**Proiect Baze de Date**

Agușoaei Alexandru

Grupa 133

An universitar 2023-2024

**Baza de date a unei platforme de socializare**

**Cerințe**

1. ***Descrierea modelului real, a utilității acestuia și a regulilor de funcționare.***
2. ***Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului.***
3. ***Descrierea entităților, incluzând precizarea cheii primare.***
4. ***Descrierea relațiilor, incluzând precizarea cardinalității acestora.***
5. ***Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor.***
6. ***Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.***
7. ***Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6. Diagrama conceptuală obținută trebuie să conțină minimum 7 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ.***
8. ***Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7.***
9. ***Realizarea normalizării până la forma normală 3 (FN1-FN3).***
10. ***Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).***
11. ***Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative; maxim 30 de înregistrări în fiecare tabel).*** 
    1. ***12. Formulați în limbaj natural și implementați 5 cereri SQL complexe ce vor utiliza, în ansamblul lor, următoarele elemente:***
    2. ***a) subcereri sincronizate în care intervin cel puțin 3 tabele***
    3. ***b) subcereri nesincronizate în clauza FROM***
    4. ***c) grupări de date, funcții grup, filtrare la nivel de grupuri cu subcereri nesincronizate (în clauza de HAVING) în care intervin cel puțin 3 tabele (in cadrul aceleiași cereri)***
    5. ***d) ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)***
    6. ***e) utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE***
    7. ***f) utilizarea a cel puțin 1 bloc de cerere (clauza WITH)***

***Observație: Într-o cerere se vor regăsi mai multe elemente dintre cele enumerate mai sus, astfel încât cele 5 cereri să le cuprindă pe toate.***

***13. Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.***

***14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.***

***15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n.***

***Observație: Cele 3 cereri sunt diferite de cererile de la exercițiul 12.***

***16. Optimizarea unei cereri, aplicând regulile de optimizare ce derivă din proprietățile operatorilor algebrei relaționale. Cererea va fi exprimată prin expresie algebrică, arbore algebric și limbaj (SQL), atât anterior cât și ulterior optimizării.***

***17. a. Realizarea normalizării BCNF, FN4, FN5.***

***b. Aplicarea denormalizării, justificând necesitatea acesteia.***

1. ***Descrierea modelului real***

În acest proiect voi ilustra cum arata o baza de date a unei platforme de socializare. În era digitală actuală, platformele de socializare au devenit componente esențiale ale vieții cotidiene, conectând milioane de utilizatori din întreaga lume. Aceste platforme permit oamenilor să interacționeze, să împărtășească idei, să formeze comunități și să creeze conținut, transformând modul în care comunicăm și ne organizăm societatea. În spatele funcționării eficiente a acestor platforme complexe se află un element tehnologic critic: baza de date. O bază de date bine proiectată și gestionată este fundamentul pe care se sprijină orice platformă de socializare. Aceasta joacă un rol vital în stocarea, gestionarea și accesarea informațiilor utilizatorilor, facilitând o varietate de funcționalități esențiale pentru experiența utilizatorilor.

***2. Prezentarea constrângerilor***

-Un utilizator poate avea mai multe albume de fotografii, iar un album este asociat unui singur utilizator.

-Un album de fotografii poate avea mai multe fotografii, iar o fotografie aparține unui singur album.

-Un utilizator poate avea mai multe playlist-uri muzicale, iar un playlist aparține unui singur utilizator.

-O piesă poate aparține mai multor playlist-uri, iar un playlist poate avea mai multe piese.

-Un utilizator are o unică parolă, iar unei parole îi este asociată un singur utilizator.

-Un istoric de parole poate cuprinde mai multe parole, iar o parolă aparține unui singur istoric de parole.

-Un utilizator poate să fie în mai multe grup-uri, iar un grup poate să conțină mai mulți utilizatori.

-Un utilizator are o unică locație, iar o locație poate fi atribuită mai multor utilizatori.

-O locație are o unică adresă, iar o adresă poate fi asociată mai multor locații.

***3. Descrierea entităților***

Pentru modelul de date al aplicației de socializare, Utilizatori, Albume\_fotografii, Fotografii, Parole, Istoric\_parole, Grupuri, Grupuri\_și\_utilizatori, Playlisturi\_muzicale, Piese, Playlisturi\_și\_piese, Locații și Adrese reprezintă entitațile.

Despre entități:

* UTILIZATORI – Persoanele care își crează un cont pe platformă (pk – utilizator\_id.
* ALBUME\_FOTOGRAFII – Albumele adună la un loc mai multe fotografii ale utilizatorului (pk – album\_id).
* FOTOGRAFII – O imagine postată de către utilizator (pk – fotografie\_id).
* PAROLE – O parolă unică a unui utilizator pentru a putea accesa contul pe platformă (parola actuală pe care utilizatorul o folosește, pk – parola\_id).
* ISTORIC\_PAROLE – Istoricul de parole grupează parolele unui utilizator pe care acesta le-a schimbat în timp (parole pe care nu le mai folosește, pk – utilizator\_id + data\_schimbare).
* GRUPURI – Grupurile conțin mai mulți utilizatori pentru a face mai eficientă comunicarea si socializarea grupurilor de prieteni (pk – grup\_id).
* GRUPURI\_ȘI\_UTILIZATORI – Pentru rezolvarea relației many to many între grupuri si utilizatori (pk – utilizator\_id + grup\_id).
* PLAYLISTURI\_MUZICALE – Playlisturile asimilează la un loc mai multe piese pe placul utilizatorului (pk – playlist\_id).
* PIESE – Piesele muzicale care pot fi ascultate de toți utilizatorii (pk – piesa\_id).
* PLAYLISTURI\_ȘI\_PIESE – Pentru rezolvarea relației many to many între piese și playlisturi (pk – piesa\_id + playlist\_id).
* LOCAȚII – Datele despre utilizatori în privința locației lor (pk – locatie\_id).
* ADRESE – Adresele unde locuiesc utilizatorii (pk – adresa\_id).

***4. Descrierea relațiilor***

***Utilizatori – Parole (1:1)***

Un utilizator trebuie să aibă o parolă, iar o parolă trebuie să fie atribuită unui unic utilizator. Relația are ca scop asocierea unică necesară pentru un utilizator de a avea o parolă pentru a se conecta la platforma de socializare.

***Utilizatori – Istoric\_parole (1:1)***

Un utilizator are un unic istoric de parole. Relația are ca scop stocarea parolelor vechi ale utilizatorului.

***Parole – Istoric\_parole (m:0)***

O parolă schimbată poate fi stocată într-un istoric de parole asociat utilizatorului, iar un istoric de parole poate conține mai multe parole sau chiar nici una.

***Utilizatori – Albume\_fotografii (0:m)***

Un utilizator poate avea mai multe albume sau nici unul, iar un album trebuie să fie asociat unui unic utilizator.

***Albume\_fotografii - Fotografii (m:1)***

Un album foto trebuie sa aibă minim o fotografie, iar o fotografie trebuie sa fie asociată unui album de fotografii. Relația are ca scop gruparea eficientă a fotografiilor utilizatorilor.

***Utilizatori – Playlisturi\_și\_piese (1:m)***

Un utilizator poate avea mai multe playlisturi sau poate să nu aibă deloc, iar un playlist trebuie să fie asociat unui unic utilizator.

***Piese – Playlisturi\_muzicale (m:m)***

O piesă poate fi asociată unuia sau mai multor playlisturi, iar un playlist poate avea una sau mai multe piese.

***Utilizatori – Locații (m:1)***

Un utilizator trebuie sa aibă o locație asociată, iar o locație poate fi asociată cu mai mulți utilizatori.

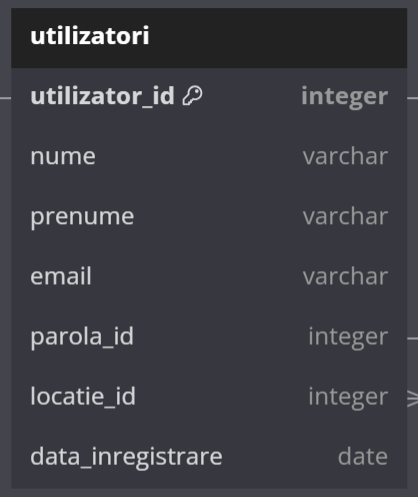
***Locații – Adrese (1:1)***

O locație trebuie sa aibă o unică adresă, iar o adresă trebuie sa fie asociată unei locații.

***Utilizatori – Grupuri (m:m)***

Un utilizator poate să fie în mai multe grupuri, iar un grup poate să conțină mai mulți utilizatori.

***5. Descrierea atributelor***



***1. Utilizatori***

***Utilizator\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare utilizator.

***Nume (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând numele utilizatorului.

***Prenume (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând prenumele utilizatorului.

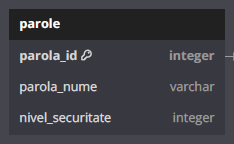
***Email (Not Null, Unique****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând emailul personal al utilizatorului.

***Parola\_id (FK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând id ul de parola din tabela Parole.

***Locatie\_id (FK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând id ul de locație din tabela Locații.

***Data\_intregistrare (Not Null****)* – Variabilă de tip dată calendaristică, reprezentând data la care s-a facut contul utilizatorului.

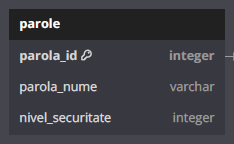
***2. Parole***

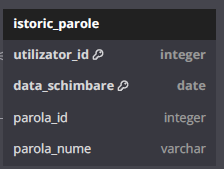


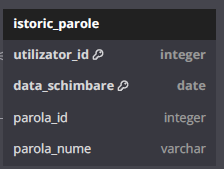
***Parola\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru parola fiecarui utilizator.

