*1. Descrierea modelului real, a utilității acestuia și a regulilor de funcționare.*

Aplicatia este proiectata pentru e exemplifica utilizarea unei baze de date a unui magazine de muzica (cd-uri, casete, vinyl-uri), care gestioneaza situatia produselor aflate spre vanzare si inchiriere (albume muzicale = colectii de melodii), tine o evidenta a melodiilor componente ale fiecarui album si asociaza cu fiecare melodie un artist, respectiv unul sau multe genuri muzicale. De asemenea, in baza de date se stocheaza evidenta clientilor care au inchiriat albume, evidenta angajatiilor care au procesat inchirierile si vanzarile si detaliile asociate componentelor albumelor (casa de discuri, tara artistului).

**Utilitatea** unei astfel de aplicatii este, pe langa component evidenta de management al unui magazin cu acest specific, de asemenea si component de organizare a unei colectii muzicale complexe, ce poate sta la baza unei platforme de tip biblioteca muzicala, unde se pot cauta, dupa nume/gen/artisti/album, orice melodie aflata in evidenta.

Aplicatia va functiona dupa **reguli** restranse de functionare: interogare (cautare date), adaugare (inserare date in tabele), actualizare date din fiecare table si stergere date in metoda necascadata (dintr-o singura tabele la un anumit moment).

*2. Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului.*

Modelul relational prezent va impune constrangeri de tip

1. cheie primara

2. cheie straina

3. constrangere de unicitate

4. constrangere de tip check

**Constrangerile de tip cheie primara** sunt implementate cu scopul inregistrarii unice a fiecarei intrare (rand) din fiecare tabel. Aceasta constrangere se aplica asupra unor campuri de tip numeric, ce vor constitui ID-uri pentru inregistrari, iar ID-urile vor fi elementele de legatura (relatie) intre tabele.

**Constrangerile de tip cheie straina** constituie reguli de asociere a datelor intre tabele, prin metoda ID-ului referinta. Astfel, anumite tabele au impusa asocierea unui ID existent in alta tabela, logic proiectata in legatura cu tabela referinta. Se pot realiza, astfel, relatii si interogari complexe asupra bazei.

**Constrangerea de unicitate** este neobligatorie intr-un model functional in care exista constrangerea de tip cheie primara, dar este utila in modelul real unde atributele cu legatura in viata reala dispun de identificatori unici (ex: CNP). Aceasta constrangere va impiedica adaugarea unui client/angajat de doua ori (acelasi cod numeric personal), dar si existenta titlurilor identice in tabelele de tip nomenclator (unicitate pe nume).

**Constrangerea de tip check** este folosita pentru validarea modelului real de functionare a unei afaceri de tipul celei aflate in discutie (ex: Vanzarea unui produs trebuie sa se faca pentru un pret mai mare de 0 unitati monetare, situatia existentei unui card de fidelitate poate fi ori adevarata =1, ori falsa =0). Astfel, baza de date interpreteaza date de tip boolean prin valori numerice limitate (0 si 1), dar si impiedica stocarea datelor logic-gresite in alte campuri.

*3. Descrierea entităților, incluzând precizarea cheii primare.*

|  |  |  |
| --- | --- | --- |
| **ENTITATE** | **CHEIE PRIMARE** | **DESCRIERE** |
| TARA | PK\_TARA | Nomenclator de tari din care provin artistii aflati in evidenta |
| ARTIST | PK\_ARTIST | Lista de artisti care interpreteaza melodiile de pe albumele vandute in magazin, asociati cu o tara de origine |
| GEN | PK\_GEN | Lista genuri muzicale asociate cu melodiile de pe albume |
| TIP\_ALBUM | PK\_TIP\_ALBUM | Formatul in care se vinde albumul (cd,caseta,vinyl) |
| CASA\_DISCURI | PK\_CASA\_DISCURI | Lista de case de discuri, producatoarea albumelor vandute sau inchiriate |
| ALBUM | PK\_ALBUM | Lista cu albume unice efectiv comercializate de magazin, asociate cu un artist, o casa de discuri si un tip de album (daca acelasi album se vinde in mai multe forme, ex: cd si vinyl, acesta va aparea de doua ori) |
| MELODIE | PK\_MELODIE | Lista cu melodii unice, asociate cu un artist si un album |
| MELODIE\_GEN | PK\_MELODIE\_GEN | Tabelul de asociere dintre melodii si genuri muzicale (un gen poate fi asociat cu mai multe melodii, o melodie poate fi asociata cu mai multe genuri) |
| CLIENT | PK\_CLIENT | Lista cu clientii inregistrati care au efectuat inchirieri de albume |
| ANGAJAT | PK\_ANGAJAT | Lista cu angajatii magazinului, asociati cu vanzari si inchirieri de albume |
| VANZARE | PK\_VANZARE | Evidenta vanzarilor de albume (ce album s-a vandut, ce angajat a vandut) |
| INCHIRIERE | PK\_INCHIRIERE | Evidenta inchirierilor de albume (ce album s-a inchiriat, ce angajat a efectuat inchirierea, ce client l-a inchiriat) |

*4. Descrierea relațiilor, incluzând precizarea cardinalității acestora.*

* TARA – ARTIST ( UNUL – 0,1 SAU MAI MULTI )
* ARTIST – MELODIE ( UNUL – 0,1 SAU MULTI )
* MELODIE – MELODIE\_GEN ( UNUL – MAI MULTI )
* GEN – MELODIE\_GEN ( UNUL – MAI MULTI )
* ARTIST – ALBUM ( UNUL – MAI MULTI )
* ALBUM – MELODIE ( 0, UNUL – MAI MULTI )
* TIP\_ALBUM – ALBUM ( UNUL – MAI MULTI )
* CASA\_DISCURI – ALBUM ( UNUL – MAI MULTI )
* CLIENT – INCHIRIERE ( UNUL – MAI MULTI )
* ALBUM – INCHIRIERE ( UNUL – MAI MULTI )
* ANGAJAT – INCHIRIERE ( UNUL – MAI MULTI )
* ANGAJAT – VANZARE ( UNUL – MAI MULTI )
* ALBUM – VANZARE ( UNUL – MAI MULTI )

*5. Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor.*

*CREATE TABLE TARA (*

*PK\_TARA NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*NUME NVARCHAR2(50) NOT NULL, --(Romania,USA)*

***CONSTRAINT*** *PK\_TARA* ***PRIMARY******KEY*** *( PK\_TARA ),*

***CONSTRAINT*** *UQ\_NUME\_TARA* ***UNIQUE*** *( NUME )*

*);*

*CREATE TABLE ARTIST (*

*PK\_ARTIST NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*NUME NVARCHAR2(50) NOT NULL, --(Lou Bega,Louis Armstrong,Maria Tanase)*

*FK\_TARA NUMBER(18,0) NOT NULL, --(1,2,3,4)*

***CONSTRAINT*** *PK\_ARTIST* ***PRIMARY******KEY*** *( PK\_ARTIST ),*

***CONSTRAINT*** *UQ\_NUME\_ARTIST* ***UNIQUE*** *(NUME)*

*);*

*CREATE TABLE GEN (*

*PK\_GEN NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*NUME NVARCHAR2(50) NOT NULL, --(Jazz,Rock)*

***CONSTRAINT*** *PK\_GEN* ***PRIMARY******KEY*** *( PK\_GEN ),*

***CONSTRAINT*** *UQ\_NUME\_GEN* ***UNIQUE*** *(NUME)*

*);*

*CREATE TABLE TIP\_ALBUM (*

*PK\_TIP\_ALBUM NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*NUME NVARCHAR2(50) NOT NULL, --(CD,Vinyl,Caseta)*

***CONSTRAINT*** *PK\_TIP\_ALBUM* ***PRIMARY******KEY*** *( PK\_TIP\_ALBUM ),*

***CONSTRAINT*** *UQ\_NUME\_TIP\_ALBUM* ***UNIQUE*** *(NUME)*

*);*

*CREATE TABLE CASA\_DISCURI (*

*PK\_CASA\_DISCURI NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*NUME NVARCHAR2(50) NOT NULL, --(Universal, Electrocord)*

***CONSTRAINT*** *PK\_CASA\_DISCURI* ***PRIMARY******KEY*** *( PK\_CASA\_DISCURI ),*

***CONSTRAINT*** *UQ\_NUME\_CASA\_DISCURI* ***UNIQUE*** *(NUME)*

*);*

*CREATE TABLE ALBUM (*

*PK\_ALBUM NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*NUME NVARCHAR2(50) NOT NULL, --(Hello Dolly, Ciuleandra)*

*AN NUMBER(18,0) NOT NULL, --(1964,1970)*

*STOC NUMBER*

***DEFAULT*** *0 NOT NULL, --(0,1,10)*

*PRET\_VANZARE NUMBER NULL, --(100,250)*

*PRET\_INCHIRIERE NUMBER NULL, --(10,15)*

*DATA\_ADAUGARE Timestamp(3)*

***DEFAULT*** *CURRENT\_TIMESTAMP NOT NULL, --(2021-07-01)*

*FK\_ARTIST NUMBER(18,0) NOT NULL, --(1,2,3,4)*

*FK\_TIP\_ALBUM NUMBER(18,0) NOT NULL, --(1,2,3,4)*

*FK\_CASA\_DISCURI NUMBER(18,0) NOT NULL, --(1,2,3,4)*

***CONSTRAINT*** *PK\_ALBUM* ***PRIMARY******KEY*** *( PK\_ALBUM ),*

***CONSTRAINT*** *CHECK\_PRET\_VANZARE\_ALBUM* ***CHECK*** *(PRET\_VANZARE IS NULL OR PRET\_VANZARE > 0),*

***CONSTRAINT*** *CHECK\_STOC\_ALBUM* ***CHECK*** *(STOC >= 0),*

***CONSTRAINT*** *UQ\_ALBUM* ***UNIQUE*** *(NUME,FK\_ARTIST,FK\_TIP\_ALBUM)*

*);*

*CREATE TABLE MELODIE (*

*PK\_MELODIE NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*NUME NVARCHAR2(50) NOT NULL, --(What a wonderful life,Bun e vinul ghiurghiuliu)*

