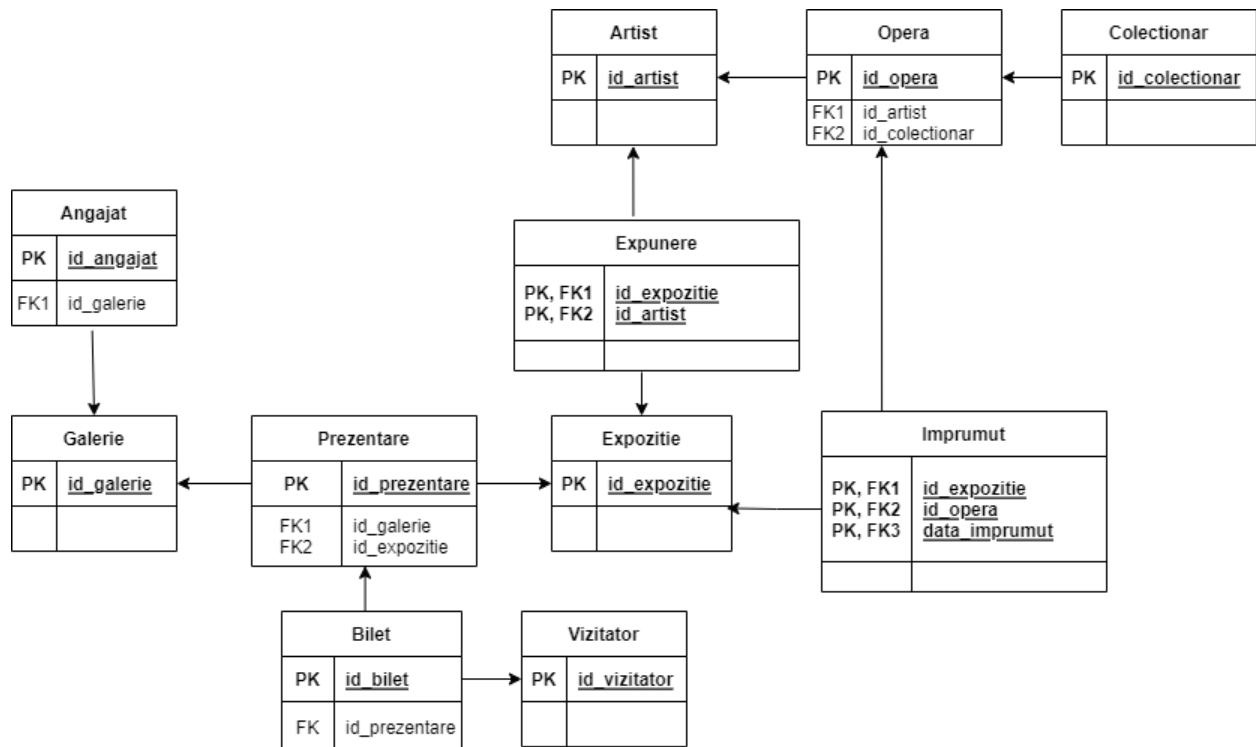
În ceea ce privește relațiile, acestea pot fi stabilite între entități, cum ar fi faptul că operele de artă sunt create de artiști sau că o expoziție cuprinde una sau mai multe opere. De asemenea, colecționarii pot achiziționa opere de artă de la galerie, iar angajații pot fi responsabili de administrarea și gestionarea operelor de artă și a relațiilor cu clienții. În plus, vizitatorii pot achiziționa bilete și pot lua parte la expozițiile galeriei.

Astfel, o bază de date a unei galerii de artă poate fi un instrument valoros pentru a gestiona cu succes opera de artă, artiștii, colecționarii și clienții, oferind o perspectivă mai clară și organizată asupra activităților galeriei de artă.

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



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



4 și 5. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, definind toate constrângerile de integritate necesare (chei primare, cheile externe etc).

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

```

CREATE TABLE Artist (
    id_artist INT PRIMARY KEY,
    nume VARCHAR(255) NOT NULL,
    data_nasterii DATE,
    nationalitate VARCHAR(255)
);
    
```

```

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (1, 'Leonardo da Vinci', TO_DATE('1452-04-15', 'YYYY-MM-DD'), 'italian');

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (2, 'Vincent van Gogh', TO_DATE('1853-03-30', 'YYYY-MM-DD'), 'olandez');

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (3, 'Pablo Picasso', TO_DATE('1881-10-25', 'YYYY-MM-DD'), 'spaniol');

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (4, 'Claude Monet', TO_DATE('1840-11-14', 'YYYY-MM-DD'), 'francez');

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (5, 'Frida Kahlo', TO_DATE('1907-07-06', 'YYYY-MM-DD'), 'mexican');

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (6, 'Michelangelo', TO_DATE('1475-03-06', 'YYYY-MM-DD'), 'italian');

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (7, 'Auguste Rodin', TO_DATE('1840-11-12', 'YYYY-MM-DD'), 'francez');

INSERT INTO Artist (id_artist, nume, data_nasterii, nationalitate)
VALUES
  (8, 'Constantin Brancusi', TO_DATE('1876-02-19', 'YYYY-MM-DD'), 'roman');

```

	ID_ARTIST	NUME	DATA_NASTERII	NATIONALITATE
1	1	Leonardo da Vinci	15-APR-52	italian
2	2	Vincent van Gogh	30-MAR-53	olandez
3	3	Pablo Picasso	25-OCT-81	spaniol
4	4	Claude Monet	14-NOV-40	francez
5	5	Frida Kahlo	06-JUL-07	mexican
6	6	Michelangelo	06-MAR-75	italian
7	7	Auguste Rodin	12-NOV-40	francez
8	8	Constantin Brancusi	19-FEB-76	roman

```

CREATE TABLE Opera (
    id_opera INT PRIMARY KEY,
    titlu VARCHAR(255) NOT NULL,
    id_artist INT,
    tip VARCHAR(255) NOT NULL,
    an_creatie INT,
    id_colectionar INT,
    FOREIGN KEY (id_artist) REFERENCES Artist(id_artist),
    FOREIGN KEY (id_colectionar) REFERENCES Artist(id_colectionar)
);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (1, 'Mona Lisa', 1, 'Pictura', 1503, 1);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (2, 'Starry Night', 2, 'Pictura', 1889, 2);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (3, 'Guernica', 3, 'Pictura', 1937, 1);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (4, 'Water Lilies', 4, 'Pictura', 1919, 3);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (5, 'The Two Fridas', 5, 'Pictura', 1939, 4);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (6, 'David', 6, 'Sculptura', 1504, 2);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (7, 'The Thinker', 7, 'Sculptura', 1902, NULL);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (8, 'Pieta', 6, 'Sculptura', 1499, NULL);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (9, 'Bird in Space', 8, 'Sculptura', 1923, 3);

INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
    (10, 'The Kiss', 7, 'Sculptura', 1889, 5);
INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES

```

```

(11, 'Cina cea de Taina', 1, 'Pictura', NULL, 2);
INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
(12, 'Peisaj', 5, 'Pictura', 1869, 1);
INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie, id_colectionar)
VALUES
(13, 'The Thinker', 5, 'Pictura', 1869, 4);

```

ID_OPERA	TITLU	ID_ARTIST	TIP	AN_CREATIE
1	1 Mona Lisa		1 Pictura	1503
2	2 Starry Night		2 Pictura	1889
3	3 Guernica		3 Pictura	1937
4	4 Water Lilies		4 Pictura	1919
5	5 The Two Fridas		5 Pictura	1939
6	6 David		6 Sculptura	1504
7	7 The Thinker		7 Sculptura	1902
8	8 Pieta		6 Sculptura	1499
9	9 Bird in Space		8 Sculptura	1923
10	10 The Kiss		7 Sculptura	1889
11	11 Cina cea de Taina		1 Pictura	(null)
12	12 Peisaj		1 Pictura	1898
13	13 Peisaj		1 Pictura	1898

```

CREATE TABLE Colectionar (
  id_colectionar INT PRIMARY KEY,
  nume VARCHAR(255) NOT NULL,
  prenume VARCHAR(255) NOT NULL,
  telefon VARCHAR(255),
  adresa_mail VARCHAR(255) NOT NULL,
);

INSERT INTO Colectionar (id_colectionar, nume, prenume, telefon, adresa_mail)
VALUES
(1, 'Dirjan', 'Alexandra', '0701073147', 'dirjan.alexandra@gmail.com', 1);

INSERT INTO Colectionar (id_colectionar, nume, prenume, telefon, adresa_mail)
VALUES
(2, 'Nitu', 'Paul', '0248292876', 'nitu.paul@gmail.com', 3);

INSERT INTO Colectionar (id_colectionar, nume, prenume, telefon, adresa_mail)
VALUES
(3, 'Marin', 'Andrei', '0785282188', 'marin.andrei@gmail.com', 5);

INSERT INTO Colectionar (id_colectionar, nume, prenume, telefon, adresa_mail)
VALUES
(4, 'Cristea', 'Ionut', '0270491235', 'cristea.ionut@gmail.com', 2);

```

```

INSERT INTO Colectionar (id_colectionar, nume, prenume, telefon, adresa_mail)
VALUES
    (5, 'Dumitru', 'Clara', '0264719906 ', 'dumitru.clara@gmail.com', 4);

INSERT INTO Colectionar (id_colectionar, nume, prenume, telefon, adresa_mail)
VALUES
    (6, 'Tumulescu', 'Andreea', '0365884320 ', 'tumulescu.andreea@gmail.com', 7);

```

	ID_COLECTIONAR	NUME	PRENUME	TELEFON	ADRESA_MAIL	ID_OPERA
1	1	Dirjan	Alexandra	0701073147	dirjan.alexandra@gmail.com	1
2	2	Nitu	Paul	0248292876	nitu.paul@gmail.com	3
3	3	Marin	Andrei	0785282188	marin.andrei@gmail.com	5
4	4	Cristea	Ionut	0270491235	cristea.ionut@gmail.com	2
5	5	Dumitru	Clara	0264719906	dumitru.clara@gmail.com	4
6	6	Tumulescu	Andreea	0365884320	tumulescu.andreea@gmail.com	7

```

CREATE TABLE Galerie (
    id_galerie INT PRIMARY KEY,
    denumire VARCHAR(255) NOT NULL,
    locatie VARCHAR(255)
);

INSERT INTO Galerie (id_galerie, denumire, locatie)
VALUES
    (1, 'Galeria Nationala', 'Bucuresti');

INSERT INTO Galerie (id_galerie, denumire, locatie)
VALUES
    (2, 'Museum of Modern Art', 'New York');

INSERT INTO Galerie (id_galerie, denumire, locatie)
VALUES
    (3, 'Louvre Museum', 'Paris');

INSERT INTO Galerie (id_galerie, denumire, locatie)
VALUES
    (4, 'Tate Modern', 'Londra');

INSERT INTO Galerie (id_galerie, denumire, locatie)
VALUES
    (5, 'Uffizi Gallery', 'Florenta');

INSERT INTO Galerie (id_galerie, denumire, locatie)
VALUES
    (6, 'Museum of Modern Art', 'Bruxelles');