***Parola\_nume (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând parola în sine.

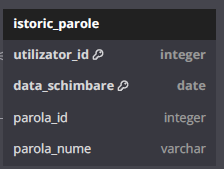
***Nivel\_securitate (Not Null****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând nivelul de securitate al parolei reprezentat pe o scară de la 1 la 10.

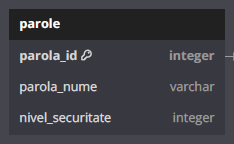
***3. Istoric\_parole***

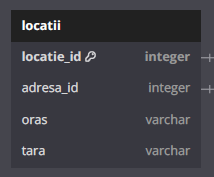
***Utilizator\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unui utilizator.

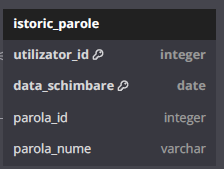
***Data\_schimbare (PK****)* – Variabilă de tip dată calendaristică, reprezentând data la care s-a schimbat o anumită parolă.

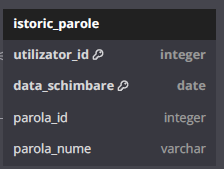
***Parola\_id (FK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru parola schimbată de utilizator.

***Parola\_nume (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând parola veche a utilizatorului.

***4. Locații***



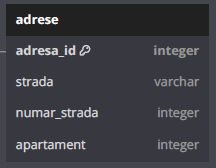
***Locatie\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unei locații.

***Adresa\_id (FK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unei adrese.

***Oras (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând orașul locației.

***Tara (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând țara locației.

***5. Adrese***



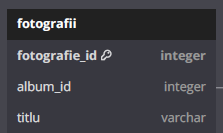
***Adresa\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unei adrese.

***Strada (-****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând numele străzii.

***Numar\_strada (-****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând numărul strazii.

***Apartament (-****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând numărul apartamentului.

***6. Fotografii***

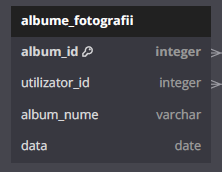


***Fotografie\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unei fotografii.

***Album\_id (FK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unui album.

***Titlu (-****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând titlul fotografiei.

***7. Albume\_fotografii***



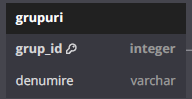
***Album\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unui album.

***Utilizator\_id (FK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare utilizator.

***Album\_nume (-****)* – Variabilă de tip caracter lungime maxim 100, reprezentând titlul albumului.

***Data (Not Null****)* – Variabilă de tip dată calendaristică, reprezentând data când s-a creat albumul.

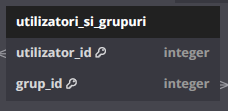
***8. Grupuri***



***Grup\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unui grup.

***Denumire (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând denumirea grupului.

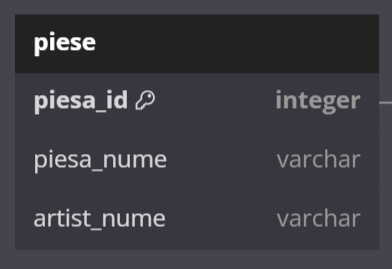
***9. Utilizatori\_si\_grupuri***



***Utilizator\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare utilizator.

***Grup\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic al unui grup.

***10. Piese***

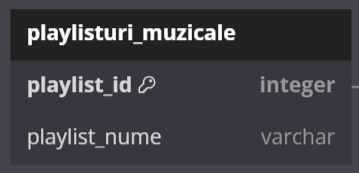


***Piesa\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare piesă.

***Piesa\_nume (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând numele piesei.

***Artist\_nume (Not Null****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând numele artistului.

***11. Playlisturi\_muzicale***



***Playlist\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare playlist.

***Playlist\_nume (-****)* – Variabilă de tip caracter de lungime maxim 100, reprezentând numele playlistului.

***12. Playlisturi\_si\_piese***

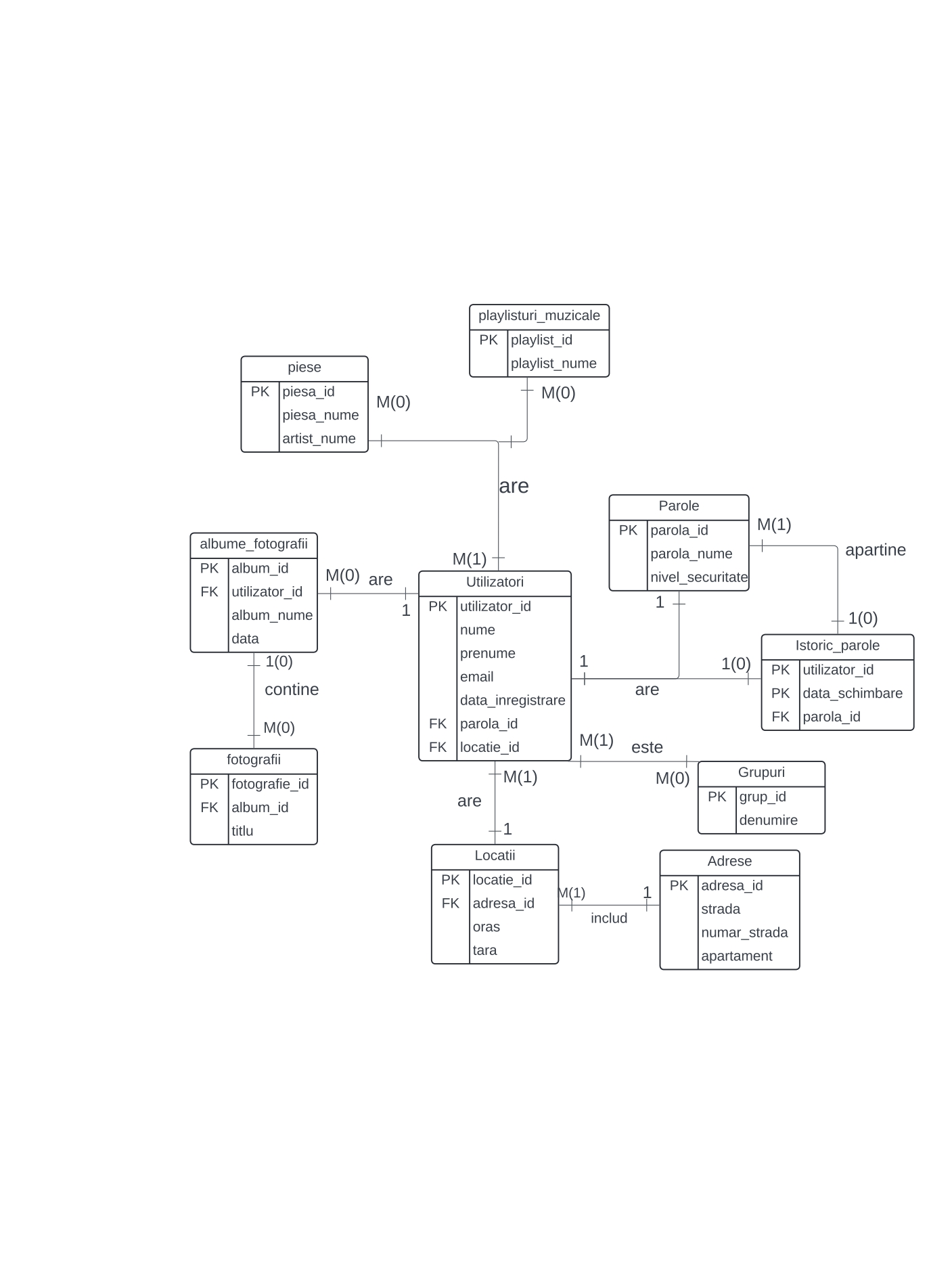


***Piesa\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare piesă.

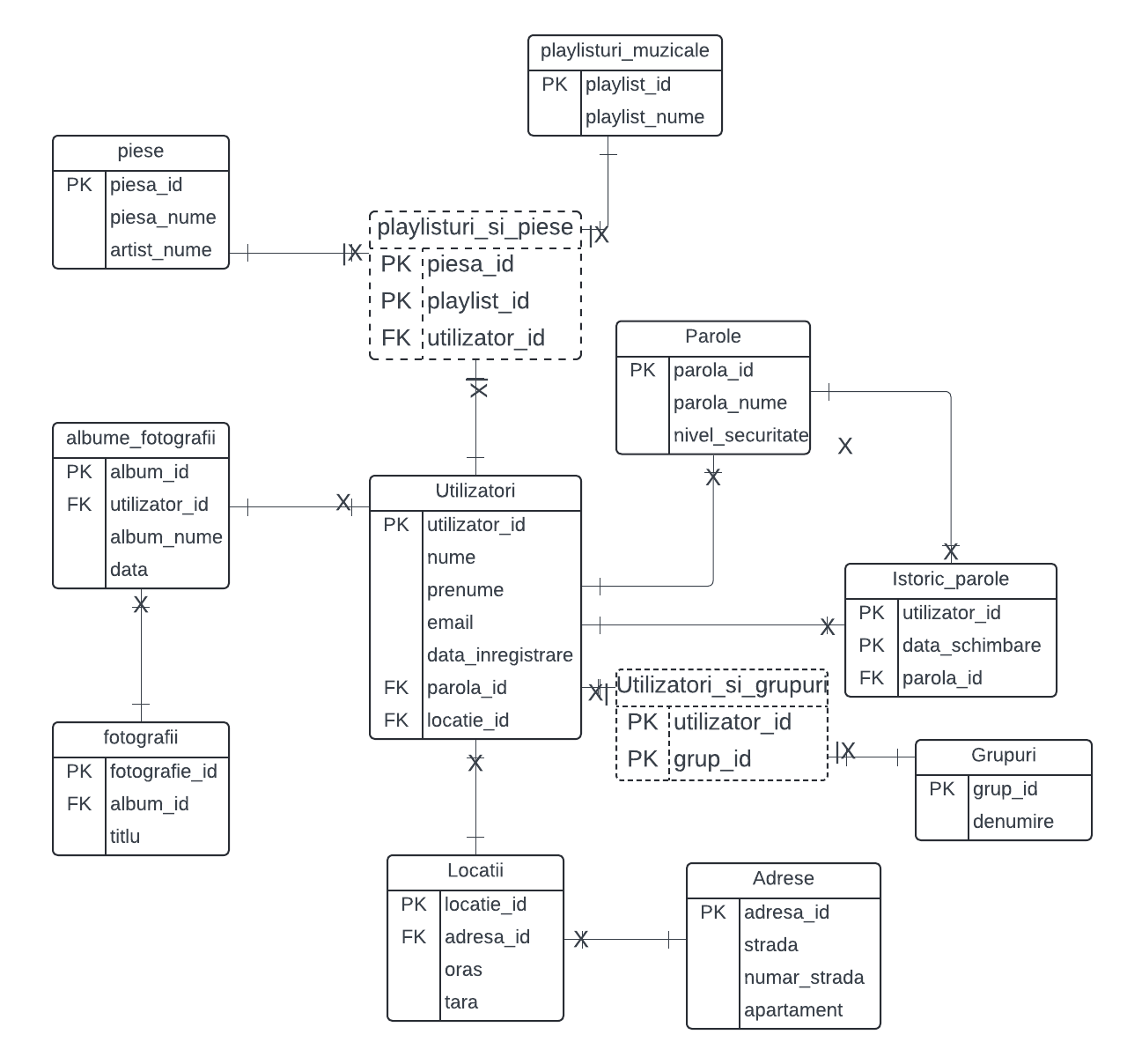
***Playlist\_id (PK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare playlist.

