

Sisteme de Gestiune a Bazelor de Date

Proiect

Sistem de Gestiune a Restaurării Patrimoniului Cultural

Student: Voaides-Negustor Robert-Ionuț

Grupa: 251

Seria: 25

Anul Universitar: 2025-2026

Cuprins

1	Descrierea Bazei de Date	3
1.1	Scenariul Real Modelat	3
1.2	Utilitatea Bazei de Date	3
2	Diagrama Entitate-Relație (ERD)	4
3	Diagrama Conceptuală	5
4	Implementarea Structurii	6
5	Popularea Bazei de Date	9
6	Utilizarea Colecțiilor (Tablouri/Vectori)	12
7	Utilizarea Cursorilor	15
8	Funcție Stocată	17
9	Procedură Stocată	18
10	Trigger LMD la Nivel de Comandă	19
11	Trigger LMD la Nivel de Linie	20
12	Trigger LDD	21
13	Pachet PL/SQL (Opțional)	22

Introducere

Infrastructura utilizata

- **Sistem de operare:** Fedora Linux
- **Containerizare:** Docker
- **SGBD:** Oracle Database XE 21c (container `oracle-xe`, imagine `gvenzl/oracle-xe:21-slim`)
- **Mediu de lucru:** Visual Studio Code și DataGrip

1. Descrierea Bazei de Date

1.1 Scenariul Real Modelat

Această bază de date a fost proiectată pentru a gestiona activitatea unei instituții sau companii specializate în conservarea și restaurarea patrimoniului istoric. Scenariul real modelat urmărește ciclul de viață complet al unui proiect de restaurare, de la identificarea obiectivului (monumentul) și asigurarea surselor de finanțare, până la execuția efectivă și monitorizarea calității lucrărilor.

Baza de date surprinde interacțiunile complexe dintre diversele entități implicate într-un șantier de restaurare: monumentele istorice, echipele multidisciplinare de experți, resursele materiale necesare, instituțiile statului care avizează lucrările și sursele de finanțare.

1.2 Utilitatea Bazei de Date

Implementarea acestui sistem informatic răspunde nevoii de a avea o evidență clară, transparentă și centralizată asupra modului în care este conservat patrimoniul. Principalele funcționalități și beneficii sunt:

- **Trasabilitatea intervențiilor:** Sistemul permite crearea unui istoric detaliat al tuturor restaurărilor efectuate asupra unui monument. Se poate verifica oricând ce materiale s-au utilizat și ce experți au participat, informații vitale pentru viitoarele lucrări de întreținere.
- **Gestionarea resurselor:** Baza de date rezolvă problema alocării eficiente a resurselor. Deoarece experții și stocurile de materiale pot fi partajate între mai multe șantiere simultan, sistemul evidențiază clar distribuția acestora.
- **Monitorizarea financiară și legală:** Permite urmărirea bugetelor alocate din diverse surse de finanțare, precum și starea avizelor necesare de la autoritățile competente (ex: Ministerul Culturii, Primărie) și rezultatele inspecțiilor de șantier.

