Tema 1 - Laborator

Aelenei Alex

Cuprins

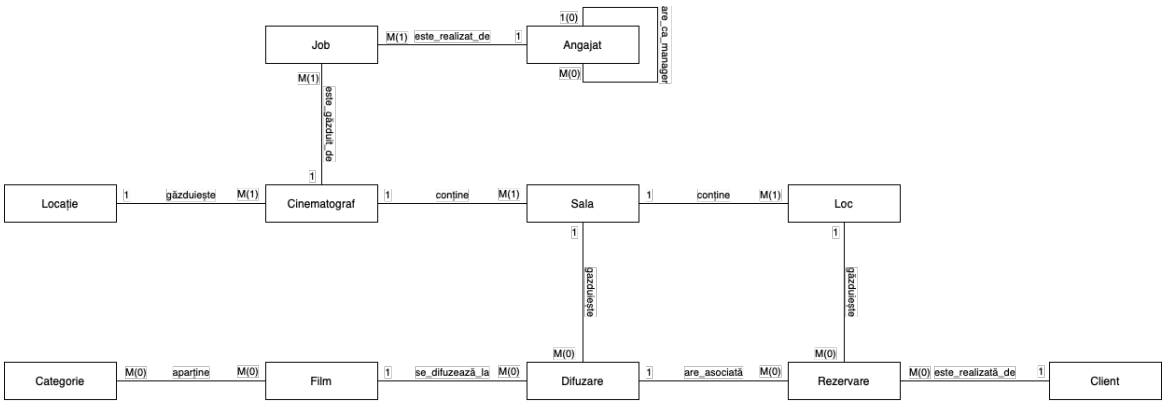

Figura 1: Diagrama conceptuală

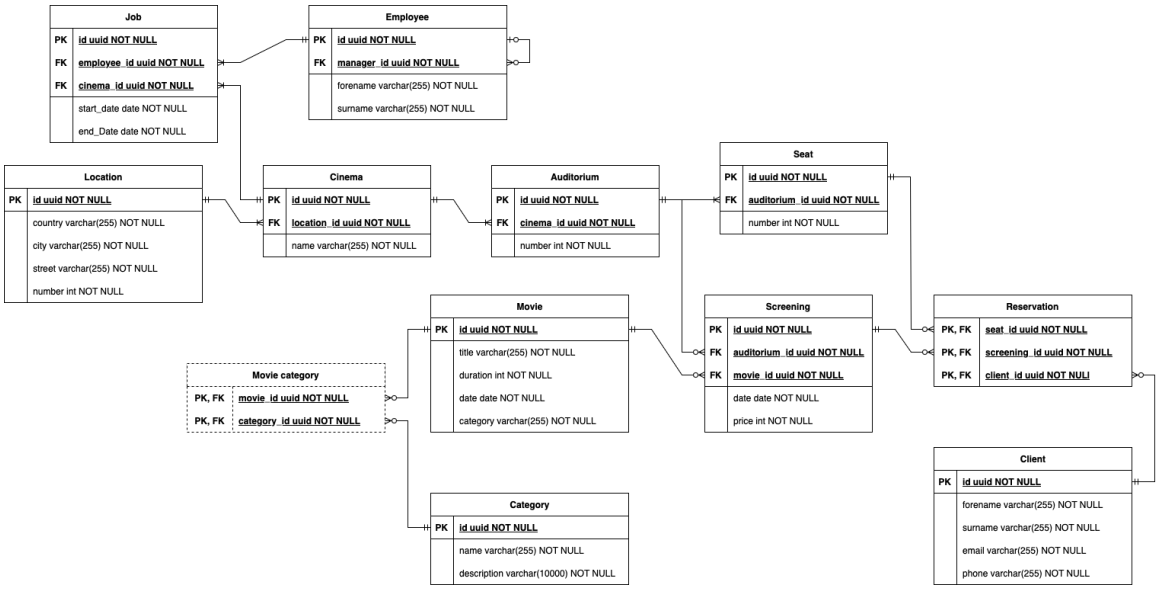

Figura 2: Diagrama entitate-relație

1 Exercițiul 1

1.1 a)

Vom folosi diagrama conceptuală din figura 3.