***Utilizator\_id (FK****)* – Variabilă de tip întreg de lungime maxim 5, reprezentând identificatorul unic pentru fiecare utilizator.

***6. Diagrama E/R***



***7. Diagrama Conceptuală***



***8. Scheme relaționale***

Schemele relaționale corespunzătoare diagramei conceptuale sunt următoarele:

* UTILIZATORI (utilizator\_id#, nume, prenume, email, parola\_id, locatie\_id, data\_inregistrare)
* FOTOGRAFII (fotografie\_id#, album\_id, titlu)
* ALBUME\_FOTOGRAFII (album\_id#, utilizator\_id, album\_nume, data)
* PIESE (piesa\_id#, piesa\_nume, artist\_nume)
* PLAYLISTURI\_MUZICALE (playlist\_id#, playlist\_nume)
* PLAYLISTURI\_SI\_PIESE (piesa\_id#, playlist\_id#, utilizator\_id)
* PAROLE (parola\_id#, parola\_nume, nivel\_securitate)
* ISTORIC\_PAROLE (utilizator\_id#, data\_schimbare#, parola\_id, parola\_nume)
* GRUPURI (grup\_id#, denumire)
* UTILIZATORI\_SI\_GRUPURI (utilizator\_id#, grup\_id#)
* LOCAȚII (locatie\_id#, adresa\_id, oras, tara)
* ADRESE (adresa\_id#, strada, numar\_strada, apartament)

***9. Realizarea normalizării***

**Prima Formă Normală (1NF)**

**Definiție**: O tabelă este în prima formă normală (1NF) dacă:

1. Toate coloanele conțin doar valori atomice (indivizibile).
2. Toate coloanele din tabelă conțin un singur tip de valoare.
3. Toate valorile dintr-o coloană sunt de același tip de date.
4. Fiecare rând este unic.

**Exemplu (***albume\_fotografii***):**

Poate exista cazul în care un utilizator crează 3 albume de fotografii in aceeași zi. Fară FN1 am avea în tabela albume\_fotografii o înregistrare de genul:

|  |  |  |
| --- | --- | --- |
| Utilizator\_id | Album\_nume | Data |
| 100 | “Primele poze”, “La mare”, “Ziua de naștere“ | 03.08.2023 |

Astfel, se vor introduce 3 înregistrări diferite:

|  |  |  |  |
| --- | --- | --- | --- |
| Album\_id | Utilizator\_id | Album\_nume | Data |
| 1 | 100 | “Primele poze” | 03.08.2023 |
| 2 | 100 | “La mare” | 03.08.2023 |
| 3 | 100 | “Ziua de naștere“ | 03.08.2023 |

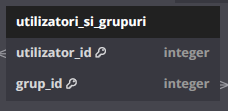
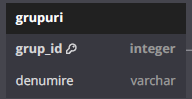
**A Doua Formă Normală (2NF)**

**Definiție:** O tabelă este în a doua formă normală (2NF) dacă**:**

1. Este în 1NF.
2. Toate atributele non-cheie sunt funcțional dependente de întreaga cheie primară (elimină dependențele parțiale).

**Exemplu (***utilizatori\_si\_grupuri***):**

În tabela utilizatori\_si\_grupuri avem cheie primară compusă utilizator\_id+grup\_id, fară să avem alte atribute, deci este în FN2. Ar fi fost greșit să fie in tabela utilizatori\_si\_grupuri și atributul grup\_nume, deoarece numele grupului este legat strict de grup, nu și de utilizator. Astfel, modul corect de a proceda este de a pune atributul grup\_nume în tabela GRUPURI.



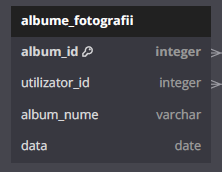
**A Treia Formă Normală (3NF)**

**Definiție**: O tabelă este în a treia formă normală (3NF) dacă:

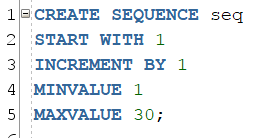
1. Este în 2NF.
2. Toate atributele non-cheie sunt funcțional dependente doar de cheie primară (elimină dependențele tranzitive).

**Exemplu (***albume\_fotografii***):**

Deoarece toate atributele care nu sunt chei (primary sau foreign) nu depind decât de chei, și nu depind unele de altele, ne aflăm în FN3. Ar fi fost greșit să avem în tabela albume\_fotografii un camp legat de foreign key, în cazul de față utilizator\_id (cum ar fi utilizator\_nume, email, nume, prenume, etc.)



***10. Crearea unei secvențe utilizată în inserarea înregistrărilor în tabele***



CREATE SEQUENCE seq

START WITH 1

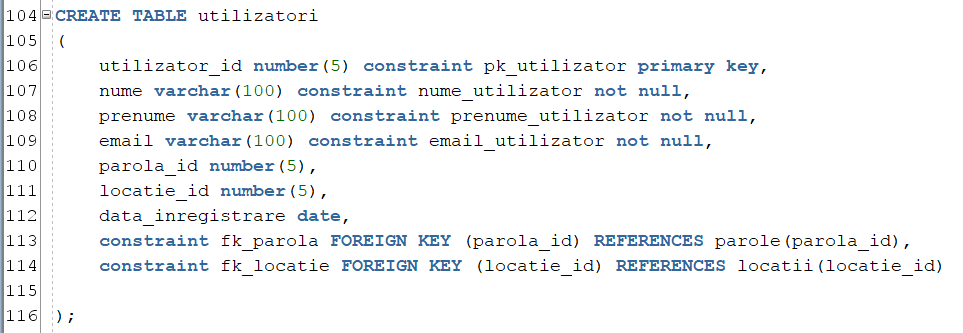
INCREMENT BY 1

MINVALUE 1

MAXVALUE 30;

***11. Crearea tabelelor și inserarea datelor.***

***Utilizatori***



CREATE TABLE utilizatori

(

utilizator\_id number(5) constraint pk\_utilizator primary key,

nume varchar(100) constraint nume\_utilizator not null,

prenume varchar(100) constraint prenume\_utilizator not null,

email varchar(100) constraint email\_utilizator not null,

parola\_id number(5),

locatie\_id number(5),

data\_inregistrare date,

constraint fk\_parola FOREIGN KEY (parola\_id) REFERENCES parole(parola\_id),

constraint fk\_locatie FOREIGN KEY (locatie\_id) REFERENCES locatii(locatie\_id)

);



INSERT INTO utilizatori

VALUES (seq.NEXTVAL, 'Popescu', 'Andrei', 'popescu\_andrei@gmail.com', 1, 1, TO\_DATE('04/12/2023', 'DD/MM/YYYY'));

INSERT INTO utilizatori

VALUES (seq.NEXTVAL, 'Dimitrie', 'Roxana', 'dimitrie\_roxx@gmail.com', 3, 4, TO\_DATE('12/06/2023', 'DD/MM/YYYY'));

INSERT INTO utilizatori

VALUES (seq.NEXTVAL, 'Ionescu', 'Maria', 'ionescu\_maria@gmail.com', 2, 2, TO\_DATE('05/11/2023', 'DD/MM/YYYY'));

INSERT INTO utilizatori

VALUES (seq.NEXTVAL, 'Georgescu', 'Mihai', 'georgescu\_mihai@gmail.com', 5, 3, TO\_DATE('06/10/2023', 'DD/MM/YYYY'));

INSERT INTO utilizatori

VALUES (seq.NEXTVAL, 'Vasilescu', 'Elena', 'vasilescu\_elena@gmail.com', 4, 3, TO\_DATE('22/12/2023', 'DD/MM/YYYY'));

INSERT INTO utilizatori

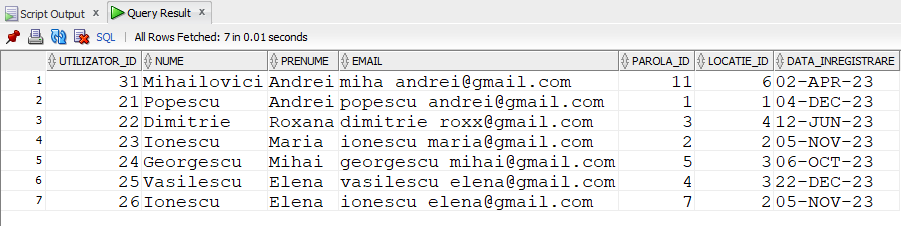
VALUES (seq.NEXTVAL, 'Ionescu', 'Elena', 'ionescu\_elena@gmail.com', 7, 2, TO\_DATE('05/11/2023', 'DD/MM/YYYY'));

