Sistem de gestiune al unui lanț de cinematografe

Aelenei Alex, Seria 25, Grupa 252, Anul universitar 2024-2025

Table of Contents

[Introducere 2](#_Toc187443329)

[Configurație 2](#_Toc187443330)

[Inițializare 2](#_Toc187443331)

[Cerința 1 4](#_Toc187443332)

[Cerința 2 5](#_Toc187443333)

[Cerința 3 5](#_Toc187443334)

[Cerința 4 5](#_Toc187443335)

[Cerința 5 8](#_Toc187443336)

[Cerința 6 – rezervările unui utilizator 17](#_Toc187443337)

[Cerința 7 – ocuparea locurilor la o difuzare 21](#_Toc187443338)

[Cerința 8 – filmele dintr-o categorie 24](#_Toc187443339)

[Cerința 9 – difuzarile unui film într-un interval de timp 27](#_Toc187443340)

[Cerința 10 – numărul mediu de rezervări la difuzările dintr-un cinematograf 36](#_Toc187443341)

[Cerința 11 – validare rezervare nouă 42](#_Toc187443342)

# Introducere

## Configurație

Pentru acest proiect vom folosi Docker și o imagine de Oracle Database 19c Enterprise Edition. Proiectul a fost testat și rulat pe MacOS Sequoia 15.1.1, procesor M2 Pro, 16GB RAM. Proiectul a fost testat și pe Linux și Windows folosind [această imagine](https://hub.docker.com/layers/emhui/oracle/19.3.0-ee-slim-faststart/images/sha256-2b9e8eedc21286599bb1765771d647a5e9c7ba68fd5808ece9669f9ac437ad66). Întrucât dezvoltarea s-a realizat pe un procesor cu arhitectura ARM, această imagine nu a funcționat, așadar am construit utilizând codul sursă oficial de la Oracle o imagine pentru ARM urmărind [acest ghid](https://gist.github.com/miccheng/8120d2e17818ba2a2d227554b70cd34e).

Pentru a configura și a porni cu ușurință proiectul, vom folosi docker compose:

|  |
| --- |
| services:  database:  image: oracle/database:19.3.0-ee-slim-faststart  environment:  ORACLE\_PASSWORD: ProiectSgbd2024  ports:  - 1522:1521 |

Figura 1 - docker-compose.yml

Resursele alocate docker-ului sunt următoarele: CPU limit: 10 (maxim), memory limit: 8GB, swap: 1GB, virtual disk limit: 64GB.

## Inițializare

Utilizând comanda docker compose -f docker-compose.yml up putem porni containerul. Trebuie menționat că în continuare vom folosi Oracle SQL Developer 23.1.1.345. Ne putem conecta din SQL Developer la baza de date ca și utilizator sistem astfel:

A screenshot of a computer

Description automatically generated

Figura 2 - Conexiunea la baza de date ca sys

Ulterior, vom crea un alt utilizator de pe care vom realiza operațiile asupra bazei de date, conform [ghidului de instalare și inițializare](https://drive.google.com/drive/folders/1o2zgjXiz2eb0u-BayL3Awetv-yMUJGWv). Script-ul următor creează un rol nou cu toate permisiunile necesare, creează un utilizator, îi atribuie rolul și orice alte permisiuni sau privilegii necesare:

|  |
| --- |
| -- Query-ul trebuie rulat ca SYSDBA  -- Creare rol nou  CREATE ROLE SGBD\_USER\_ROLE;  -- Atribuire privilegii si roluri noului rol  GRANT CONNECT TO SGBD\_USER\_ROLE;  GRANT RESOURCE TO SGBD\_USER\_ROLE;  GRANT CREATE TABLE TO SGBD\_USER\_ROLE;  GRANT CREATE VIEW TO SGBD\_USER\_ROLE;  GRANT CREATE MATERIALIZED VIEW TO SGBD\_USER\_ROLE;  GRANT CREATE SYNONYM TO SGBD\_USER\_ROLE;  GRANT CREATE PROCEDURE TO SGBD\_USER\_ROLE;  GRANT CREATE SEQUENCE TO SGBD\_USER\_ROLE;  GRANT CREATE TRIGGER TO SGBD\_USER\_ROLE;  GRANT CREATE TYPE TO SGBD\_USER\_ROLE;  GRANT QUERY REWRITE TO SGBD\_USER\_ROLE;  GRANT SELECT\_CATALOG\_ROLE TO SGBD\_USER\_ROLE;  GRANT ALTER SESSION TO SGBD\_USER\_ROLE;  GRANT SELECT ANY DICTIONARY TO SGBD\_USER\_ROLE;  GRANT CREATE PUBLIC DATABASE LINK TO SGBD\_USER\_ROLE;  GRANT CREATE PUBLIC SYNONYM TO SGBD\_USER\_ROLE;  -- Creare utilizator nou  CREATE USER SGBD\_USER IDENTIFIED BY SgbdProjectUser123  PROFILE DEFAULT  DEFAULT TABLESPACE USERS  QUOTA UNLIMITED ON USERS  ACCOUNT UNLOCK;  -- Atribuire rol la utilizator  GRANT SGBD\_USER\_ROLE TO SGBD\_USER;  -- Atribuire privilegiu  GRANT UNLIMITED TABLESPACE TO SGBD\_USER; |

Figura 3 - Script pentru crearea utilizatorului sgbd\_user

În continuare ne putem conecta din SQL Developer la baza de date ca și sgbd\_user astfel:

A screenshot of a computer

Description automatically generated

Figura 4 - Conexiunea la baza de date ca sgbd\_user

# Cerința 1

Această bază de date modelează sistemul intern de gestiune al unui lanț de cinematografe. Cinematografele au asociat un nume și o locație, iar o locație poate găzdui mai multe cinematografe. O locație este descrisă prin țară, oraș, adresă și cod poștal. În cinematografe lucrează angajați, care pot avea un manager. Pentru fiecare angajat se reține numele și prenumele. Un angajat poate să aibă mai multe posturi în mai multe cinematografe. Un post are o dată de început obligatorie și o dată de sfârșit opțională (postul poate fi asociat pe perioada determinată sau nedeterminată). De asemenea există săli de cinematograf, care au asociat un cinematograf și un cod de identificare.

În sistemul de gestiune al lanțului de cinematografe se găsesc și mai multe filme. Un film are un titlu, o durată în minute și o dată a apariției. De asemenea, există și mai multe categorii de filme, fiecare categorie având asociat un nume și o descriere. Un film poate aparține mai multor categorii, iar o categorie poate include mai multe filme. O difuzare a unui film are asociat filmul, o sală în care se va difuza filmul, o dată a difuzării și un preț per bilet. De asemenea, există și locuri în sălile de cinematograf, fiecare loc având o sală asociată și un cod de identificare.

Lanțul de cinematografe are în sistemul de gestiune și o listă de clienți. Fiecare client are asociat un nume, un prenume, un email unic și un număr de telefon. Un client poate să facă mai multe rezervări la o difuzare. O rezervare are asociată o difuzare, un loc, un client și o dată.

# Cerința 2

A black and white diagram

Description automatically generated

Figura 5 - Diagrama entitate-relație

# Cerința 3

A black screen with white text

Description automatically generated

Figura 6 - Diagrama conceptuală

# Cerința 4

|  |
| --- |
| CREATE TABLE LOCATII (  ID\_LOCATIE NUMBER(10) NOT NULL,  TARA VARCHAR2(255) NOT NULL,  ORAS VARCHAR2(255) NOT NULL,  ADRESA VARCHAR2(255) NOT NULL,  COD\_POSTAL VARCHAR2(10) NOT NULL,  CONSTRAINT LOCATII\_PK PRIMARY KEY (ID\_LOCATIE)  );  CREATE TABLE CINEMATOGRAFE (  ID\_CINEMATOGRAF NUMBER(10) NOT NULL,  ID\_LOCATIE NUMBER(10) NOT NULL,  NUME VARCHAR2(255) NOT NULL,  CONSTRAINT CINEMATOGRAFE\_PK PRIMARY KEY (ID\_CINEMATOGRAF),  CONSTRAINT CINEMATOGRAFE\_LOCATIE\_FK FOREIGN KEY (ID\_LOCATIE) REFERENCES LOCATII(ID\_LOCATIE)  );  CREATE TABLE ANGAJATI (  ID\_ANGAJAT NUMBER(10) NOT NULL,  ID\_MANAGER NUMBER(10),  NUME VARCHAR2(255) NOT NULL,  PRENUME VARCHAR2(255) NOT NULL,  CONSTRAINT ANGAJATI\_PK PRIMARY KEY (ID\_ANGAJAT)  );  ALTER TABLE ANGAJATI  ADD CONSTRAINT ANGAJATI\_MANAGER\_FK FOREIGN KEY (  ID\_MANAGER  )  REFERENCES ANGAJATI(  ID\_ANGAJAT  );  CREATE TABLE POSTURI (  ID\_POST NUMBER(10) NOT NULL,  ID\_ANGAJAT NUMBER(10) NOT NULL,  ID\_CINEMATOGRAF NUMBER(10) NOT NULL,  DATA\_INCEPUT DATE NOT NULL,  DATA\_SFARSIT DATE,  CONSTRAINT POSTURI\_PK PRIMARY KEY (ID\_POST),  CONSTRAINT POSTURI\_ANGAJAT\_FK FOREIGN KEY (ID\_ANGAJAT) REFERENCES ANGAJATI(ID\_ANGAJAT),  CONSTRAINT POSTURI\_CINEMATOGRAF\_FK FOREIGN KEY (ID\_CINEMATOGRAF) REFERENCES CINEMATOGRAFE(ID\_CINEMATOGRAF)  );  CREATE TABLE SALI (  ID\_SALA NUMBER(10) NOT NULL,  ID\_CINEMATOGRAF NUMBER(10) NOT NULL,  COD VARCHAR2(255) NOT NULL,  CONSTRAINT SALI\_PK PRIMARY KEY (ID\_SALA),  CONSTRAINT SALI\_CINEMATOGRAF\_FK FOREIGN KEY (ID\_CINEMATOGRAF) REFERENCES CINEMATOGRAFE(ID\_CINEMATOGRAF)  );  CREATE TABLE LOCURI (  ID\_LOC NUMBER(10) NOT NULL,  ID\_SALA NUMBER(10) NOT NULL,  COD VARCHAR2(20) NOT NULL,  CONSTRAINT LOCURI\_PK PRIMARY KEY (ID\_LOC),  CONSTRAINT LOCURI\_SALA\_FK FOREIGN KEY (ID\_SALA) REFERENCES SALI(ID\_SALA)  );  CREATE TABLE CATEGORII (  ID\_CATEGORIE NUMBER(10) NOT NULL,  NUME VARCHAR2(255) NOT NULL,  DESCRIERE VARCHAR2(1000),  CONSTRAINT CATEGORII\_PK PRIMARY KEY (ID\_CATEGORIE)  );  CREATE TABLE FILME (  ID\_FILM NUMBER(10) NOT NULL,  TITLU VARCHAR2(255) NOT NULL,  DURATA NUMBER(10) NOT NULL,  DATA\_PUBLICARE DATE NOT NULL,  CONSTRAINT FILME\_PK PRIMARY KEY (ID\_FILM)  );  CREATE TABLE CATEGORII\_FILME (  ID\_FILM NUMBER(10) NOT NULL,  ID\_CATEGORIE NUMBER(10) NOT NULL,  CONSTRAINT CATEGORII\_FILME\_PK PRIMARY KEY (ID\_FILM, ID\_CATEGORIE)  );  CREATE TABLE DIFUZARI (  ID\_DIFUZARE NUMBER(10) NOT NULL,  ID\_SALA NUMBER(10) NOT NULL,  ID\_FILM NUMBER(10) NOT NULL,  DATA DATE NOT NULL,  PRET NUMBER(10) NOT NULL,  CONSTRAINT DIFUZARI\_PK PRIMARY KEY (ID\_DIFUZARE),  CONSTRAINT DIFUZARI\_SALA\_FK FOREIGN KEY (ID\_SALA) REFERENCES SALI(ID\_SALA),  CONSTRAINT DIFUZARI\_FILM\_FK FOREIGN KEY (ID\_FILM) REFERENCES FILME(ID\_FILM)  );  CREATE TABLE CLIENTI (  ID\_CLIENT NUMBER(10) NOT NULL,  NUME VARCHAR2(255) NOT NULL,  PRENUME VARCHAR2(255) NOT NULL,  EMAIL VARCHAR2(255) NOT NULL,  TELEFON VARCHAR2(255) NOT NULL,  CONSTRAINT CLIENTI\_PK PRIMARY KEY (ID\_CLIENT),  CONSTRAINT CLIENTI\_EMAIL\_UNIQUE UNIQUE (EMAIL)  );  CREATE TABLE REZERVARI (  ID\_LOC NUMBER(10) NOT NULL,  ID\_DIFUZARE NUMBER(10) NOT NULL,  ID\_CLIENT NUMBER(10) NOT NULL,  DATA DATE DEFAULT SYSDATE NOT NULL,  CONSTRAINT REZERVARI\_PK PRIMARY KEY (ID\_LOC, ID\_DIFUZARE, ID\_CLIENT)  );  COMMIT; |