*DURATA NUMBER NOT NULL, --(187,194 ) secunde*

*FK\_ARTIST NUMBER(18,0) NOT NULL, --(1,2,3,4)*

*FK\_ALBUM NUMBER(18,0)*

***DEFAULT*** *NULL NULL, --(1,2,3,4)*

***CONSTRAINT*** *PK\_MELODIE* ***PRIMARY******KEY*** *( PK\_MELODIE ),*

***CONSTRAINT*** *CHECK\_DURATA\_MELODIE* ***CHECK*** *(DURATA > 0),*

***CONSTRAINT*** *UQ\_MELODIE* ***UNIQUE*** *(NUME,FK\_ARTIST,FK\_ALBUM)*

*);*

*CREATE TABLE MELODIE\_GEN (*

*PK\_MELODIE\_GEN NUMBER(18,0) NOT NULL, --(1,2,3,4….)*

*FK\_MELODIE NUMBER(18,0) NOT NULL, --(1,2,3,4)*

*FK\_GEN NUMBER(18,0) NOT NULL, --(1,2,3,4)*

***CONSTRAINT*** *PK\_MELODIE\_GEN* ***PRIMARY******KEY*** *( PK\_MELODIE\_GEN ),*

***CONSTRAINT*** *UQ\_MELODIE\_GEN* ***UNIQUE*** *( FK\_MELODIE, FK\_GEN ),*

*);*

*CREATE TABLE CLIENT (*

*PK\_CLIENT Number(18,0) NOT NULL, --(1,2,3,4….)*

*NUME Nvarchar2(50) NOT NULL, --(Popescu,Ionescu)*

*PRENUME Nvarchar2(50) NOT NULL, --(Vasile,Dumitru)*

*CNP Nvarchar2(13) NOT NULL, --(1950102345678)*

*ARE\_CARD\_FIDELITATE Number(1)*

***DEFAULT*** *0 NOT NULL, --(0,1)*

*MAIL Nvarchar2(80) NULL, --(vasile.popescu@gmail.com)*

*TELEFON Nvarchar2(15) NOT NULL, --(0712345678)*

***CONSTRAINT*** *PK\_CLIENT* ***PRIMARY******KEY*** *( PK\_CLIENT ),*

***CONSTRAINT*** *CHECK\_ARE\_CARD\_FIDELITATE* ***CHECK*** *( ARE\_CARD\_FIDELITATE in (0,1) ),*

***CONSTRAINT*** *UQ\_CNP\_CLIENT* ***UNIQUE*** *(CNP)*

*);*

*CREATE TABLE ANGAJAT (*

*PK\_ANGAJAT Number(18,0) NOT NULL, --(1,2,3,4….)*

*NUME Nvarchar2(50) NOT NULL, --(Popa)*

*PRENUME Nvarchar2(50) NOT NULL, --(Mirela)*

*CNP Nvarchar2(50) NOT NULL, --(1960304567890)*

*DATA\_ANGAJARE Timestamp(3) NOT NULL, --(2020-01-01)*

***CONSTRAINT*** *PK\_ANGAJATI* ***PRIMARY******KEY*** *( PK\_ANGAJAT ),*

***CONSTRAINT*** *UQ\_CNP\_ANGAJAT* ***UNIQUE*** *(CNP)*

*);*

*CREATE TABLE VANZARE (*

*PK\_VANZARE Number(18,0) NOT NULL, --(1,2,3,4….)*

*DATA\_VANZARE Timestamp(3)*

***DEFAULT*** *CURRENT\_TIMESTAMP NOT NULL, --(2021-07-15)*

*CANTITATE NUMBER*

***DEFAULT*** *1 NOT NULL, --(1,2)*

*DISCOUNT NUMBER*

***DEFAULT*** *0 NOT NULL, --(0,0.25,0.5)*

*FK\_ALBUM NUMBER(18,0) NOT NULL, --(1,2,3,4)*

*FK\_ANGAJAT NUMBER(18,0) NOT NULL, --(1,2,3,4)*

***CONSTRAINT*** *PK\_VANZARE* ***PRIMARY******KEY*** *( PK\_VANZARE )*

*);*

*CREATE TABLE INCHIRIERE(*

*PK\_INCHIRIERE Number(18,0) NOT NULL, --(1,2,3,4….)*

*DATA\_INCHIRIERE Timestamp(3)*

***DEFAULT*** *CURRENT\_TIMESTAMP NOT NULL, --(2021-08-01)*

*DATE\_RETUR Timestamp(3)*

***DEFAULT*** *NULL NULL, --(2021-09-01)*

*CANTITATE NUMBER*

***DEFAULT*** *1 NOT NULL, --(1,2)*

*DISCOUNT NUMBER*

***DEFAULT*** *0 NOT NULL, --(0,0.35)*

*PENALITATI NUMBER*

***DEFAULT*** *0 NOT NULL, --(15,10)*

*FK\_ALBUM NUMBER(18,0) NOT NULL, --(1,2,3,4)*

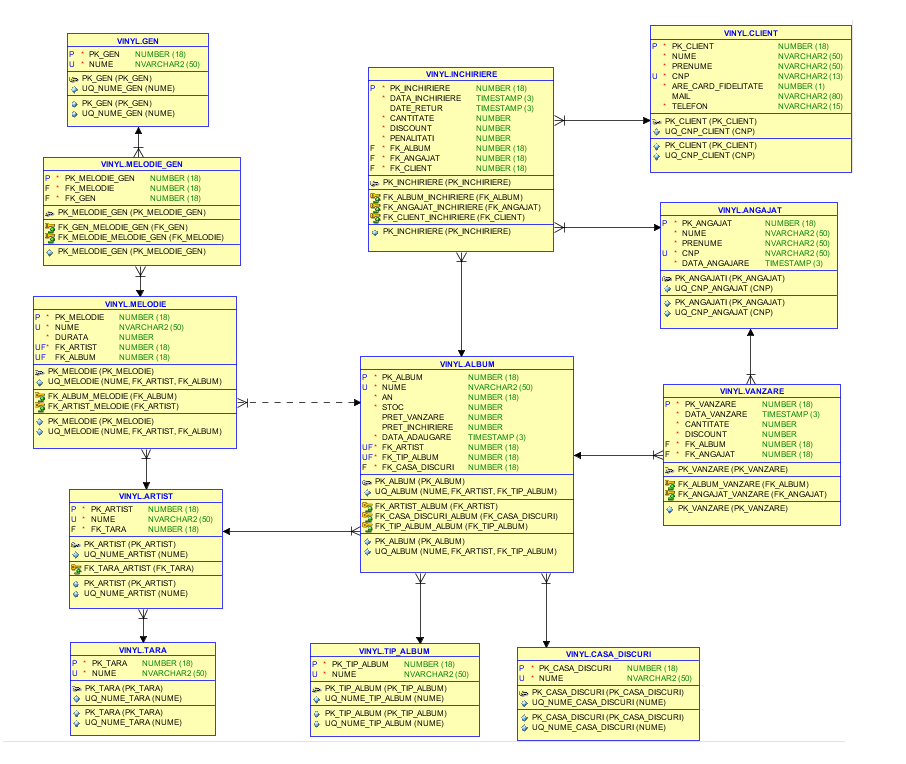
*FK\_ANGAJAT NUMBER(18,0) NOT NULL, --(1,2,3,4)*

*FK\_CLIENT NUMBER(18,0) NOT NULL, --(1,2,3,4)*

***CONSTRAINT*** *PK\_INCHIRIERE* ***PRIMARY******KEY*** *( PK\_INCHIRIERE )*

*);*

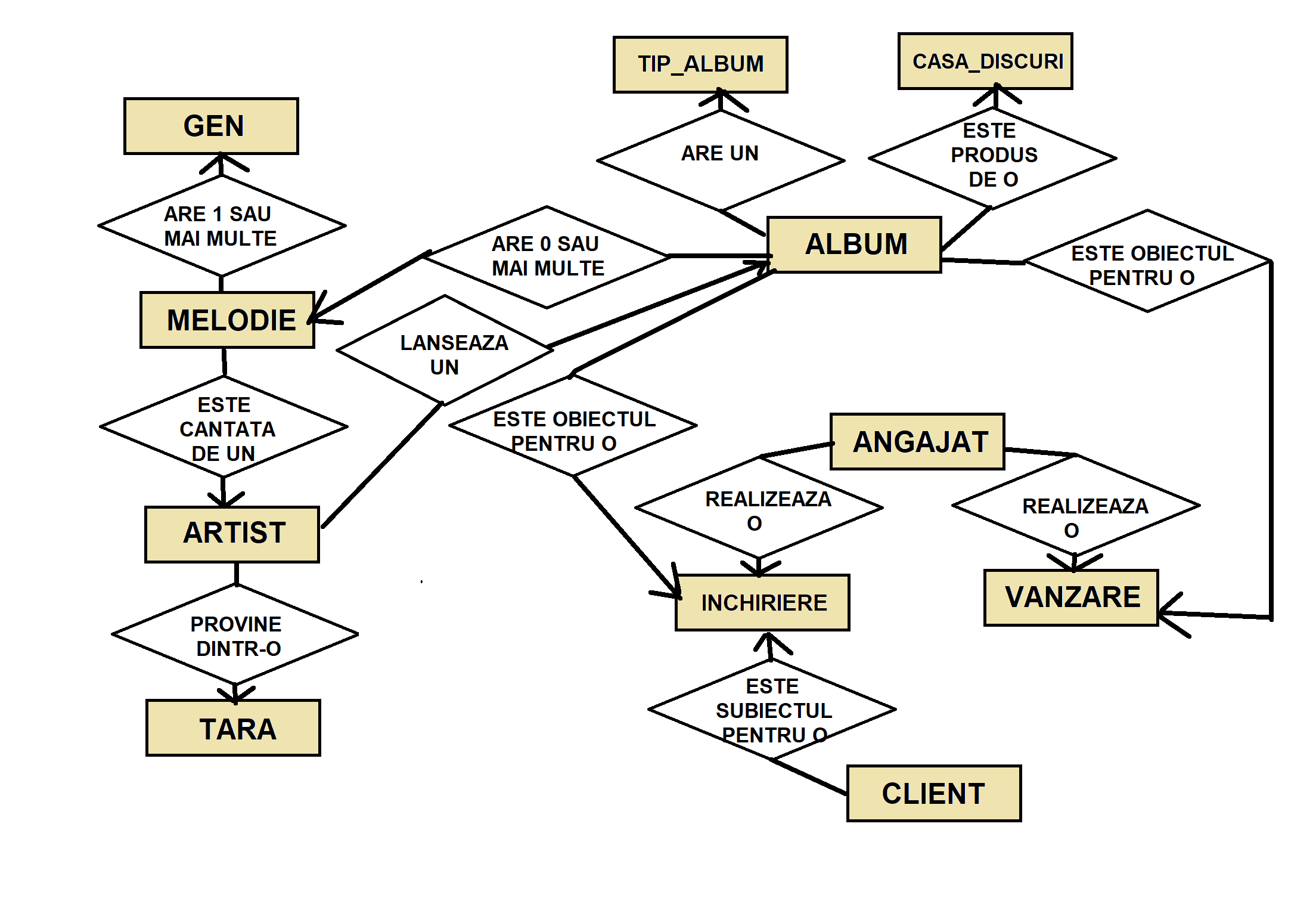
*6. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.*

**

*7. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectata la punctul 6. Diagrama conceptuală obținută trebuie să conțină minimum 6 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ.*