```



	ID_GALERIE	DENUMIRE	LOCATIE
1	1	Galeria Nationala	Bucuresti
2	2	Museum of Modern Art	New York
3	3	Louvre Museum	Paris
4	4	Tate Modern	Londra
5	5	Uffizi Gallery	Florenta
6	6	Museum of Modern Art	Bruxelles

```

CREATE TABLE Expozitie (
  id_expozitie INT PRIMARY KEY,
  nume_expozitie VARCHAR(255) NOT NULL,
  data_inceput TIMESTAMP NOT NULL,
  data_final TIMESTAMP NOT NULL,
  pret_bilet INT NOT NULL
);

INSERT INTO Expozitie (id_expozitie, nume_expozitie, data_inceput, data_final,
pret_bilet)
VALUES
  (1, 'Semne si simboluri', TO_TIMESTAMP('2022-06-01 08:00:00', 'YYYY-MM-DD
HH24:MI:SS'),
  TO_TIMESTAMP('2022-06-30 21:59:00', 'YYYY-MM-DD HH24:MI:SS'), 25);

INSERT INTO Expozitie (id_expozitie,nume_expozitie, data_inceput, data_final,
pret_bilet)
VALUES
  (2, 'One Night Exhibition', TO_TIMESTAMP('2023-07-15 07:00:00', 'YYYY-MM-DD
HH24:MI:SS'),
  TO_TIMESTAMP('2023-07-16 00:00:00', 'YYYY-MM-DD HH24:MI:SS'), 40);

INSERT INTO Expozitie (id_expozitie, nume_expozitie, data_inceput, data_final,
pret_bilet)
VALUES
  (3, 'Between Worlds and Traditions', TO_TIMESTAMP('2022-09-01 08:00:00',
'YYYY-MM-DD HH24:MI:SS'),
  TO_TIMESTAMP('2022-09-30 23:59:00', 'YYYY-MM-DD HH24:MI:SS'), 15);

INSERT INTO Expozitie (id_expozitie, nume_expozitie, data_inceput, data_final,
pret_bilet)
VALUES
  (4, 'Gardens', TO_TIMESTAMP('2023-10-15 10:00:00', 'YYYY-MM-DD HH24:MI:SS'),
  TO_TIMESTAMP('2023-11-15 17:00:00', 'YYYY-MM-DD HH24:MI:SS'), 25);

INSERT INTO Expozitie (id_expozitie, nume_expozitie, data_inceput, data_final,
pret_bilet)
VALUES

```

```
(5, 'Retrospectiva', TO_TIMESTAMP('2023-12-01 09:30:00 ', 'YYYY-MM-DD HH24:MI:SS'),
TO_TIMESTAMP('2023-12-31 18:00:00', 'YYYY-MM-DD HH24:MI:SS'), 45);
```

ID_EXPOZITIE	NUME_EXPOZITIE	DATA_INCEPUT	DATA_FINAL	PRET_BILET
1	1 Semne si simboluri	01-JUN-22 08.00.00.000000000 AM	30-JUN-22 09.59.00.000000000 PM	25
2	2 One Night Exhibition	15-JUL-23 07.00.00.000000000 AM	16-JUL-23 12.00.00.000000000 AM	40
3	3 Between Worlds and Traditions	01-SEP-22 08.00.00.000000000 AM	30-SEP-22 11.59.00.000000000 PM	15
4	4 Gardens	15-OCT-23 10.00.00.000000000 AM	15-NOV-23 05.00.00.000000000 PM	25
5	5 Retrospectiva	01-DEC-23 09.30.00.000000000 AM	31-DEC-23 06.00.00.000000000 PM	45

```
CREATE TABLE Angajat (
  id_angajat INT PRIMARY KEY,
  nume VARCHAR(255) NOT NULL,
  pozitie VARCHAR(255) NOT NULL,
  salariu INT NOT NULL,
  id_galerie INT,
  FOREIGN KEY (id_galerie) REFERENCES Galerie(id_galerie)
);

INSERT INTO Angajat (id_angajat, nume, pozitie, salariu, id_galerie)
VALUES
  (1, 'Dimitrescu', 'Manager', 6000, 1);

INSERT INTO Angajat (id_angajat, nume, pozitie, salariu, id_galerie)
VALUES
  (2, 'Barbu', 'Ghid', 3500, 2);

INSERT INTO Angajat (id_angajat, nume, pozitie, salariu, id_galerie)
VALUES
  (3, 'Gheorghe', 'Director', 8000, 3);

INSERT INTO Angajat (id_angajat, nume, pozitie, salariu, id_galerie)
VALUES
  (4, 'Iacob', 'Curator', 4500, 4);

INSERT INTO Angajat (id_angajat, nume, pozitie, salariu, id_galerie)
VALUES
  (5, 'Stanciu', 'Ghid', 3500, 5);

INSERT INTO Angajat (id_angajat, nume, pozitie, salariu, id_galerie)
VALUES
  (6, 'Simion', 'Manager', 6000, 2);

INSERT INTO Angajat (id_angajat, nume, pozitie, salariu, id_galerie)
VALUES
  (7, 'Palade', 'Ghid', 3500, 3);
```

	ID_ANGAJAT	NUME	POZITIE	SALARIU	ID_GALERIE
1	1	Dimitrescu	Manager	6000	1
2	2	Barbu	Ghid	3500	2
3	3	Gheorghe	Director	8000	3
4	4	Iacob	Curator	4500	4
5	5	Stanciu	Ghid	3500	5
6	6	Simion	Manager	6000	2
7	7	Palade	Ghid	3500	3

```

CREATE TABLE Vizitator (
  id_vizitator INT PRIMARY KEY,
  nume VARCHAR(255) NOT NULL,
  prenume VARCHAR(255) NOT NULL,
  adresa_mail VARCHAR(255)
);

INSERT INTO Vizitator (id_vizitator, nume, prenume, adresa_mail)
VALUES
  (1, 'Diaconescu', 'Cosmin', 'd.cosmin@gmail.com');

INSERT INTO Vizitator (id_vizitator, nume, prenume, adresa_mail)
VALUES
  (2, 'Voinea', 'David', 'david.v@gmail.com');

INSERT INTO Vizitator (id_vizitator, nume, prenume, adresa_mail)
VALUES
  (3, 'Vlasceanu', 'Luminita', 'v.luminita@gmail.com');

INSERT INTO Vizitator (id_vizitator, nume, prenume, adresa_mail)
VALUES
  (4, 'Paul', 'Marin', 'marin.paul@gmail.com');

INSERT INTO Vizitator (id_vizitator, nume, prenume, adresa_mail)
VALUES
  (5, 'Dragan', 'Andreea', 'andreea.dragan@gmail.com');

```

	ID_VIZITATOR	NUME	PRENUME	ADRESA_MAIL
1	2	Voinea	David	david.v@gmail.com
2	3	Vlasceanu	Luminita	v.luminita@gmail.com
3	4	Paul	Marin	marin.paul@gmail.com
4	5	Dragan	Andreea	maria.martinez@gmail.com
5	1	Diaconescu	Cosmin	d.cosmin@gmail.com

```

CREATE TABLE Expunere (
    ID INT,
    id_expozitie INT,
    id_artist INT,
    PRIMARY KEY (ID),
    FOREIGN KEY (id_expozitie) REFERENCES Expozitie(id_expozitie),
    FOREIGN KEY (id_artist) REFERENCES Artist(id_artist),
    UNIQUE (id_expozitie, id_artist)
);

INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (1, 1, 1);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (2, 1, 6);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (3, 1, 5);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (4, 2, 4);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (5, 2, 7);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (6, 2, 2);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (7, 2, 5);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (8, 3, 3);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (9, 4, 3);
INSERT INTO Expunere (ID, id_expozitie, id_artist)
VALUES
    (10, 4, 1);

```

ID	ID_EXPOZITIE	ID_ARTIST
1	1	1
2	1	6
3	1	5
4	2	4
5	2	7
6	2	2
7	2	5
8	3	3
9	4	3
10	4	1

```

CREATE TABLE Bilet (
    id_bilet INT PRIMARY KEY,
    data_vizita DATE NOT NULL,
    id_vizitator INT,
    id_prezentare INT,
    FOREIGN KEY (id_vizitator) REFERENCES Vizitator(id_vizitator),
    FOREIGN KEY (id_prezentare) REFERENCES Prezentare(id_prezentare),
    UNIQUE (id_prezentare, id_vizitator, data_vizita)
);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (1, TO_DATE('2023-06-15', 'YYYY-MM-DD'), 1, 5);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (2, TO_DATE('2023-07-01', 'YYYY-MM-DD'), 2, 3);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (3, TO_DATE('2023-08-10', 'YYYY-MM-DD'), 3, 1);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (4, TO_DATE('2023-09-20', 'YYYY-MM-DD'), 4, 3);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (5, TO_DATE('2023-10-05', 'YYYY-MM-DD'), 5, 4);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (6, TO_DATE('2022-01-15', 'YYYY-MM-DD'), 2, 5);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (7, TO_DATE('2022-02-23', 'YYYY-MM-DD'), 2, 2);

INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (8, TO_DATE('2022-03-15', 'YYYY-MM-DD'), 3, 1);

    INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (9, TO_DATE('2022-03-15', 'YYYY-MM-DD'), 3, 5);

    INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES
    (10, TO_DATE('2022-03-15', 'YYYY-MM-DD'), 4, 1);
INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)
VALUES

```

```
(11, TO_DATE('2023-06-15', 'YYYY-MM-DD'), 1, 2);
```

```
INSERT INTO Bilet (id_bilet, data_vizita, id_vizitator, id_prezentare)  
VALUES  
(12, TO_DATE('2023-06-15', 'YYYY-MM-DD'), 5, 3);
```

	ID_BILET	DATA_VIZITA	ID_VIZITATOR	ID_PREZENTARE
1	8	15-MAR-22	3	1
2	3	10-AUG-23	3	1
3	10	15-MAR-22	4	1
4	11	15-JUN-23	1	2
5	7	23-FEB-22	2	2
6	2	01-JUL-23	2	3
7	4	20-SEP-23	4	3
8	12	15-JUN-23	5	3
9	5	05-OCT-23	5	4
10	1	15-JUN-23	1	5
11	6	15-JAN-22	2	5
12	9	15-MAR-22	3	5

```
CREATE TABLE Prezentare (  
    id_prezentare INT,  
    id_expozitie INT,  
    id_galerie INT,  
    PRIMARY KEY (ID),  
    FOREIGN KEY (id_expozitie) REFERENCES Expozitie(id_expozitie),  
    FOREIGN KEY (id_galerie) REFERENCES Galerie(id_galerie),  
    UNIQUE (id_expozitie, id_galerie)  
);
```

```
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)  
VALUES  
(0, 1, 2);  
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)  
VALUES  
(1, 1, 3);  
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)  
VALUES  
(2, 1, 4);  
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)  
VALUES  
(3, 2, 4);  
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)  
VALUES  
(4, 3, 3);  
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)  
VALUES
```

```

(5, 4, 5);
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)
VALUES
(6, 5, 3);
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)
VALUES
(7, 5, 2);
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)
VALUES
(8, 5, 1);
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)
VALUES
(9, 5, 5);
INSERT INTO Prezentare (IDid_prezentare id_expozitie, id_galerie)
VALUES
(10, 5, 4);
INSERT INTO Prezentare (id_prezentare, id_expozitie, id_galerie)
VALUES
(11, 4, 1);