A screenshot of a computer

Description automatically generated

Figura 7 - Script pentru definirea tabelelor și a constrângerilor

# Cerința 5

În ceea ce privește datele de test, vom include o cantitate semnificativă de date de test pentru a putea testa toate cerințele următoare. Astfel, datele vor fi generate în două părți: o parte manual, printr-un script SQL și o parte automat și aleator, printr-un script de PL/SQL.

În datele de test din fișierul SQL sunt incluse: 5 locații, 5 cinematografe, 20 de angajați, 20 de posturi, 15 săli, 50 de difuzări, 5 filme, 5 categorii de. În plus, sunt incluse 450 de locuri (câte 30 de locuri per sală) și un număr aleator de rezervări (aproximativ 300, se generează aleator în script-ul de PL/SQL).

Trebuie menționat, de asemenea, că toate datele personale ale utilizatorilor și ale angajaților sunt generate aleator, folosind [generatedata](https://generatedata.com/). În plus, toate celelalte date sunt disponibile publicului și nu au nici o legătură cu realitatea, având rol pur orientativ în acest proiect.

|  |
| --- |
| INSERT INTO CATEGORII VALUES (1, 'Horror', NULL);  INSERT INTO CATEGORII VALUES (2, 'Action', NULL);  INSERT INTO CATEGORII VALUES (3, 'Romance', NULL);  INSERT INTO CATEGORII VALUES (4, 'History', NULL);  INSERT INTO CATEGORII VALUES (5, 'Science-fiction', NULL);  INSERT INTO FILME VALUES ( 1, 'Palm Springs', 90, TO\_DATE('2020/07/10', 'yyyy/mm/dd') );  INSERT INTO FILME VALUES ( 2, 'Beau Is Afraid', 179, TO\_DATE('2023/04/21', 'yyyy/mm/dd') );  INSERT INTO FILME VALUES ( 3, 'Aliens', 137, TO\_DATE('1986/07/14', 'yyyy/mm/dd') );  INSERT INTO FILME VALUES ( 4, 'Everything Everywhere All at Once', 132, TO\_DATE('2022/07/14', 'yyyy/mm/dd') );  INSERT INTO FILME VALUES ( 5, 'Pan''s Labyrinth', 109, TO\_DATE('2006/09/11', 'yyyy/mm/dd') );  INSERT INTO CATEGORII\_FILME VALUES (1, 3);  INSERT INTO CATEGORII\_FILME VALUES (1, 5);  INSERT INTO CATEGORII\_FILME VALUES (2, 1);  INSERT INTO CATEGORII\_FILME VALUES (2, 5);  INSERT INTO CATEGORII\_FILME VALUES (3, 1);  INSERT INTO CATEGORII\_FILME VALUES (3, 2);  INSERT INTO CATEGORII\_FILME VALUES (3, 5);  INSERT INTO CATEGORII\_FILME VALUES (4, 2);  INSERT INTO CATEGORII\_FILME VALUES (4, 5);  INSERT INTO CATEGORII\_FILME VALUES (5, 5);  INSERT INTO LOCATII VALUES ( 1, 'Romania', 'Bucuresti', 'Calea Vacaresti', '100575' );  INSERT INTO LOCATII VALUES ( 2, 'Romania', 'Bucuresti', 'Strada Liviu Rebreanu', '410095' );  INSERT INTO LOCATII VALUES ( 3, 'Romania', 'Bucuresti', 'Bulevardul Pierre de Coubertin', '041082' );  INSERT INTO LOCATII VALUES ( 4, 'Romania', 'Bucuresti', 'Bulevardul Doina Cornea', '450043' );  INSERT INTO LOCATII VALUES ( 5, 'Romania', 'Bucuresti', 'Bulevardul 1 Decembrie 1918', '330088' );  INSERT INTO CINEMATOGRAFE VALUES (1, 1, 'Cinema City');  INSERT INTO CINEMATOGRAFE VALUES (2, 2, 'Cinema City');  INSERT INTO CINEMATOGRAFE VALUES (3, 3, 'Cinema City');  INSERT INTO CINEMATOGRAFE VALUES (4, 4, 'Cinema City');  INSERT INTO CINEMATOGRAFE VALUES (5, 4, 'IMAX');  INSERT INTO ANGAJATI VALUES (1, NULL, 'Karly', 'Bush');  INSERT INTO ANGAJATI VALUES (2, 1, 'Dante', 'Brien');  INSERT INTO ANGAJATI VALUES (3, 1, 'Bertha', 'Poole');  INSERT INTO ANGAJATI VALUES (4, 1, 'Deacon', 'Herrera');  INSERT INTO ANGAJATI VALUES (5, NULL, 'Rama', 'Taylor');  INSERT INTO ANGAJATI VALUES (6, 5, 'Jolie', 'Mooney');  INSERT INTO ANGAJATI VALUES (7, 5, 'Dale', 'Munoz');  INSERT INTO ANGAJATI VALUES (8, 5, 'Raymond', 'Atkinson');  INSERT INTO ANGAJATI VALUES (9, NULL, 'Kylee', 'Puckett');  INSERT INTO ANGAJATI VALUES (10, 9, 'Raymond', 'Hudson');  INSERT INTO ANGAJATI VALUES (11, 9, 'Ingrid', 'Grant');  INSERT INTO ANGAJATI VALUES (12, 9, 'Maile', 'Ramirez');  INSERT INTO ANGAJATI VALUES (13, NULL, 'Zorita', 'Morris');  INSERT INTO ANGAJATI VALUES (14, 13, 'Halee', 'Houston');  INSERT INTO ANGAJATI VALUES (15, 13, 'Winifred', 'Rush');  INSERT INTO ANGAJATI VALUES (16, 13, 'Anastasia', 'Ewing');  INSERT INTO ANGAJATI VALUES (17, NULL, 'Lance', 'Sosa');  INSERT INTO ANGAJATI VALUES (18, 17, 'Tarik', 'Valdez');  INSERT INTO ANGAJATI VALUES (19, 17, 'Yoko', 'Newton');  INSERT INTO ANGAJATI VALUES (20, 17, 'Cleo', 'Rose');  INSERT INTO POSTURI VALUES (1, 1, 1, TO\_DATE('2023/04/11', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (2, 2, 1, TO\_DATE('2023/12/23', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (3, 3, 1, TO\_DATE('2024/10/20', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (4, 4, 1, TO\_DATE('2023/09/26', 'yyyy/mm/dd'), TO\_DATE('2025/09/26', 'yyyy/mm/dd'));  INSERT INTO POSTURI VALUES (5, 5, 2, TO\_DATE('2024/08/13', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (6, 6, 2, TO\_DATE('2022/09/22', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (7, 7, 2, TO\_DATE('2023/09/23', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (8, 8, 2, TO\_DATE('2023/01/17', 'yyyy/mm/dd'), TO\_DATE('2025/01/17', 'yyyy/mm/dd'));  INSERT INTO POSTURI VALUES (9, 9, 3, TO\_DATE('2023/10/29', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (10, 10, 3, TO\_DATE('2023/08/05', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (11, 11, 3, TO\_DATE('2024/10/14', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (12, 12, 3, TO\_DATE('2024/06/02', 'yyyy/mm/dd'), TO\_DATE('2024/06/02', 'yyyy/mm/dd'));  INSERT INTO POSTURI VALUES (13, 13, 4, TO\_DATE('2023/11/27', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (14, 14, 4, TO\_DATE('2022/05/22', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (15, 15, 4, TO\_DATE('2023/01/06', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (16, 16, 4, TO\_DATE('2022/12/01', 'yyyy/mm/dd'), TO\_DATE('2025/12/01', 'yyyy/mm/dd'));  INSERT INTO POSTURI VALUES (17, 17, 5, TO\_DATE('2021/11/29', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (18, 18, 5, TO\_DATE('2024/08/28', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (19, 19, 5, TO\_DATE('2021/11/29', 'yyyy/mm/dd'), NULL);  INSERT INTO POSTURI VALUES (20, 20, 5, TO\_DATE('2024/8/28', 'yyyy/mm/dd'), TO\_DATE('2025/8/28', 'yyyy/mm/dd'));  INSERT INTO SALI VALUES (1, 1, 'Londra');  INSERT INTO SALI VALUES (2, 1, 'Syndey');  INSERT INTO SALI VALUES (3, 1, 'Ankara');  INSERT INTO SALI VALUES (4, 2, 'Beijing');  INSERT INTO SALI VALUES (5, 2, 'Cairo');  INSERT INTO SALI VALUES (6, 2, 'Ottawa');  INSERT INTO SALI VALUES (7, 3, 'Moscova');  INSERT INTO SALI VALUES (8, 3, 'Lima');  INSERT INTO SALI VALUES (9, 3, 'Bangkok');  INSERT INTO SALI VALUES (10, 4, 'Berlin');  INSERT INTO SALI VALUES (11, 4, 'Seoul');  INSERT INTO SALI VALUES (12, 4, 'Madrid');  INSERT INTO SALI VALUES (13, 5, 'Sofia');  INSERT INTO SALI VALUES (14, 5, 'Paris');  INSERT INTO SALI VALUES (15, 5, 'Varsovia');  INSERT INTO DIFUZARI VALUES ( 1, 10, 1, TO\_DATE('2025/09/20', 'yyyy/mm/dd'), 16 );  INSERT INTO DIFUZARI VALUES ( 2, 8, 4, TO\_DATE('2025/09/5', 'yyyy/mm/dd'), 31 );  INSERT INTO DIFUZARI VALUES ( 3, 1, 5, TO\_DATE('2025/09/1', 'yyyy/mm/dd'), 18 );  INSERT INTO