Tabele asociative

* VANZARE ( legatura ANGAJAT – ALBUM )
* INCHIRIERE ( legarua ANGAJAT – ALBUM – CLIENT )
* MELODIE\_GEN ( legatura MELODIE – GEN )

**

*8. Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectata la punctul 7.*

ALTER TABLE ARTIST ADD CONSTRAINT **FK\_TARA\_ARTIST** FOREIGN KEY(FK\_TARA) REFERENCES TARA (PK\_TARA);

ALTER TABLE ALBUM ADD CONSTRAINT **FK\_ARTIST\_ALBUM** FOREIGN KEY(FK\_ARTIST) REFERENCES ARTIST (PK\_ARTIST);

ALTER TABLE ALBUM ADD CONSTRAINT **FK\_TIP\_ALBUM\_ALBUM** FOREIGN KEY(FK\_TIP\_ALBUM) REFERENCES TIP\_ALBUM (PK\_TIP\_ALBUM);

ALTER TABLE ALBUM ADD CONSTRAINT **FK\_CASA\_DISCURI\_ALBUM** FOREIGN KEY(FK\_CASA\_DISCURI) REFERENCES CASA\_DISCURI (PK\_CASA\_DISCURI);

ALTER TABLE MELODIE ADD CONSTRAINT **FK\_ARTIST\_MELODIE** FOREIGN KEY(FK\_ARTIST) REFERENCES ARTIST (PK\_ARTIST);

ALTER TABLE MELODIE ADD CONSTRAINT **FK\_ALBUM\_MELODIE** FOREIGN KEY(FK\_ALBUM) REFERENCES ALBUM (PK\_ALBUM);

ALTER TABLE MELODIE\_GEN ADD CONSTRAINT **FK\_MELODIE\_MELODIE\_GEN** FOREIGN KEY(FK\_MELODIE) REFERENCES MELODIE (PK\_MELODIE);

ALTER TABLE MELODIE\_GEN ADD CONSTRAINT **FK\_GEN\_MELODIE\_GEN** FOREIGN KEY(FK\_GEN) REFERENCES GEN (PK\_GEN);

ALTER TABLE VANZARE ADD CONSTRAINT **FK\_ALBUM\_VANZARE** FOREIGN KEY(FK\_ALBUM) REFERENCES ALBUM (PK\_ALBUM);

ALTER TABLE VANZARE ADD CONSTRAINT **FK\_ANGAJAT\_VANZARE** FOREIGN KEY(FK\_ANGAJAT) REFERENCES ANGAJAT (PK\_ANGAJAT);

ALTER TABLE INCHIRIERE ADD CONSTRAINT **FK\_ALBUM\_INCHIRIERE** FOREIGN KEY(FK\_ALBUM) REFERENCES ALBUM (PK\_ALBUM);

ALTER TABLE INCHIRIERE ADD CONSTRAINT **FK\_ANGAJAT\_INCHIRIERE** FOREIGN KEY(FK\_ANGAJAT) REFERENCES ANGAJAT (PK\_ANGAJAT);

ALTER TABLE INCHIRIERE ADD CONSTRAINT **FK\_CLIENT\_INCHIRIERE** FOREIGN KEY(FK\_CLIENT) REFERENCES CLIENT (PK\_CLIENT);

*9. Realizarea normalizării până la forma normală 3 (FN1-FN3).*

|  |  |  |
| --- | --- | --- |
| **FORMA**  **NORMALA** | **STATUS REALIZARE** | **EXPLICATIE** |
| FN1 | REALIZATA DIN PROIECTARE | - toate tabelele sunt reduse la forma atomica indivizibila (entitate = tabel)  - toate tabelele prezinta atribute unice pentru identificare (chei primare, constrangeri de unicitate pe anumite campuri) |
| FN2 | REALIZATA DIN PROIECTARE | - fiecare atribut non-cheie este dependent de cheile primare (informatiile sunt asociate unic cu id-uri reprezentand cheile primare)  - fiecare tabel dependent de alt tabel formeaza o legatura unica prin campul de cheie straina (ex: tabelul MELODIE\_GEN formeaza atribute pereche FK\_MELODIE, FK\_GEN unice) |
| FN3 | REALIZATA DIN PROIECTARE | - toate campurile din tabele sunt independente (nu exista campuri calculate sau dependente de alte coloane)  - nu exista dependente tranzitive intre atributele non-cheie (datele introduse in fiecare tabel, cu exceptia cheilor straine, sunt independente de datele din alte tabele) |

*Exemplu NON-FN1:* In loc de cele doua tabele ALBUM (pk\_album) si MELODIE (pk\_melodie, fk\_album) -> vom avea un singur tabel ALBUME\_MELODII (pk\_album\_melodie, id\_album, id\_melodie…) cu obiecte de tip ALBUM non-atomice (id-uri de albume repetitive si entitati neseparate)

*Exemplu NON-FN2:* In loc de tabelele ALBUM (pk\_album), MELODIE (pk\_melodie, fk\_album, fk\_artitst) si ARTIST (pk\_artist) -> am avea doar ALBUM (pk\_album) si MELODII\_ARTISTI (pk\_melodie, fk\_album, nume\_artist, detalii\_artist …) fara identificarea unica a artistilor printr-un id (artistii pot fi asociati cu mai multe melodii, astfel facandu-se referinta de la melodie la un artist unic inregistrat, fara a se introduce detaliile artistului la fiecare melodie)

*Exemplu NON-FN3:* In loc de tabelele ALBUM (pk\_album,fk\_casa\_discuri), MELODIE (pk\_melodie, fk\_album, fk\_artitst), ARTIST (pk\_artist) si CASA\_DISCURI (pk\_casa\_discuri) -> am avea doar ALBUM (pk\_album, nume\_casa\_discuri …), MELODIE (pk\_melodie, fk\_album, fk\_artitst) si ARTIST (pk\_artist), astfel nefiind asociate casele de discuri unic inregistrate cu fiecare album (o casa de discuri poate produce mai multe albume din cele din magazin, deci nu trebuie introduce detalii de fiecare data)

*10. 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).*

INSERT INTO TARA(PK\_TARA,NUME) VALUES (sequence\_tara.nextval,'Romania');

INSERT INTO TARA(PK\_TARA,NUME) VALUES (sequence\_tara.nextval,'SUA');

INSERT INTO TARA(PK\_TARA,NUME) VALUES (sequence\_tara.nextval,'Marea Britanie');

INSERT INTO TARA(PK\_TARA,NUME) VALUES (sequence\_tara.nextval,'Franta');

INSERT INTO TARA(PK\_TARA,NUME) VALUES (sequence\_tara.nextval,'Germania');

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'Maria Tanase',1);

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'Guns N Roses',2);

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'Louis Armstrong',2);

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'Elvis Presley',2);

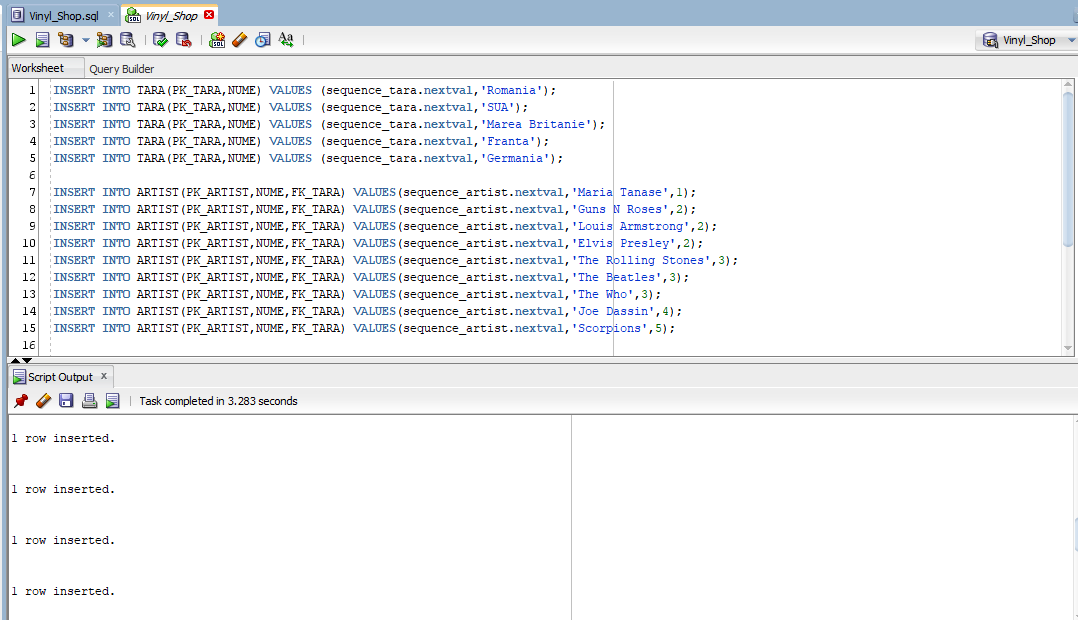
INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'The Rolling Stones',3);

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'The Beatles',3);

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'The Who',3);

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'Joe Dassin',4);

INSERT INTO ARTIST(PK\_ARTIST,NUME,FK\_TARA) VALUES(sequence\_artist.nextval,'Scorpions',5);



INSERT INTO GEN(PK\_GEN,NUME) VALUES (sequence\_gen.nextval,'Folclor');

INSERT INTO GEN(PK\_GEN,NUME) VALUES (sequence\_gen.nextval,'Rock');

INSERT INTO GEN(PK\_GEN,NUME) VALUES (sequence\_gen.nextval,'Jazz');

INSERT INTO GEN(PK\_GEN,NUME) VALUES (sequence\_gen.nextval,'Rock & Roll');

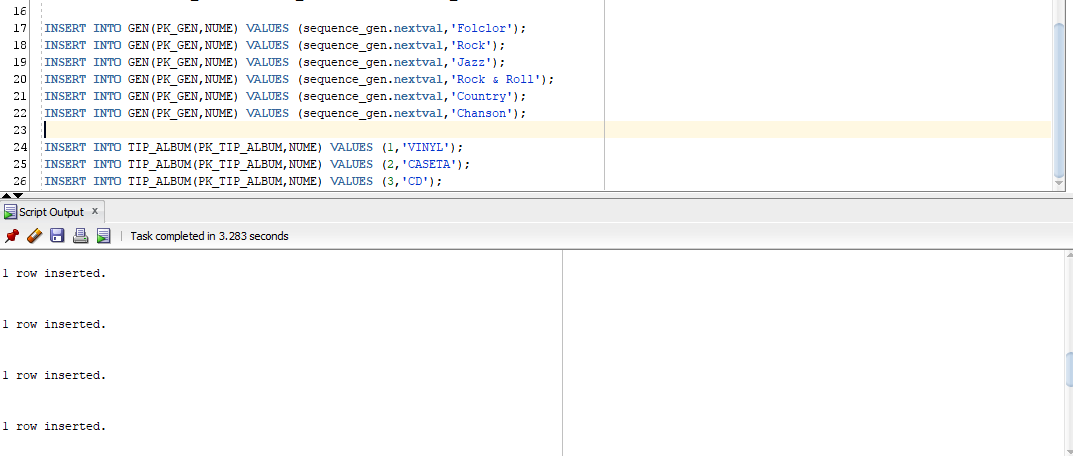
INSERT INTO GEN(PK\_GEN,NUME) VALUES (sequence\_gen.nextval,'Country');