2. Diagrama Entitate-Relație (ERD)

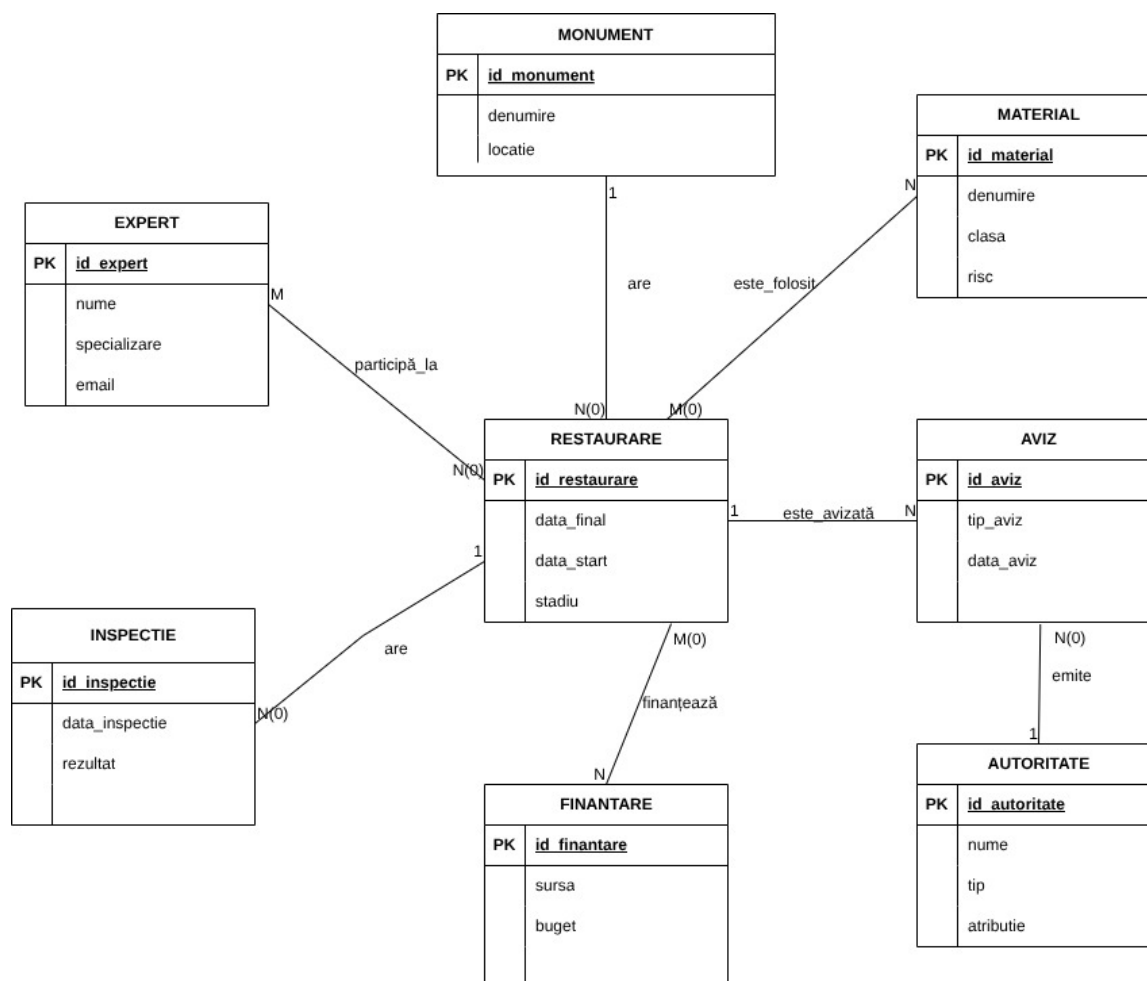


Figura 2.1: Diagrama ERD a bazei de date

3. Diagrama Conceptuală

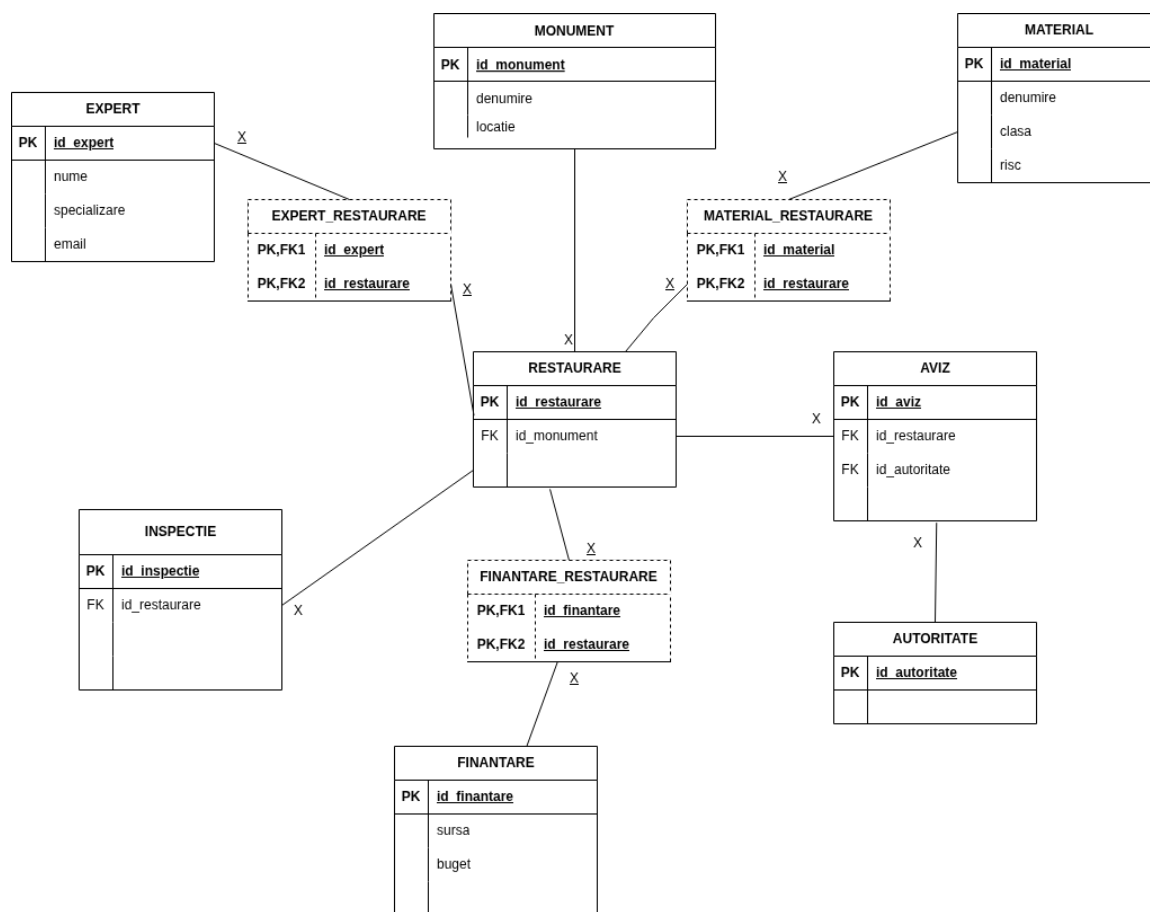


Figura 3.1: Diagrama Conceptuală

4. Implementarea Structurii

Mai jos sunt prezentate instrucțiunile de creare a tabelelor și constrângerile de integritate.

```
1 CREATE TABLE monument (
2     id_monument NUMBER(5) CONSTRAINT pk_monument PRIMARY KEY,
3     denumire     VARCHAR2(100) NOT NULL,
4     locatie      VARCHAR2(100) NOT NULL
5 );
6
7
8 CREATE TABLE expert (
9     id_expert     NUMBER(5) CONSTRAINT pk_expert PRIMARY KEY,
10    nume           VARCHAR2(50) NOT NULL,
11    specializare   VARCHAR2(50) NOT NULL,
12    email          VARCHAR2(100) CONSTRAINT uq_expert_email UNIQUE
13 );
14
15
16 CREATE TABLE material (
17     id_material   NUMBER(5) CONSTRAINT pk_material PRIMARY KEY,
18     denumire      VARCHAR2(50) NOT NULL,
19     clasa         VARCHAR2(20),
20     risc          VARCHAR2(20)
21 );
22
23
24 CREATE TABLE finantare (
25     id_finantare  NUMBER(5) CONSTRAINT pk_finantare PRIMARY KEY,
26     sursa         VARCHAR2(50) NOT NULL,
27     buget         NUMBER(12, 2) CONSTRAINT ck_buget_pozitiv CHECK (buget
28 > 0)
29 );
30
31 CREATE TABLE autoritate (
32     id_autoritate NUMBER(5) CONSTRAINT pk_autoritate PRIMARY KEY,
33     nume          VARCHAR2(100) NOT NULL,
34     tip           VARCHAR2(50),
35     atributie     VARCHAR2(100)
36 );
37
38 CREATE TABLE restaurare (