DIFUZARI VALUES ( 4, 2, 1, TO\_DATE('2025/09/11', 'yyyy/mm/dd'), 21 );  INSERT INTO DIFUZARI VALUES ( 5, 8, 4, TO\_DATE('2025/09/8', 'yyyy/mm/dd'), 23 );  INSERT INTO DIFUZARI VALUES ( 6, 4, 1, TO\_DATE('2025/09/15', 'yyyy/mm/dd'), 20 );  INSERT INTO DIFUZARI VALUES ( 7, 13, 1, TO\_DATE('2025/09/5', 'yyyy/mm/dd'), 31 );  INSERT INTO DIFUZARI VALUES ( 8, 10, 2, TO\_DATE('2025/09/21', 'yyyy/mm/dd'), 35 );  INSERT INTO DIFUZARI VALUES ( 9, 11, 2, TO\_DATE('2025/09/1', 'yyyy/mm/dd'), 35 );  INSERT INTO DIFUZARI VALUES ( 10, 14, 5, TO\_DATE('2025/09/27', 'yyyy/mm/dd'), 30 );  INSERT INTO DIFUZARI VALUES ( 11, 6, 1, TO\_DATE('2025/09/12', 'yyyy/mm/dd'), 24 );  INSERT INTO DIFUZARI VALUES ( 12, 13, 4, TO\_DATE('2025/09/7', 'yyyy/mm/dd'), 31 );  INSERT INTO DIFUZARI VALUES ( 13, 14, 4, TO\_DATE('2025/09/3', 'yyyy/mm/dd'), 28 );  INSERT INTO DIFUZARI VALUES ( 14, 8, 1, TO\_DATE('2025/09/9', 'yyyy/mm/dd'), 29 );  INSERT INTO DIFUZARI VALUES ( 15, 9, 1, TO\_DATE('2025/09/7', 'yyyy/mm/dd'), 33 );  INSERT INTO DIFUZARI VALUES ( 16, 3, 2, TO\_DATE('2025/09/21', 'yyyy/mm/dd'), 23 );  INSERT INTO DIFUZARI VALUES ( 17, 2, 4, TO\_DATE('2025/09/28', 'yyyy/mm/dd'), 20 );  INSERT INTO DIFUZARI VALUES ( 18, 11, 5, TO\_DATE('2025/09/23', 'yyyy/mm/dd'), 27 );  INSERT INTO DIFUZARI VALUES ( 19, 1, 1, TO\_DATE('2025/09/27', 'yyyy/mm/dd'), 20 );  INSERT INTO DIFUZARI VALUES ( 20, 6, 5, TO\_DATE('2025/09/23', 'yyyy/mm/dd'), 21 );  INSERT INTO DIFUZARI VALUES ( 21, 12, 3, TO\_DATE('2025/09/9', 'yyyy/mm/dd'), 29 );  INSERT INTO DIFUZARI VALUES ( 22, 8, 1, TO\_DATE('2025/09/27', 'yyyy/mm/dd'), 20 );  INSERT INTO DIFUZARI VALUES ( 23, 1, 5, TO\_DATE('2025/09/30', 'yyyy/mm/dd'), 24 );  INSERT INTO DIFUZARI VALUES ( 24, 1, 1, TO\_DATE('2025/09/1', 'yyyy/mm/dd'), 28 );  INSERT INTO DIFUZARI VALUES ( 25, 8, 1, TO\_DATE('2025/09/27', 'yyyy/mm/dd'), 22 );  INSERT INTO DIFUZARI VALUES ( 26, 10, 2, TO\_DATE('2025/09/18', 'yyyy/mm/dd'), 18 );  INSERT INTO DIFUZARI VALUES ( 27, 7, 2, TO\_DATE('2025/09/10', 'yyyy/mm/dd'), 18 );  INSERT INTO DIFUZARI VALUES ( 28, 9, 1, TO\_DATE('2025/09/25', 'yyyy/mm/dd'), 15 );  INSERT INTO DIFUZARI VALUES ( 29, 12, 5, TO\_DATE('2025/09/4', 'yyyy/mm/dd'), 31 );  INSERT INTO DIFUZARI VALUES ( 30, 2, 2, TO\_DATE('2025/09/16', 'yyyy/mm/dd'), 21 );  INSERT INTO DIFUZARI VALUES ( 31, 12, 2, TO\_DATE('2025/09/28', 'yyyy/mm/dd'), 22 );  INSERT INTO DIFUZARI VALUES ( 32, 10, 4, TO\_DATE('2025/09/10', 'yyyy/mm/dd'), 32 );  INSERT INTO DIFUZARI VALUES ( 33, 7, 3, TO\_DATE('2025/09/6', 'yyyy/mm/dd'), 24 );  INSERT INTO DIFUZARI VALUES ( 34, 12, 4, TO\_DATE('2025/09/8', 'yyyy/mm/dd'), 27 );  INSERT INTO DIFUZARI VALUES ( 35, 10, 4, TO\_DATE('2025/09/4', 'yyyy/mm/dd'), 17 );  INSERT INTO DIFUZARI VALUES ( 36, 13, 5, TO\_DATE('2025/09/1', 'yyyy/mm/dd'), 20 );  INSERT INTO DIFUZARI VALUES ( 37, 9, 2, TO\_DATE('2025/09/21', 'yyyy/mm/dd'), 22 );  INSERT INTO DIFUZARI VALUES ( 38, 14, 4, TO\_DATE('2025/09/1', 'yyyy/mm/dd'), 24 );  INSERT INTO DIFUZARI VALUES ( 39, 12, 4, TO\_DATE('2025/09/7', 'yyyy/mm/dd'), 24 );  INSERT INTO DIFUZARI VALUES ( 40, 14, 5, TO\_DATE('2025/09/4', 'yyyy/mm/dd'), 15 );  INSERT INTO DIFUZARI VALUES ( 41, 7, 2, TO\_DATE('2025/09/13', 'yyyy/mm/dd'), 26 );  INSERT INTO DIFUZARI VALUES ( 42, 12, 4, TO\_DATE('2025/09/13', 'yyyy/mm/dd'), 23 );  INSERT INTO DIFUZARI VALUES ( 43, 5, 2, TO\_DATE('2025/09/17', 'yyyy/mm/dd'), 33 );  INSERT INTO DIFUZARI VALUES ( 44, 11, 3, TO\_DATE('2025/09/5', 'yyyy/mm/dd'), 35 );  INSERT INTO DIFUZARI VALUES ( 45, 9, 3, TO\_DATE('2025/09/16', 'yyyy/mm/dd'), 34 );  INSERT INTO DIFUZARI VALUES ( 46, 8, 2, TO\_DATE('2025/09/13', 'yyyy/mm/dd'), 22 );  INSERT INTO DIFUZARI VALUES ( 47, 10, 4, TO\_DATE('2025/09/13', 'yyyy/mm/dd'), 20 );  INSERT INTO DIFUZARI VALUES ( 48, 13, 1, TO\_DATE('2025/09/24', 'yyyy/mm/dd'), 23 );  INSERT INTO DIFUZARI VALUES ( 49, 3, 3, TO\_DATE('2025/09/3', 'yyyy/mm/dd'), 20 );  INSERT INTO DIFUZARI VALUES ( 50, 11, 2, TO\_DATE('2025/09/23', 'yyyy/mm/dd'), 27 );  INSERT INTO CLIENTI VALUES ( 1, 'Yuli', 'Pruitt', 'donec.elementum@outlook.com', '+40758768308' );  INSERT INTO CLIENTI VALUES ( 2, 'Abra', 'Adkins', 'scelerisque.neque@icloud.com', '+40755268230' );  INSERT INTO CLIENTI VALUES ( 3, 'James', 'Schneider', 'urna.suscipit@protonmail.ca', '+40717954725' );  INSERT INTO CLIENTI VALUES ( 4, 'Kellie', 'Mcgowan', 'magna@yahoo.com', '+40712564297' );  INSERT INTO CLIENTI VALUES ( 5, 'Callum', 'Mercado', 'ac@hotmail.couk', '+40764531440' );  INSERT INTO CLIENTI VALUES ( 6, 'Indigo', 'Gay', 'lacus.cras@outlook.org', '+40747540129' );  INSERT INTO CLIENTI VALUES ( 7, 'Emery', 'Delgado', 'velit.egestas.lacinia@google.ca', '+40736986198' );  INSERT INTO CLIENTI VALUES ( 8, 'Aimee', 'Barnes', 'aliquam.ornare@protonmail.edu', '+40710863075' );  INSERT INTO CLIENTI VALUES ( 9, 'Nicole', 'Cox', 'eleifend.cras.sed@yahoo.ca', '+40731160680' );  INSERT INTO CLIENTI VALUES ( 10, 'Michelle', 'Haley', 'magna.ut@hotmail.net', '+40753547175' );  INSERT INTO CLIENTI VALUES ( 11, 'Jack', 'Rodriguez', 'cursus@protonmail.net', '+40784139038' );  INSERT INTO CLIENTI VALUES ( 12, 'Alice', 'Francis', 'et.magnis.dis@protonmail.com', '+40786644439' );  INSERT INTO CLIENTI VALUES ( 13, 'Kuame', 'Bowers', 'blandit.mattis@yahoo.couk', '+40703022781' );  INSERT INTO CLIENTI VALUES ( 14, 'Zelenia', 'Baird', 'enim.gravida.sit@outlook.org', '+40724766722' );  INSERT INTO CLIENTI VALUES ( 15, 'Jacob', 'Burks', 'nisi.aenean.eget@google.ca', '+40713368272' );  INSERT INTO CLIENTI VALUES ( 16, 'Quintessa', 'Jenkins', 'parturient.montes.nascetur@icloud.org', '+40756782126' );  INSERT INTO CLIENTI VALUES ( 17, 'Simon', 'Grant', 'mi.lorem.vehicula@aol.ca', '+40741312116' );  INSERT INTO CLIENTI VALUES ( 18, 'Ainsley', 'Klein', 'donec.porttitor@yahoo.couk', '+40743829582' );  INSERT INTO CLIENTI VALUES ( 19, 'Ahmed', 'Richards', 'odio.tristique@protonmail.com', '+40705621229' );  INSERT INTO CLIENTI VALUES ( 20, 'Kiayada', 'Hickman', 'erat.vivamus@aol.couk', '+40778627852' );  INSERT INTO CLIENTI VALUES ( 21, 'Kermit', 'Vinson', 'semper.rutrum@icloud.org', '+40776584532' );  INSERT INTO CLIENTI VALUES ( 22, 'Urielle', 'Casey', 'donec@protonmail.edu', '+40756359226' );  INSERT INTO CLIENTI VALUES ( 23, 'Dennis', 'Hoover', 'dis.parturient.montes@hotmail.edu', '+40788253227' );  INSERT INTO CLIENTI VALUES ( 24, 'Allistair', 'Pate', 'ut.mi@protonmail.net', '+40781658326' );  INSERT INTO CLIENTI VALUES ( 25, 'Tucker', 'Foster', 'sed.dolor.fusce@outlook.com', '+40778668913' );  INSERT INTO CLIENTI VALUES ( 26, 'Sasha', 'Dillard', 'integer.sem@google.com', '+40784986874' );  INSERT INTO CLIENTI VALUES ( 27, 'Barclay', 'Tillman', 'felis.purus@google.org', '+40720635786' );  INSERT INTO CLIENTI VALUES ( 28, 'Chastity', 'Guthrie', 'ultrices.sit@google.com', '+40772921276' );  INSERT INTO CLIENTI VALUES ( 29, 'Kato', 'Mckinney', 'convallis.convallis.dolor@yahoo.com', '+40746644650' );  INSERT INTO CLIENTI VALUES ( 30, 'Rogan', 'Stark', 