```

	ID_PREZEN...	ID_EXPOZITIE	ID_GALERIE
1	0	1	2
2	1	1	3
3	2	1	4
4	3	2	4
5	4	3	3
6	5	4	5
7	6	5	3
8	7	5	2
9	8	5	1
10	9	5	5
11	10	5	4

```

CREATE TABLE Imprumut (
  ID INT,
  id_expozitie INT,
  id_opera INT,
  PRIMARY KEY (ID),
  FOREIGN KEY (id_expozitie) REFERENCES Expozitie(id_expozitie),
  FOREIGN KEY (id_opera) REFERENCES Opera(id_opera),
  UNIQUE (id_expozitie, id_opera)
);

INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
(1, 1, 3);
INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES

```

```

    (2, 1, 2);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (3, 1, 8);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (4, 2, 2);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (5, 2, 4);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (6, 2, 7);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (7, 3, 5);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (8, 4, 4);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (9, 4, 9);
    INSERT INTO Imprumut (ID, id_expozitie, id_opera)
VALUES
    (10, 5, 5);

```

	ID	ID_EXPOZITIE	ID_OPERA
1	1	1	3
2	2	1	2
3	3	1	8
4	4	2	2
5	5	2	4
6	6	2	7
7	7	3	5
8	8	4	4
9	9	4	9
10	10	5	5



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.

```
--Afisati operele care au fost expuse in cele 3 galerii cu cele mai multe  
expozitii, impreuna cu expozitia din care au facut parte. Daca intr-o galerie nu a  
fost expusa nicio opera, afisati mesajul "nu exista".
```

```
SET SERVEROUTPUT ON;  
CREATE OR REPLACE PROCEDURE EX6 AS  
    --pentru tabelul expozitii  
    TYPE EXPOZITIIREC IS RECORD (  
        id_galerie GALERIE.ID_GALERIE%TYPE,  
        id_expozitie EXPOZITIE.ID_EXPOZITIE%TYPE,  
        nume_expozitie EXPOZITIE.NUME_EXPOZITIE%TYPE  
    );  
  
    --salvam primele 3 galerii in varray  
    TYPE vector IS VARRAY(3) OF GALERIE%ROWTYPE;  
    GALERII vector:= vector();  
  
    --in expozitii salvam expozitia si galeria din care face parte  
    TYPE imbricat IS TABLE OF EXPOZITIIREC;  
    EXPOZITII imbricat := imbricat();  
  
    --salvam operele intr-un tablou indexat  
    TYPE idx IS TABLE OF VARCHAR(200) INDEX BY PLS_INTEGER;  
    OPERE idx;  
    NR_EXPOZITII NUMBER(6);  
  
BEGIN  
    --primele 3 galerii in functie de numarul de expozitii  
    WITH TABEL AS (SELECT G.DENUMIRE, COUNT(E.id_expozitie)  
                    FROM GALERIE G  
                    JOIN PREZENTARE P ON P.ID_GALERIE = G.ID_GALERIE  
                    JOIN EXPOZITIE E ON E.ID_EXPOZITIE = P.ID_EXPOZITIE  
                    GROUP BY G.DENUMIRE  
                    order by COUNT(E.id_expozitie) desc)  
    SELECT G.ID_GALERIE, G.DENUMIRE, G.LOCATIE  
    BULK COLLECT INTO GALERII  
    FROM GALERIE G  
    JOIN TABEL T ON T.DENUMIRE = G.DENUMIRE  
    WHERE ROWNUM<=3;  
  
    --am salvat "expozitii" drept nested, deci mai intai aflam dimensiunea listei  
    SELECT COUNT(*)  
    INTO NR_EXPOZITII  
    FROM EXPOZITIE E
```

```

JOIN PREZENTARE P ON P.ID_EXPOZITIE = E.ID_EXPOZITIE
JOIN GALERIE G ON G.ID_GALERIE = P.ID_GALERIE;

expozitii.extend(nr_expozitii + 1);

--expozitii
SELECT G.ID_GALERIE, E.ID_EXPOZITIE, E.NUME_EXPOZITIE
BULK COLLECT INTO EXPOZITII
FROM EXPOZITIE E
JOIN PREZENTARE P ON P.ID_EXPOZITIE = E.ID_EXPOZITIE
JOIN GALERIE G ON G.ID_GALERIE = P.ID_GALERIE
ORDER BY G.DENUMIRE;

--luam fiecare galerie din cele 3 si ii afisam expozitiile
FOR I IN GALERII.first..GALERII.last LOOP
DBMS_OUTPUT.PUT_LINE('GALERIA ' || GALERII(I).denumire);
DBMS_OUTPUT.PUT_LINE('-----');

    FOR J IN EXPOZITII.first..EXPOZITII.last LOOP
        IF EXPOZITII(J).ID_GALERIE = GALERII(I).ID_GALERIE THEN
            DBMS_OUTPUT.PUT_LINE('EXPOZITIA ' || EXPOZITII(J).NUME_EXPOZITIE
||': ');

                SELECT O.TITLU
                BULK COLLECT INTO OPERE
                FROM OPERA O
                JOIN IMPRUMUT I ON I.ID_OPERA = O.ID_OPERA
                JOIN EXPOZITIE E ON E.ID_EXPOZITIE = I.ID_EXPOZITIE
                WHERE E.ID_EXPOZITIE = EXPOZITII(J).ID_EXPOZITIE;

                FOR K IN OPERE.FIRST..OPERE.LAST LOOP
                    DBMS_OUTPUT.PUT_LINE(OPERE(K));
                END LOOP;
                DBMS_OUTPUT.PUT_LINE(' ');
            END IF;
        END LOOP;
        DBMS_OUTPUT.PUT_LINE(' ');
    END LOOP;

END;
/
BEGIN
    Ex6;
END;

```

SQL Worksheet | History

project\_Silaghi\_Mara

Worksheet | Query Builder

```
DBMS_OUTPUT.PUT_LINE(OPERE(K));  
END LOOP;  
DBMS_OUTPUT.PUT_LINE(' ');  
END IF;  
END LOOP;  
DBMS_OUTPUT.PUT_LINE(' ');  
END LOOP;  
  
END;  
/  
BEGIN  
  Ex6;  
END;
```

Query Result x | Script Output x

Task completed in 0.277 seconds

PL/SQL procedure successfully completed.

project\_Silaghi\_Mara x

GALERIA Tate Modern

EXPOZITIA Semne si simboluri:  
Starry Night  
Guernica  
Pieta

EXPOZITIA One Night Exhibition:  
Starry Night  
Water Lilies  
The Thinker

EXPOZITIA Retrospectiva:  
The Two Fridas

GALERIA Louvre Museum

EXPOZITIA Retrospectiva:  
The Two Fridas

EXPOZITIA Between Worlds and Traditions:  
The Two Fridas

EXPOZITIA Semne si simboluri:  
Starry Night  
Guernica  
Pieta

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.

```
--pentru top 2 cele mai vizitate expozitii, sa se afiseze operele
--expuse si artistii lor

CREATE OR REPLACE PROCEDURE Ex7 AS
    -- Primul cursor: Selectează expozițiile bazate pe numărul de vizitatori
    CURSOR Cursor1 IS
        WITH TABEL AS (
            SELECT DISTINCT COUNT(B.id_bilet) AS NR
            FROM Expozitie E
            LEFT JOIN Presentare P ON P.id_expozitie = E.id_expozitie
            --exista expozitii care nu apartin niciunei prezentari
            LEFT JOIN Bilet B ON P.id_prezentare =
B.id_prezentare
            GROUP BY E.id_expozitie, E.ume_expozitie
            ORDER BY NR DESC
        )
        SELECT E.id_expozitie, E.ume_expozitie, COUNT(B.id_bilet) as nr_vizitatori
        FROM Expozitie E
        LEFT JOIN Presentare P ON P.id_expozitie = E.id_expozitie
        LEFT JOIN Bilet B ON P.id_prezentare = B.id_prezentare
        GROUP BY E.id_expozitie, E.ume_expozitie
        HAVING COUNT(B.id_bilet) IN (SELECT NR FROM TABEL WHERE ROWNUM <= 2)
        ORDER BY COUNT(B.ID_BILET) DESC;

    -- Al doilea cursor: Obține informații despre operele de artă și artiști pentru
o expoziție specifică
    CURSOR Cursor2(idExpozitie EXPOZITIE.ID_EXPOZITIE%TYPE) IS
        SELECT O.titlu, A.ume
        FROM Opera O
        JOIN Artist A ON O.id_artist = A.id_artist
        JOIN Imprumut I ON O.id_opera = I.id_opera
        WHERE I.id_expozitie = idExpozitie;

    recExpozitie Cursor1%ROWTYPE;
    recOpArt Cursor2%ROWTYPE;
BEGIN
    OPEN Cursor1;
    LOOP
        FETCH Cursor1 INTO recExpozitie;
        EXIT WHEN Cursor1%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Expozitie: ' || recExpozitie.ume_expozitie ||
            ', Vizitatori: ' || recExpozitie.nr_vizitatori);

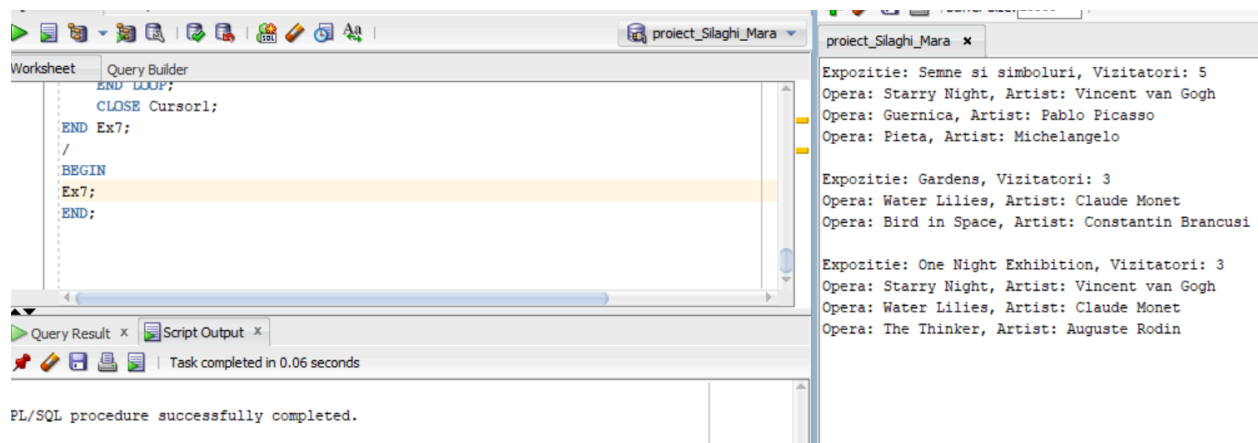
        OPEN Cursor2(recExpozitie.id_expozitie);
        LOOP
```

```

        FETCH Cursor2 INTO recOpArt;
        EXIT WHEN Cursor2%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Opera: ' || recOpArt.titlu ||
                               ', Artist: ' || recOpArt.num);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
    CLOSE Cursor2;
END LOOP;
CLOSE Cursor1;
END Ex7;
/
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 definite. Definiți minim 2 excepții proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.