INSERT INTO utilizatori

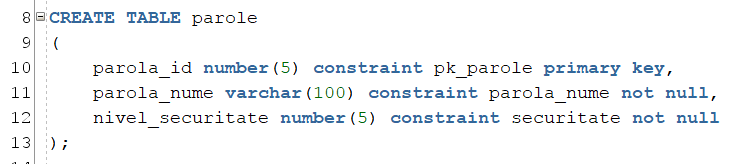
VALUES (31, 'Mihailovici', 'Andrei', 'miha\_andrei@gmail.com', 11, 6, TO\_DATE('02/04/2023', 'DD/MM/YYYY'));

select \*

from utilizatori;



***Parole***



CREATE TABLE parole

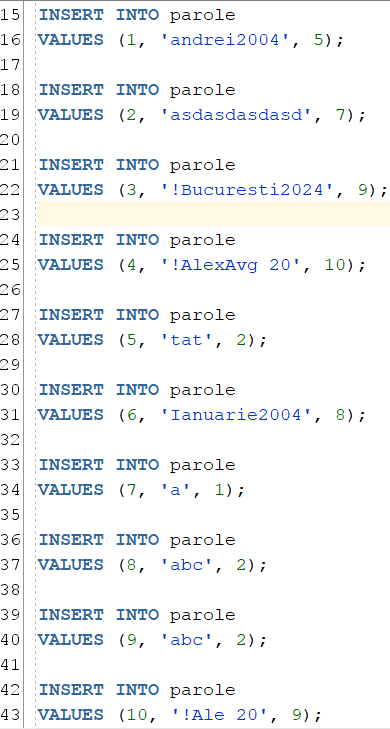
(

parola\_id number(5) constraint pk\_parole primary key,

parola\_nume varchar(100) constraint parola\_nume not null,

nivel\_securitate number(5) constraint securitate not null

);



INSERT INTO parole

VALUES (1, 'andrei2004', 5);

INSERT INTO parole

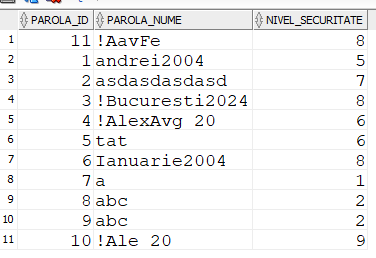
VALUES (2, 'asdasdasdasd', 7);

INSERT INTO parole

VALUES (3, '!Bucuresti2024', 9);

INSERT INTO parole

VALUES (4, '!AlexAvg 20', 10);



INSERT INTO parole

VALUES (5, 'tat', 2);

INSERT INTO parole

VALUES (6, 'Ianuarie2004', 8);

INSERT INTO parole

VALUES (7, 'a', 1);

INSERT INTO parole

VALUES (8, 'abc', 2);

INSERT INTO parole

VALUES (9, 'abc', 2);

INSERT INTO parole

VALUES (10, '!Ale 20', 9);

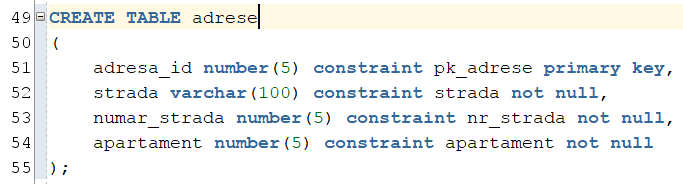
INSERT INTO parole

VALUES (11, '!AavFe', 8);

select \*

from parole;

***Adrese***



CREATE TABLE adrese

(

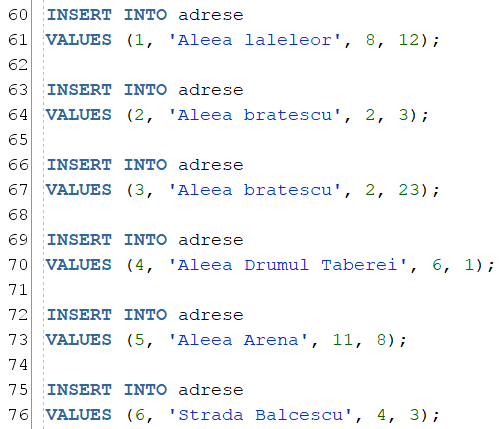
adresa\_id number(5) constraint pk\_adrese primary key,

strada varchar(100) constraint strada not null,

numar\_strada number(5) constraint nr\_strada not null,

apartament number(5) constraint apartament not null

);



INSERT INTO adrese

VALUES (1, 'Aleea laleleor', 8, 12);

INSERT INTO adrese

VALUES (2, 'Aleea bratescu', 2, 3);

INSERT INTO adrese

VALUES (3, 'Aleea bratescu', 2, 23);

INSERT INTO adrese

VALUES (4, 'Aleea Drumul Taberei', 6, 1);

INSERT INTO adrese

VALUES (5, 'Aleea Arena', 11, 8);

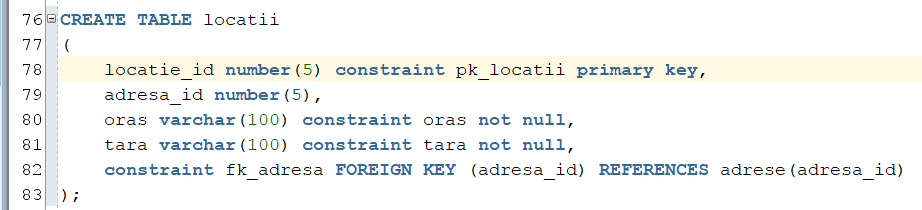
INSERT INTO adrese

VALUES (6, 'Strada Balcescu', 4, 3);

select \*

from adrese;

***Locatii***



CREATE TABLE locatii

(

locatie\_id number(5) constraint pk\_locatii primary key,

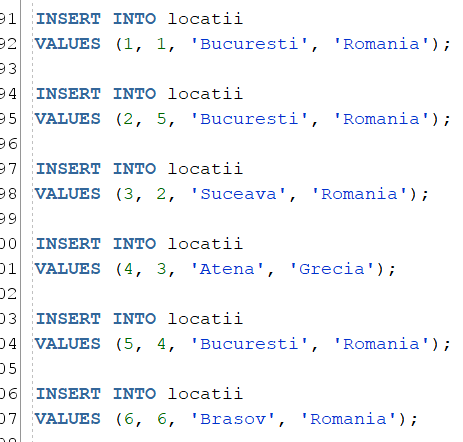
adresa\_id number(5),

oras varchar(100) constraint oras not null,

tara varchar(100) constraint tara not null,

constraint fk\_adresa FOREIGN KEY (adresa\_id) REFERENCES adrese(adresa\_id)

);



INSERT INTO locatii

VALUES (1, 1, 'Bucuresti', 'Romania');

INSERT INTO locatii

VALUES (2, 5, 'Bucuresti', 'Romania');

INSERT INTO locatii

VALUES (3, 2, 'Suceava', 'Romania');

INSERT INTO locatii

VALUES (4, 3, 'Atena', 'Grecia');

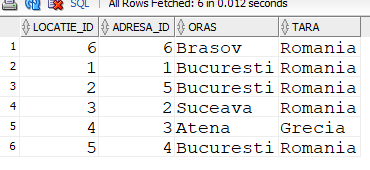
INSERT INTO locatii

VALUES (5, 4, 'Bucuresti', 'Romania');

INSERT INTO locatii

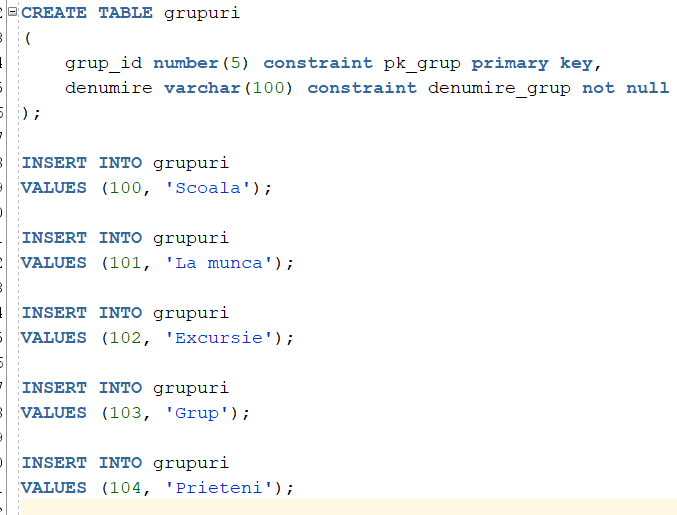
VALUES (6, 6, 'Brasov', 'Romania');

select \*



from locatii;

***Grupuri***



CREATE TABLE grupuri

(

grup\_id number(5) constraint pk\_grup primary key,

denumire varchar(100) constraint denumire\_grup not null

);

INSERT INTO grupuri

VALUES (100, 'Scoala');

INSERT INTO grupuri

VALUES (101, 'La munca');

INSERT INTO grupuri

VALUES (102, 'Excursie');

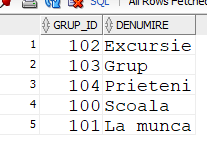
INSERT INTO grupuri

VALUES (103, 'Grup');