'in.mi@yahoo.ca', '+40777656596' );  INSERT INTO CLIENTI VALUES ( 31, 'Nell', 'Owens', 'dapibus.quam@outlook.couk', '+40783261285' );  INSERT INTO CLIENTI VALUES ( 32, 'Ebony', 'Hale', 'donec.vitae@aol.org', '+40743321578' );  INSERT INTO CLIENTI VALUES ( 33, 'Willa', 'Klein', 'dolor.elit@hotmail.com', '+40714674918' );  INSERT INTO CLIENTI VALUES ( 34, 'Isabelle', 'Watkins', 'et.ipsum@icloud.ca', '+40783488866' );  INSERT INTO CLIENTI VALUES ( 35, 'Melinda', 'Giles', 'scelerisque.neque@icloud.couk', '+40757857016' );  INSERT INTO CLIENTI VALUES ( 36, 'Joseph', 'Cannon', 'ipsum.non@protonmail.couk', '+40734935878' );  INSERT INTO CLIENTI VALUES ( 37, 'Paul', 'Kaufman', 'elit.sed@yahoo.net', '+40734445756' );  INSERT INTO CLIENTI VALUES ( 38, 'Henry', 'Green', 'tristique.pharetra.quisque@google.org', '+40718367721' );  INSERT INTO CLIENTI VALUES ( 39, 'Cameron', 'Vega', 'praesent.luctus@protonmail.ca', '+40706568186' );  INSERT INTO CLIENTI VALUES ( 40, 'Omar', 'Carter', 'cras.lorem.lorem@yahoo.net', '+40715832304' );  INSERT INTO CLIENTI VALUES ( 41, 'Jerome', 'Doyle', 'ut.odio.vel@outlook.com', '+40766825917' );  INSERT INTO CLIENTI VALUES ( 42, 'Odessa', 'Mendoza', 'a.nunc@yahoo.edu', '+40722547934' );  INSERT INTO CLIENTI VALUES ( 43, 'Thaddeus', 'Mack', 'nisi.nibh@aol.org', '+40774956357' );  INSERT INTO CLIENTI VALUES ( 44, 'Wyatt', 'Blackburn', 'praesent.interdum.ligula@yahoo.com', '+40762596159' );  INSERT INTO CLIENTI VALUES ( 45, 'Adam', 'Jennings', 'molestie.orci.tincidunt@outlook.ca', '+40756424464' );  INSERT INTO CLIENTI VALUES ( 46, 'Lara', 'Kramer', 'proin.ultrices.duis@hotmail.ca', '+40708662777' );  INSERT INTO CLIENTI VALUES ( 47, 'Natalie', 'Reyes', 'laoreet.posuere@protonmail.com', '+40785532322' );  INSERT INTO CLIENTI VALUES ( 48, 'Macaulay', 'Humphrey', 'interdum.feugiat.sed@yahoo.org', '+40777568211' );  INSERT INTO CLIENTI VALUES ( 49, 'Aurelia', 'Avila', 'id.enim@protonmail.couk', '+40785768536' );  INSERT INTO CLIENTI VALUES ( 50, 'Skyler', 'Burt', 'nunc@hotmail.net', '+40786685345' );  INSERT INTO CLIENTI VALUES ( 51, 'Rogan', 'Neal', 'dolor@icloud.edu', '+40705844348' );  INSERT INTO CLIENTI VALUES ( 52, 'Flynn', 'Gordon', 'sed.pede.cum@aol.ca', '+40732577924' );  INSERT INTO CLIENTI VALUES ( 53, 'Jenette', 'Weiss', 'risus.donec@protonmail.ca', '+40733318216' );  INSERT INTO CLIENTI VALUES ( 54, 'Evan', 'Burton', 'egestas@protonmail.net', '+40724479622' );  INSERT INTO CLIENTI VALUES ( 55, 'Craig', 'Berry', 'enim.condimentum@aol.org', '+40772690573' );  INSERT INTO CLIENTI VALUES ( 56, 'Rhiannon', 'Chambers', 'lacus.ut@google.org', '+40736321124' );  INSERT INTO CLIENTI VALUES ( 57, 'Charles', 'Lyons', 'cras.interdum.nunc@icloud.com', '+40714375233' );  INSERT INTO CLIENTI VALUES ( 58, 'Bruce', 'Wilkinson', 'dapibus@hotmail.net', '+40778212516' );  INSERT INTO CLIENTI VALUES ( 59, 'Asher', 'Pickett', 'at.risus@yahoo.net', '+40725846393' );  INSERT INTO CLIENTI VALUES ( 60, 'Xyla', 'Gamble', 'turpis.nulla@hotmail.net', '+40757112489' );  INSERT INTO CLIENTI VALUES ( 61, 'Hayfa', 'Le', 'eget.lacus.mauris@google.ca', '+40794305574' );  INSERT INTO CLIENTI VALUES ( 62, 'Adrian', 'Nielsen', 'risus.nulla@google.edu', '+40792772677' );  INSERT INTO CLIENTI VALUES ( 63, 'Dustin', 'Burns', 'lorem.vehicula.et@outlook.net', '+40741812616' );  INSERT INTO CLIENTI VALUES ( 64, 'Ifeoma', 'Beach', 'eu@yahoo.org', '+40713193251' );  INSERT INTO CLIENTI VALUES ( 65, 'Stuart', 'Clarke', 'malesuada@aol.com', '+40763515534' );  INSERT INTO CLIENTI VALUES ( 66, 'Isadora', 'Mccullough', 'curae.phasellus@yahoo.edu', '+40738311611' );  INSERT INTO CLIENTI VALUES ( 67, 'Melyssa', 'Sears', 'odio.etiam@aol.net', '+40726668251' );  INSERT INTO CLIENTI VALUES ( 68, 'Wendy', 'Goodwin', 'sed@outlook.com', '+40772325312' );  INSERT INTO CLIENTI VALUES ( 69, 'Sydney', 'Foreman', 'eu.eros@hotmail.org', '+40718557612' );  INSERT INTO CLIENTI VALUES ( 70, 'Venus', 'Atkinson', 'nunc.est@icloud.ca', '+40738262437' );  INSERT INTO CLIENTI VALUES ( 71, 'Rhonda', 'Slater', 'duis.at@hotmail.couk', '+40728892666' );  INSERT INTO CLIENTI VALUES ( 72, 'Xantha', 'Greer', 'ac.ipsum.phasellus@aol.edu', '+40771728282' );  INSERT INTO CLIENTI VALUES ( 73, 'Pearl', 'Bowen', 'congue.elit@icloud.ca', '+40736220754' );  INSERT INTO CLIENTI VALUES ( 74, 'Ayanna', 'Soto', 'libero.est@hotmail.org', '+40727461815' );  INSERT INTO CLIENTI VALUES ( 75, 'Amber', 'Huff', 'egestas.fusce@aol.couk', '+40797767237' );  INSERT INTO CLIENTI VALUES ( 76, 'Jelani', 'Weiss', 'turpis.nulla@hotmail.couk', '+40797163779' );  INSERT INTO CLIENTI VALUES ( 77, 'Brynn', 'Figueroa', 'etiam.gravida@aol.net', '+40766627418' );  INSERT INTO CLIENTI VALUES ( 78, 'Raphael', 'Bailey', 'vitae@hotmail.ca', '+40782898695' );  INSERT INTO CLIENTI VALUES ( 79, 'Colt', 'Burt', 'consectetuer.rhoncus@protonmail.edu', '+40714335364' );  INSERT INTO CLIENTI VALUES ( 80, 'Garrett', 'Lawrence', 'vivamus@outlook.org', '+40745671926' );  INSERT INTO CLIENTI VALUES ( 81, 'Brody', 'Flores', 'molestie.tortor@aol.couk', '+40738893921' );  INSERT INTO CLIENTI VALUES ( 82, 'Alfonso', 'Burt', 'aenean.euismod.mauris@aol.com', '+40715365135' );  INSERT INTO CLIENTI VALUES ( 83, 'Sydnee', 'Everett', 'lorem.sit@protonmail.edu', '+40784176453' );  INSERT INTO CLIENTI VALUES ( 84, 'Ella', 'Gutierrez', 'arcu@protonmail.edu', '+40701122957' );  INSERT INTO CLIENTI VALUES ( 85, 'Keith', 'Floyd', 'dictum.magna@outlook.org', '+40781133729' );  INSERT INTO CLIENTI VALUES ( 86, 'Ashton', 'Horne', 'ullamcorper.eu.euismod@aol.net', '+40776174643' );  INSERT INTO CLIENTI VALUES ( 87, 'Tobias', 'Schwartz', 'nunc.pulvinar@google.ca', '+40795878248' );  INSERT INTO CLIENTI VALUES ( 88, 'Fleur', 'Bray', 'rutrum.lorem@yahoo.edu', '+40737433604' );  INSERT INTO CLIENTI VALUES ( 89, 'Dominique', 'Whitley', 'lectus.cum@protonmail.couk', '+40725015143' );  INSERT INTO CLIENTI VALUES ( 90, 'Nyssa', 'Montoya', 'dapibus.rutrum@yahoo.org', '+40739570986' );  INSERT INTO CLIENTI VALUES ( 91, 'Lewis', 'Patton', 'egestas.lacinia.sed@icloud.edu', '+40714486851' );  INSERT INTO CLIENTI VALUES ( 92, 'Dieter', 'Saunders', 'libero.proin.mi@icloud.com', '+40711612286' );  INSERT INTO CLIENTI VALUES ( 93, 'David', 'Collier', 'class.aptent@protonmail.edu', '+40716767315' );  INSERT INTO CLIENTI VALUES ( 94, 'Judah', 'Carson', 'congue.turpis@google.couk', '+40778666149' );  INSERT INTO CLIENTI VALUES ( 95, 'Keiko', 'Melton', 'neque.in@google.net', '+40786480086' );  INSERT INTO CLIENTI VALUES ( 96, 'Samuel', 'Crosby', 'amet.ultricies.sem@aol.com', '+40793224590' );  INSERT INTO CLIENTI VALUES ( 97, 'Liberty', 'Fuentes', 'congue@google.ca', '+40766379721' );  INSERT INTO CLIENTI VALUES ( 98, 'Kylynn', 'Santos', 'sed.facilisis@protonmail.com', '+40702743184' );  INSERT INTO CLIENTI VALUES ( 99, 'Emery', 'Wilkerson', 'erat.nonummy@outlook.couk', '+40758242443' );  INSERT INTO CLIENTI VALUES ( 100, 'Echo', 'Cohen', 'a.aliquet@google.net', '+40735244327' ); |