```
--Pentru o opera data ca parametru, sa se afiseze expozitiile de care a fost
--imprumutata. Daca nu a fost imprumutata de nicio expozitie, sa se afiseze un
mesaj.
```

```
CREATE OR REPLACE FUNCTION Ex8 (TitluOp Opera.titlu%type)
RETURN VARCHAR2 IS
    TYPE lista IS TABLE OF VARCHAR2(255);
    ListaExpozitii lista;
    Rezultat VARCHAR2(4000);
    ContorOpera NUMBER;
    --exceptii
    ERR_1 EXCEPTION;
    ERR_2 EXCEPTION;
    ERR_3 EXCEPTION;
    ERR_4 EXCEPTION;
BEGIN
    IF TitluOp IS NULL OR TitluOp = '' THEN
        RAISE ERR_1;    --TITLU NUL
    END IF;

    SELECT COUNT(*)
    INTO ContorOpera
    FROM Opera
    WHERE UPPER(titlu) = UPPER(TitluOp);

    IF ContorOpera = 0 THEN
        RAISE ERR_2;    --OPERA NU EXISTA IN BAZA DE DATE
    END IF;

    IF ContorOpera >1 THEN
        RAISE ERR_3;    --EXISTA MAI MULTE OPERE CU ACELASI TITLU
    END IF;

    SELECT E.ume_expozitie
    BULK COLLECT INTO ListaExpozitii
    FROM Expozitie E
    JOIN Imprumut I ON E.id_expozitie = I.id_expozitie
    JOIN Opera O ON I.id_opera = O.id_opera
    WHERE UPPER(O.titlu) = UPPER(TitluOp);

    Rezultat := 'Opera "' || TitluOp || '" se afla in: ';

    FOR i IN 1..ListaExpozitii.COUNT LOOP
        Rezultat := Rezultat || ListaExpozitii(i) || ', ';
    END LOOP;
    Rezultat := Rezultat || ' ';
END;
```

```

END LOOP;

IF ListaExpozitiei.COUNT = 0 THEN
    RAISE ERR_4;    --OPERA CAUTATA NU FACE PARTE DIN NICIO EXPOZITIE
END IF;

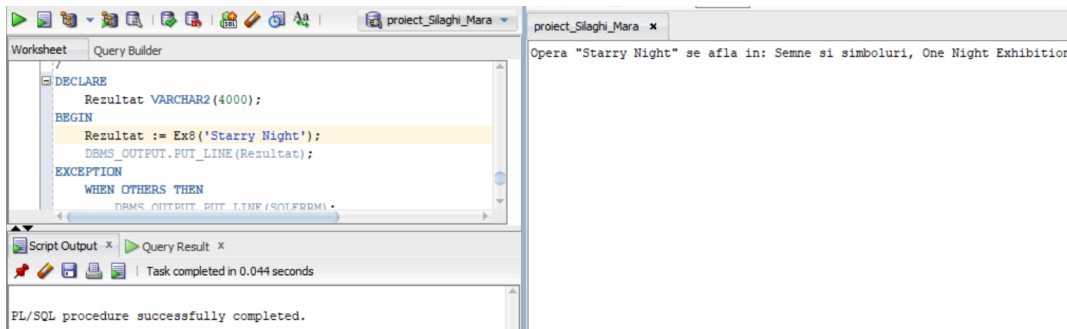
-- Eliminăm ultima virgula
Rezultat := RTRIM(Rezultat, ', ');

RETURN Rezultat;
EXCEPTION
    WHEN ERR_1 THEN
        DBMS_OUTPUT.PUT_LINE('Titlul nu poate fi gol!');
        RETURN -1;
    WHEN ERR_2 THEN
        DBMS_OUTPUT.PUT_LINE('Opera nu exista în baza de date!');
        RETURN -1;
    WHEN ERR_3 THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multe opere cu acest
titlu!');
        RETURN -1;
    WHEN ERR_4 THEN
        DBMS_OUTPUT.PUT_LINE('Opera cautata nu face parte din
nicio expozitie!');
        RETURN -1;
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20003, SQLERRM);
        RETURN -1;

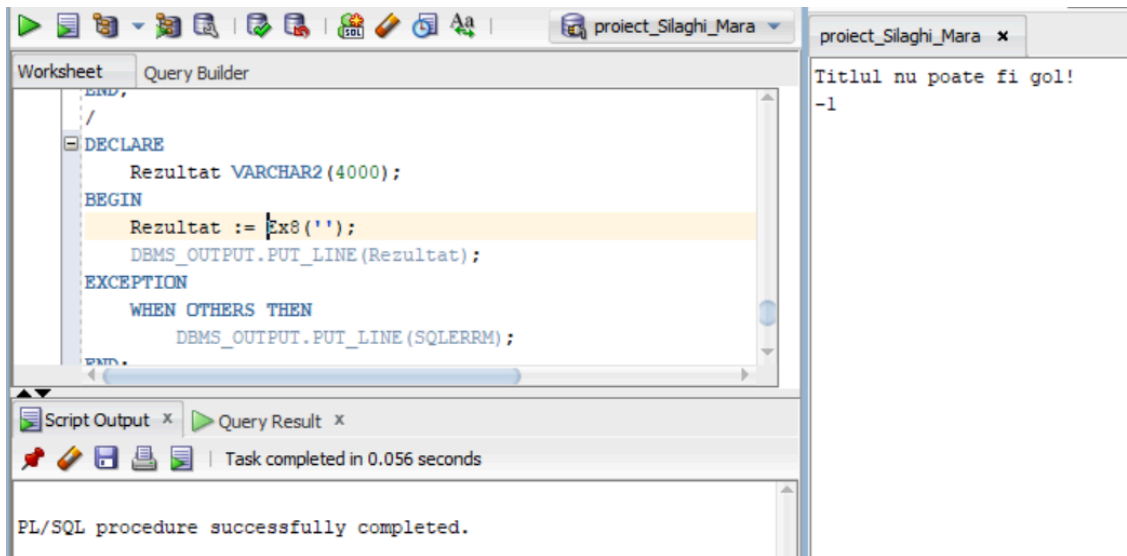
END;
/
DECLARE
    Rezultat VARCHAR2(4000);
BEGIN
    Rezultat := Ex8('Starry Night');
    DBMS_OUTPUT.PUT_LINE(Rezultat);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE(SQLERRM);
END;
/

```

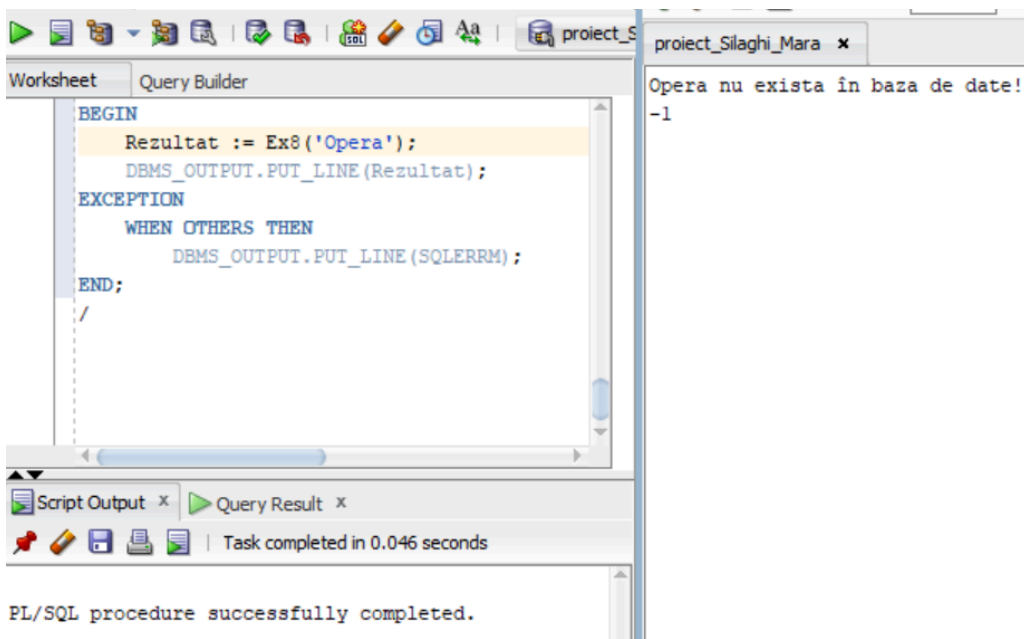
Pentru input-ul ‘Starry Night’:



Pentru input-ul ‘’:

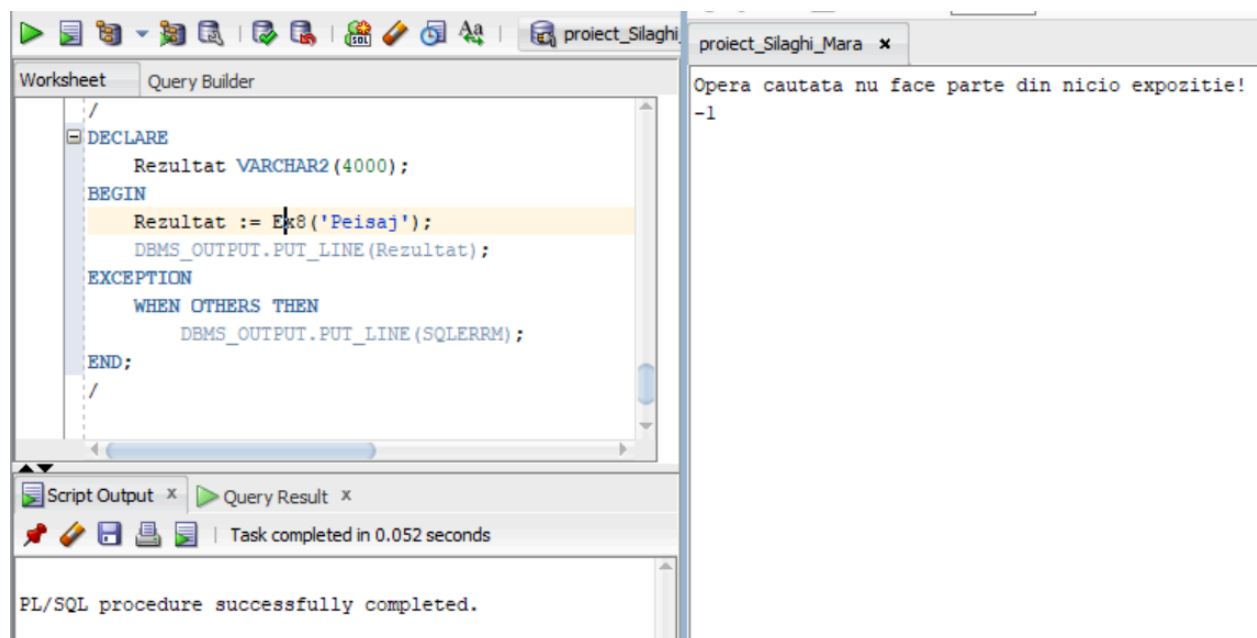


Pentru input-ul ‘Opera’:





Pentru input-ul 'Peisaj':



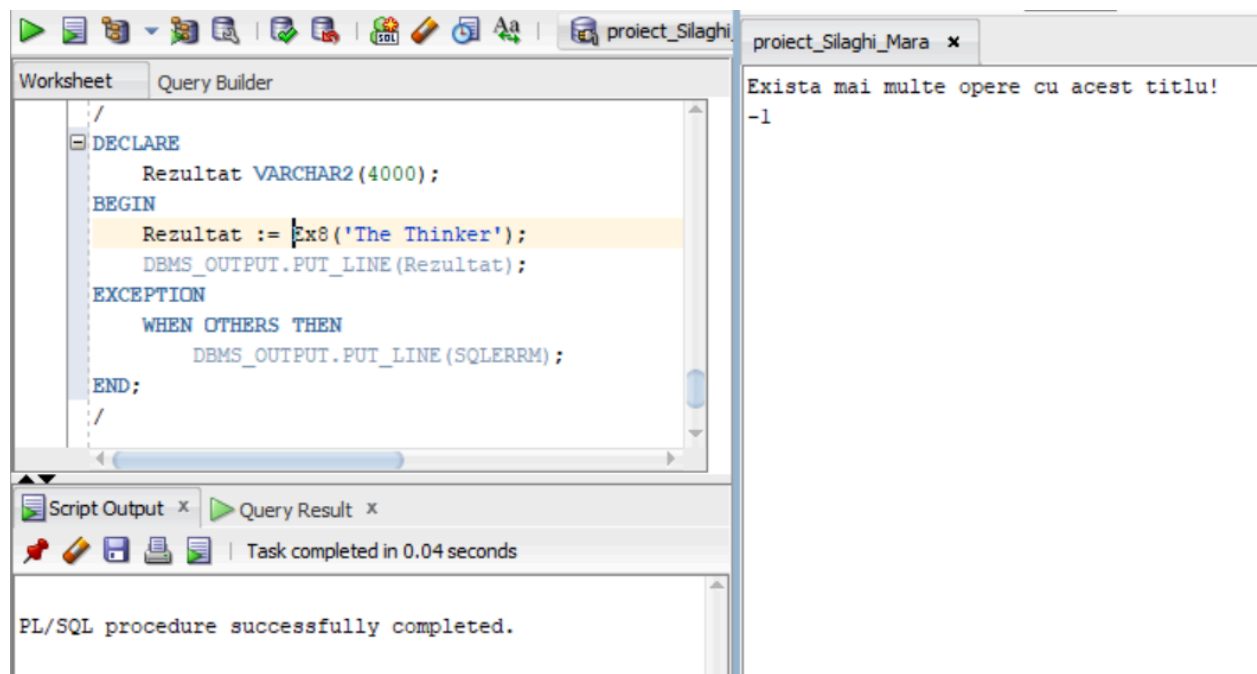
The screenshot shows the Oracle SQL Developer interface. The 'Query Builder' tab is active, displaying a PL/SQL procedure. The procedure declares a variable 'Rezultat' of type VARCHAR2(4000). In the 'BEGIN' block, it assigns the value 'Peisaj' to 'Rezultat' using the 'Ex8' function, then prints it with 'DBMS\_OUTPUT.PUT\_LINE'. An 'EXCEPTION' block handles 'OTHERS' by printing the SQL error message. The status bar indicates 'Task completed in 0.052 seconds' and 'PL/SQL procedure successfully completed.' The 'Script Output' pane on the right shows the output: 'Opera cautata nu face parte din nicio expozitie!' followed by a line separator '-1'.