1.2 b)

Bineînțeles, vom folosi *SPOOL* și alte setări pentru a transforma aceste date în fișiere *SQL* valide.

```
01 |  SELECT
02 |      'DROP TABLE '
03 |      || TABLE_NAME
04 |      || ' CASCADE CONSTRAINTS;' AS QUERY
05 |  FROM
06 |      USER_TABLES;
```

Listing 1: Query pentru generarea unui script care sterge toate tabelele

```
SELECT
    'INSERT INTO MOVIE VALUES
('
    || M.MOVIE_ID
    || ', '''
    || M.TITLE
    || ''', '
    || M.DURATION
    || ', TO_DATE('''
    || TO_CHAR(M.RELEASE_DATE, 'yyyy/mm/dd')
       || ''', ''yyyy/mm/dd''));' AS "Inserare date"
FROM
    MOVIE M;
```

Listing 2: Query pentru generarea datelor din tabelul MOVIE

2 Exercițiul 2

2.1 a)

Această diagramă conceptuală descrie structura organizațională și operativă al unui lanț de cinematografe. Angajații au nume, prenume și un superior direct. De asemenea, pot exista angajați care nu au un superior direct. Locațiile sunt descrise de țară, oraș, stradă și numărul străzii. Cinematografele se află într-o locație și au un nume. Angajații au un rol, care este asociat unui cinematograf. De asemenea, fiecare rol are o dată de început și una de sfârșit. Fiecare cinematograf are mai multe săli, fiecare având un număr. Fiecare sală are mai multe locuri, care de asemenea au nu număr. O categorie de film este descrisă de numele categoriei și descrierea acesteia. Un film este descris de numele filmului, durata în minute, data apariției. Un film poate face parte din mai multe categorii. Difuzarea unui film se realizează într-o sală a unui cinematograf, având o dată și un preț pentru fiecare bilet. Clienții au nume, prenume, email și număr de telefon. Aceștia pot rezerva un loc sau mai multe pentru orice difuzare activă.

2.2 b)

Diagrama conceptuală poate fi regăsită în figura 3.

2.3 c)

În primul rând, după cum se poate observa în 2, se adaugă chei primare artificiale, și se stabilesc cheile externe și celelalte atribute esențiale. Analizând cardinalitatea relațiilor dintre entități putem observa necesitatea unui tabel de legătură între filme și categorii (pentru a modela relația many-to-many). De asemenea, o cheie primară compusă este utilă pentru a descrie unicitatea unei rezervări.

2.4 d)

Diagrama entitate-relație poate fi regăsită în figura 2.