Figura 8 - Script SQL pentru inserarea datelor (o parte)

A screenshot of a computer

Description automatically generated

|  |
| --- |
| CREATE OR REPLACE PROCEDURE GENERARE\_REZERVARE\_PENTRU\_CLIENT (  ID\_CLIENT\_PARAM IN CLIENTI.ID\_CLIENT%TYPE  ) AS  ID\_DIFUZARE\_GENERAT DIFUZARI.ID\_DIFUZARE%TYPE;  DIFUZARE DIFUZARI%ROWTYPE;  ID\_LOC LOCURI.ID\_LOC%TYPE;  ZI\_REZERVARE NUMBER(10);  BEGIN  ID\_DIFUZARE\_GENERAT := ROUND(DBMS\_RANDOM.VALUE(1, 50));  SELECT  \* INTO DIFUZARE  FROM  DIFUZARI D  WHERE  D.ID\_DIFUZARE = ID\_DIFUZARE\_GENERAT;  ID\_LOC := ROUND(DBMS\_RANDOM.VALUE(1, 30));  ID\_LOC := ID\_LOC + (DIFUZARE.ID\_SALA - 1) \* 30;  ZI\_REZERVARE := ROUND(DBMS\_RANDOM.VALUE(1, TO\_NUMBER(TO\_CHAR(DIFUZARE.DATA, 'dd'))));  INSERT INTO REZERVARI VALUES (  ID\_LOC,  DIFUZARE.ID\_DIFUZARE,  ID\_CLIENT\_PARAM,  TO\_DATE('2025/09/'  || ZI\_REZERVARE, 'yyyy/mm/dd')  );  END GENERARE\_REZERVARE\_PENTRU\_CLIENT;  /  CREATE OR REPLACE FUNCTION LOC\_ALEATOR\_LA\_DIFUZARE (  ID\_DIFUZARE\_PARAM IN DIFUZARI.ID\_DIFUZARE%TYPE  ) RETURN LOCURI.ID\_LOC%TYPE IS  ID\_LOC LOCURI.ID\_LOC%TYPE;  BEGIN  SELECT  D.ID\_SALA INTO ID\_LOC  FROM  DIFUZARI D  WHERE  D.ID\_DIFUZARE = ID\_DIFUZARE\_PARAM;  ID\_LOC := (ID\_LOC - 1) \* 30 + DBMS\_RANDOM.VALUE(1, 30);  RETURN ID\_LOC;  END LOC\_ALEATOR\_LA\_DIFUZARE;  /  DECLARE  LOCURI\_PER\_SALA NUMBER(10) := 30;  REZERVARI\_PER\_CLIENT\_MAX NUMBER(10) := 5;  REZERVARI\_PER\_CLIENT NUMBER(10);  ZI\_REZERVARE NUMBER(10);  ID\_LOC LOCURI.ID\_LOC%TYPE;  COD\_LOC LOCURI.COD%TYPE;  ID\_DIFUZARE\_REZERVARE REZERVARI.ID\_DIFUZARE%TYPE;  ID\_SALA\_REZERVARE DIFUZARI.ID\_SALA%TYPE;  ID\_LOC\_REZERVARE REZERVARI.ID\_LOC%TYPE;  QUERY\_INSERT\_REZERVARE VARCHAR2(255);  QUERY\_INSERT\_LOC VARCHAR2(255);  BEGIN  FOR DATE\_SALA IN (  SELECT  \*  FROM  SALI  ) LOOP  FOR INDEX\_LOC IN 1..LOCURI\_PER\_SALA LOOP  COD\_LOC := 'LOC-'  || DATE\_SALA.ID\_SALA  || '-'  || INDEX\_LOC;  ID\_LOC := (DATE\_SALA.ID\_SALA - 1) \* LOCURI\_PER\_SALA + INDEX\_LOC;  QUERY\_INSERT\_LOC := 'INSERT INTO LOCURI VALUES('  || ID\_LOC  || ','  || DATE\_SALA.ID\_SALA  || ','''  || COD\_LOC  || ''')';  BEGIN  EXECUTE IMMEDIATE QUERY\_INSERT\_LOC;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Se incearca inserarea unui loc invalid. Sarim peste.');  END;  END LOOP;  END LOOP;  FOR DATE\_CLIENT IN (  SELECT  \*  FROM  CLIENTI  ) LOOP  REZERVARI\_PER\_CLIENT := ROUND(DBMS\_RANDOM.VALUE(1, REZERVARI\_PER\_CLIENT\_MAX));  FOR INDEX\_REZERVARE IN 1..REZERVARI\_PER\_CLIENT LOOP  BEGIN  GENERARE\_REZERVARE\_PENTRU\_CLIENT(DATE\_CLIENT.ID\_CLIENT);  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Se incearca inserarea unei rezervari invalide. Sarim peste.');  END;  END LOOP;  END LOOP;  END; |

Figura 9 - Script PL/SQL pentru inserarea datelor și alte funcții auxiliare pentru generarea de date (cealaltă parte)

A screenshot of a computer

Description automatically generated

# Cerința 6 – rezervările unui utilizator

Considerăm ca avem cunoscut ID-ul unui utilizator. Astfel, dorim să aflăm la ce difuzări are acesta rezervări, și pe ce locuri în sălile respective de cinematograf are rezervările. O difuzare va fi caracterizată, în scopul reprezentării adecvate, prin denumirea filmului și data difuzării. Pentru a rezolva această problemă, considerăm script-ul PL/SQL de mai jos:

|  |
| --- |
| CREATE TYPE LISTA\_CODURI\_LOCURI IS  VARRAY(20) OF VARCHAR2(20);  /  CREATE TYPE REZERVARI\_CLIENT\_DATE\_DIFUZARE IS  OBJECT (  TITLU\_FILM VARCHAR2(255),  DATA\_DIFUZARE DATE,  CODURI\_LOCURI LISTA\_CODURI\_LOCURI  )  /  CREATE TYPE REZERVARI\_CLIENT\_DATE\_DIFUZARI IS  TABLE OF REZERVARI\_CLIENT\_DATE\_DIFUZARE;  /  ALTER TABLE CLIENTI  ADD DATE\_REZERVARI REZERVARI\_CLIENT\_DATE\_DIFUZARI NESTED TABLE DATE\_REZERVARI STORE AS DATE\_REZERVARI\_TI;  CREATE OR REPLACE FUNCTION REZERVARI\_CLIENT (  ID\_CLIENT\_PARAM CLIENTI.ID\_CLIENT%TYPE  ) RETURN REZERVARI\_CLIENT\_DATE\_DIFUZARI AS  TYPE LOCURI\_DIFUZARE IS  TABLE OF LISTA\_CODURI\_LOCURI INDEX BY BINARY\_INTEGER;  DATE\_LOCURI\_DIFUZARE LOCURI\_DIFUZARE;  DATE\_CLIENT REZERVARI\_CLIENT\_DATE\_DIFUZARI;  DATE\_LISTA\_CODURI\_LOCURI LISTA\_CODURI\_LOCURI;  BEGIN  DATE\_CLIENT := REZERVARI\_CLIENT\_DATE\_DIFUZARI();  FOR DATE\_REZERVARE IN (  SELECT  R.ID\_DIFUZARE,  L.COD  FROM  (  SELECT  \*  FROM  REZERVARI R  WHERE  R.ID\_CLIENT = ID\_CLIENT\_PARAM  ) R  INNER JOIN LOCURI L  ON L.ID\_LOC = R.ID\_LOC  ) LOOP  BEGIN  DATE\_LISTA\_CODURI\_LOCURI := DATE\_LOCURI\_DIFUZARE(DATE\_REZERVARE.ID\_DIFUZARE);  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DATE\_LISTA\_CODURI\_LOCURI := LISTA\_CODURI\_LOCURI();  END;  DATE\_LISTA\_CODURI\_LOCURI.EXTEND;  DATE\_LISTA\_CODURI\_LOCURI(DATE\_LISTA\_CODURI\_LOCURI.COUNT) := DATE\_REZERVARE.COD;  DATE\_LOCURI\_DIFUZARE(DATE\_REZERVARE.ID\_DIFUZARE) := DATE\_LISTA\_CODURI\_LOCURI;  END LOOP;  FOR DATE\_DIFUZARE IN (  SELECT  D.ID\_DIFUZARE,  D.DATA,  F.TITLU  FROM  (  SELECT  R.ID\_DIFUZARE  FROM  REZERVARI R  WHERE  R.ID\_CLIENT = ID\_CLIENT\_PARAM  GROUP BY  R.ID\_DIFUZARE  ) DC  INNER JOIN DIFUZARI D  ON DC.ID\_DIFUZARE = D.ID\_DIFUZARE  INNER JOIN FILME F  ON F.ID\_FILM = D.ID\_FILM  ) LOOP  DATE\_LISTA\_CODURI\_LOCURI := DATE\_LOCURI\_DIFUZARE(DATE\_DIFUZARE.ID\_DIFUZARE);  DATE\_CLIENT.EXTEND;  DATE\_CLIENT(DATE\_CLIENT.COUNT) := REZERVARI\_CLIENT\_DATE\_DIFUZARE( DATE\_DIFUZARE.TITLU, DATE\_DIFUZARE.DATA, DATE\_LISTA\_CODURI\_LOCURI);  END LOOP;  UPDATE CLIENTI C  SET  DATE\_REZERVARI = DATE\_CLIENT  WHERE  C.ID\_CLIENT = ID\_CLIENT\_PARAM;  RETURN DATE\_CLIENT;  END REZERVARI\_CLIENT;  /  DECLARE  DATE\_CLIENT REZERVARI\_CLIENT\_DATE\_DIFUZARI;  BEGIN  FOR CLIENT IN (  SELECT  \*  FROM  CLIENTI  ) LOOP  DATE\_CLIENT := REZERVARI\_CLIENT(CLIENT.ID\_CLIENT);  END LOOP;  END; |

Pentru a testa script-ul de mai sus considerăm următorul script de testare:

|  |
| --- |
| DECLARE  ID\_CLIENT CLIENTI.ID\_CLIENT%TYPE := &ID\_CLIENT;  DATE\_CLIENT REZERVARI\_CLIENT\_DATE\_DIFUZARI;  DATE\_DIFUZARE REZERVARI\_CLIENT\_DATE\_DIFUZARE;  LISTA\_LOCURI VARCHAR2(255);  BEGIN  DATE\_CLIENT := REZERVARI\_CLIENT(ID\_CLIENT);  FOR INDEX\_DIFUZARE IN DATE\_CLIENT.FIRST..DATE\_CLIENT.LAST LOOP  DATE\_DIFUZARE := DATE\_CLIENT(INDEX\_DIFUZARE);  DBMS\_OUTPUT.PUT\_LINE('Film: '  || DATE\_DIFUZARE.TITLU\_FILM);  DBMS\_OUTPUT.PUT\_LINE('Data: '  || TO\_CHAR(DATE\_DIFUZARE.DATA\_DIFUZARE, 'dd.mm.yyyy'));  LISTA\_LOCURI := '';  FOR INDEX\_COD\_LOC IN DATE\_DIFUZARE.CODURI\_LOCURI.FIRST..DATE\_DIFUZARE.CODURI\_LOCURI.LAST LOOP  LISTA\_LOCURI := LISTA\_LOCURI  || DATE\_DIFUZARE.CODURI\_LOCURI(INDEX\_COD\_LOC)  || ' ';  END LOOP;  DBMS\_OUTPUT.PUT\_LINE('Coduri locuri: '  || LISTA\_LOCURI);  DBMS\_OUTPUT.NEW\_LINE();  END LOOP;  END; |