```
DECLARE
  Rezultat VARCHAR2(4000);
BEGIN
  Rezultat := Ex8('Peisaj');
  DBMS_OUTPUT.PUT_LINE(Rezultat);
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE(SQLERRM);
END;
```

Script Output x Query Result x  
Task completed in 0.052 seconds  
PL/SQL procedure successfully completed.

proiect\_Silaghi\_Mara x  
Opera cautata nu face parte din nicio expozitie!  
-1

Pentru input-ul 'The Thinker':



The screenshot shows the Oracle SQL Developer interface. The 'Query Builder' tab is active, displaying a PL/SQL procedure. The procedure declares a variable 'Rezultat' of type VARCHAR2(4000). In the 'BEGIN' block, it assigns the value 'The Thinker' to 'Rezultat' using the 'Ex8' function, then prints it with 'DBMS\_OUTPUT.PUT\_LINE'. An 'EXCEPTION' block handles 'OTHERS' by printing the SQL error message. The status bar indicates 'Task completed in 0.04 seconds' and 'PL/SQL procedure successfully completed.' The 'Script Output' pane on the right shows the output: 'Exista mai multe opere cu acest titlu!' followed by a line separator '-1'.

```
DECLARE
  Rezultat VARCHAR2(4000);
BEGIN
  Rezultat := Ex8('The Thinker');
  DBMS_OUTPUT.PUT_LINE(Rezultat);
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE(SQLERRM);
END;
```

Script Output x Query Result x  
Task completed in 0.04 seconds  
PL/SQL procedure successfully completed.

proiect\_Silaghi\_Mara x  
Exista mai multe opere cu acest titlu!  
-1

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.

```
--Pentru o galerie data, sa se afiseze veniturile si nr de angajati. Daca nu are
--venituri, sa se afiseze un mesaj.
CREATE OR REPLACE PROCEDURE Ex9(NumeGalerie Galerie.denumire%type)
IS
    v_denumire GALERIE.DENUMIRE%TYPE;
    v_venituri NUMBER;
    v_nr_angajati NUMBER;
    v_id_galerie NUMBER;
    ERR_1 EXCEPTION;
    ERR_2 EXCEPTION;
BEGIN
    IF NumeGalerie IS NULL OR NumeGalerie = '' THEN
        RAISE ERR_1;
    END IF;

    --pentru verificarea too_many_rows
    SELECT ID_GALERIE INTO v_id_galerie
    FROM GALERIE
    WHERE DENUMIRE = NumeGalerie;

    WITH TABEL AS (SELECT G.ID_GALERIE, G.DENUMIRE, COUNT(B.ID_BILET) * E.pret_bilet
    AS VENIT
                    FROM GALERIE G
                    LEFT JOIN PREZENTARE P ON P.ID_GALERIE = G.ID_GALERIE
                    LEFT JOIN EXPOZITIE E ON E.ID_EXPOZITIE = P.ID_EXPOZITIE
                    LEFT JOIN BILET B ON B.ID_PREZENTARE = P.ID_PREZENTARE
                    WHERE UPPER(G.DENUMIRE) = UPPER(NumeGalerie)
                    GROUP BY G.ID_GALERIE, G.DENUMIRE, E.pret_bilet),
    --se calculeaza venitul pentru fiecare prezentare
    NR_ANGAJATI AS (SELECT ID_GALERIE, COUNT(ID_ANGAJAT) AS NR_ANGAJATI
                    FROM ANGAJAT
                    GROUP BY ID_GALERIE)
    SELECT T.DENUMIRE, SUM(VENIT) AS VENITURI, A.NR_ANGAJATI
    --se insumeaza prezentarile din aceeasi galerie
    INTO v_denumire, v_venituri, v_nr_angajati
    FROM TABEL T
    JOIN NR_ANGAJATI A ON A.ID_GALERIE = T.ID_GALERIE
    GROUP BY T.DENUMIRE, A.NR_ANGAJATI;

    IF V_VENITURI = 0 THEN
        RAISE ERR_2;
    END IF;
```

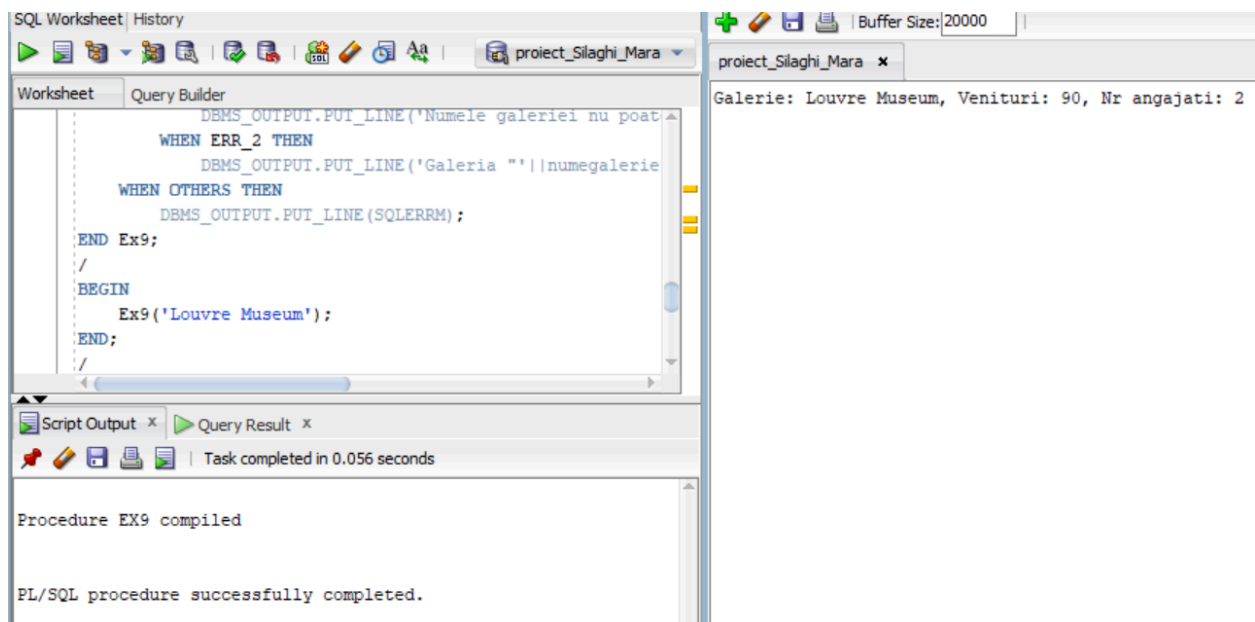
```

        DBMS_OUTPUT.PUT_LINE('Galerie: ' || v_denumire || ', Venituri: ' || v_venituri
|| ', Nr angajati: ' || v_nr_angajati);