39     id_restaurare NUMBER(5) CONSTRAINT pk_restaurare PRIMARY KEY,
40     id_monument   NUMBER(5) NOT NULL,
41     data_start    DATE DEFAULT SYSDATE,
42     data_final    DATE,
43     stadiu        VARCHAR2(20) CONSTRAINT ck_stadiu CHECK (stadiu IN ('
44 Planificat', 'In executie', 'Finalizat', 'Suspendat')),
45     CONSTRAINT fk_restaurare_monument FOREIGN KEY (id_monument)
46 REFERENCES monument(id_monument),
47     CONSTRAINT ck_date_valide CHECK (data_final >= data_start)
48 );
49
50 CREATE TABLE aviz (
51     id_aviz       NUMBER(5) CONSTRAINT pk_aviz PRIMARY KEY,
52     id_restaurare  NUMBER(5) NOT NULL,
```

```

52     id_autoritate NUMBER(5) NOT NULL,
53     tip_aviz       VARCHAR2(50) NOT NULL,
54     data_aviz      DATE DEFAULT SYSDATE,
55     CONSTRAINT fk_aviz_restaurare FOREIGN KEY (id_restaurare) REFERENCES
56     restaurare(id_restaurare),
57     CONSTRAINT fk_aviz_autoritate FOREIGN KEY (id_autoritate) REFERENCES
58     autoritate(id_autoritate)
59 );
60
61 CREATE TABLE inspectie (
62     id_inspectie   NUMBER(5) CONSTRAINT pk_inspectie PRIMARY KEY,
63     id_restaurare  NUMBER(5) NOT NULL,
64     data_inspectie DATE DEFAULT SYSDATE,
65     rezultat       VARCHAR2(200),
66     CONSTRAINT fk_inspectie_restaurare FOREIGN KEY (id_restaurare)
67     REFERENCES restaurare(id_restaurare)
68 );
69
70 CREATE TABLE expert_restaurare (
71     id_expert      NUMBER(5),
72     id_restaurare  NUMBER(5),
73     CONSTRAINT pk_expert_restaurare PRIMARY KEY (id_expert,