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Figura 10 - Rezultat script de testare pentru ID\_CLIENT=5

# Cerința 7 – ocuparea locurilor la o difuzare

Considerăm că avem cunoscut ID-ul unei difuzări. Pentru această difuzare vrem să determinăm care dintre locurile din sala de cinematograf în care se va desfășura difuzarea sunt ocupate. Un loc poate să fie neocupat, ocupat de o singură persoană, sau ocupat multiplu (doi clienți diferiți au rezervat acest loc). Pentru a prezenta într-un mod adecvat aceste date, vom caracteriza un loc prin cod, status și numele complet al celui care a rezervat acel loc (în cazul în care este ocupat). Pentru a rezolva această problemă, considerăm script-ul PL/SQL de mai jos:

|  |
| --- |
| CREATE TYPE OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOC IS  OBJECT (  COD\_LOC VARCHAR2(20),  STATUS VARCHAR2(20),  NUME\_COMPLET VARCHAR2(255)  )  /  CREATE TYPE OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOCURI IS  TABLE OF OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOC;  /  CREATE OR REPLACE FUNCTION OCUPARE\_LOCURI\_LA\_DIFUZARE (  ID\_DIFUZARE\_PARAM DIFUZARI.ID\_DIFUZARE%TYPE  ) RETURN OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOCURI AS  CURSOR REZERVARE\_LOC(  ID\_LOC\_PARAM LOCURI.ID\_LOC%TYPE  ) IS (  SELECT  \*  FROM  REZERVARI R  WHERE  R.ID\_DIFUZARE = ID\_DIFUZARE\_PARAM  AND R.ID\_LOC = ID\_LOC\_PARAM  );  DATE\_REZERVARE REZERVARI%ROWTYPE;  NUME\_COMPLET VARCHAR2(255);  REZERVARI\_PER\_LOC NUMBER(10);  LISTA\_CODURI OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOCURI;  BEGIN  LISTA\_CODURI := OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOCURI();  FOR DATE\_LOC IN (  SELECT  L.\*  FROM  (  SELECT  \*  FROM  DIFUZARI D  WHERE  D.ID\_DIFUZARE = ID\_DIFUZARE\_PARAM  ) D  INNER JOIN SALI S  ON S.ID\_SALA = D.ID\_SALA  INNER JOIN LOCURI L  ON L.ID\_SALA = S.ID\_SALA  ) LOOP  OPEN REZERVARE\_LOC(DATE\_LOC.ID\_LOC);  REZERVARI\_PER\_LOC := 0;  LOOP  FETCH REZERVARE\_LOC INTO DATE\_REZERVARE;  EXIT WHEN REZERVARE\_LOC%NOTFOUND;  REZERVARI\_PER\_LOC := REZERVARI\_PER\_LOC + 1;  SELECT  (C.NUME  || ' '  || C.PRENUME) INTO NUME\_COMPLET  FROM  CLIENTI C  WHERE  C.ID\_CLIENT = DATE\_REZERVARE.ID\_CLIENT;  END LOOP;  CLOSE REZERVARE\_LOC;  LISTA\_CODURI.EXTEND;  IF REZERVARI\_PER\_LOC = 0 THEN  LISTA\_CODURI(LISTA\_CODURI.COUNT) := OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOC(DATE\_LOC.COD, 'Neocupat', '');  ELSIF REZERVARI\_PER\_LOC = 1 THEN  LISTA\_CODURI(LISTA\_CODURI.COUNT) := OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOC(DATE\_LOC.COD, 'Ocupat', NUME\_COMPLET);  ELSE  LISTA\_CODURI(LISTA\_CODURI.COUNT) := OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOC(DATE\_LOC.COD, 'Ocupat multiplu', '');  END IF;  END LOOP;  RETURN LISTA\_CODURI;  END OCUPARE\_LOCURI\_LA\_DIFUZARE; |

A screenshot of a computer

Description automatically generated

Pentru a testa script-ul de mai sus considerăm următorul script de testare:

|  |
| --- |
| DECLARE  ID\_DIFUZARE DIFUZARI.ID\_DIFUZARE%TYPE := &ID\_DIFUZARE;  DATE\_LOCURI OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOCURI;  DATE\_LOC OCUPARE\_LOCURI\_LA\_DIFUZARE\_DATE\_LOC;  BEGIN  DATE\_LOCURI := OCUPARE\_LOCURI\_LA\_DIFUZARE(ID\_DIFUZARE);  FOR INDEX\_DATE\_LOC IN DATE\_LOCURI.FIRST..DATE\_LOCURI.LAST LOOP  DATE\_LOC := DATE\_LOCURI(INDEX\_DATE\_LOC);  DBMS\_OUTPUT.PUT\_LINE('Cod loc: '  || DATE\_LOC.COD\_LOC);  DBMS\_OUTPUT.PUT\_LINE('Status: '  || DATE\_LOC.STATUS);  IF DATE\_LOC.NUME\_COMPLET IS NOT NULL THEN  DBMS\_OUTPUT.PUT\_LINE('Nume persoana: '  || DATE\_LOC.NUME\_COMPLET);  END IF;  DBMS\_OUTPUT.NEW\_LINE();  END LOOP;  END; |

A screenshot of a computer

Description automatically generated

Figura 11 - Rezultat script de testare pentru ID\_DIFUZARE=5

# Cerința 8 – filmele dintr-o categorie

Considerăm că avem numele complet sau parțial al unei categorii de filme. Dacă putem determina o unică categorie de filme în baza denumirii date, atunci să se afișeze toate filmele care aparțin acelei categorii. Datele asociate filmelor vor fi complete, adică înregistrările complete din tabelul de filme. Pentru a rezolva această problemă, considerăm script-ul PL/SQL de mai jos:

|  |
| --- |
| CREATE OR REPLACE FUNCTION FILME\_IN\_CATEGORIE (  DENUMIRE\_CATEGORIE\_PARTIALA CATEGORII.NUME%TYPE  ) RETURN SYS\_REFCURSOR AS  ID\_CATEGORIE\_GASIT CATEGORII.ID\_CATEGORIE%TYPE;  CURSOR\_FILME SYS\_REFCURSOR;  BEGIN  SELECT  C.ID\_CATEGORIE INTO ID\_CATEGORIE\_GASIT  FROM  CATEGORII C  WHERE  LOWER(C.NUME) LIKE LOWER('%'  || DENUMIRE\_CATEGORIE\_PARTIALA  || '%');  OPEN CURSOR\_FILME FOR  SELECT  F.\*  FROM  CATEGORII C  INNER JOIN CATEGORII\_FILME CF  ON CF.ID\_CATEGORIE = C.ID\_CATEGORIE  INNER JOIN FILME F  ON F.ID\_FILM = CF.ID\_FILM  WHERE  C.ID\_CATEGORIE = ID\_CATEGORIE\_GASIT;  RETURN CURSOR\_FILME;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  RAISE\_APPLICATION\_ERROR(-20000, 'Nu exista categorii care sa se potriveasca cu denumirea data!');  WHEN TOO\_MANY\_ROWS THEN  RAISE\_APPLICATION\_ERROR(-20001, 'Exista mai multe categorii care se potrivesc cu denumirea data!');  END FILME\_IN\_CATEGORIE; |

A screenshot of a computer

Description automatically generated

Pentru a testa script-ul de mai sus considerăm următorul script de testare:

|  |
| --- |
| DECLARE  NUME\_CATEGORIE VARCHAR2(255) := '&DENUMIRE\_CATEGORIE';  CURSOR\_FILM SYS\_REFCURSOR;  DATE\_FILM FILME%ROWTYPE;  BEGIN  CURSOR\_FILM := FILME\_IN\_CATEGORIE(NUME\_CATEGORIE);  LOOP  FETCH CURSOR\_FILM INTO DATE\_FILM;  EXIT WHEN CURSOR\_FILM%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE(DATE\_FILM.TITLU);  DBMS\_OUTPUT.PUT\_LINE('Durata: '  || DATE\_FILM.DURATA);  DBMS\_OUTPUT.PUT\_LINE('Data aparitiei: '  || TO\_CHAR(DATE\_FILM.DATA\_PUBLICARE, 'dd.mm.yyyy'));  DBMS\_OUTPUT.NEW\_LINE();  END LOOP;  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Eroare! '  || SUBSTR(SQLERRM, 1, 255));  END; |

A screenshot of a computer

Description automatically generated

Figura 12 - Rezultat script de testare pentru DENUMIRE\_CATEGORIE=HORR

A computer screen shot of a blue screen

Description automatically generated

Figura 13 - Rezultat script de testare pentru DENUMIRE\_CATEGORIE=al

A screenshot of a computer

Description automatically generated

Figura 14 - Rezultat script de testare pentru DENUMIRE\_CATEGORIE=on

# Cerința 9 – difuzarile unui film într-un interval de timp

Considerăm că avem ID-ul unui film, o dată de început și una de sfârșit. Dacă datele sunt valide (data de început nu succede data de sfârșit) și există un film cu ID-ul dat, atunci să se afișeze datele asociate tuturor difuzărilor acelui film în intervalul de timp dat. Aceste date vor fi alcătuite din: codul sălii în care are loc difuzarea, titlul filmului, data difuzării, prețul unui bilet, numele complet al persoanei care a făcut rezervarea și locul rezervat de aceasta.