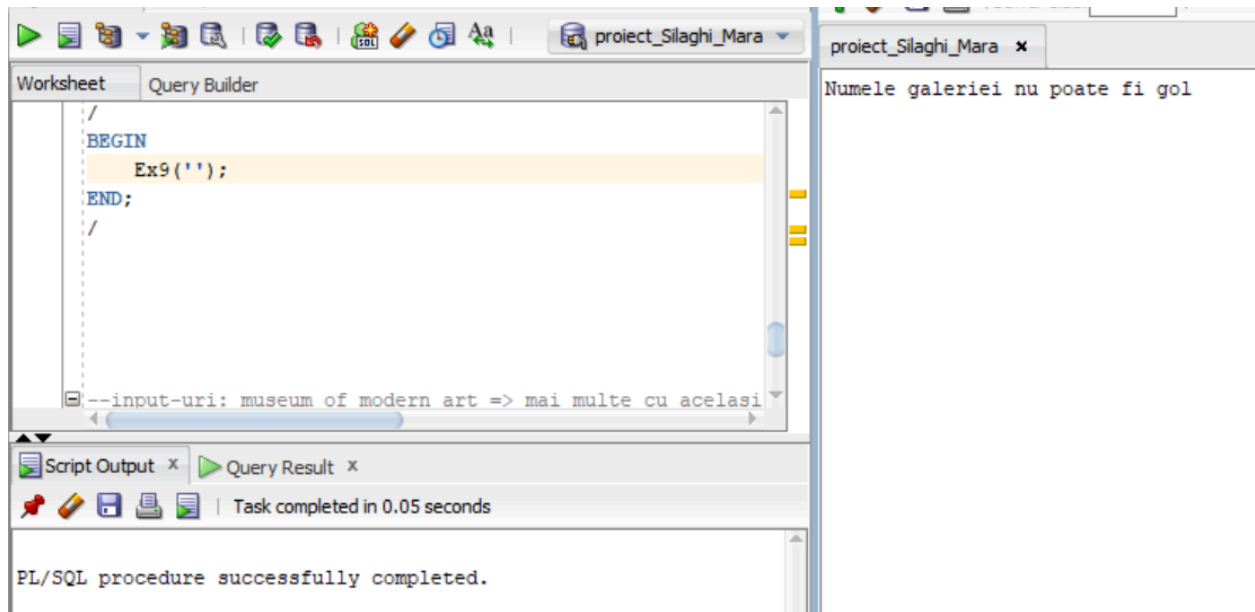
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista o galerie cu numele dat');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Eroare: Mai multe galerii cu acelasi nume');
        WHEN ERR_1 THEN
            DBMS_OUTPUT.PUT_LINE('Numele galeriei nu poate fi gol');
        WHEN ERR_2 THEN
            DBMS_OUTPUT.PUT_LINE('Galeria "' || numegalerie || '" nu are
venituri');
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE(SQLERRM);
END Ex9;
/
BEGIN
    Ex9('Galeria Nationala');
END;
/

```

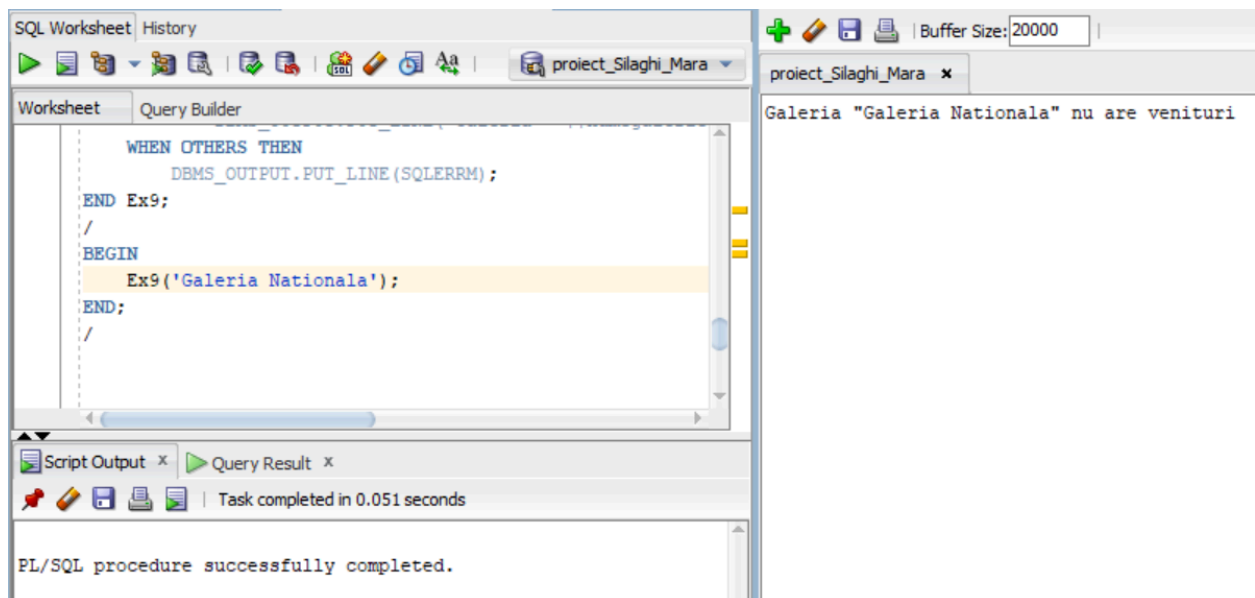
Pentru input-ul 'Louvre Museum':



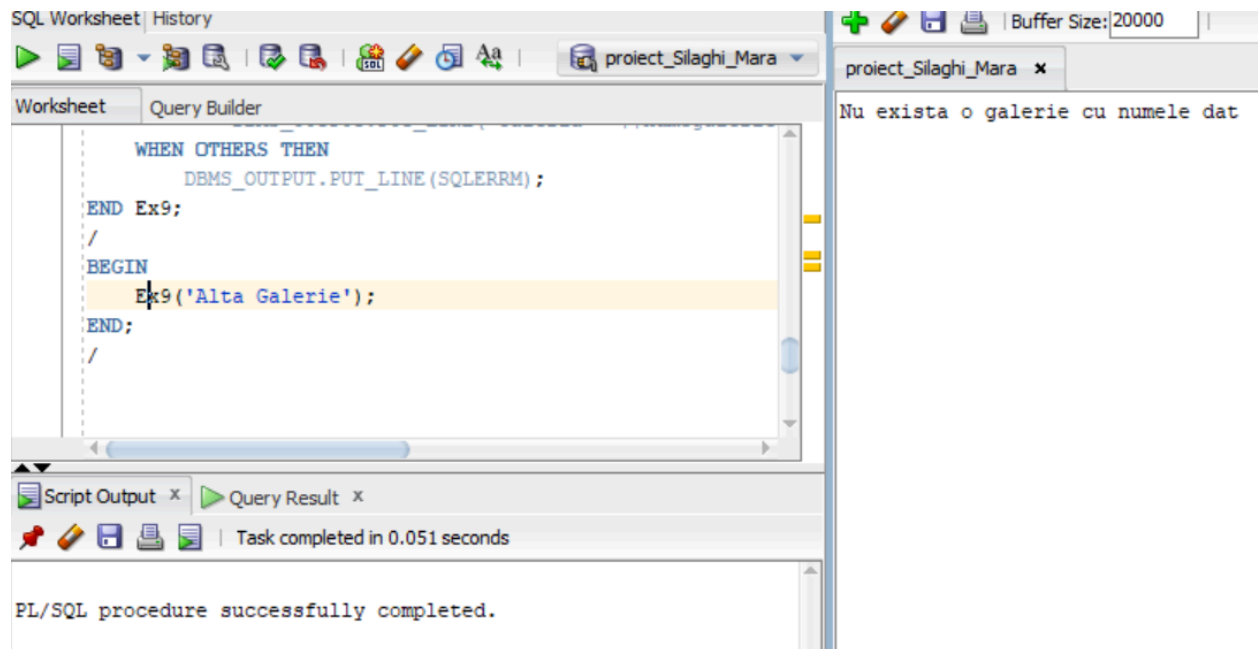
Pentru input-ul ‘’:



Pentru input-ul ‘Galeria Nationala’:



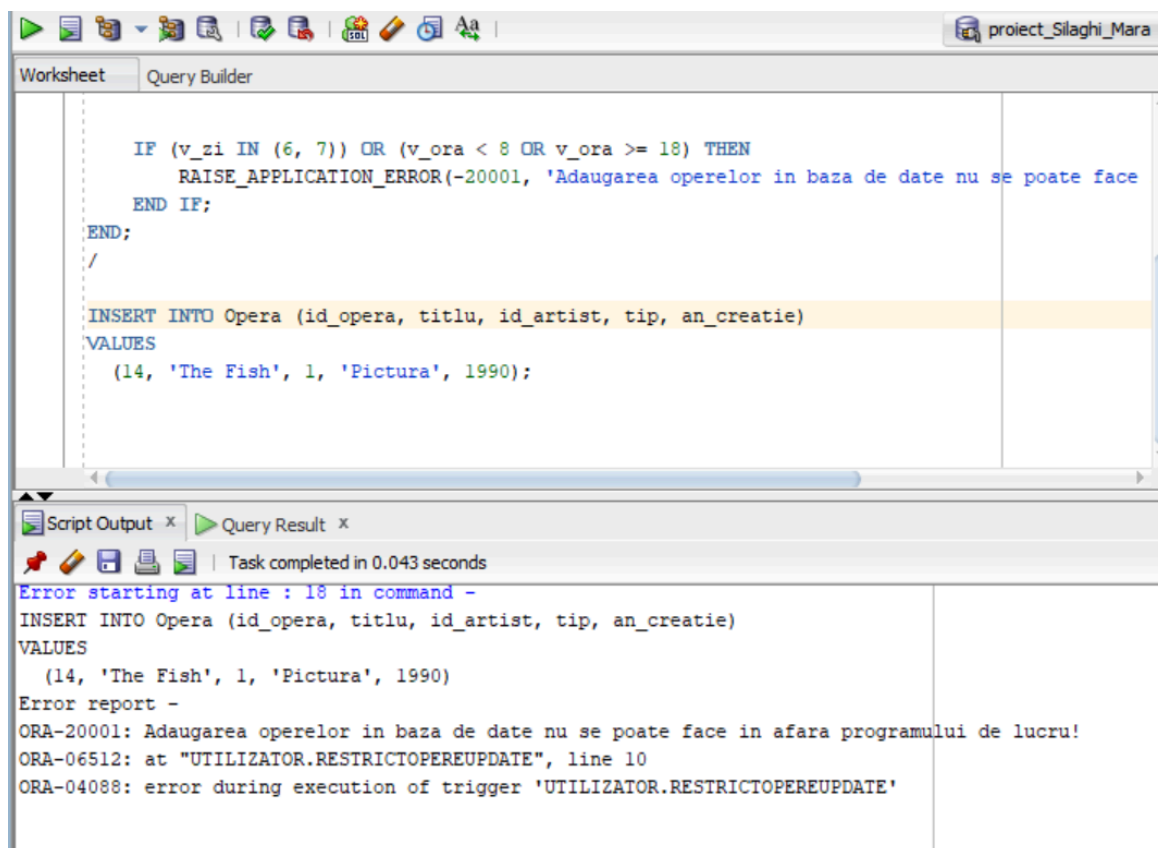
Pentru input-ul 'Alta Galerie':



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

```
--nu se pot adauga opere in afara programului
CREATE OR REPLACE TRIGGER Ex10
BEFORE INSERT OR UPDATE OR DELETE ON Opera
DECLARE
    v_ora NUMBER;
    v_zi NUMBER;
BEGIN
    v_ora := TO_NUMBER(TO_CHAR(SYSDATE, 'HH24'));
    v_zi := TO_CHAR(SYSDATE, 'D');