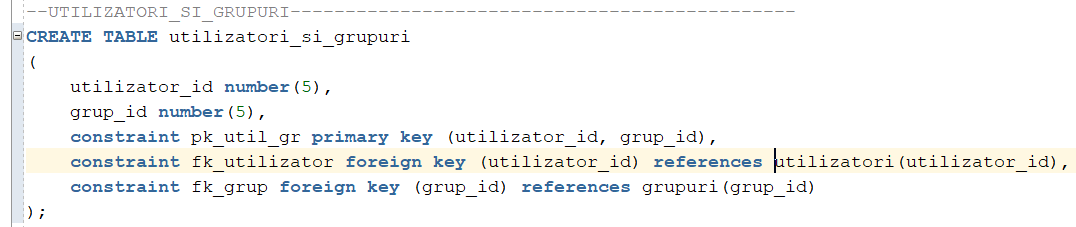
INSERT INTO grupuri

VALUES (104, 'Prieteni');select \*

from grupuri;



***Utilizatori\_si\_grupuri***



CREATE TABLE utilizatori\_si\_grupuri

(

utilizator\_id number(5),

grup\_id number(5),

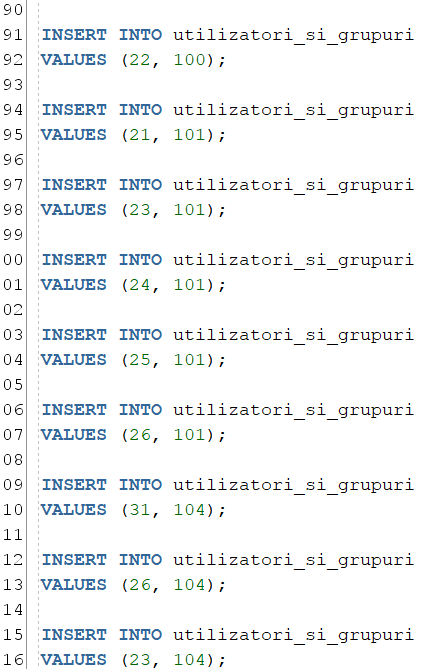
constraint pk\_util\_gr primary key (utilizator\_id, grup\_id),

constraint fk\_utilizator foreign key (utilizator\_id) references utilizatori(utilizator\_id),

constraint fk\_grup foreign key (grup\_id) references grupuri(grup\_id

);

INSERT INTO utilizatori\_si\_grupuri



VALUES (21, 100);

INSERT INTO utilizatori\_si\_grupuri

VALUES (22, 100);

INSERT INTO utilizatori\_si\_grupuri

VALUES (21, 101);

INSERT INTO utilizatori\_si\_grupuri

VALUES (23, 101);

INSERT INTO utilizatori\_si\_grupuri

VALUES (24, 101);

INSERT INTO utilizatori\_si\_grupuri

VALUES (25, 101);

INSERT INTO utilizatori\_si\_grupuri

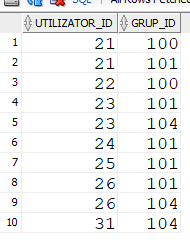
VALUES (26, 101);

INSERT INTO utilizatori\_si\_grupuri

VALUES (31, 104);

INSERT INTO utilizatori\_si\_grupuri

VALUES (26, 104);



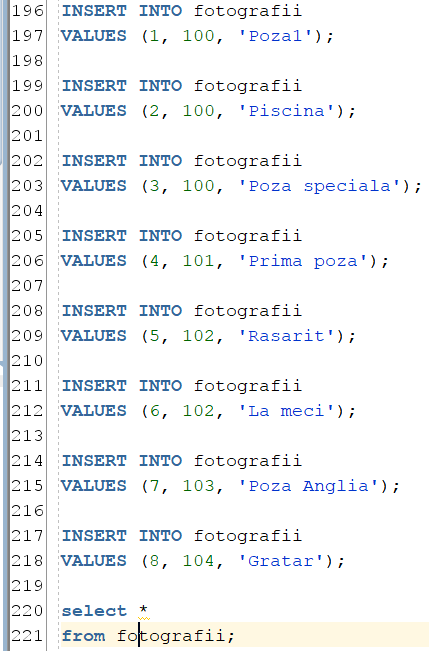
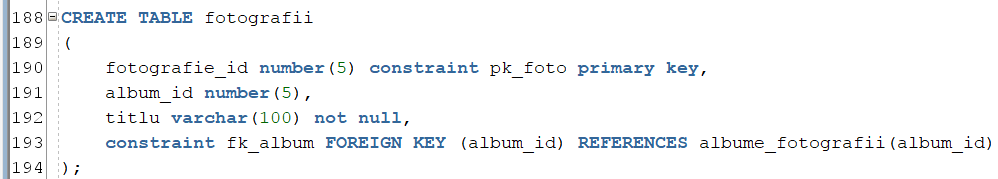
INSERT INTO utilizatori\_si\_grupuri

VALUES (23, 104);

select \*

from utilizatori\_si\_grupuri;

***Fotografii***



CREATE TABLE fotografii

(

fotografie\_id number(5) constraint pk\_foto primary key,

album\_id number(5),

titlu varchar(100) not null,

constraint fk\_album FOREIGN KEY (album\_id) REFERENCES albume\_fotografii(album\_id)

);

INSERT INTO fotografii

VALUES (1, 100, 'Poza1');

INSERT INTO fotografii

VALUES (2, 100, 'Piscina');

INSERT INTO fotografii

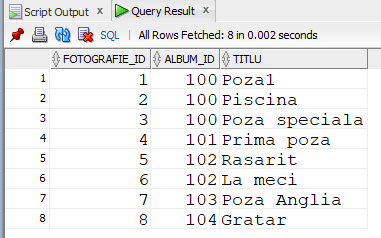
VALUES (3, 100, 'Poza speciala');

INSERT INTO fotografii

VALUES (4, 101, 'Prima poza');

INSERT INTO fotografii

VALUES (5, 102, 'Rasarit');



INSERT INTO fotografii

VALUES (6, 102, 'La meci');

INSERT INTO fotografii

VALUES (7, 103, 'Poza Anglia');

INSERT INTO fotografii

VALUES (8, 104, 'Gratar');

select \*

from fotografii;

***Albume\_fotografii***



CREATE TABLE albume\_fotografii

(

album\_id number(5) constraint pk\_album primary key,

utilizator\_id number(5),

album\_nume varchar(100) constraint nume\_album not null,

data\_album date,

constraint fk\_util FOREIGN KEY (utilizator\_id) REFERENCES utilizatori(utilizator\_id)

);

INSERT INTO albume\_fotografii

VALUES (100, 21, 'La mare', TO\_DATE('25/12/2023', 'DD/MM/YYYY'));

INSERT INTO albume\_fotografii

VALUES (101, 21, 'Ziua mea', TO\_DATE('01/02/2024', 'DD/MM/YYYY'));

INSERT INTO albume\_fotografii

VALUES (102, 23, 'Scoala', TO\_DATE('29/11/2023', 'DD/MM/YYYY'));

INSERT INTO albume\_fotografii

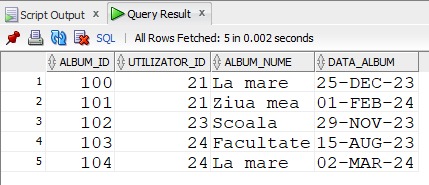
VALUES (103, 24, 'Facultate', TO\_DATE('15/08/2023', 'DD/MM/YYYY'));

INSERT INTO albume\_fotografii

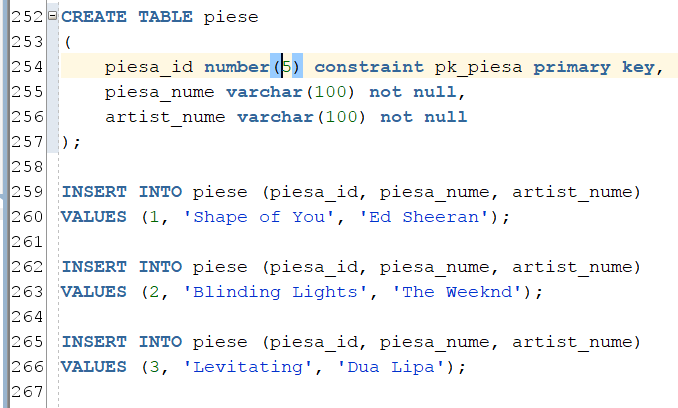
VALUES (104, 24, 'La mare', TO\_DATE('02/03/2024', 'DD/MM/YYYY'));

select \*

from albume\_fotografii;



***Piese***



CREATE TABLE piese

(

piesa\_id number(5) constraint pk\_piesa primary key,

piesa\_nume varchar(100) not null,

artist\_nume varchar(100) not null

);

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (1, 'Shape of You', 'Ed Sheeran');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (2, 'Blinding Lights', 'The Weeknd');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (3, 'Levitating', 'Dua Lipa');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (4, 'Bad Guy', 'Billie Eilish');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (5, 'Rockstar', 'Post Malone');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (6, 'Old Town Road', 'Lil Nas X');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (7, 'Senorita', 'Camila Cabello');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

VALUES (8, 'Circles', 'Post Malone');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