INSERT INTO GEN(PK\_GEN,NUME) VALUES (sequence\_gen.nextval,'Chanson');

INSERT INTO TIP\_ALBUM(PK\_TIP\_ALBUM,NUME) VALUES (1,'VINYL');

INSERT INTO TIP\_ALBUM(PK\_TIP\_ALBUM,NUME) VALUES (2,'CASETA');

INSERT INTO TIP\_ALBUM(PK\_TIP\_ALBUM,NUME) VALUES (3,'CD');



INSERT INTO CASA\_DISCURI(PK\_CASA\_DISCURI,NUME) VALUES (sequence\_casa\_discuri.nextval,'Global Records');

INSERT INTO CASA\_DISCURI(PK\_CASA\_DISCURI,NUME) VALUES (sequence\_casa\_discuri.nextval,'Universal Records');

INSERT INTO CASA\_DISCURI(PK\_CASA\_DISCURI,NUME) VALUES (sequence\_casa\_discuri.nextval,'Electrocord');

INSERT INTO CASA\_DISCURI(PK\_CASA\_DISCURI,NUME) VALUES (sequence\_casa\_discuri.nextval,'Atlantic Records');

INSERT INTO CASA\_DISCURI(PK\_CASA\_DISCURI,NUME) VALUES (sequence\_casa\_discuri.nextval,'CBS');

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Ciuleandra',1962,340,20,1,1,3);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Greatest Hits',2001,190,12,1,3,3);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Appetite for Destruction',1987,250,11,2,3,2);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Hello, Dolly!',1964,235,15,3,3,2);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Moody Blue',1977,211,24,4,1,1);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'G.I. Blues',1960,253,22,4,1,1);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Aftermath',1966,290,19,5,1,2);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Abbey Road',1969,253,18,6,1,4);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Let it be',1970,243,33,6,2,1);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'My Generation',1965,244,31,7,1,4);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Who’s next',1971,267,22,7,1,1);

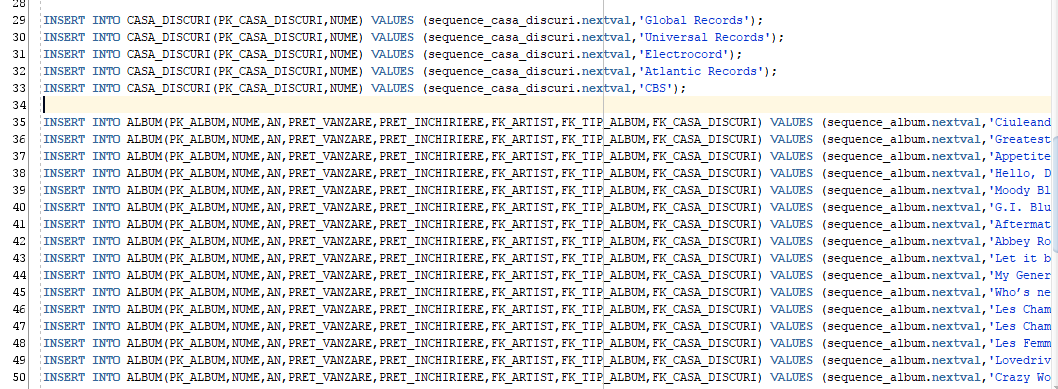
INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Les Champs-Élysées',1969,250,15,8,1,5);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Les Champs-Élysées',1969,250,15,8,3,5);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Les Femmes de ma vie',1970,240,15,8,1,5);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Lovedrive',1979,242,15,9,1,4);

INSERT INTO ALBUM(PK\_ALBUM,NUME,AN,PRET\_VANZARE,PRET\_INCHIRIERE,FK\_ARTIST,FK\_TIP\_ALBUM,FK\_CASA\_DISCURI) VALUES (sequence\_album.nextval,'Crazy World',1990,310,15,9,1,2);



INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Pe vale, tato, pe vale',180,1,1);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Trenule, masina mica',182,1,2);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Welcome to the Jungle',190,2,3);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Sweet Child O Mine',198,2,3);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Paradise City',244,2,3);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'What a Wonderful World',170,3,NULL);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Black and Blue',189,3,4);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Let Me Be There',214,4,5);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Blue Suede Shoes',194,4,6);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Doncha Bother Me',170,5,7);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Lady Jane',172,5,7);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Under My Thumb',180,5,7);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Something',150,6,8);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Two of us',230,6,9);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Across the Universe',280,6,9);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Out in the street',350,7,10);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Won’t get fooled again',250,7,11);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Baba O’Riley',210,7,11);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Les Champs-Élysées',180,8,12);

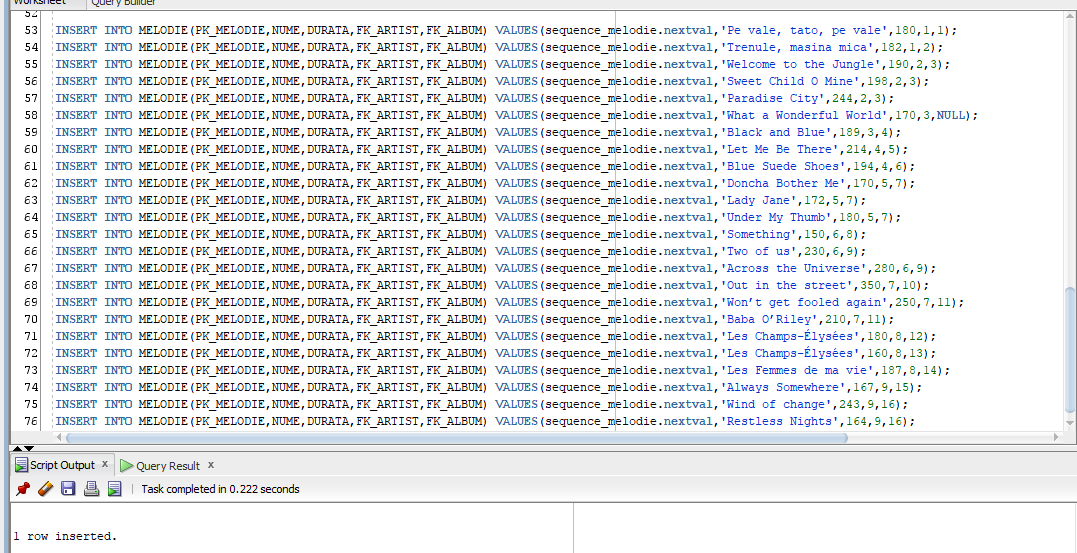
INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Les Champs-Élysées',160,8,13);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Les Femmes de ma vie',187,8,14);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Always Somewhere',167,9,15);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Wind of change',243,9,16);

INSERT INTO MELODIE(PK\_MELODIE,NUME,DURATA,FK\_ARTIST,FK\_ALBUM) VALUES(sequence\_melodie.nextval,'Restless Nights',164,9,16);



INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,1,1);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,2,1);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,3,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,4,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,5,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,6,3);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,7,3);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,8,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,8,5);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,9,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,9,4);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,10,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,10,5);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,11,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,12,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,13,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,14,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,15,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,16,1);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,17,1);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,18,1);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,19,6);

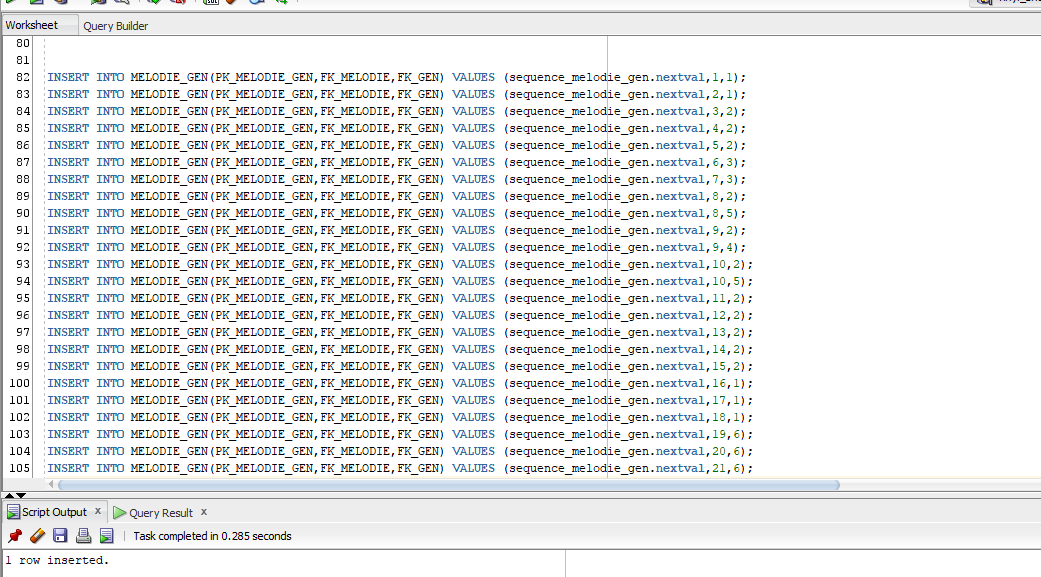
INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,20,6);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,21,6);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,22,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,23,2);

INSERT INTO MELODIE\_GEN(PK\_MELODIE\_GEN,FK\_MELODIE,FK\_GEN) VALUES (sequence\_melodie\_gen.nextval,24,2);



INSERT INTO CLIENT(PK\_CLIENT,NUME,PRENUME,CNP,MAIL,TELEFON) VALUES (sequence\_client.nextval,'Popescu','Vasile','1910123456789',NULL,'0712345678');

INSERT INTO CLIENT(PK\_CLIENT,NUME,PRENUME,CNP,MAIL,TELEFON) VALUES (sequence\_client.nextval,'Ionescu','Dumitru','1920304567890','ionescu.dumitru@gmail.com','0712739678');

INSERT INTO CLIENT(PK\_CLIENT,NUME,PRENUME,CNP,MAIL,TELEFON) VALUES (sequence\_client.nextval,'Georgescu','Ion','1870516787889','georgescu.ion@yahoo.com','0712325670');

INSERT INTO CLIENT(PK\_CLIENT,NUME,PRENUME,CNP,MAIL,TELEFON) VALUES (sequence\_client.nextval,'Enescu','Emilia','2900304567813',NULL,'0743535378');