    -- Restrictionam modificarile in weekend (sambata și duminica) si in afara
    orelor de lucru (18:00 - 08:00)
    IF (v_zi IN (6, 7)) OR (v_ora < 8 OR v_ora >= 18) THEN
        RAISE_APPLICATION_ERROR(-20001, 'Adaugarea operelor in baza de date nu se
        poate face in afara programului de lucru!');
    END IF;
END;
```



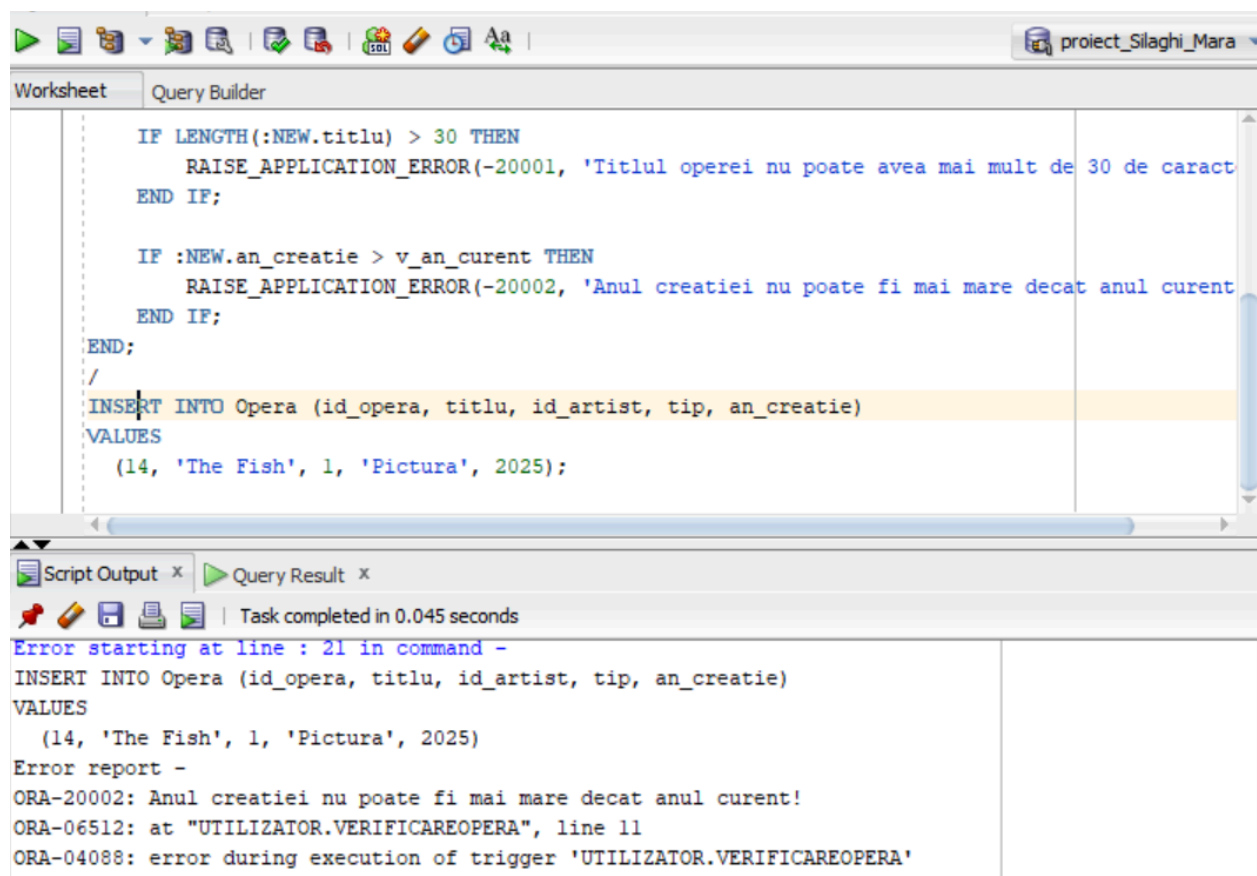
## 11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.

```
--nu se pot adauga opere cu titlu care are mai mult de 30 de caractere
--sau care are anul de creatie mai mare decat anul curent

CREATE OR REPLACE TRIGGER Ex11
BEFORE INSERT OR UPDATE ON Opera
FOR EACH ROW
DECLARE
    v_an_curent NUMBER;
BEGIN
    v_an_curent := TO_NUMBER(TO_CHAR(SYSDATE, 'YYYY'));

    IF LENGTH(:NEW.titlu) > 30 THEN
        RAISE_APPLICATION_ERROR(-20001, 'Titlul operei nu poate avea mai mult de 30
de caractere!');
    END IF;

    IF :NEW.an_creatie > v_an_curent THEN
        RAISE_APPLICATION_ERROR(-20002, 'Anul creatiei nu poate fi mai mare decat
anul curent!');
    END IF;
END;
```



The screenshot displays the Oracle SQL Developer environment. The top toolbar includes icons for running queries, saving, and other standard database operations. The main window is titled "Worksheet" and "Query Builder". The SQL editor contains the following code:

```
IF LENGTH(:NEW.titlu) > 30 THEN
    RAISE_APPLICATION_ERROR(-20001, 'Titlul operei nu poate avea mai mult de 30 de caract
END IF;

IF :NEW.an_creatie > v_an_curent THEN
    RAISE_APPLICATION_ERROR(-20002, 'Anul creatiei nu poate fi mai mare decat anul curent
END IF;

END;
/
INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie)
VALUES
    (14, 'The Fish', 1, 'Pictura', 2025);
```

The bottom panel shows the "Script Output" and "Query Result" tabs. The "Script Output" tab is active, displaying the following error message:

```
Error starting at line : 21 in command -
INSERT INTO Opera (id_opera, titlu, id_artist, tip, an_creatie)
VALUES
    (14, 'The Fish', 1, 'Pictura', 2025)
Error report -
ORA-20002: Anul creatiei nu poate fi mai mare decat anul curent!
ORA-06512: at "UTILIZATOR.VERIFICAREOPERA", line 11
ORA-04088: error during execution of trigger 'UTILIZATOR.VERIFICAREOPERA'
```

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

```
--se contorizeaza in tabelul 'istoric' fiecare operatie de tip ldd create, drop sau
--alter
CREATE TABLE ISTORIC (
UTILIZATOR VARCHAR2(50),
EVENIMENT VARCHAR2(50),
NUME_OBIECT VARCHAR2(50)
);

CREATE OR REPLACE TRIGGER Ex12
AFTER CREATE OR ALTER OR DROP ON SCHEMA
BEGIN
INSERT INTO ISTORIC
VALUES(SYS.LOGIN_USER, SYS.SYSEVENT, SYS.DICTIONARY_OBJ_NAME);
END;
/
CREATE TABLE TABEL
(COD INT PRIMARY KEY
);
ALTER TABLE TABEL
ADD ( NUME VARCHAR2(20));

DROP TABLE TABEL;
SELECT * FROM ISTORIC;
```



Worksheet Query Builder

```

CREATE OR REPLACE TRIGGER Ex12
AFTER CREATE OR ALTER OR DROP ON SCHEMA
BEGIN
INSERT INTO ISTARIC
VALUES(SYS.LOGIN_USER, SYS.SYSEVENT, SYS.DICTIONARY_OBJ_NAME);
END;
/
CREATE TABLE TABEL
(COD INT PRIMARY KEY
);
ALTER TABLE TABEL
ADD ( NUME VARCHAR2(20));

DROP TABLE TABEL;
SELECT * FROM ISTARIC;

```

Script Output x Query Result x

SQL | All Rows Fetched: 4 in 0.007 seconds

	UTILIZATOR	EVENIMENT	NUME_OBJECT
1	UTILIZATOR	CREATE	SYS_C008424
2	UTILIZATOR	CREATE	TABEL
3	UTILIZATOR	ALTER	TABEL
4	UTILIZATOR	DROP	TABEL

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

```

CREATE OR REPLACE PACKAGE PROIECT AS
PROCEDURE Ex6;
PROCEDURE Ex7;
FUNCTION Ex8 (TitluOp Opera.titlu%type) RETURN VARCHAR2;
PROCEDURE Ex9 (NumeGalerie Galerie.denumire%type);
END PROIECT;
/

CREATE OR REPLACE PACKAGE BODY PROIECT
AS

PROCEDURE EX6 AS
--pentru tabelul expozitii
TYPE EXPOZITIIREC IS RECORD (
    id_galerie GALERIE.ID_GALERIE%TYPE,
    id_expozitie EXPOZITIE.ID_EXPOZITIE%TYPE,
    nume_expozitie EXPOZITIE.NUME_EXPOZITIE%TYPE
);

--salvam primele 3 galerii in varray
TYPE vector IS VARRAY(3) OF GALERIE%ROWTYPE;
GALERII vector:= vector();

--in expozitii salvam expozitia si galeria din care face parte

```

```

TYPE imbricat IS TABLE OF EXPOZITIIREC;
EXPOZITII imbricat := imbricat();

--salvam operele intr-un tablou indexat
TYPE idx IS TABLE OF VARCHAR(200) INDEX BY PLS_INTEGER;
OPERE idx;
NR_EXPOZITII NUMBER(6);

BEGIN
    --primele 3 galerii in functie de numarul de expozitii
    WITH TABEL AS (SELECT G.DENUMIRE, COUNT(E.id_expozitie)
                     FROM GALERIE G
                     JOIN PREZENTARE P ON P.ID_GALERIE = G.ID_GALERIE
                     JOIN EXPOZITIE E ON E.ID_EXPOZITIE = P.ID_EXPOZITIE
                     GROUP BY G.DENUMIRE
                     order by COUNT(E.id_expozitie) desc)
    SELECT G.ID_GALERIE, G.DENUMIRE, G.LOCATIE
    BULK COLLECT INTO GALERII
    FROM GALERIE G
    JOIN TABEL T ON T.DENUMIRE = G.DENUMIRE
    WHERE ROWNUM<=3;

    --am salvat "expozitii" drept nested, deci mai intai aflam dimensiunea listei
    SELECT COUNT(*)
    INTO NR_EXPOZITII
    FROM EXPOZITIE E
    JOIN PREZENTARE P ON P.ID_EXPOZITIE = E.ID_EXPOZITIE
    JOIN GALERIE G ON G.ID_GALERIE = P.ID_GALERIE;

    expozitii.extend(nr_expozitii + 1);

    --expozitii
    SELECT G.ID_GALERIE, E.ID_EXPOZITIE, E.NUME_EXPOZITIE
    BULK COLLECT INTO EXPOZITII
    FROM EXPOZITIE E
    JOIN PREZENTARE P ON P.ID_EXPOZITIE = E.ID_EXPOZITIE
    JOIN GALERIE G ON G.ID_GALERIE = P.ID_GALERIE
    ORDER BY G.DENUMIRE;

    --luam fiecare galerie din cele 3 si ii afisam expozitiile
    FOR I IN GALERII.first..GALERII.last LOOP
        DBMS_OUTPUT.PUT_LINE('GALERIA ' || GALERII(I).denumire);
        DBMS_OUTPUT.PUT_LINE('-----');

        FOR J IN EXPOZITII.first..EXPOZITII.last LOOP
            IF EXPOZITII(J).ID_GALERIE = GALERII(I).ID_GALERIE THEN
                DBMS_OUTPUT.PUT_LINE('EXPOZITIA ' || EXPOZITII(J).NUME_EXPOZITIE
                ||': ');

                SELECT O.TITLU
                BULK COLLECT INTO OPERE
                FROM OPERA O

```

```

        JOIN IMPRUMUT I ON I.ID_OPERA = O.ID_OPERA
        JOIN EXPOZITIE E ON E.ID_EXPOZITIE = I.ID_EXPOZITIE
        WHERE E.ID_EXPOZITIE = EXPOZITII(J).ID_EXPOZITIE;

        FOR K IN OPERE.FIRST..OPERE.LAST LOOP
            DBMS_OUTPUT.PUT_LINE(OPERE(K));
        END LOOP;
        DBMS_OUTPUT.PUT_LINE(' ');
    END IF;
END LOOP;
DBMS_OUTPUT.PUT_LINE(' ');
END LOOP;

END Ex6;