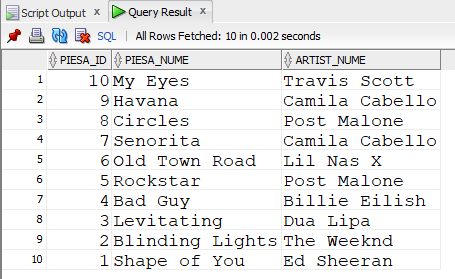
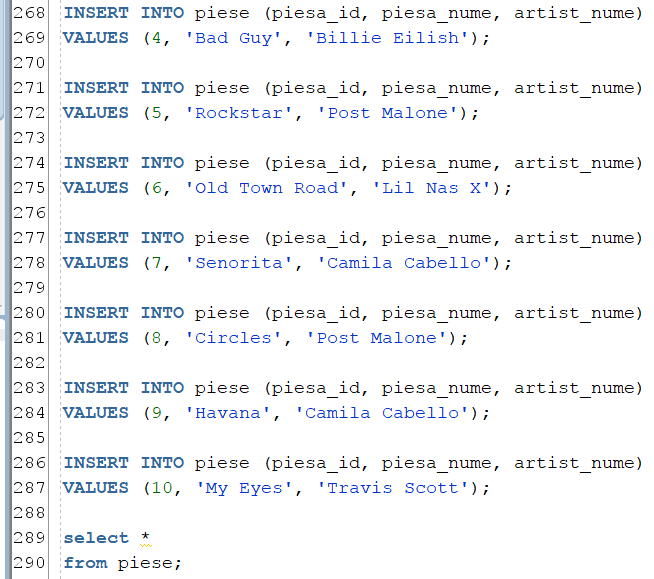
VALUES (9, 'Havana', 'Camila Cabello');

INSERT INTO piese (piesa\_id, piesa\_nume, artist\_nume)

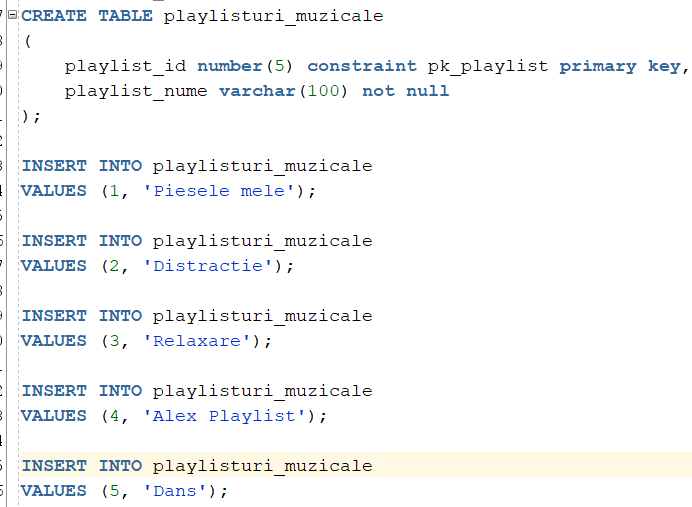
VALUES (10, 'My Eyes', 'Travis Scott');

select \*

from piese;



***Playlisturi\_muzicale***



CREATE TABLE playlisturi\_muzicale

(

playlist\_id number(5) constraint pk\_playlist primary key,

playlist\_nume varchar(100) not null

);

INSERT INTO playlisturi\_muzicale

VALUES (1, 'Piesele mele');

INSERT INTO playlisturi\_muzicale

VALUES (2, 'Distractie');

INSERT INTO playlisturi\_muzicale

VALUES (3, 'Relaxare');

INSERT INTO playlisturi\_muzicale

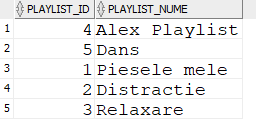
VALUES (4, 'Alex Playlist');

INSERT INTO playlisturi\_muzicale

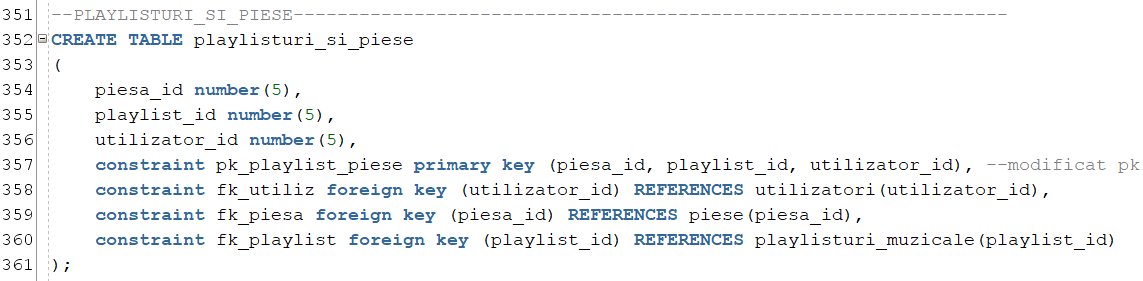
VALUES (5, 'Dans');

select \*

from playlisturi\_muzicale;



***Playlisturi\_si\_piese***



CREATE TABLE playlisturi\_si\_piese

(

piesa\_id number(5),

playlist\_id number(5),

utilizator\_id number(5),

constraint pk\_playlist\_piese primary key (piesa\_id, playlist\_id, utilizator\_id), --modificat pk

constraint fk\_utiliz foreign key (utilizator\_id) REFERENCES utilizatori(utilizator\_id),

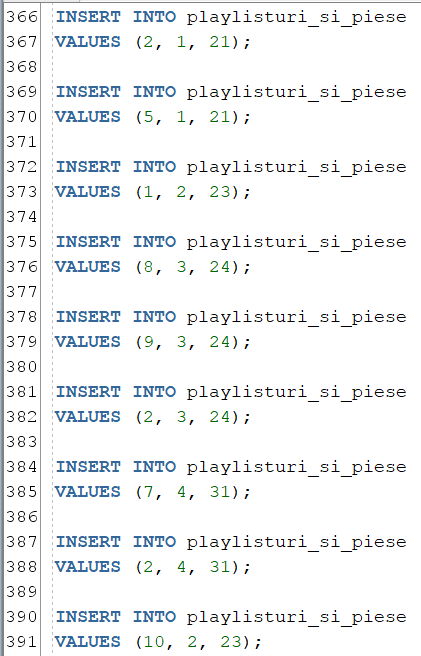
constraint fk\_piesa foreign key (piesa\_id) REFERENCES piese(piesa\_id),

constraint fk\_playlist foreign key (playlist\_id) REFERENCES playlisturi\_muzicale(playlist\_id)

);

INSERT INTO playlisturi\_si\_piese

VALUES (1, 1, 21);



INSERT INTO playlisturi\_si\_piese

VALUES (2, 1, 21);

INSERT INTO playlisturi\_si\_piese

VALUES (5, 1, 21);

INSERT INTO playlisturi\_si\_piese

VALUES (1, 2, 23);

INSERT INTO playlisturi\_si\_piese

VALUES (8, 3, 24);

INSERT INTO playlisturi\_si\_piese

VALUES (9, 3, 24);

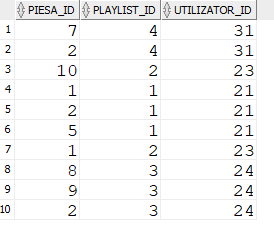
INSERT INTO playlisturi\_si\_piese

VALUES (2, 3, 24);

INSERT INTO playlisturi\_si\_piese

VALUES (7, 4, 31);

INSERT INTO playlisturi\_si\_piese



VALUES (2, 4, 31);

INSERT INTO playlisturi\_si\_piese

VALUES (10, 2, 23);

select \*

from playlisturi\_si\_piese;

***Istoric\_parole***



CREATE TABLE istoric\_parole

(

utilizator\_id number(5),

data\_schimbare date,

parola\_id number(5),

constraint pk\_istoric primary key(utilizator\_id, data\_schimbare),

constraint fk\_parola\_istoric foreign key (parola\_id) references parole(parola\_id)

);

INSERT INTO istoric\_parole

VALUES (22, TO\_DATE('27/12/2023', 'DD/MM/YYYY'), 8);

INSERT INTO istoric\_parole

VALUES (22, TO\_DATE('01/01/2024', 'DD/MM/YYYY'), 9);

INSERT INTO istoric\_parole

VALUES (24, TO\_DATE('30/12/2023', 'DD/MM/YYYY'), 10);

INSERT INTO istoric\_parole

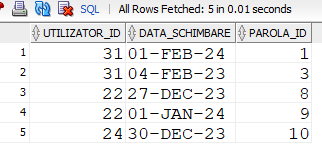
VALUES (31, TO\_DATE('01/02/2024', 'DD/MM/YYYY'), 1);

INSERT INTO istoric\_parole

VALUES (31, TO\_DATE('04/02/2023', 'DD/MM/YYYY'), 3);

select \*

from istoric\_parole;



***12. SQL***

* 1. Formulați în limbaj natural și implementați 5 cereri SQL complexe ce vor utiliza, în ansamblul lor, următoarele elemente:
  2. a) subcereri sincronizate în care intervin cel puțin 3 tabele
  3. b) subcereri nesincronizate în clauza FROM
  4. c) grupări de date, funcții grup, filtrare la nivel de grupuri cu subcereri nesincronizate (în clauza de HAVING) în care intervin cel puțin 3 tabele (in cadrul aceleiași cereri)
  5. d) ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)
  6. e) utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE
  7. f) utilizarea a cel puțin 1 bloc de cerere (clauza WITH)

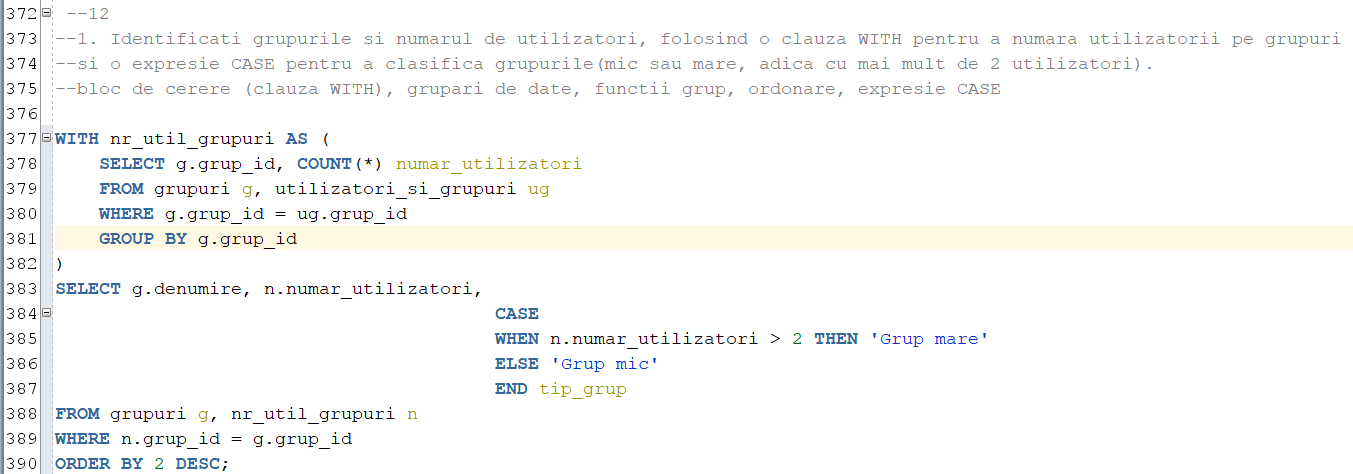
**Observație:** Într-o cerere se vor regăsi mai multe elemente dintre cele enumerate mai sus, astfel încât cele 5 cereri să le cuprindă pe toate.