INSERT INTO CLIENT(PK\_CLIENT,NUME,PRENUME,CNP,MAIL,TELEFON) VALUES (sequence\_client.nextval,'Gavrilescu','Ileana','2851011345678',NULL,'0713355668');

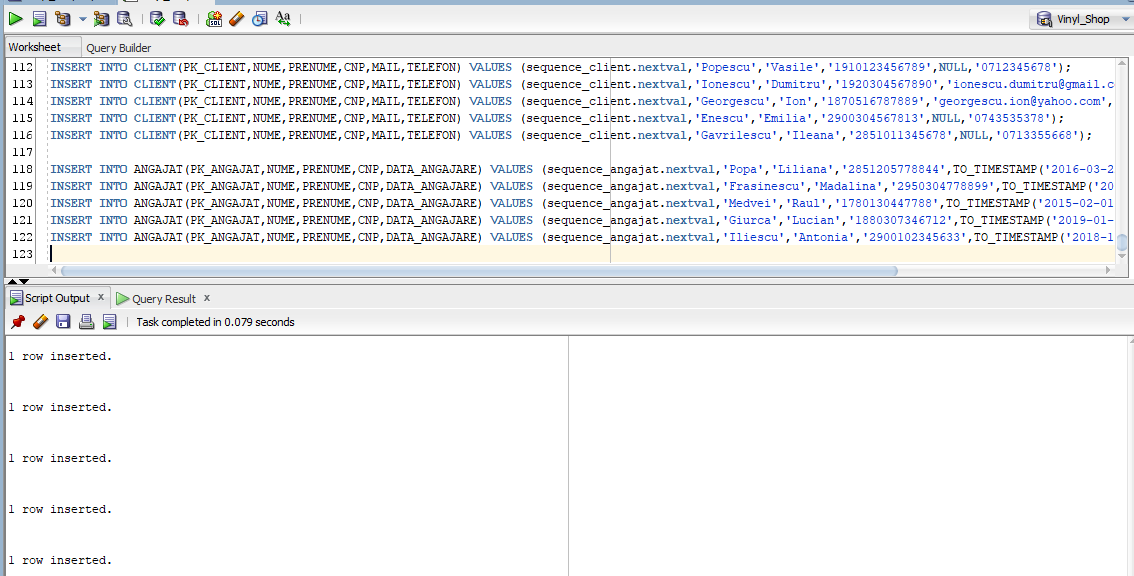
INSERT INTO ANGAJAT(PK\_ANGAJAT,NUME,PRENUME,CNP,DATA\_ANGAJARE) VALUES (sequence\_angajat.nextval,'Popa','Liliana','2851205778844',TO\_TIMESTAMP('2016-03-22','YYYY-MM-DD HH24:MI:SS'));

INSERT INTO ANGAJAT(PK\_ANGAJAT,NUME,PRENUME,CNP,DATA\_ANGAJARE) VALUES (sequence\_angajat.nextval,'Frasinescu','Madalina','2950304778899',TO\_TIMESTAMP('2018-09-09','YYYY-MM-DD HH24:MI:SS'));

INSERT INTO ANGAJAT(PK\_ANGAJAT,NUME,PRENUME,CNP,DATA\_ANGAJARE) VALUES (sequence\_angajat.nextval,'Medvei','Raul','1780130447788',TO\_TIMESTAMP('2015-02-01','YYYY-MM-DD HH24:MI:SS'));

INSERT INTO ANGAJAT(PK\_ANGAJAT,NUME,PRENUME,CNP,DATA\_ANGAJARE) VALUES (sequence\_angajat.nextval,'Giurca','Lucian','1880307346712',TO\_TIMESTAMP('2019-01-01','YYYY-MM-DD HH24:MI:SS'));

INSERT INTO ANGAJAT(PK\_ANGAJAT,NUME,PRENUME,CNP,DATA\_ANGAJARE) VALUES (sequence\_angajat.nextval,'Iliescu','Antonia','2900102345633',TO\_TIMESTAMP('2018-10-11','YYYY-MM-DD HH24:MI:SS'));



INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-05-11','YYYY-MM-DD HH24:MI:SS'),2,1);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-06-12','YYYY-MM-DD HH24:MI:SS'),3,2);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-04-18','YYYY-MM-DD HH24:MI:SS'),12,5);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-04-19','YYYY-MM-DD HH24:MI:SS'),10,4);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-05-10','YYYY-MM-DD HH24:MI:SS'),1,2);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-05-25','YYYY-MM-DD HH24:MI:SS'),11,2);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-06-04','YYYY-MM-DD HH24:MI:SS'),8,1);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-06-08','YYYY-MM-DD HH24:MI:SS'),9,3);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-06-23','YYYY-MM-DD HH24:MI:SS'),12,3);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-07-12','YYYY-MM-DD HH24:MI:SS'),2,2);

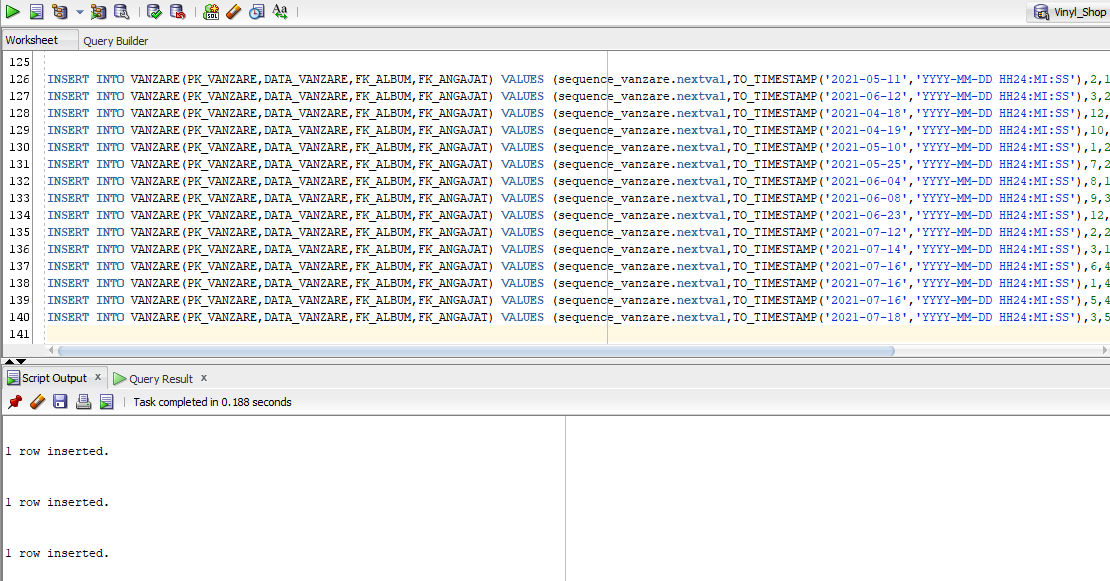
INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-07-14','YYYY-MM-DD HH24:MI:SS'),3,1);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-07-16','YYYY-MM-DD HH24:MI:SS'),14,4);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-07-16','YYYY-MM-DD HH24:MI:SS'),1,4);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-07-16','YYYY-MM-DD HH24:MI:SS'),5,4);

INSERT INTO VANZARE(PK\_VANZARE,DATA\_VANZARE,FK\_ALBUM,FK\_ANGAJAT) VALUES (sequence\_vanzare.nextval,TO\_TIMESTAMP('2021-07-18','YYYY-MM-DD HH24:MI:SS'),3,5);



INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-07-18','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-08-18','YYYY-MM-DD HH24:MI:SS'),3,5,1);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-06-12','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-07-12','YYYY-MM-DD HH24:MI:SS'),3,2,3);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-04-18','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-05-18','YYYY-MM-DD HH24:MI:SS'),12,5,4);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-04-19','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-05-18','YYYY-MM-DD HH24:MI:SS'),10,4,2);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-05-10','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-06-09','YYYY-MM-DD HH24:MI:SS'),1,2,2);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-05-25','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-06-25','YYYY-MM-DD HH24:MI:SS'),7,2,2);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-06-04','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-07-12','YYYY-MM-DD HH24:MI:SS'),8,1,2);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-06-08','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-07-12','YYYY-MM-DD HH24:MI:SS'),9,3,1);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-06-23','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-07-22','YYYY-MM-DD HH24:MI:SS'),12,3,4);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-07-12','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-08-12','YYYY-MM-DD HH24:MI:SS'),2,2,4);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-07-14','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-08-12','YYYY-MM-DD HH24:MI:SS'),3,1,2);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-07-16','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-08-12','YYYY-MM-DD HH24:MI:SS'),6,4,1);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-07-16','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-08-12','YYYY-MM-DD HH24:MI:SS'),1,4,1);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-07-16','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-08-12','YYYY-MM-DD HH24:MI:SS'),5,4,5);

INSERT INTO INCHIRIERE(PK\_INCHIRIERE,DATA\_INCHIRIERE,DATE\_RETUR,FK\_ALBUM,FK\_ANGAJAT,FK\_CLIENT) VALUES (sequence\_inchiriere.nextval,TO\_TIMESTAMP('2021-07-18','YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2021-08-12','YYYY-MM-DD HH24:MI:SS'),3,5,5);



*11. Formulați în limbaj natural și implementați 5 cereri SQL complexe ce vor utiliza, în ansamblul lor, următoarele elemente:*

*• operație join pe cel puțin 4 tabele*

*• filtrare la nivel de linii*

*• subcereri sincronizate în care intervin cel puțin 3 tabele*

*• subcereri nesincronizate în care intervin cel puțin 3 tabele*

*• grupări de date, funcții grup, filtrare la nivel de grupuri*

*• ordonări*

*• utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a funcțiilor NVL și DECODE, a cel puțin unei expresii CASE*