2.5 e)

```
CREATE TABLE LOCATION (
    LOCATION_ID NUMBER(10) NOT NULL,
    COUNTRY VARCHAR2(255) NOT NULL,
    CITY VARCHAR2(255) NOT NULL,
    STREET VARCHAR2(255) NOT NULL,
    "NUMBER" NUMBER(10) NOT NULL,
    CONSTRAINT LOCATION_PK PRIMARY KEY (LOCATION_ID)
);

CREATE TABLE CINEMA (
    CINEMA_ID NUMBER(10) NOT NULL,
    LOCATION_ID NUMBER(10) NOT NULL,
    NAME VARCHAR2(255) NOT NULL,
    CONSTRAINT CINEMA_PK PRIMARY KEY (CINEMA_ID),
```

```
15 |       CONSTRAINT CINEMA_LOCATION_PK FOREIGN KEY (LOCATION_ID) REFERENCES LOCATION(
       LOCATION_ID)
16 |   );
17 |
18 |   CREATE TABLE EMPLOYEE (
19 |       EMPLOYEE_ID NUMBER(10) NOT NULL,
20 |       MANAGER_ID NUMBER(10),
21 |       FIRST_NAME VARCHAR2(255) NOT NULL,
22 |       LAST_NAME VARCHAR2(255) NOT NULL,
23 |       CONSTRAINT EMPLOYEE_PK PRIMARY KEY (EMPLOYEE_ID)
24 |   );
25 |
26 |   ALTER TABLE EMPLOYEE
27 |       ADD CONSTRAINT EMPLOYEE_MANAGER_PK FOREIGN KEY (
28 |           MANAGER_ID
29 |       )
30 |           REFERENCES EMPLOYEE(
31 |               EMPLOYEE_ID
32 |           );
33 |
34 |   CREATE TABLE JOB (
35 |       JOB_ID NUMBER(10) NOT NULL,
36 |       EMPLOYEE_ID NUMBER(10) NOT NULL,
37 |       CINEMA_ID NUMBER(10) NOT NULL,
38 |       START_DATE DATE NOT NULL,
39 |       END_DATE DATE NOT NULL,
40 |       CONSTRAINT JOB_PK PRIMARY KEY (JOB_ID),
41 |       CONSTRAINT JOB_EMPLOYEE_FK FOREIGN KEY (EMPLOYEE_ID) REFERENCES EMPLOYEE(
       EMPLOYEE_ID),
42 |       CONSTRAINT JOB_CINEMA_FK FOREIGN KEY (CINEMA_ID) REFERENCES CINEMA(CINEMA_ID)
43 |   );
44 |
45 |   CREATE TABLE AUDITORIUM (
46 |       AUDITORIUM_ID NUMBER(10) NOT NULL,
47 |       CINEMA_ID NUMBER(10) NOT NULL,
48 |       "NUMBER" NUMBER(10) NOT NULL,
49 |       CONSTRAINT AUDITORIUM_PK PRIMARY KEY (AUDITORIUM_ID),
50 |       CONSTRAINT AUDITORIUM_CINEMA_FK FOREIGN KEY (CINEMA_ID) REFERENCES CINEMA(
       CINEMA_ID)
51 |   );
52 |
53 |   CREATE TABLE SEAT (
54 |       SEAT_ID NUMBER(10) NOT NULL,
55 |       AUDITORIUM_ID NUMBER(10) NOT NULL,
56 |       "NUMBER" NUMBER(10) NOT NULL,
57 |       CONSTRAINT SEAT_PK PRIMARY KEY (SEAT_ID),
58 |       CONSTRAINT SEAT_AUDITORIUM_FK FOREIGN KEY (AUDITORIUM_ID) REFERENCES AUDITORIUM(
       AUDITORIUM_ID)
59 |   );
60 |
61 |   CREATE TABLE CATEGORY (
62 |       CATEGORY_ID NUMBER(10) NOT NULL,
63 |       "NAME" VARCHAR2(255) NOT NULL,
64 |       DESCRIPTION VARCHAR2(1000) NOT NULL,
65 |       CONSTRAINT CATEGORY_PK PRIMARY KEY (CATEGORY_ID)
66 |   );
67 |
68 |   CREATE TABLE MOVIE (
69 |       MOVIE_ID NUMBER(10) NOT NULL,
70 |       TITLE VARCHAR2(255) NOT NULL,
71 |       DURATION NUMBER(10) NOT NULL,
72 |       RELEASE_DATE DATE NOT NULL,
73 |       CONSTRAINT MOVIE_PK PRIMARY KEY (MOVIE_ID)
74 |   );
75 |
76 |   CREATE TABLE MOVIE_CATEGORY (
77 |       MOVIE_ID NUMBER(10) NOT NULL,
78 |       CATEGORY_ID NUMBER(10) NOT NULL,
79 |       CONSTRAINT MOVIE_CATEGORY_PK PRIMARY KEY (MOVIE_ID, CATEGORY_ID)
80 |   );
81 |
82 |   CREATE TABLE SCREENING (
83 |       SCREENING_ID NUMBER(10) NOT NULL,
84 |       AUDITORIUM_ID NUMBER(10) NOT NULL,
85 |       MOVIE_ID NUMBER(10) NOT NULL,
86 |       "DATE" DATE NOT NULL,
```

```
87 |       PRICE NUMBER(10) NOT NULL,
88 |       CONSTRAINT SCREENING_PK PRIMARY KEY (SCREENING_ID),
89 |       CONSTRAINT SCREENING_AUDITORIUM_FK FOREIGN KEY (AUDITORIUM_ID) REFERENCES
          AUDITORIUM(AUDITORIUM_ID),
90 |       CONSTRAINT SCREENING_MOVIE_FK FOREIGN KEY (MOVIE_ID) REFERENCES MOVIE(MOVIE_ID)
91 |   );
92 |
93 |   CREATE TABLE CLIENT (
94 |       CLIENT_ID NUMBER(10) NOT NULL,
95 |       FIRST_NAME VARCHAR2(255) NOT NULL,
96 |       LAST_NAME VARCHAR2(255) NOT NULL,
97 |       EMAIL VARCHAR2(255) NOT NULL,
98 |       PHONE_NUMBER VARCHAR2(255) NOT NULL,
99 |       CONSTRAINT CLIENT_PK PRIMARY KEY (CLIENT_ID)
100 |  );
101 |
102 |  CREATE TABLE RESERVATION (
103 |      SEAT_ID NUMBER(10) NOT NULL,
104 |      SCREENING_ID NUMBER(10) NOT NULL,
105 |      CLIENT_ID NUMBER(10) NOT NULL,
106 |      CONSTRAINT SEAT_SCREENING_CLIENT_PK PRIMARY KEY (SEAT_ID, SCREENING_ID, CLIENT_ID)
107 |  );
108 |
109 |  COMMIT;
```

Listing 3: Script pentru generarea tabelelor

2.6 f)