**Cerința 1 (clauza WITH, grupari, functii grup, CASE)**

--1. Identificati grupurile si numarul de utilizatori, folosind o clauza WITH pentru a numara utilizatorii pe grupuri

--si o expresie CASE pentru a clasifica grupurile(mic sau mare, adica cu mai mult de 2 utilizatori).

--bloc de cerere (clauza WITH), grupari de date, functii grup, ordonare, expresie CASE



WITH nr\_util\_grupuri AS (

SELECT g.grup\_id, COUNT(\*) numar\_utilizatori

FROM grupuri g, utilizatori\_si\_grupuri ug

WHERE g.grup\_id = ug.grup\_id

GROUP BY g.grup\_id

)

SELECT g.denumire, n.numar\_utilizatori,

CASE

WHEN n.numar\_utilizatori > 2 THEN 'Grup mare'

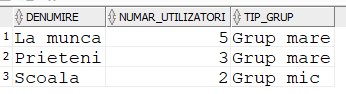
ELSE 'Grup mic'

END tip\_grup

FROM grupuri g, nr\_util\_grupuri n

WHERE n.grup\_id = g.grup\_id

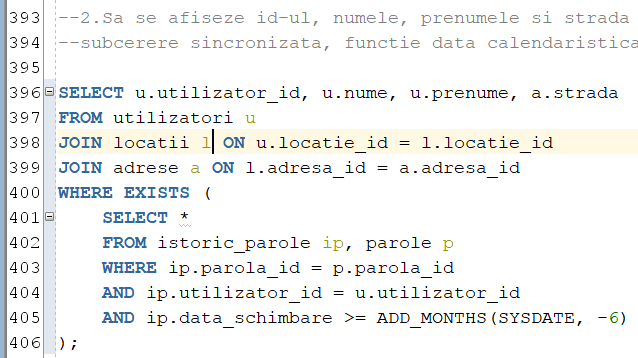
ORDER BY 2 DESC;



**Cerința 2 (subcerere sincronizata, functie data calendaristica, exists)**

--2.Sa se afiseze id-ul, numele, prenumele si strada pe care locuiesc utilizatorii care si-au schimbat parola in ultimele 6 luni.

--subcerere sincronizata, functie data calendaristica, exists



SELECT u.utilizator\_id, u.nume, u.prenume, a.strada

FROM utilizatori u

JOIN locatii l ON u.locatie\_id = l.locatie\_id

JOIN adrese a ON l.adresa\_id = a.adresa\_id

WHERE EXISTS (

SELECT \*

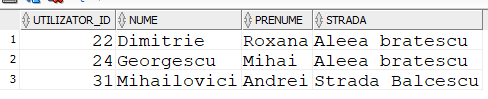
FROM istoric\_parole ip, parole p

WHERE ip.parola\_id = p.parola\_id

AND ip.utilizator\_id = u.utilizator\_id

AND ip.data\_schimbare >= ADD\_MONTHS(SYSDATE, -6)

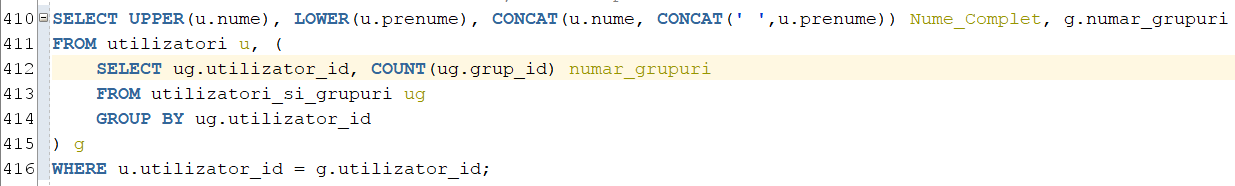
);



**Cerința 3 (subcerere nesincronizata, functii pe siruri de caractere)**

--3.Sa se afiseze numele și prenumele utilizatorilor impreuna cu numarul de grupuri din care fac parte.

--subcerere nesincronizata in clauza from, functii pe siruri de caractere



SELECT UPPER(u.nume), LOWER(u.prenume), CONCAT(u.nume, CONCAT(' ',u.prenume)) Nume\_Complet, g.numar\_grupuri

FROM utilizatori u, (

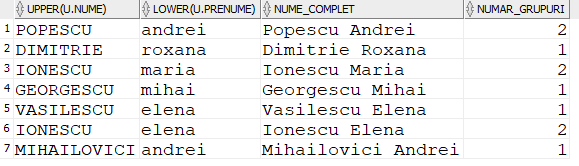
SELECT ug.utilizator\_id, COUNT(ug.grup\_id) numar\_grupuri

FROM utilizatori\_si\_grupuri ug

GROUP BY ug.utilizator\_id

) g

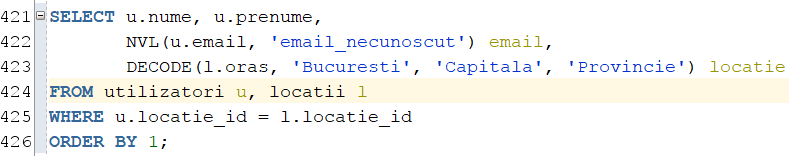
WHERE u.utilizator\_id = g.utilizator\_id;



**Cerința 4 (nvl, decode, ordonare)**

--4.Afisati id-ul, numele si prenumele utilizatorilor, impr euna cu mail-ul lor in cazul in care exista si tipul de locatie.

--nvl, decode, ordonare



SELECT u.nume, u.prenume,

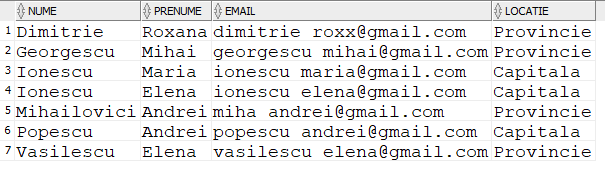
NVL(u.email, 'email\_necunoscut') email,

DECODE(l.oras, 'Bucuresti', 'Capitala', 'Provincie') locatie

FROM utilizatori u, locatii l

WHERE u.locatie\_id = l.locatie\_id

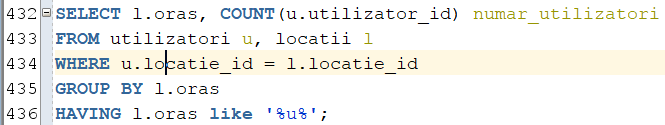
ORDER BY 1;



**Cerința 5 (grupari de date, functii grup)**

--5.Sa se afișeze orasele și numarul de utilizatori din fiecare oraș. Filtreaza rezultatele pentru a avea doar orase care contin litera 'u'.

--grupari de date, functii grup



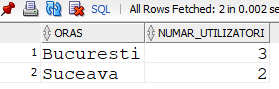
SELECT l.oras, COUNT(u.utilizator\_id) numar\_utilizatori

FROM utilizatori u, locatii l

WHERE u.locatie\_id = l.locatie\_id

GROUP BY l.oras

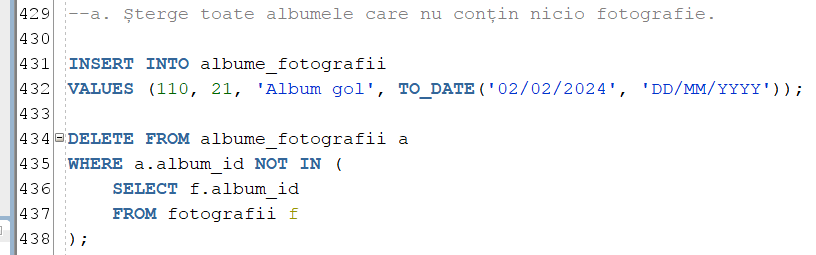
HAVING l.oras like '%u%';



***13. Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.***

***Cerința 1***

Șterge toate albumele care nu conțin nicio fotografie.



INSERT INTO albume\_fotografii

VALUES (110, 21, 'Album gol', TO\_DATE('02/02/2024', 'DD/MM/YYYY'));

DELETE FROM albume\_fotografii a

WHERE a.album\_id NOT IN (

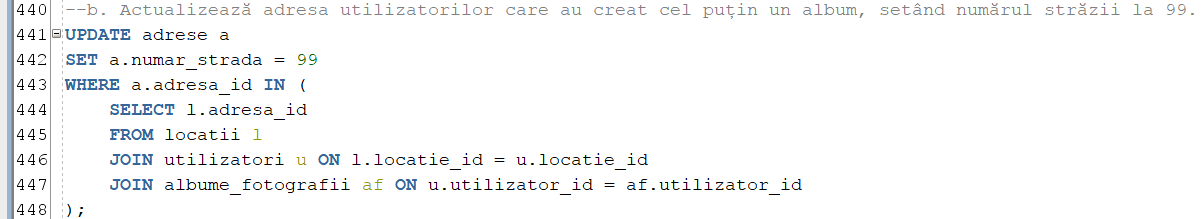
SELECT f.album\_id

FROM fotografii f

);

***Cerința 2***

Actualizeaza adresa utilizatorilor care au creat cel putin un album, setand numarul strazii la 99.