Pentru a rezolva această problemă, considerăm script-ul PL/SQL de mai jos:

|  |
| --- |
| CREATE TYPE DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARE IS  OBJECT (  ID\_DIFUZARE NUMBER(10),  COD\_SALA VARCHAR2(255),  TITLU\_FILM VARCHAR2(255),  DATA\_DIFUZARE DATE,  PRET\_DIFUZARE NUMBER,  NUME\_PRENUME VARCHAR2(255),  COD\_LOC VARCHAR2(255)  )  /  CREATE TYPE DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARI IS  TABLE OF DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARE  /  CREATE OR REPLACE PROCEDURE DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP (  ID\_FILM\_PARAM IN FILME.ID\_FILM%TYPE,  DATA\_INCEPUT\_PARAM IN DIFUZARI.DATA%TYPE,  DATA\_SFARSIT\_PARAM IN DIFUZARI.DATA%TYPE,  DATE\_DIFUZARI OUT DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARI  ) AS  EXCEPTIE\_DATE\_INVALIDE EXCEPTION;  EXCEPTIE\_REZERVARE\_INVALIDA EXCEPTION;  BEGIN  IF DATA\_INCEPUT\_PARAM > DATA\_SFARSIT\_PARAM THEN  RAISE EXCEPTIE\_DATE\_INVALIDE;  END IF;  DATE\_DIFUZARI := DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARI();  FOR DATE\_DIFUZARE IN (  SELECT  FD.TITLU,  FD.DATA,  FD.ID\_SALA,  FD.COD\_SALA,  FD.PRET,  FD.ID\_DIFUZARE,  L.COD,  L.ID\_SALA AS ID\_SALA\_LOC,  C.NUME  || ' '  || C.PRENUME AS NUME\_COMPLET  FROM  (  SELECT  F.TITLU,  D.ID\_DIFUZARE,  D.DATA,  D.PRET,  S.ID\_SALA,  S.COD AS COD\_SALA  FROM  (  SELECT  \*  FROM  FILME F  WHERE  F.ID\_FILM = ID\_FILM\_PARAM  ) F  INNER JOIN DIFUZARI D  ON D.ID\_FILM = F.ID\_FILM  INNER JOIN SALI S  ON S.ID\_SALA = D.ID\_SALA  WHERE  D.DATA BETWEEN DATA\_INCEPUT\_PARAM AND DATA\_SFARSIT\_PARAM  ) FD  INNER JOIN REZERVARI R  ON FD.ID\_DIFUZARE = R.ID\_DIFUZARE  INNER JOIN LOCURI L  ON L.ID\_LOC = R.ID\_LOC  INNER JOIN CLIENTI C  ON C.ID\_CLIENT = R.ID\_CLIENT  ) LOOP  IF DATE\_DIFUZARE.ID\_SALA != DATE\_DIFUZARE.ID\_SALA\_LOC THEN  RAISE EXCEPTIE\_REZERVARE\_INVALIDA;  END IF;  DATE\_DIFUZARI.EXTEND;  DATE\_DIFUZARI(DATE\_DIFUZARI.COUNT) := DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARE( DATE\_DIFUZARE.ID\_DIFUZARE, DATE\_DIFUZARE.COD\_SALA, DATE\_DIFUZARE.TITLU, DATE\_DIFUZARE.DATA, DATE\_DIFUZARE.PRET, DATE\_DIFUZARE.NUME\_COMPLET, DATE\_DIFUZARE.COD );  END LOOP;  EXCEPTION  WHEN EXCEPTIE\_DATE\_INVALIDE THEN  RAISE\_APPLICATION\_ERROR(-20002, 'Data de inceput succede data de sfarsit.');  WHEN EXCEPTIE\_REZERVARE\_INVALIDA THEN  RAISE\_APPLICATION\_ERROR(-20003, 'O rezervare invalida a fost gasita in lista rezervarilor asociate difuzarilor.');  END DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP; |

A computer screen shot of a blue screen

Description automatically generated

Pentru a testa script-ul de mai sus considerăm următorul script de testare:

|  |
| --- |
| DECLARE  ID\_FILM\_1 FILME.ID\_FILM%TYPE := 1;  DATA\_INCEPUT\_1 DIFUZARI.DATA%TYPE := TO\_DATE('2025/09/15', 'yyyy/mm/dd');  DATA\_SFARSIT\_1 DIFUZARI.DATA%TYPE := TO\_DATE('2025/09/15', 'yyyy/mm/dd');  ID\_FILM\_2 FILME.ID\_FILM%TYPE := 2;  DATA\_INCEPUT\_2 DIFUZARI.DATA%TYPE := TO\_DATE('2025/09/25', 'yyyy/mm/dd');  DATA\_SFARSIT\_2 DIFUZARI.DATA%TYPE := TO\_DATE('2025/09/15', 'yyyy/mm/dd');  ID\_LOC\_REZERVARE REZERVARI.ID\_LOC%TYPE;  ID\_DIFUZARE\_REZERVARE REZERVARI.ID\_DIFUZARE%TYPE;  ID\_CLIENT\_REZERVARE REZERVARI.ID\_CLIENT%TYPE := ROUND(DBMS\_RANDOM.VALUE(1, 100));  DATE\_DIFUZARI DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARI;  DATE\_DIFUZARE DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARE;  PROCEDURE AFISARE\_DATE\_DIFUZARI (  DATE\_DIFUZARI IN DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP\_DATE\_DIFUZARI  ) IS  BEGIN  FOR INDEX\_DATE\_DIFUZARE IN DATE\_DIFUZARI.FIRST..DATE\_DIFUZARI.LAST LOOP  DATE\_DIFUZARE := DATE\_DIFUZARI(INDEX\_DATE\_DIFUZARE);  DBMS\_OUTPUT.PUT\_LINE('Inregistrarea '  || INDEX\_DATE\_DIFUZARE);  DBMS\_OUTPUT.PUT\_LINE('Sala: '  || DATE\_DIFUZARE.COD\_SALA);  DBMS\_OUTPUT.PUT\_LINE('Film: '  || DATE\_DIFUZARE.TITLU\_FILM);  DBMS\_OUTPUT.PUT\_LINE('Difuzare: '  || DATE\_DIFUZARE.ID\_DIFUZARE);  DBMS\_OUTPUT.PUT\_LINE('Data difuzarii: '  || TO\_CHAR(DATE\_DIFUZARE.DATA\_DIFUZARE, 'dd.mm.yyyy'));  DBMS\_OUTPUT.PUT\_LINE('Pret pentru bilet: '  || DATE\_DIFUZARE.PRET\_DIFUZARE);  DBMS\_OUTPUT.PUT\_LINE('Nume complet: '  || DATE\_DIFUZARE.NUME\_PRENUME);  DBMS\_OUTPUT.PUT\_LINE('Cod loc: '  || DATE\_DIFUZARE.COD\_LOC);  DBMS\_OUTPUT.NEW\_LINE();  END LOOP;  END AFISARE\_DATE\_DIFUZARI;  BEGIN  SELECT  D.ID\_DIFUZARE INTO ID\_DIFUZARE\_REZERVARE  FROM  (  SELECT  D.ID\_DIFUZARE  FROM  DIFUZARI D  WHERE  D.ID\_FILM = ID\_FILM\_1  AND D.DATA BETWEEN DATA\_INCEPUT\_1 AND DATA\_SFARSIT\_1  ) D  WHERE  ROWNUM <= 1;  ID\_LOC\_REZERVARE := LOC\_ALEATOR\_LA\_DIFUZARE(ID\_DIFUZARE\_REZERVARE);    -- Caz simplu  DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP(ID\_FILM\_1, DATA\_INCEPUT\_1, DATA\_SFARSIT\_1, DATE\_DIFUZARI);  AFISARE\_DATE\_DIFUZARI(DATE\_DIFUZARI);    -- Caz in care datele sunt invalide  BEGIN  DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP(ID\_FILM\_2, DATA\_INCEPUT\_2, DATA\_SFARSIT\_2, DATE\_DIFUZARI);  AFISARE\_DATE\_DIFUZARI(DATE\_DIFUZARI);  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Eroare! '  || SUBSTR(SQLERRM, 1, 255));  END;    -- Caz in care se insereaza o rezervare noua  INSERT INTO REZERVARI(  ID\_LOC,  ID\_DIFUZARE,  ID\_CLIENT  ) VALUES (  ID\_LOC\_REZERVARE,  ID\_DIFUZARE\_REZERVARE,  ID\_CLIENT\_REZERVARE  );  BEGIN  DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP(ID\_FILM\_1, DATA\_INCEPUT\_1, DATA\_SFARSIT\_1, DATE\_DIFUZARI);  AFISARE\_DATE\_DIFUZARI(DATE\_DIFUZARI);  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Eroare! '  || SUBSTR(SQLERRM, 1, 255));  END;    -- Caz in care se detecteaza on rezervare invalida  UPDATE REZERVARI R  SET  R.ID\_LOC = ID\_LOC\_REZERVARE + 30  WHERE  R.ID\_LOC = ID\_LOC\_REZERVARE  AND R.ID\_DIFUZARE = ID\_DIFUZARE\_REZERVARE  AND R.ID\_CLIENT = ID\_CLIENT\_REZERVARE;  BEGIN  DIFUZARI\_FILM\_IN\_INTERVAL\_DE\_TIMP(ID\_FILM\_1, DATA\_INCEPUT\_1, DATA\_SFARSIT\_1, DATE\_DIFUZARI);  AFISARE\_DATE\_DIFUZARI(DATE\_DIFUZARI);  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('Eroare! '  || SUBSTR(SQLERRM, 1, 255));  END;  DELETE FROM REZERVARI R  WHERE  R.ID\_LOC = ID\_LOC\_REZERVARE  AND R.ID\_DIFUZARE = ID\_DIFUZARE\_REZERVARE  AND R.ID\_CLIENT = ID\_CLIENT\_REZERVARE;  END; |

A screenshot of a computer

Description automatically generated

|  |
| --- |
| Inregistrarea 1  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Dominique Whitley  Cod loc: LOC-4-5  Inregistrarea 2  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Fleur Bray  Cod loc: LOC-4-6  Inregistrarea 3  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Ebony Hale  Cod loc: LOC-4-7  Inregistrarea 4  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Wendy Goodwin  Cod loc: LOC-4-7  Inregistrarea 5  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Garrett Lawrence  Cod loc: LOC-4-10  Inregistrarea 6  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Dennis Hoover  Cod loc: LOC-4-14  Inregistrarea 7  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Allistair Pate  Cod loc: LOC-4-21  Inregistrarea 8  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Fleur Bray  Cod loc: LOC-4-27  Eroare! ORA-20002: Data de inceput succede data de sfarsit.  Inregistrarea 1  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Dominique Whitley  Cod loc: LOC-4-5  Inregistrarea 2  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Fleur Bray  Cod loc: LOC-4-6  Inregistrarea 3  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Ebony Hale  Cod loc: LOC-4-7  Inregistrarea 4  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Wendy Goodwin  Cod loc: LOC-4-7  Inregistrarea 5  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Garrett Lawrence  Cod loc: LOC-4-10  Inregistrarea 6  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Dennis Hoover  Cod loc: LOC-4-14  Inregistrarea 7  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Allistair Pate  Cod loc: LOC-4-21  Inregistrarea 8  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Melyssa Sears  Cod loc: LOC-4-21  Inregistrarea 9  Sala: Beijing  Film: Palm Springs  Difuzare: 6  Data difuzarii: 15.09.2025  Pret pentru bilet: 20  Nume complet: Fleur Bray  Cod loc: LOC-4-27  Eroare! ORA-20003: O rezervare invalida a fost gasita in lista rezervarilor asociate difuzarilor.  PL/SQL procedure successfully completed. |

Figura 15 - Rezultat script de testare

# Cerința 10 – numărul mediu de rezervări la difuzările dintr-un cinematograf

Considerăm o nouă statistică pentru toate cinematografele: numărul mediu de rezervări la difuzările din cinematograf. Această statistică se va actualiza de fiecare dată când se creează, șterge sau editează o rezervare.