În primul rând, vom utiliza datele inserate prin query-ul următor pentru a testa cerința 26, adaptate tabelelor cu filme și categorii.

```
01 |  INSERT INTO CATEGORY VALUES (
02 |      1,
03 |      'Horror',
04 |      'The horror category is ...'
05 |  );
06 |
07 |  INSERT INTO CATEGORY VALUES (
08 |      2,
09 |      'Action',
10 |      'The action category is ...'
11 |  );
12 |
13 |  INSERT INTO CATEGORY VALUES (
14 |      3,
15 |      'Romance',
16 |      'The romance category is ...'
17 |  );
18 |
19 |  INSERT INTO CATEGORY VALUES (
20 |      4,
21 |      'History',
22 |      'The history category is ...'
23 |  );
24 |
25 |  INSERT INTO CATEGORY VALUES (
26 |      5,
27 |      'Science-fiction',
28 |      'The science-fiction genre is...'
29 |  );
30 |
31 |  INSERT INTO MOVIE VALUES (
32 |      1,
33 |      'Palm Springs',
34 |      90,
35 |      TO_DATE('2020/07/10', 'yyyy/mm/dd')
36 |  );
37 |
38 |  INSERT INTO MOVIE VALUES (
39 |      2,
40 |      'Beau Is Afraid',
41 |      179,
```

```
    TO_DATE('2023/04/21', 'yyyy/mm/dd')
);

INSERT INTO MOVIE VALUES (
    3,
    'Aliens',
    137,
    TO_DATE('1986/07/14', 'yyyy/mm/dd')
);

INSERT INTO MOVIE VALUES (
    4,
    'Everything Everywhere All at Once',
    132,
    TO_DATE('2022/07/14', 'yyyy/mm/dd')
);

INSERT INTO MOVIE VALUES (
    5,
    'Pan''s Labyrinth',
    109,
    TO_DATE('2006/09/11', 'yyyy/mm/dd')
);

INSERT INTO MOVIE_CATEGORY VALUES (
    1,
    3
);

INSERT INTO MOVIE_CATEGORY VALUES (
    1,
    5
);

INSERT INTO MOVIE_CATEGORY VALUES (
    2,
    1
);

INSERT INTO MOVIE_CATEGORY VALUES (
    2,
    5
);

INSERT INTO MOVIE_CATEGORY VALUES (
    3,
    1
);

INSERT INTO MOVIE_CATEGORY VALUES (
    3,
    2
);

INSERT INTO MOVIE_CATEGORY VALUES (
    3,
    5
);

INSERT INTO MOVIE_CATEGORY VALUES (
    4,
    2
);

INSERT INTO MOVIE_CATEGORY VALUES (
    4,
    5
);

INSERT INTO MOVIE_CATEGORY VALUES (
    5,
    5
);
```

Listing 4: Query pentru generarea tabelelor

Cerința 26 afirmă: *Să se obţină numărul departamentelor care au cel puţin 15 angajaţi.* Adaptând la aceste două tabele (filme și categorii), vom determina numărul de filme care fac parte din cel puțin două categorii.

```
01 |  WITH CATEGORY_COUNTS AS (
02 |      SELECT
03 |          COUNT(M.MOVIE_ID) AS CATEGORY_COUNT
04 |      FROM
05 |          MOVIE           M
06 |          INNER JOIN MOVIE_CATEGORY MC
07 |          ON M.MOVIE_ID = MC.MOVIE_ID
08 |      GROUP BY
09 |          M.MOVIE_ID
10 |  )
11 |  SELECT
12 |      COUNT(*) AS "Movies with more than 2 categories"
13 |  FROM
14 |      CATEGORY_COUNTS CC
15 |  WHERE
16 |      CC.CATEGORY_COUNT >= 2;
```

Listing 5: Query pentru generarea tabelelor

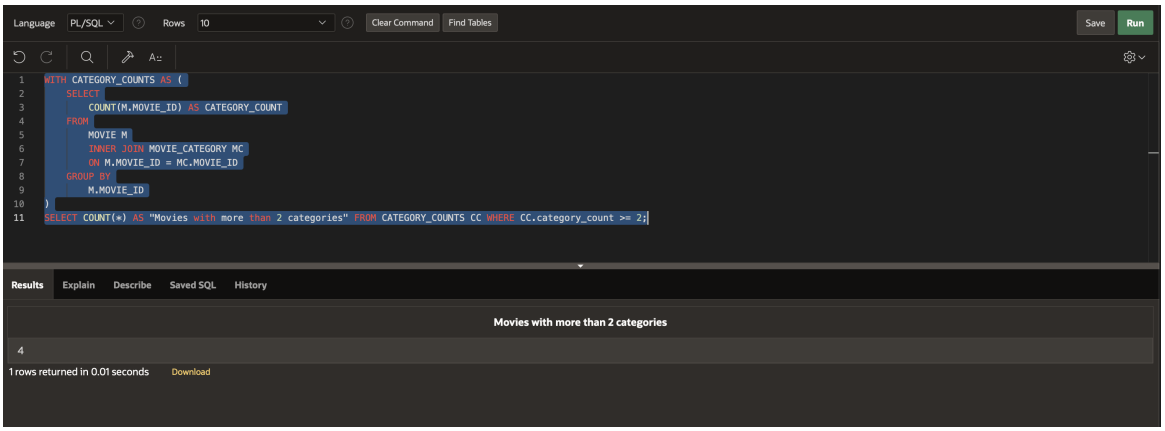

Figura 3: Rezultat query