PROCEDURE Ex7 AS
    -- Primul cursor: Selectează expozițiile bazate pe numărul de vizitatori
    CURSOR CursorExpozitii IS
        WITH TABEL AS (
            SELECT DISTINCT COUNT(B.id_bilet) AS NR
            FROM Expozitie E
            LEFT JOIN Prezentare P ON P.id_expozitie = E.id_expozitie
            --exista expoziții care nu aparțin niciunei prezentări
            LEFT JOIN Bilet B ON P.id_prezentare =
B.id_prezentare
            GROUP BY E.id_expozitie, E.ume_expozitie
            ORDER BY NR DESC
        )
        SELECT E.id_expozitie, E.ume_expozitie, COUNT(B.id_bilet) as nr_vizitatori
        FROM Expozitie E
        LEFT JOIN Prezentare P ON P.id_expozitie = E.id_expozitie
        LEFT JOIN Bilet B ON P.id_prezentare = B.id_prezentare
        GROUP BY E.id_expozitie, E.ume_expozitie
        HAVING COUNT(B.id_bilet) IN (SELECT NR FROM TABEL WHERE ROWNUM <= 2)
        ORDER BY COUNT(B.ID_BILET) DESC;

    -- Al doilea cursor: Obține informații despre operele de artă și artiști pentru
o expoziție specifică
    CURSOR CursorOpereArtisti(idExpozitie EXPOZITIE.ID_EXPOZITIE%TYPE) IS
        SELECT O.titlu, A.ume
        FROM Opera O
        JOIN Artist A ON O.id_artist = A.id_artist
        JOIN Imprumut I ON O.id_opera = I.id_opera
        WHERE I.id_expozitie = idExpozitie;

    recExpozitie CursorExpozitii%ROWTYPE;
    recOperaArtist CursorOpereArtisti%ROWTYPE;
BEGIN

```

```

OPEN CursorExpozitii;
LOOP
    FETCH CursorExpozitii INTO recExpozitie;
    EXIT WHEN CursorExpozitii%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Expozitie: ' || recExpozitie.ume_expozitie ||
                          ', Vizitatori: ' || recExpozitie.nr_vizitatori);

    OPEN CursorOpereArtisti(recExpozitie.id_expozitie);
    LOOP
        FETCH CursorOpereArtisti INTO recOperaArtist;
        EXIT WHEN CursorOpereArtisti%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Opera: ' || recOperaArtist.titlu ||
                              ', Artist: ' || recOperaArtist.ume);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
    CLOSE CursorOpereArtisti;
END LOOP;
CLOSE CursorExpozitii;
END Ex7;

```

```

FUNCTION Ex8 (TitluOp Opera.titlu%type)
RETURN VARCHAR2 IS
    TYPE lista IS TABLE OF VARCHAR2(255);
    ListaExpozitii lista;
    Rezultat VARCHAR2(4000);
    ContorOpera NUMBER;
    --exceptii
    ERR_1 EXCEPTION;
    ERR_2 EXCEPTION;
    ERR_3 EXCEPTION;
    ERR_4 EXCEPTION;
BEGIN
    IF TitluOp IS NULL OR TitluOp = '' THEN
        RAISE ERR_1;    --TITLU NUL
    END IF;

    SELECT COUNT(*)
    INTO ContorOpera
    FROM Opera
    WHERE UPPER(titlu) = UPPER(TitluOp);

    IF ContorOpera = 0 THEN
        RAISE ERR_2;    --OPERA NU EXISTA IN BAZA DE DATE
    END IF;

    IF ContorOpera >1 THEN
        RAISE ERR_3;    --EXISTA MAI MULTE OPERE CU ACELASI TITLU
    END IF;

```

```

END IF;

SELECT E.ume_expozitie
BULK COLLECT INTO ListaExpozitii
FROM Expozitie E
JOIN Imprumut I ON E.id_expozitie = I.id_expozitie
JOIN Opera O ON I.id_opera = O.id_opera
WHERE UPPER(O.titlu) = UPPER(TitluOp);

Rezultat := 'Opera "' || TitluOp || '" se afla in: ';

FOR i IN 1..ListaExpozitii.COUNT LOOP
    Rezultat := Rezultat || ListaExpozitii(i) || ', ';
END LOOP;

IF ListaExpozitii.COUNT = 0 THEN
    RAISE ERR_4;    --OPERA CAUTATA NU FACE PARTE DIN NICIO EXPOZITIE
END IF;

-- Eliminăm ultima virgula
Rezultat := RTRIM(Rezultat, ', ');

RETURN Rezultat;
EXCEPTION
    WHEN ERR_1 THEN
        DBMS_OUTPUT.PUT_LINE('Titlul nu poate fi gol!');
        RETURN -1;
    WHEN ERR_2 THEN
        DBMS_OUTPUT.PUT_LINE('Opera nu exista în baza de date!');
        RETURN -1;
    WHEN ERR_3 THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multe opere cu acest
titlu!');
        RETURN -1;
    WHEN ERR_4 THEN
        DBMS_OUTPUT.PUT_LINE('Opera cautata nu face parte din
nicio expozitie!');
        RETURN -1;
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE(SQLERRM);
        RETURN -1;
END Ex8;

```

```

PROCEDURE Ex9(NumeGalerie Galerie.denumire%type)
IS
    v_denumire GALERIE.DENUMIRE%TYPE;
    v_venituri NUMBER;
    v_nr_angajati NUMBER;
    v_id_galerie NUMBER;
    ERR_1 EXCEPTION;

```

```

        ERR_2 EXCEPTION;
BEGIN
    IF NumeGalerie IS NULL OR NumeGalerie = '' THEN
        RAISE ERR_1;
    END IF;

    --pentru verificarea too_many_rows
    SELECT ID_GALERIE INTO v_id_galerie
    FROM GALERIE
    WHERE DENUMIRE = NumeGalerie;

    WITH TABEL AS (SELECT G.ID_GALERIE, G.DENUMIRE, COUNT(B.ID_BILET) * E.pret_bilet
    AS VENIT
                    FROM GALERIE G
                    LEFT JOIN PREZENTARE P ON P.ID_GALERIE = G.ID_GALERIE
                    LEFT JOIN EXPOZITIE E ON E.ID_EXPOZITIE = P.ID_EXPOZITIE
                    LEFT JOIN BILET B ON B.ID_PREZENTARE = P.ID_PREZENTARE
                    WHERE UPPER(G.DENUMIRE) = UPPER(NumeGalerie)
                    GROUP BY G.ID_GALERIE, G.DENUMIRE, E.pret_bilet),
        --se calculeaza venitul pentru fiecare prezentare
    NR_ANGAJATI AS (SELECT ID_GALERIE, COUNT(ID_ANGAJAT) AS NR_ANGAJATI
                    FROM ANGAJAT
                    GROUP BY ID_GALERIE)
    SELECT T.DENUMIRE, SUM(VENIT) AS VENITURI, A.NR_ANGAJATI
    --se insumeaza prezentarile din aceeasi galerie
    INTO v_denumire, v_venituri, v_nr_angajati
    FROM TABEL T
    JOIN NR_ANGAJATI A ON A.ID_GALERIE = T.ID_GALERIE
    GROUP BY T.DENUMIRE, A.NR_ANGAJATI;

    IF V_VENITURI = 0 THEN
        RAISE ERR_2;
    END IF;

    DBMS_OUTPUT.PUT_LINE('Galerie: ' || v_denumire || ', Venituri: ' || v_venituri
    || ', Nr angajati: ' || v_nr_angajati);

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista o galerie cu numele dat');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Eroare: Mai multe galerii cu acelasi nume');
        WHEN ERR_1 THEN
            DBMS_OUTPUT.PUT_LINE('Numele galeriei nu poate fi gol');
        WHEN ERR_2 THEN
            DBMS_OUTPUT.PUT_LINE('Galeria "' || numegalerie || '" nu are
venituri');
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE(SQLERRM);
    END Ex9;

END PROIECT;

```

```

/
-- testare package
EXECUTE PROIECT.EX6;
/
EXECUTE PROIECT.EX7;
/
BEGIN
DBMS_OUTPUT.PUT_LINE(PROIECT.Ex8('Starry Night'));
END;
/
EXECUTE PROIECT.Ex9('Galeria Nationala');

```

SQL Worksheet History

project\_Silaghi\_Mara

Worksheet Query Builder

```

END PROIECT;
/
-- testare package
EXECUTE PROIECT.EX6;
/
EXECUTE PROIECT.EX7;
/
BEGIN
DBMS_OUTPUT.PUT_LINE(PROIECT.Ex8('Starry Night'));
END;
/
EXECUTE PROIECT.Ex9('Galeria Nationala');
/

```

Script Output x Query Result x

Task completed in 0.057 seconds

PL/SQL procedure successfully completed.

project\_Silaghi\_Mara x

GALERIA Tate Modern

---

EXPOZITIA Semne si simboluri:  
Starry Night  
Guernica  
Pieta

EXPOZITIA One Night Exhibition:  
Starry Night  
Water Lilies  
The Thinker

EXPOZITIA Retrospectiva:  
The Two Fridas

---

GALERIA Louvre Museum

---

EXPOZITIA Retrospectiva:  
The Two Fridas

EXPOZITIA Between Worlds and Traditions:  
The Two Fridas

SQL Worksheet History

project\_Silaghi\_Mara

Worksheet Query Builder

```

END PROIECT;
/
-- testare package
EXECUTE PROIECT.EX6;
/
EXECUTE PROIECT.EX7;
/
BEGIN
DBMS_OUTPUT.PUT_LINE(PROIECT.Ex8('Starry Night'));
END;
/
EXECUTE PROIECT.Ex9('Galeria Nationala');
/

```

Script Output x Query Result x

Task completed in 0.062 seconds

PL/SQL procedure successfully completed.

project\_Silaghi\_Mara x

Expozitie: Semne si simboluri, Vizitatori: 5  
Opera: Starry Night, Artist: Vincent van Gogh  
Opera: Guernica, Artist: Pablo Picasso  
Opera: Pieta, Artist: Michelangelo

Expozitie: Gardens, Vizitatori: 3  
Opera: Water Lilies, Artist: Claude Monet  
Opera: Bird in Space, Artist: Constantin Brancusi

Expozitie: One Night Exhibition, Vizitatori: 3  
Opera: Starry Night, Artist: Vincent van Gogh  
Opera: Water Lilies, Artist: Claude Monet  
Opera: The Thinker, Artist: Auguste Rodin

Worksheet Query Builder

```
END PROIECT;  
/  
-- testare package  
EXECUTE PROIECT.EX6;  
/  
EXECUTE PROIECT.EX7;  
/  
BEGIN  
DBMS_OUTPUT.PUT_LINE(PROIECT.Ex8('Starry Night'));  
END;  
/  
EXECUTE PROIECT.Ex9('Galeria Nationala');
```

Script Output x Query Result x

Task completed in 0.055 seconds

PL/SQL procedure successfully completed.

proiect\_Silaghi\_Mara x

Opera "Starry Night" se afla in: Semne si simboluri, One Night Exhibition

Worksheet Query Builder

```
END PROIECT;  
/  
-- testare package  
EXECUTE PROIECT.EX6;  
/  
EXECUTE PROIECT.EX7;  
/  
BEGIN  
DBMS_OUTPUT.PUT_LINE(PROIECT.Ex8('Starry Night'));  
END;  
/  
EXECUTE PROIECT.Ex9('Galeria Nationala');
```

Script Output x Query Result x

Task completed in 0.058 seconds

PL/SQL procedure successfully completed.

proiect\_Silaghi\_Mara x

Galeria "Galeria Nationala" nu are venituri