Pentru a rezolva această problemă, considerăm script-ul PL/SQL de mai jos:

|  |
| --- |
| ALTER TABLE CINEMATOGRAFE  ADD MEDIE\_REZERVARI\_PER\_DIFUZARE NUMBER(  10,  4  );  /  CREATE OR REPLACE PROCEDURE ACTUALIZEAZA\_MEDIE\_REZERVARI\_PER\_DIFUZARE AS  BEGIN  FOR DATE\_REZERVARI IN (  SELECT  S.ID\_CINEMATOGRAF,  AVG(NUMAR\_REZERVARI) AS MEDIE\_REZERVARI\_PER\_DIFUZARE  FROM  (  SELECT  D.ID\_DIFUZARE,  D.ID\_SALA,  COUNT(R.ID\_LOC) AS NUMAR\_REZERVARI  FROM  DIFUZARI D  INNER JOIN REZERVARI R  ON R.ID\_DIFUZARE = D.ID\_DIFUZARE  GROUP BY  D.ID\_DIFUZARE,  D.ID\_SALA  ) DR  INNER JOIN SALI S  ON S.ID\_SALA = DR.ID\_SALA  GROUP BY  S.ID\_CINEMATOGRAF  ) LOOP  UPDATE CINEMATOGRAFE C  SET  C.MEDIE\_REZERVARI\_PER\_DIFUZARE = DATE\_REZERVARI.MEDIE\_REZERVARI\_PER\_DIFUZARE  WHERE  C.ID\_CINEMATOGRAF = DATE\_REZERVARI.ID\_CINEMATOGRAF;  END LOOP;  END ACTUALIZEAZA\_MEDIE\_REZERVARI\_PER\_DIFUZARE;  /  BEGIN  ACTUALIZEAZA\_MEDIE\_REZERVARI\_PER\_DIFUZARE();  END;  /  CREATE OR REPLACE TRIGGER ACTUALIZEAZA\_MEDIE\_REZERVARI\_PER\_DIFUZARE\_TRIGGER AFTER  DELETE OR UPDATE OR INSERT OF ID\_DIFUZARE ON REZERVARI  BEGIN  ACTUALIZEAZA\_MEDIE\_REZERVARI\_PER\_DIFUZARE();  END;  A screenshot of a computer  Description automatically generated |

Pentru a testa script-ul de mai sus considerăm următorul script de testare:

|  |
| --- |
| DECLARE  ID\_DIFUZARE\_REZERVARE REZERVARI.ID\_DIFUZARE%TYPE := 1;  ID\_DIFUZARE\_REZERVARE\_NOU REZERVARI.ID\_DIFUZARE%TYPE := 2;  ID\_LOC\_REZERVARE REZERVARI.ID\_LOC%TYPE;  ID\_LOC\_REZERVARE\_NOU REZERVARI.ID\_LOC%TYPE;  ID\_CLIENT\_REZERVARE REZERVARI.ID\_CLIENT%TYPE := ROUND(DBMS\_RANDOM.VALUE(1, 100));  PROCEDURE AFISARE\_CINEMA (  MESAJ\_INCEPUT IN VARCHAR2  ) AS  BEGIN  DBMS\_OUTPUT.PUT\_LINE(MESAJ\_INCEPUT);  DBMS\_OUTPUT.NEW\_LINE();  FOR DATE\_CINEMA IN (  SELECT  \*  FROM  CINEMATOGRAFE C  INNER JOIN LOCATII L  ON L.ID\_LOCATIE = C.ID\_LOCATIE  ) LOOP  DBMS\_OUTPUT.PUT\_LINE(DATE\_CINEMA.NUME);  DBMS\_OUTPUT.PUT\_LINE('Adresa: '  || DATE\_CINEMA.ADRESA);  DBMS\_OUTPUT.PUT\_LINE('Medie rezervari per difuzare: '  || DATE\_CINEMA.MEDIE\_REZERVARI\_PER\_DIFUZARE);  DBMS\_OUTPUT.NEW\_LINE();  END LOOP;  END AFISARE\_CINEMA;  BEGIN  ID\_LOC\_REZERVARE := LOC\_ALEATOR\_LA\_DIFUZARE(ID\_DIFUZARE\_REZERVARE);  ID\_LOC\_REZERVARE\_NOU := LOC\_ALEATOR\_LA\_DIFUZARE(ID\_DIFUZARE\_REZERVARE\_NOU);  AFISARE\_CINEMA('Initial:');    -- Dupa crearea unei rezervari noi  INSERT INTO REZERVARI(  ID\_LOC,  ID\_DIFUZARE,  ID\_CLIENT  ) VALUES (  ID\_LOC\_REZERVARE,  ID\_DIFUZARE\_REZERVARE,  ID\_CLIENT\_REZERVARE  );  AFISARE\_CINEMA('Dupa crearea unei rezervari noi:');    -- Dupa modificarea rezervarii create  UPDATE REZERVARI R  SET  R.ID\_LOC = ID\_LOC\_REZERVARE\_NOU,  R.ID\_DIFUZARE = ID\_DIFUZARE\_REZERVARE\_NOU  WHERE  R.ID\_LOC = ID\_LOC\_REZERVARE  AND R.ID\_DIFUZARE = ID\_DIFUZARE\_REZERVARE  AND R.ID\_CLIENT = ID\_CLIENT\_REZERVARE;  AFISARE\_CINEMA('Dupa modificarea rezervarii:');    -- Dupa stergerea rezervarii modificate  DELETE FROM REZERVARI R  WHERE  R.ID\_LOC = ID\_LOC\_REZERVARE\_NOU  AND R.ID\_DIFUZARE = ID\_DIFUZARE\_REZERVARE\_NOU  AND R.ID\_CLIENT = ID\_CLIENT\_REZERVARE;  AFISARE\_CINEMA('Dupa stergerea rezervarii:');  END; |

A screenshot of a computer

Description automatically generated

|  |
| --- |
| Initial:  Cinema City  Adresa: Calea Vacaresti  Medie rezervari per difuzare: 7  Cinema City  Adresa: Strada Liviu Rebreanu  Medie rezervari per difuzare: 6.75  Cinema City  Adresa: Bulevardul Pierre de Coubertin  Medie rezervari per difuzare: 7.3077  Cinema City  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 5.4375  IMAX  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 4.5  Dupa crearea unei rezervari noi:  Cinema City  Adresa: Calea Vacaresti  Medie rezervari per difuzare: 7  Cinema City  Adresa: Strada Liviu Rebreanu  Medie rezervari per difuzare: 6.75  Cinema City  Adresa: Bulevardul Pierre de Coubertin  Medie rezervari per difuzare: 7.3077  Cinema City  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 5.5  IMAX  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 4.5  Dupa modificarea rezervarii:  Cinema City  Adresa: Calea Vacaresti  Medie rezervari per difuzare: 7  Cinema City  Adresa: Strada Liviu Rebreanu  Medie rezervari per difuzare: 6.75  Cinema City  Adresa: Bulevardul Pierre de Coubertin  Medie rezervari per difuzare: 7.3846  Cinema City  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 5.4375  IMAX  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 4.5  Dupa stergerea rezervarii:  Cinema City  Adresa: Calea Vacaresti  Medie rezervari per difuzare: 7  Cinema City  Adresa: Strada Liviu Rebreanu  Medie rezervari per difuzare: 6.75  Cinema City  Adresa: Bulevardul Pierre de Coubertin  Medie rezervari per difuzare: 7.3077  Cinema City  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 5.4375  IMAX  Adresa: Bulevardul Doina Cornea  Medie rezervari per difuzare: 4.5    PL/SQL procedure successfully completed. |

Figura 16 - Rezultat script de testare

# Cerința 11 – validare rezervare nouă

Pentru fiecare rezervare nouă care este creată sau pentru fiecare rezervare care este editată trebuie verificat că difuzarea la care se face verificare are loc după data rezervării și că locul rezervării se află în sala în care are loc difuzarea.

Pentru a rezolva această problemă, considerăm script-ul PL/SQL de mai jos:

|  |
| --- |
| CREATE OR REPLACE TRIGGER VALIDARE\_LOC\_REZERVARE\_SALA BEFORE  UPDATE OR INSERT OF ID\_LOC ON REZERVARI FOR EACH ROW  DECLARE  DATE\_DIFUZARE DIFUZARI%ROWTYPE;  ID\_SALA\_LOC SALI.ID\_SALA%TYPE;  BEGIN  SELECT  L.ID\_SALA INTO ID\_SALA\_LOC  FROM  LOCURI L  WHERE  L.ID\_LOC = :NEW.ID\_LOC;  SELECT  \* INTO DATE\_DIFUZARE  FROM  DIFUZARI D  WHERE  D.ID\_DIFUZARE = :NEW.ID\_DIFUZARE;  IF ID\_SALA\_LOC != DATE\_DIFUZARE.ID\_SALA THEN  RAISE\_APPLICATION\_ERROR(-20004, 'Locul asociat rezervarii nu se afla in sala unde se difuzeaza filmul!');  END IF;  IF DATE\_DIFUZARE.DATA < :NEW.DATA THEN  RAISE\_APPLICATION\_ERROR(-20005, 'Data rezervarii succede data difuzarii!');  END IF;  END; |

A screenshot of a computer

Description automatically generated

Pentru a testa script-ul de mai sus considerăm următorul script de testare:

A screenshot of a computer

Description automatically generated

Figura 17 - Rezultat script de testare

# Cerința 12 – tabele de backup

Daca un tabel este șters sau modificat, atunci înainte să se realizeze modificările, trebuie sa se facă un backup la acest tabel.

Pentru a rezolva această problemă, considerăm script-ul PL/SQL de mai jos:

|  |
| --- |
| CREATE SEQUENCE SEQ\_BACKUP\_TABLE\_INDEX START WITH 1 INCREMENT BY 1 MINVALUE 1 NOMAXVALUE NOCYCLE CACHE 2;  CREATE OR REPLACE TRIGGER BACKUP\_TABEL\_MODIFICAT\_SAU\_STERS BEFORE ALTER OR DROP ON SCHEMA  DECLARE  NUME\_OBIECT VARCHAR2(255);  TIP\_OBIECT VARCHAR2(255);  QUERY\_BACKUP\_TABEL VARCHAR2(255);  BEGIN  NUME\_OBIECT := UPPER(TRIM(ORA\_DICT\_OBJ\_NAME));  TIP\_OBIECT := UPPER(TRIM(ORA\_DICT\_OBJ\_TYPE));  IF TIP\_OBIECT = 'TABLE' THEN  QUERY\_BACKUP\_TABEL := 'CREATE TABLE BACKUP\_'  || SEQ\_BACKUP\_TABLE\_INDEX.NEXTVAL  || '\_'  || NUME\_OBIECT  || ' AS (SELECT \* FROM '  || NUME\_OBIECT  || ')';  EXECUTE IMMEDIATE QUERY\_BACKUP\_TABEL;  DBMS\_OUTPUT.PUT\_LINE('S-a realizat backup la tabelul '  ||NUME\_OBIECT );  END IF;  END; |

A screenshot of a computer

Description automatically generated

Pentru a testa script-ul de mai sus considerăm următorul script de testare:

|  |
| --- |
| CREATE TABLE CINEMATOGRAFE\_AUXILIAR AS (  SELECT  \*  FROM  SGBD\_USER.CINEMATOGRAFE  );  SELECT  \*  FROM  CINEMATOGRAFE\_AUXILIAR;  ALTER TABLE CINEMATOGRAFE\_AUXILIAR  ADD STATISTICA\_AUXILIARA NUMBER(  10,  4  );  DROP TABLE CINEMATOGRAFE\_AUXILIAR;  SELECT \* FROM BACKUP\_1\_CINEMATOGRAFE\_AUXILIAR;  SELECT \* FROM BACKUP\_2\_CINEMATOGRAFE\_AUXILIAR; |

A screenshot of a computer

Description automatically generated