74     id_restaurare),
75     CONSTRAINT fk_er_expert FOREIGN KEY (id_expert) REFERENCES expert(
76     id_expert),
77     CONSTRAINT fk_er_restaurare FOREIGN KEY (id_restaurare) REFERENCES
78     restaurare(id_restaurare)
79 );
80
81 CREATE TABLE material_restaurare (
82     id_material    NUMBER(5),
83     id_restaurare  NUMBER(5),
84     CONSTRAINT pk_material_restaurare PRIMARY KEY (id_material,
85     id_restaurare),
86     CONSTRAINT fk_mr_material FOREIGN KEY (id_material) REFERENCES
87     material(id_material),
88     CONSTRAINT fk_mr_restaurare FOREIGN KEY (id_restaurare) REFERENCES
89     restaurare(id_restaurare)
90 );
91
92 CREATE TABLE finantare_restaurare (
93     id_finantare   NUMBER(5),
94     id_restaurare  NUMBER(5),
95     CONSTRAINT pk_finantare_restaurare PRIMARY KEY (id_finantare,
96     id_restaurare),
97     CONSTRAINT fk_fr_finantare FOREIGN KEY (id_finantare) REFERENCES
98     finantare(id_finantare),
99     CONSTRAINT fk_fr_restaurare FOREIGN KEY (id_restaurare) REFERENCES
100    restaurare(id_restaurare)
101 );

```

Listing 4.1: Crearea tabelelor

Dovada rulării în Oracle:

```
Table MONUMENT created.  
  
Table EXPERT created.  
  
Table MATERIAL created.  
  
Table FINANTARE created.  
  
Table AUTORITATE created.  
  
Table RESTAURARE created.  
  
Table AVIZ created.  
  
Table INSPECTIE created.  
  
Table EXPERT_RESTAURARE created.  
  
Table MATERIAL_RESTAURARE created.  
  
Table FINANTARE_RESTAURARE created.
```