UPDATE adrese a

SET a.numar\_strada = 99

WHERE a.adresa\_id IN (

SELECT l.adresa\_id

FROM locatii l

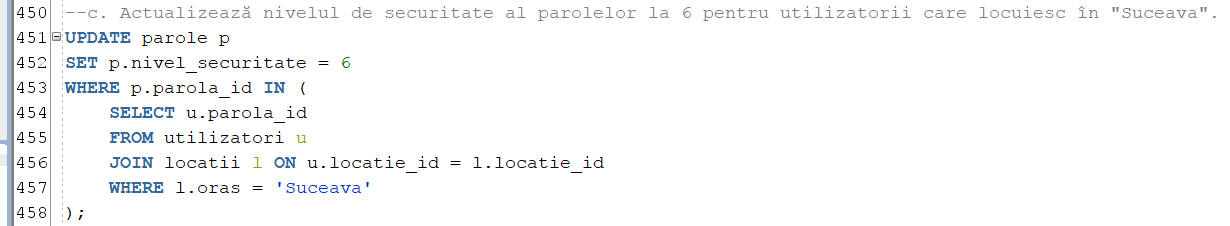
JOIN utilizatori u ON l.locatie\_id = u.locatie\_id

JOIN albume\_fotografii af ON u.utilizator\_id = af.utilizator\_id

);

***Cerința 3***

Actualizează nivelul de securitate al parolelor la 6 pentru utilizatorii care locuiesc în "Suceava".



UPDATE parole p

SET p.nivel\_securitate = 6

WHERE p.parola\_id IN (

SELECT u.parola\_id

FROM utilizatori u

JOIN locatii l ON u.locatie\_id = l.locatie\_id

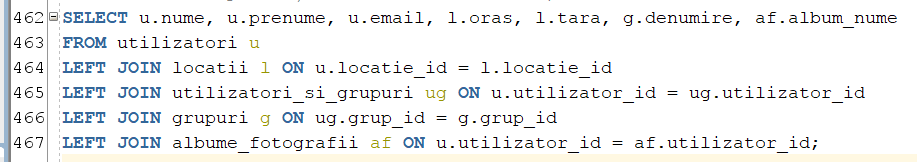
WHERE l.oras = 'Suceava'

);

***15. Formulați în limbaj natural și implementați în SQL:***

**A) O *cerere ce utilizează operația outer-join pe minimum 4 tabele.***

***Cerinta: Afisati toti utilizatorii, locatiile lor, grupurile din care fac parte si numele albumelor lor, inclusiv utilizatorii care nu au locatii, grupuri sau albume asociate.***



SELECT u.nume, u.prenume, u.email, l.oras, l.tara, g.denumire, af.album\_nume

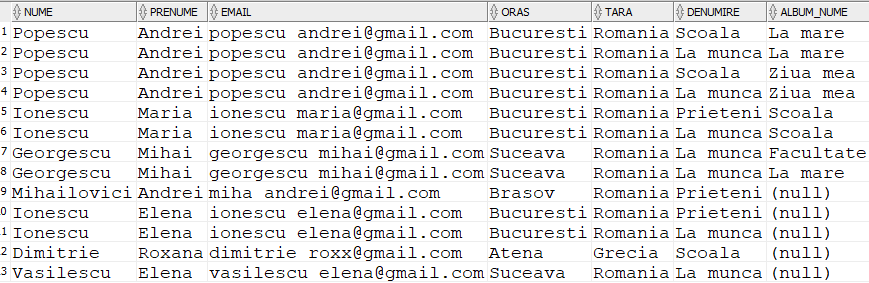
FROM utilizatori u

LEFT JOIN locatii l ON u.locatie\_id = l.locatie\_id

LEFT JOIN utilizatori\_si\_grupuri ug ON u.utilizator\_id = ug.utilizator\_id

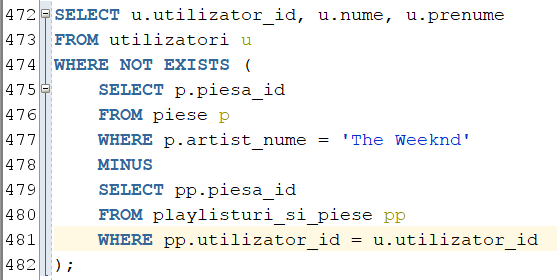
LEFT JOIN grupuri g ON ug.grup\_id = g.grup\_id

LEFT JOIN albume\_fotografii af ON u.utilizator\_id = af.utilizator\_id;



**B) *O cerere ce utilizează operația division***

***Cerinta: Gasiti utilizatorii care au adaugat toate piesele artistului 'The Weeknd' in oricare dintre playlisturile lor.***



SELECT u.utilizator\_id, u.nume, u.prenume

FROM utilizatori u

WHERE NOT EXISTS (

SELECT p.piesa\_id

FROM piese p

WHERE p.artist\_nume = 'The Weeknd'

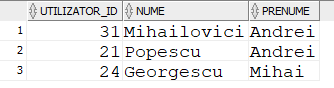
MINUS

SELECT pp.piesa\_id

FROM playlisturi\_si\_piese pp

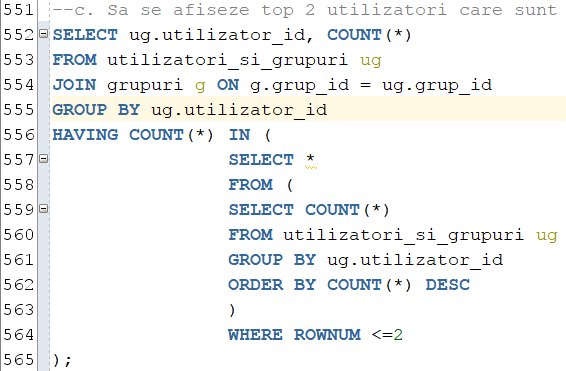
WHERE pp.utilizator\_id = u.utilizator\_id

);



**C) *O cerere care implementează analiza top-n.***

***Cerinta: Sa se afiseze top 2 utilizatori care sunt in cele mai multe grupuri.***



SELECT ug.utilizator\_id, COUNT(\*)

FROM utilizatori\_si\_grupuri ug

JOIN grupuri g ON g.grup\_id = ug.grup\_id

GROUP BY ug.utilizator\_id

HAVING COUNT(\*) IN (

SELECT \*

FROM (

SELECT COUNT(\*)

FROM utilizatori\_si\_grupuri ug

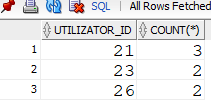
GROUP BY ug.utilizator\_id

ORDER BY COUNT(\*) DESC

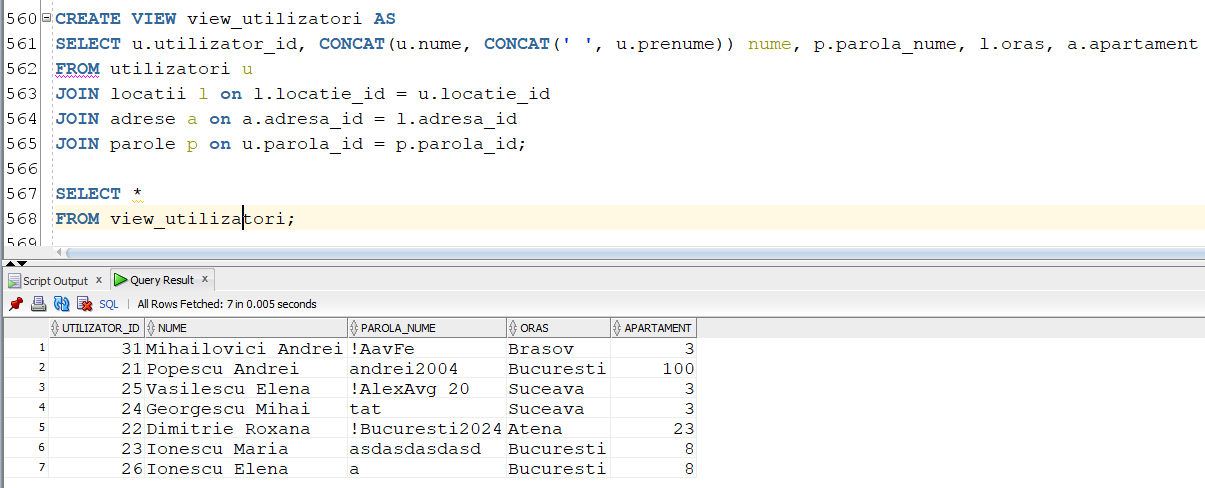
)

WHERE ROWNUM <=2

);



***14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă***



CREATE VIEW view\_utilizatori AS

SELECT u.utilizator\_id, CONCAT(u.nume, CONCAT(' ', u.prenume)) nume, p.parola\_nume, l.oras, a.apartament

FROM utilizatori u

JOIN locatii l on l.locatie\_id = u.locatie\_id

JOIN adrese a on a.adresa\_id = l.adresa\_id

JOIN parole p on u.parola\_id = p.parola\_id;

SELECT \*

FROM view\_utilizatori;

--permis - update

UPDATE view\_utilizatori

SET apartament = 100

WHERE nume = 'Popescu Andrei';

--actiune nepermisa inserarea de date

INSERT INTO view\_utilizatori

VALUES (50, 'Agusoaei Alexandru', 'parola','Bucuresti', 10);