*• utilizarea a cel puțin 1 bloc de cerere (clauza WITH)*

- Angajatii si numarul de albume inchiriate, pe zile, ordonate descrescator dupa numarul de albume si crescator dupa data

select

UPPER(CONCAT(ang.nume,CONCAT(' ',ang.prenume))) as "angajat",

TRUNC(i.data\_inchiriere) as "data\_inchiriere",

COUNT(DISTINCT alb.pk\_album)

from angajat ang

inner join inchiriere i on ang.pk\_angajat = i.fk\_angajat

inner join album alb on i.fk\_album = alb.pk\_album

group by UPPER(CONCAT(ang.nume,CONCAT(' ',ang.prenume))), TRUNC(i.data\_inchiriere)

order by 3 desc, 2 ASC

**

- Albumele inchiriate (impreuna cu artistul albumului si angajatul care a inchiriat albumul), returnate sau in stadiu de inchiriate, care s-au si vandut (se afiseaza data vanzarii) sau nevandute (se afiseaza FARA VANAZARE)

select

alb.nume as "album",

art.nume as "artist",

vanz."nume\_angajat",

DECODE(i.date\_retur,NULL,'RETURNAT','INCHIRIAT') as "status",

NVL(TO\_CHAR(vanz.data\_vanzare, 'dd-MON-yy'),'FARA VANZARE') as "data\_vanzare"

from album alb

inner join artist art on alb.fk\_artist = art.pk\_artist

inner join inchiriere i on alb.pk\_album = i.fk\_album

left join (

select

v.pk\_vanzare,

a.pk\_album,

v.data\_vanzare,

ang.nume as "nume\_angajat"

from vanzare v

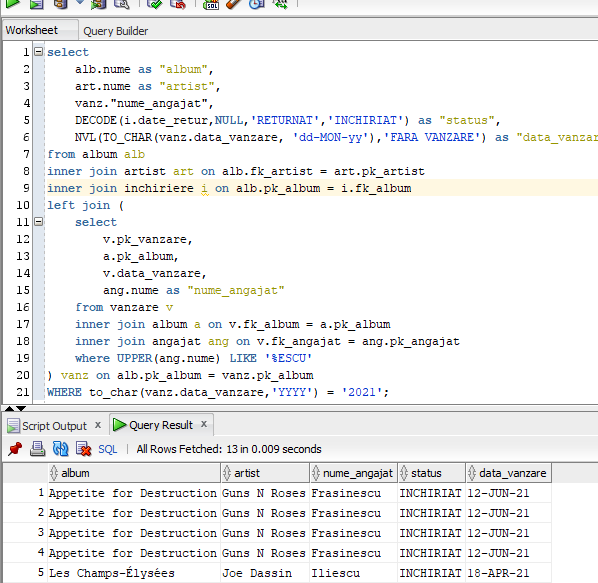
inner join album a on v.fk\_album = a.pk\_album

inner join angajat ang on v.fk\_angajat = ang.pk\_angajat

where UPPER(ang.nume) LIKE '%ESCU'

) vanz on alb.pk\_album = vanz.pk\_album

WHERE to\_char(vanz.data\_vanzare,'YYYY') = '2021';



*-* Albumele artistilor din SUA si pretul maxim al unui album cu acelasi artist

WITH MaxPretAlbumPeArtist

AS (

SELECT

art.pk\_artist,

t.nume as "nume\_tara",

art.nume as "nume\_artist",

MAX(alb.pret\_vanzare) as "Maxim"

FROM album alb

INNER JOIN artist art on alb.fk\_artist = art.pk\_artist

INNER JOIN tara t on art.fk\_tara = t.pk\_tara

GROUP BY art.pk\_artist, t.nume, art.nume

)

SELECT DISTINCT

alb.nume as "album",

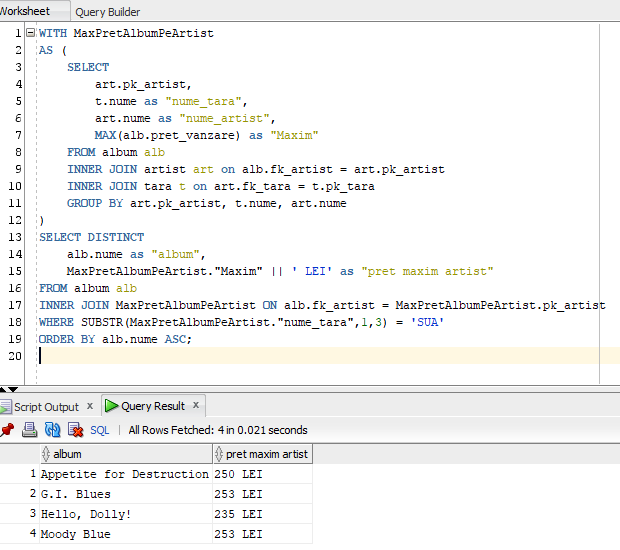
MaxPretAlbumPeArtist."Maxim" || ' LEI' as "pret maxim artist"

FROM album alb

INNER JOIN MaxPretAlbumPeArtist ON alb.fk\_artist = MaxPretAlbumPeArtist.pk\_artist

WHERE SUBSTR(MaxPretAlbumPeArtist."nume\_tara",1,3) = 'SUA'

ORDER BY alb.nume ASC;

**

*-* Daca exista albume mai scumpe de 250 lei si mai noi de anul 1985, se afiseaza albumul, daca nu, toate albumele tip vinyl

IF ( 250 < ANY ( SELECT alb.pret\_vanzare

FROM album alb

WHERE alb.an >= 1985 ) )

THEN

SELECT alb.nume as "album", alb.an, art.nume as "artist", ta.nume "tip\_album", alb.pret\_vanzare

FROM album alb

INNER JOIN artist art on alb.fk\_artist = art.pk\_artist

INNER JOIN tip\_album ta on alb.fk\_tip\_album = ta.pk\_tip\_album

WHERE alb.an >= 1985 AND alb.pret\_vanzare > 250 ;

ELSE

SELECT alb.nume as "album", alb.an, art.nume as "artist", ta.nume "tip\_album", alb.pret\_vanzare

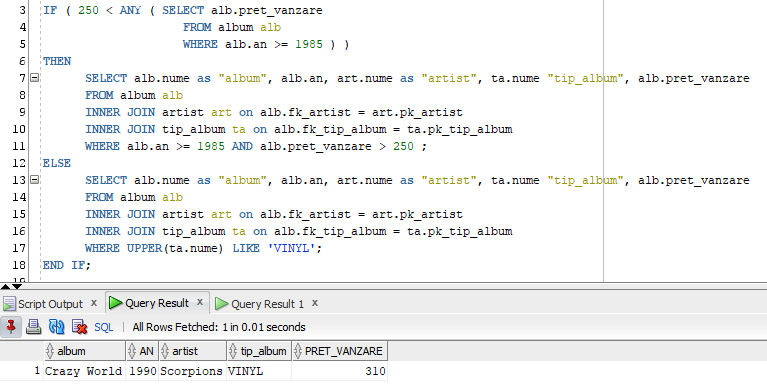
FROM album alb

INNER JOIN artist art on alb.fk\_artist = art.pk\_artist

INNER JOIN tip\_album ta on alb.fk\_tip\_album = ta.pk\_tip\_album

WHERE UPPER(ta.nume) LIKE 'VINYL';

END IF;



- Artistii (si originea acestora: SUA, Europa) si numarul de albume vandute pentru cei care au mai mult de 1 album vandut

select

a.nume as "artist",

a."origine",

count(v.fk\_album) as "nr.albume vandute"

from (

select

art.pk\_artist,

art.nume,

case to\_char(t.nume)

when 'SUA' then to\_char('SUA')

else to\_char('Europa')

end as "origine"

from artist art

inner join tara t on art.fk\_tara = t.pk\_tara

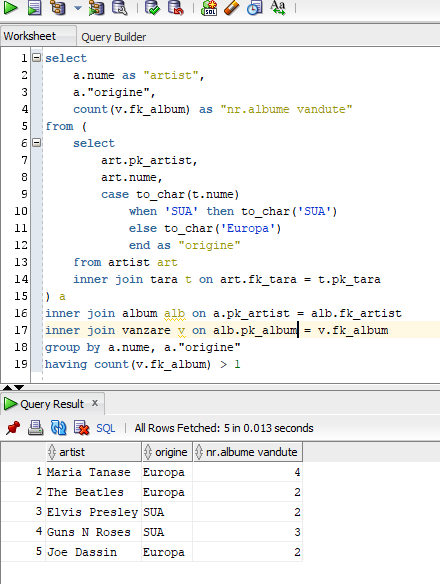
) a

inner join album alb on a.pk\_artist = alb.fk\_artist

inner join vanzare v on alb.pk\_album = v.fk\_album

group by a.nume, a."origine"

having count(v.fk\_album) > 1

**

*12. Implementarea a 3 operații de actualizare sau suprimare a datelor utilizând subcereri.*

* Actualizare stocuri albume: Adaugare cate 2 bucati din fiecare album care s-a vandut de mai mult de o data

update album

set stoc = stoc + 2

where pk\_album in (

select

a.pk\_album

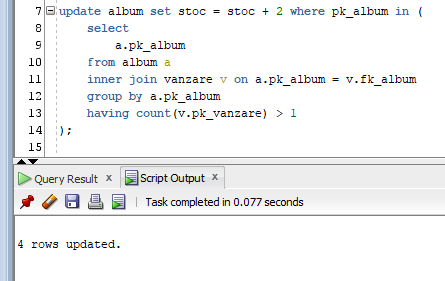
from album a

inner join vanzare v on a.pk\_album = v.fk\_album

group by a.pk\_album

having count(v.pk\_vanzare) > 1

);

**

* Actualizare status existent card de fidelitate, pentru angajatii care au cel putin 3 inchirieri de albume: se va actualiza valoarea in 1 (true).

update client c

set are\_card\_fidelitate = 1

where c.pk\_client in (

select distinct

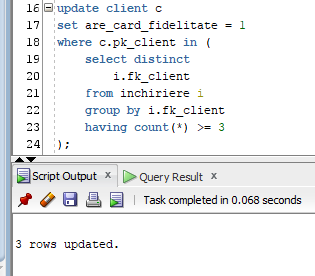
i.fk\_client

from inchiriere i

group by i.fk\_client

having count(\*) >= 3

);



* Stergere melodii care nu se afla pe niciun album ( fk\_album IS NULL ) sau de pe albume care nu se mai afla in stoc ( stoc = 0 ): stergere asociere melodie\_gen + stergere melodii

delete

from melodie\_gen

where fk\_melodie in

(

select m.pk\_melodie

from melodie m

left join album alb on m.fk\_album = alb.pk\_album

where m.fk\_album is null

or alb.stoc = 0

);

delete

from melodie

where pk\_melodie in

(

select m.pk\_melodie

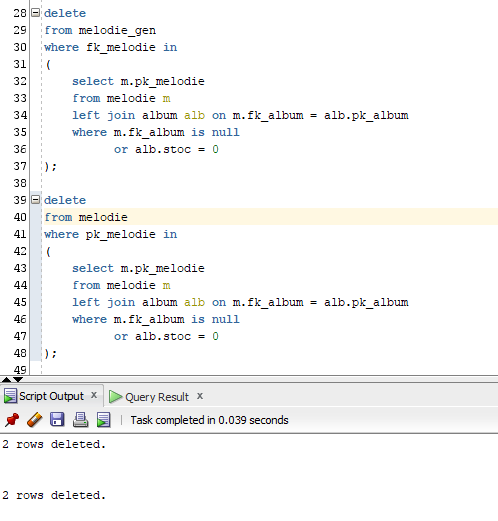
from melodie m

left join album alb on m.fk\_album = alb.pk\_album

where m.fk\_album is null

or alb.stoc = 0

);