Figura 4.1: Rularea comenzilor CREATE TABLE

5. Popularea Bazei de Date

Au fost inserate minim 5 înregistrări pentru entitățile independente și 10 pentru cele asociative.

```
1 INSERT INTO monument VALUES (1, 'Castelul Peles', 'Sinaia');
2 INSERT INTO monument VALUES (2, 'Biserica Neagra', 'Brasov');
3 INSERT INTO monument VALUES (3, 'Cetatea de Scaun', 'Suceava');
4 INSERT INTO monument VALUES (4, 'Manastirea Voronet', 'Gura Humorului');
5 INSERT INTO monument VALUES (5, 'Cazinoul', 'Constanta');
6
7 INSERT INTO expert VALUES (101, 'Popescu Ion', 'Arhitect', 'popescu.
  i@expert.ro');
8 INSERT INTO expert VALUES (102, 'Ionescu Maria', 'Inginer Structurist',
  'maria.i@expert.ro');
9 INSERT INTO expert VALUES (103, 'Georgescu Vlad', 'Restaurator Pictura',
  'vlad.g@expert.ro');
10 INSERT INTO expert VALUES (104, 'Dumitru Ana', 'Arheolog', 'ana.d@expert
  .ro');
11 INSERT INTO expert VALUES (105, 'Stanescu Dan', 'Manager Proiect', 'dan.
  s@expert.ro');
12
13 INSERT INTO material VALUES (201, 'Piatra de rau', 'Natural', 'Scazut');
14 INSERT INTO material VALUES (202, 'Mortar hidraulic', 'Sintetic', 'Mediu
  ');
15 INSERT INTO material VALUES (203, 'Lemn de stejar tratat', 'Natural', '
  Ridicat');
16 INSERT INTO material VALUES (204, 'Pigment mineral', 'Chimic', 'Mediu');
17 INSERT INTO material VALUES (205, 'Caramida arsa', 'Compozit', 'Scazut')
  ;
18
19 INSERT INTO finantare VALUES (301, 'Ministerul Culturii', 5000000);
20 INSERT INTO finantare VALUES (302, 'Fonduri Europene REGIO', 12000000);
21 INSERT INTO finantare VALUES (303, 'Buget Local Sinaia', 200000);
22 INSERT INTO finantare VALUES (304, 'Donatii Private ONG', 50000);
23 INSERT INTO finantare VALUES (305, 'Granturi Norvegiene', 3500000);
24
25 INSERT INTO autoritate VALUES (401, 'Ministerul Culturii', '
  Guvernamental', 'Avizare patrimoniu');
26 INSERT INTO autoritate VALUES (402, 'Primaria Sinaia', 'Local', '
  Certificat Urbanism');
27 INSERT INTO autoritate VALUES (403, 'ISC Brasov', 'Inspectie', 'Control
  Calitate');
28 INSERT INTO autoritate VALUES (404, 'Directia Judeteana Cultura', '
  Judetean', 'Monitorizare');
29 INSERT INTO autoritate VALUES (405, 'Primaria Constanta', 'Local', '
  Autorizatie Constructie');
30
31 INSERT INTO restaurare VALUES (1001, 1, TO_DATE('01-03-2024','DD-MM-YYYY
  '), TO_DATE('01-12-2025','DD-MM-YYYY'), 'In executie');
32 INSERT INTO restaurare VALUES (1002, 2, TO_DATE('15-05-2023','DD-MM-YYYY
  '), TO_DATE('15-05-2024','DD-MM-YYYY'), 'Finalizat');
33 INSERT INTO restaurare VALUES (1003, 5, TO_DATE('01-01-2020','DD-MM-YYYY
  '), TO_DATE('01-01-2026','DD-MM-YYYY'), 'In executie');
34 INSERT INTO restaurare VALUES (1004, 3, TO_DATE('10-10-2025','DD-MM-YYYY
  '), NULL, 'Planificat');
```

```

35 INSERT INTO restaurare VALUES (1005, 4, TO_DATE('01-06-2024','DD-MM-YYYY'),
    TO_DATE('01-09-2024','DD-MM-YYYY'), 'Suspendat');
36
37 INSERT INTO aviz VALUES (1, 1001, 401, 'Aviz Favorabil', TO_DATE('
    20-02-2024','DD-MM-YYYY'));
38 INSERT INTO aviz VALUES (2, 1001, 402, 'Autorizatie Constructie',
    TO_DATE('25-02-2024','DD-MM-YYYY'));
39 INSERT INTO aviz VALUES (3, 1003, 405, 'Prelungire Autorizatie', TO_DATE
    ('10-01-2024','DD-MM-YYYY'));
40
41
42 INSERT INTO inspectie VALUES (1, 1001, TO_DATE('01-04-2024','DD-MM-YYYY'
    ), 'Conform cu proiectul');
43 INSERT INTO inspectie VALUES (2, 1003, TO_DATE('15-04-2024','DD-MM-YYYY'
    ), 'Degradari neprevazute la fatada');
44 INSERT INTO inspectie VALUES (3, 1002, TO_DATE('20-03-2024','DD-MM-YYYY'
    ), 'Inspectie finala, lucrari conforme');
45 INSERT INTO inspectie VALUES (4, 1004, TO_DATE('12-11-2025','DD-MM-YYYY'
    ), 'Verificare preliminara a santierului');
46 INSERT INTO inspectie VALUES (5, 1005, TO_DATE('25-07-2024','DD-MM-YYYY'
    ), 'Necesita remedieri la acoperis');
47
48 INSERT INTO expert_restaurare VALUES (101, 1001);
49 INSERT INTO expert_restaurare VALUES (102, 1001);
50 INSERT INTO expert_restaurare VALUES (105, 1001);
51 INSERT INTO expert_restaurare VALUES (101, 1003);
52 INSERT INTO expert_restaurare VALUES (102, 1003);
53 INSERT INTO expert_restaurare VALUES (103, 1005);
54 INSERT INTO expert_restaurare VALUES (104, 1004);
55 INSERT INTO expert_restaurare VALUES (105, 1002);
56 INSERT INTO expert_restaurare VALUES (102, 1002);
57 INSERT INTO expert_restaurare VALUES (101, 1002);
58
59 INSERT INTO material_restaurare VALUES (203, 1001);
60 INSERT INTO material_restaurare VALUES (205, 1001);
61 INSERT INTO material_restaurare VALUES (201, 1002);
62 INSERT INTO material_restaurare VALUES (202, 1002);
63 INSERT INTO material_restaurare VALUES (201, 1003);
64 INSERT INTO material_restaurare VALUES (202, 1003);
65 INSERT INTO material_restaurare VALUES (204, 1003);
66 INSERT INTO material_restaurare VALUES (201, 1004);
67 INSERT INTO material_restaurare VALUES (204, 1005);
68 INSERT INTO material_restaurare VALUES (202, 1005);
69
70 INSERT INTO finantare_restaurare VALUES (301, 1001);
71 INSERT INTO finantare_restaurare VALUES (303, 1001);
72 INSERT INTO finantare_restaurare VALUES (304, 1002);
73 INSERT INTO finantare_restaurare VALUES (305, 1002);
74 INSERT INTO finantare_restaurare VALUES (302, 1003);
75 INSERT INTO finantare_restaurare VALUES (301, 1003);
76 INSERT INTO finantare_restaurare VALUES (305, 1003);
77 INSERT INTO finantare_restaurare VALUES (302, 1004);
78 INSERT INTO finantare_restaurare VALUES (301, 1005);
79 INSERT INTO finantare_restaurare VALUES (304, 1005);
80
81 COMMIT;