*13. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 10).*

CREATE SEQUENCE sequence\_artist

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_tara

INCREMENT BY 1

START WITH 1

MAXVALUE 200

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_gen

INCREMENT BY 1

START WITH 1

MAXVALUE 1000

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_melodie\_gen

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_melodie

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_album

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_casa\_discuri

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_angajat

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_client

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_vanzare

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

CREATE SEQUENCE sequence\_inchiriere

INCREMENT BY 1

START WITH 1

NOMAXVALUE

MINVALUE 1

NOCYCLE

NOCACHE

ORDER;

*16. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele și două cereri ce utilizează operația division.*

- Extragerea tuturor albumelor (cu detalii despre casa de discuri si tipurile de album), a clientilor si a inchirierilor (fie ca albumele au fost inchiriate sau nu)

select

alb.nume,

alb.an,

cd.nume as "casa\_discuri",

ta.nume as "tip\_album",

CONCAT(alb.pret\_inchiriere , ' lei'),

i.data\_inchiriere,

i.date\_retur,

i.discount,

i.penalitati,

CONCAT(c.nume , c.prenume) as "client",

c.are\_card\_fidelitate

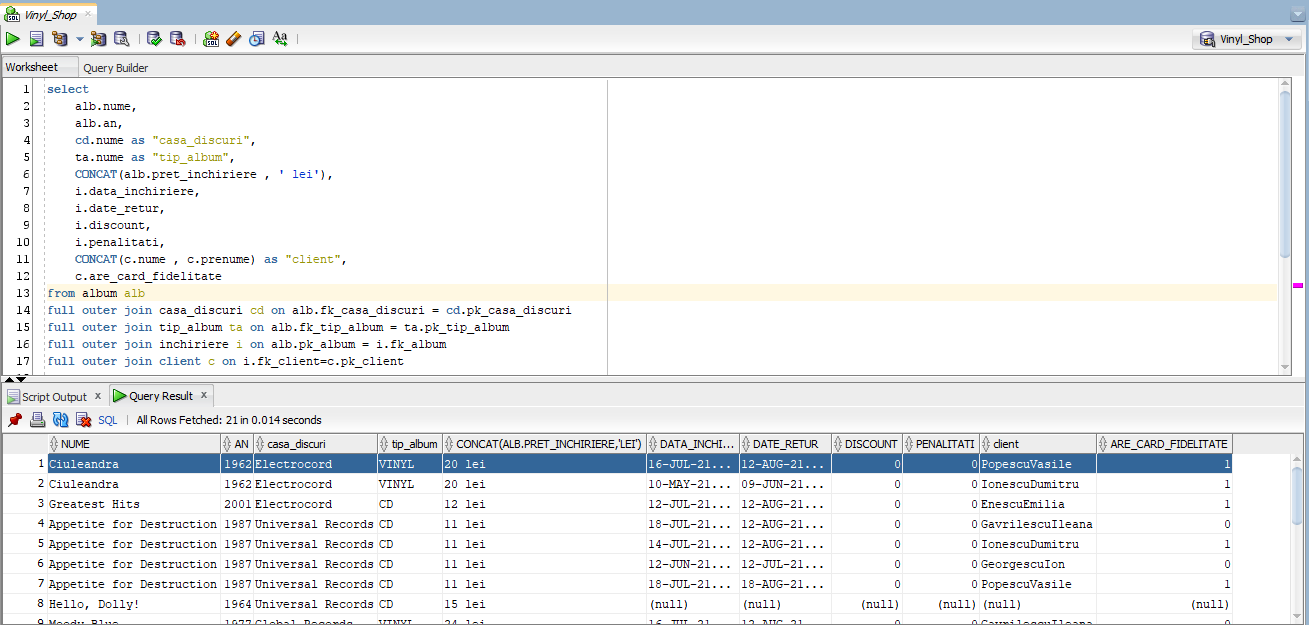
from album alb

full outer join casa\_discuri cd on alb.fk\_casa\_discuri = cd.pk\_casa\_discuri

full outer join tip\_album ta on alb.fk\_tip\_album = ta.pk\_tip\_album

full outer join inchiriere i on alb.pk\_album = i.fk\_album

full outer join client c on i.fk\_client=c.pk\_client



- Extragerea detaliilor despre toate albumele inchiriate, mai putin cele care s-au si vandut intre timp

select

alb.nume,

alb.an,

ta.nume as "tip\_album",

i.data\_inchiriere as "data"

from album alb

inner join tip\_album ta on alb.fk\_tip\_album = ta.pk\_tip\_album

inner join inchiriere i on alb.pk\_album = i.fk\_album

MINUS

select

alb.nume,

alb.an,

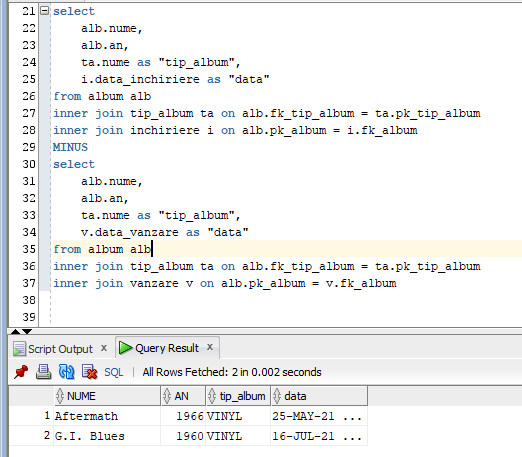
ta.nume as "tip\_album",

v.data\_vanzare as "data"

from album alb

inner join tip\_album ta on alb.fk\_tip\_album = ta.pk\_tip\_album

inner join vanzare v on alb.pk\_album = v.fk\_album



- Extragerea detaliilor despre clientii care au inchiriat albume, mai putin cei care au inchiriat acelasi album de mai mult de doua ori

select

c.nume,

c.prenume,

c.telefon,

LISTAGG(i.data\_inchiriere,',') WITHIN GROUP ( ORDER BY c.nume, c.prenume )

from client c

inner join inchiriere i on c.pk\_client = i.fk\_client

group by c.nume, c.prenume, c.telefon

MINUS

select

c.nume,

c.prenume,

c.telefon,

LISTAGG(i.data\_inchiriere,',') WITHIN GROUP ( ORDER BY c.nume, c.prenume )

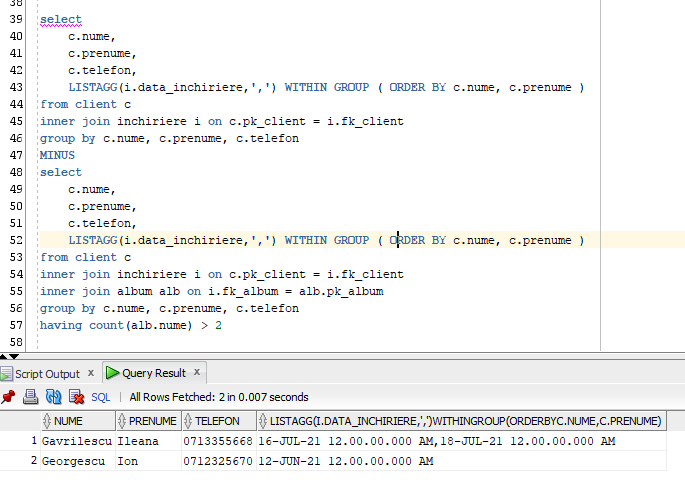
from client c

inner join inchiriere i on c.pk\_client = i.fk\_client

inner join album alb on i.fk\_album = alb.pk\_album

group by c.nume, c.prenume, c.telefon

having count(alb.nume) > 2

**

*17. 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.*

*EX*: ALBUMELE VANDUTE ANUL ACESTA, CU DISCOUNT, CARE AU FOST SI INCHIRIATE ANTERIOR

*INAINTE DE OPTIMIZARE:*

SELECT

alb.pk\_album,

v.pk\_vanzare,

i.pk\_inchiriere

FROM album alb

INNER JOIN vanzare v

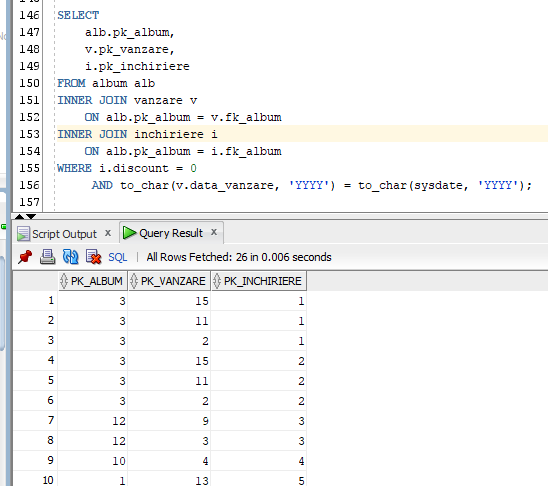
ON alb.pk\_album = v.fk\_album

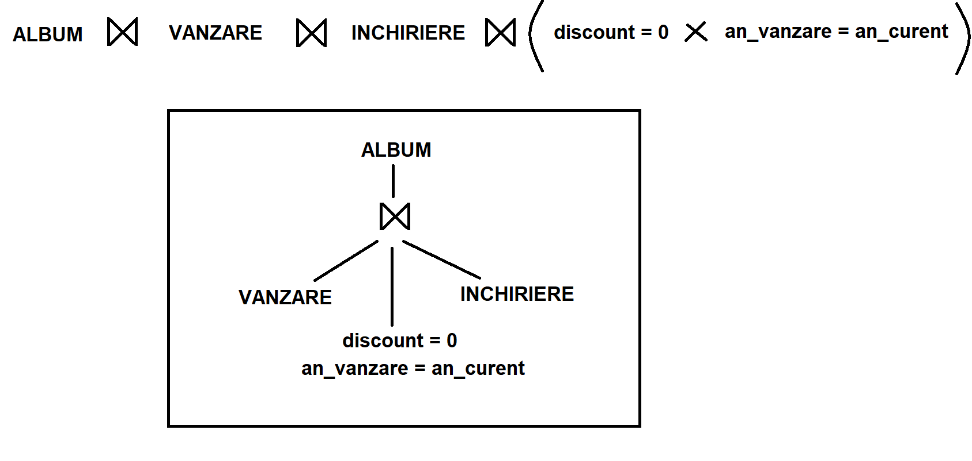
INNER JOIN inchiriere i

ON alb.pk\_album = i.fk\_album

WHERE i.discount = 0

AND to\_char(v.data\_vanzare, 'YYYY') = to\_char(sysdate, 'YYYY');