```

Listing 5.1: Inserarea datelor

Dovada rulării în Oracle:

```
1 row inserted.  
  
1 row inserted.  
  
1 row inserted.  
  
1 row inserted.  
  
1 row inserted.  
  
Commit complete.
```

Figura 5.1: Rezultatul inserării datelor (nu sunt afișate toate insert-urile)

6. Utilizarea Colecțiilor (Tablouri/-Vectori)

Enunțul Problemei

Să se scrie o procedură stocată numită `analiza_complexa_proiect` care primește ca parametru ID-ul unui proiect de restaurare. Procedura trebuie să realizeze următoarele acțiuni folosind colecții:

Să identifice primele 5 materiale folosite în proiect și să le stocheze într-un Varray.

Să colecteze numele tuturor experților care lucrează la acest proiect într-un Nested Table.

Să calculeze un bonus simbolic pentru fiecare expert (bazat pe ID-ul expertului) și să stocheze aceste valori într-un Associative Array , având ID-ul expertului drept cheie. La final, să afișeze un raport sintetic.

Rezolvare PL/SQL

```
1 CREATE OR REPLACE PROCEDURE analiza_complexa_proiect (
2   p_id_restaurare IN restaurare.id_restaurare%TYPE
3 ) IS
4
5   TYPE t_materiale_va IS VARRAY(5) OF VARCHAR2(50);
6   v_lista_materiale t_materiale_va;
7
8   TYPE t_experti_nt IS TABLE OF VARCHAR2(50);
9   v_lista_experti t_experti_nt := t_experti_nt();
10
11  TYPE t_bonusuri_aa IS TABLE OF NUMBER INDEX BY PLS_INTEGER;
12  v_map_bonusuri t_bonusuri_aa;
13
14  v_id_expert expert.id_expert%TYPE;
15
16 BEGIN
17   DBMS_OUTPUT.PUT_LINE('Analiza Proiectului ID: ' || p_id_restaurare);
18
19   SELECT m.denumire
20   BULK COLLECT INTO v_lista_materiale
21   FROM material m
22   JOIN material_restaurare mr ON m.id_material = mr.id_material
23   WHERE mr.id_restaurare = p_id_restaurare
24   AND ROWNUM <= 5;
25
26   DBMS_OUTPUT.PUT_LINE('> Materiale principale (Varray):');
27   IF v_lista_materiale.COUNT > 0 THEN
28     FOR i IN 1..v_lista_materiale.COUNT LOOP
29       DBMS_OUTPUT.PUT_LINE(' ' || i || ' ' || v_lista_materiale(
30 i));
31     END LOOP;
32   ELSE
33     DBMS_OUTPUT.PUT_LINE(' Niciun material gasit.');
```

```

34
35     FOR r IN (
36         SELECT e.id_expert, e.nume
37         FROM expert e
38         JOIN expert_restaurare er ON e.id_expert = er.id_expert
39         WHERE er.id_restaurare = p_id_restaurare
40     ) LOOP
41
42         v_lista_experti.EXTEND;
43         v_lista_experti(v_lista_experti.LAST) := r.nume;
44
45         v_map_bonusuri(r.id_expert) := 1500;
46     END LOOP;
47
48     DBMS_OUTPUT.PUT_LINE('> Lista Experti (Nested Table):');
49     IF v_lista_experti.COUNT > 0 THEN
50         FOR i IN 1..v_lista_experti.COUNT LOOP
51             DBMS_OUTPUT.PUT_LINE('  - ' || v_lista_experti(i));
52         END LOOP;
53     ELSE
54         DBMS_OUTPUT.PUT_LINE('  Niciun expert alocat.');
```

Listing 6.1: Subprogram cu colectii

Apel și Rezultat

```

1 SET SERVEROUTPUT ON;
2 BEGIN
3     analiza_complexa_proiect(1002);
4 END;
```

```
Procedure ANALIZA_COMPLEXA_PROIECT compiled

No errors.
Analiza Proiectului ID: 1002
> Materiale principale (Varray):
  1. Piatra de rau
  2. Mortar hidraulic
> Lista Experți (Nested Table):
  - Popescu Ion
  - Ionescu Maria
  - Stanescu Dan
> Calcul Bonusuri (Associative Array):
  Expert ID 101 primește bonus: 1500 RON
  Expert ID 102 primește bonus: 1500 RON
  Expert ID 105 primește bonus: 1500 RON

PL/SQL procedure successfully completed.
```

Figura 6.1: Execuția cerinței 6

7. Utilizarea Cursorurilor

Enunțul Problemei

Sa se scrie o procedura stocata numita `raport_financiar_locatie` care primeste ca parametru numele unei locatii (oras). Procedura va afisa monumentele din acea locatie si sursele lor de finantare, folosind doua cursoruri:

Cursorul Principal (Independent): Va itera prin toate monumentele care se afla in locatie specificata.

Cursorul Secundar (Parametrizat/Dependent): Pentru fiecare monument gasit de primul cursor, acest al doilea cursor va cauta toate sursele de finantare ale restaurarilor asociate acelui monument, primind ca parametru ID-ul monumentului.

Rezolvare PL/SQL

```
1 CREATE OR REPLACE PROCEDURE raport_financiar_locatie (
2   p_locatie IN monument.locatie%TYPE
3 ) IS
4   CURSOR c_monumente IS
5     SELECT id_monument, denumire
6     FROM monument
7     WHERE UPPER(locatie) = UPPER(p_locatie);
8
9   CURSOR c_finantari (p_id_mon NUMBER) IS
10    SELECT f.sursa, f.buget, r.stadiu
11    FROM finantare f
12    JOIN finantare_restaurare fr ON f.id_finantare = fr.id_finantare
13    JOIN restaurare r ON fr.id_restaurare = r.id_restaurare
14    WHERE r.id_monument = p_id_mon;
15
16   v_total_monument NUMBER;
17
18 BEGIN
19   DBMS_OUTPUT.PUT_LINE('Raport Financiar pentru: ' || p_locatie);
20
21   FOR r_mon IN c_monumente LOOP
22     DBMS_OUTPUT.PUT_LINE('Monument: ' || r_mon.denumire || ' (ID: '
23     || r_mon.id_monument || ')');
24
25     v_total_monument := 0;
26
27     FOR r_fin IN c_finantari(r_mon.id_monument) LOOP
28       DBMS_OUTPUT.PUT_LINE('    > Sursa: ' || r_fin.sursa ||
29       ' | Buget: ' || r_fin.buget ||
30       ' | Stadiu: ' || r_fin.stadiu);
31       v_total_monument := v_total_monument + r_fin.buget;
32     END LOOP;
33
34     IF v_total_monument = 0 THEN
35       DBMS_OUTPUT.PUT_LINE('    > Nu exista finantari inregistrate
36       .');
37     ELSE
```



```

36         DBMS_OUTPUT.PUT_LINE('    > TOTAL MONUMENT: ' ||
    v_total_monument || ' RON');
37     END IF;
38
39     END LOOP;
40 END;
41 /

```

Listing 7.1: Subprogram cu cursoare

Apel și Rezultat

Figura 7.1: Execuția cerinței 7

8. Funcție Stocată

Enunțul Problemei

[Funcție care utilizează 3 tabele și tratează excepțiile...]

Rezolvare PL/SQL

```
1 -- Codul tau aici
```

Listing 8.1: Functia PL/SQL

Apel și Rezultat (Tratare Excepții)

Figura 8.1: Apelul funcției și tratarea excepțiilor

9. Procedură Stocată

Enunțul Problemei

[Procedură cu 5 tabele și excepții proprii...]

Rezolvare PL/SQL

```
1 -- Codul tau aici
```

Listing 9.1: Procedura PL/SQL

Apel și Rezultat

Figura 9.1: Execuția procedurii

10. Trigger LMD la Nivel de Comandă

Definire Trigger

```
1 -- Codul trigger-ului
```

Declanșare

Figura 10.1: Demonstrație declanșare trigger

11. Trigger LMD la Nivel de Linie

Definire Trigger

```
1 -- Codul trigger-ului
```

Declanșare

Figura 11.1: Demonstrație declanșare trigger

12. Trigger LDD

Definire Trigger

```
1 -- Codul trigger-ului pentru comenzi de tip CREATE/DROP/ALTER
```

Declanșare

Figura 12.1: Demonstrație declanșare trigger LDD

13. Pachet PL/SQL (Opțional)

Specificația Pachetului

```
1 -- Codul specificatiei
```

Corpul Pachetului

```
1 -- Codul body-ului
```

Apel Funcționalități

Figura 13.1: Execuția elementelor din pachet