**

**

*DUPA OPTIMIZARE:*

SELECT

alb.pk\_album,

v.pk\_vanzare,

i.pk\_inchiriere

FROM album alb

INNER JOIN vanzare v

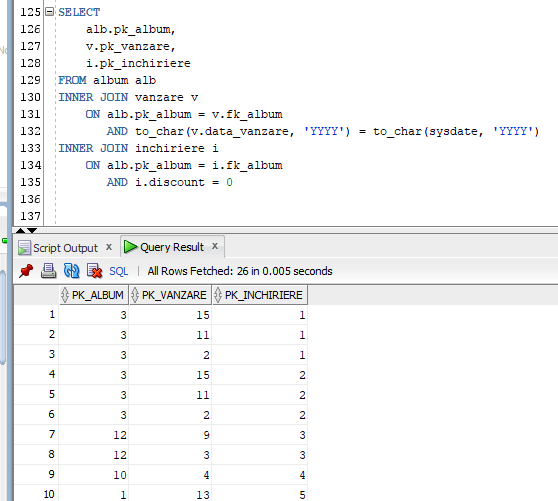
ON alb.pk\_album = v.fk\_album

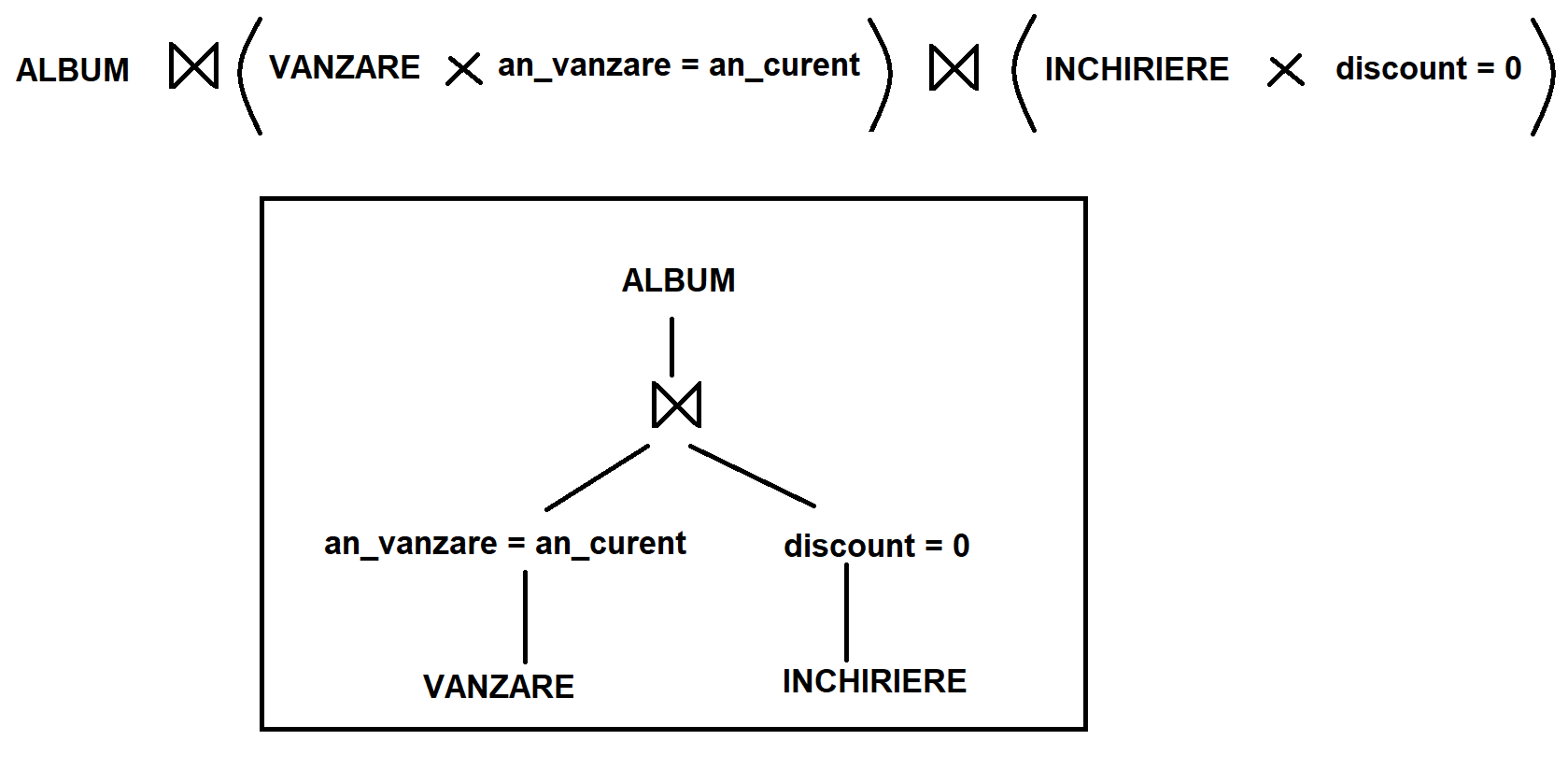
AND to\_char(v.data\_vanzare, 'YYYY') = to\_char(sysdate, 'YYYY')

INNER JOIN inchiriere i

ON alb.pk\_album = i.fk\_album

AND i.discount = 0

**

**

*18.*

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

|  |  |  |
| --- | --- | --- |
| **FORMA**  **NORMALA** | **STATUS REALIZARE** | **EXPLICATIE** |
| BCNF | REALIZATA DIN PROIECTARE | - toate cheile primare sunt super-chei pentru tabelul lor (fiecare tabel are o cheie primare numerica, unic stabilita de valoarea secventei specifice)  - nu exista perechi de chei straine redundante, care sa formeze identificatori unici (ex: in tabelul INCHIRIERE exista identificator unic pentru fiecare inregistrare, cheia primare fiind o valoare oarecare incrementala, nu perechea FK\_CLIENT – FK\_ALBUM) |
| FN4 | REALIZATA DIN PROIECTARE | - campurile dependente de alte campuri sunt stocate independent in tabele asupra caror inregistrari exista constrangere de unicitate (ex: fiecare album are asociate mai multe melodii, dar fiecare album are de asemenea mai multe inchirieri si vanzari -> entiatile sunt stocate separat, independent) |
| FN5 | REALIZATA PENTRU CAZURI PARTICULARE  (**o MELODIE are un singur ARTIST asociat**) | - presupune eliminarea oricarei dependente de asociere (uniune) care nu este implicata de o cheie (ex: asocierea unica dintre ARTIST, ALBUM si MELODIE nu implica o cheie unica pentru ARTISTUL titular al ALBUMULUI, ci doar pentru ARTISTII titulari ai unei MELODII de pe un eventual ALBUM) |

*Exemplu NON-BCNF:* In loc de cele 3 tabele ALBUM (pk\_album), CLIENT (pk\_client) si INCHIRIERE (pk\_inchiriere,fk\_album,fk\_client) -> vom avea tabelele CLIENT (pk\_client) si INCHIRIERE\_ALBUME (fk\_client, id\_album, detalii\_inchiriere…) prin care pot aparea redundante si ambiguitati cand, spre exemplu, se inchiriaza acelasi album, cu aceleasi detalii de inchiriere (data,detalii\_album) de catre 2 persoane diferite: inchirierile nu pot fi catalogate ca fiind unice in acest caz.

*Exemplu NON-FN4:* In loc de tabelele ALBUM (pk\_album), MELODIE (pk\_melodie, fk\_album) si VANZARE (pk\_vanzare,fk\_album) -> am avea ALBUM (pk\_album) si MELODII\_VANDUTE (fk\_album, fk\_melodie, fk\_vanzare) pentru care nu exista o asociere logica (albumele se vand ca intreg de melodii, respective vanzarea este asociata cu un album nu cu o singura melodie)

*Exemplu NON-FN5:* In loc de tabelele ALBUM (pk\_album,fk\_artist), MELODIE (pk\_melodie, fk\_album, fk\_artitst), ARTIST (pk\_artist) -> vom avea MELODIE (pk\_melodie), ARTIST (pk\_artist), ALBUM (pk\_album) si MELODII\_ALBUME\_ARTISTI (fk\_artist, fk\_album, fk\_melodie) unde fiecare artist ar fi asociat atat cu albumul, cat si cu melodia, dar exista cazul in care o melodie de pe un album apartine altui artist decat cel ce este titularul albumului (ex: albume compilatii de artisti, mai multi interpreti la o singura melodie): in acest caz apare ambiguitatea asocierii artistului cu melodia careia ii este titular (artist principal sau secundar), dar si cu albumul caruia nu ii este titular (participa doar la o melodie, nu la tot albumul).

Rezolvarea FN5 se va realiza prin:

* eliminarea campului fk\_artist din tabelul MELODIE la crearea acestuia

CREATE TABLE **MELODIE** (

**PK\_MELODIE** NUMBER(18,0) NOT NULL,

NUME NVARCHAR2(50) NOT NULL,

DURATA NUMBER NOT NULL,

-- ***~~FK\_ARTIST~~*** *~~NUMBER(18,0) NOT NULL,~~*

**FK\_ALBUM** NUMBER(18,0)

DEFAULT NULL NULL,

CONSTRAINT PK\_MELODIE PRIMARY KEY ( PK\_MELODIE ),

CONSTRAINT CHECK\_DURATA\_MELODIE CHECK (DURATA > 0)

);

ALTER TABLE MELODIE ADD CONSTRAINT FK\_ALBUM\_MELODIE FOREIGN KEY(FK\_ALBUM) REFERENCES ALBUM (PK\_ALBUM);

* crearea tabelului MELODIE\_ARTIST (unde se vor asocia unul sau mai multi artisti, unic, cu o singura melodie, fara a afecta informatia despre titularul albumului pe care se afla eventual respectiva melodie, informative stocata in campul **fk\_artist** din tabela **ALBUM**)

CREATE TABLE **MELODIE\_ARTIST** (

**PK\_MELODIE\_ARTIST** NUMBER(18,0) NOT NULL,

**FK\_MELODIE** NUMBER(18,0) NOT NULL,

**FK\_ARTIST** NUMBER(18,0) NOT NULL,

CONSTRAINT PK\_MELODIE\_ARTIST PRIMARY KEY (PK\_MELODIE\_ARTIST)

);

ALTER TABLE **MELODIE\_ARTIST** ADD CONSTRAINT UQ\_MELODIE\_ARTIST **UNIQUE** (**FK\_MELODIE,FK\_ARTIST**);

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

*IMPORTANTA*:

Denormalizarea datelor permite vizualizarea sumarizata si usor deductibila a infomatiilor din aplicatie.

Datorita stocarii cu respectare a tuturor formelor normale, datele din fiecare tabel, separate, nu oferta raspunsul la interogarile uzuale pentru functionarea normala a unei aplicatii de acest fel.

*APLICAREA DENORMALIZARII*:

Se vor crea legaturile (JOIN) intre toate tabelel de interes pentru afisarea tuturor informatiilor necesare.

Se va folosi FULL OUTER JOIN / LEFT JOIN / RIGHT JOIN pentru returnarea datelor chiar si in cazul cand nu exista asocieri in toate tabelele mentionate, respctiv INNER JOIN pentru obigativitatea returnarii de informatii cu intrari in toate tabelel mentionate.

Se vor grupa informatiile dupa anumite campuri, pentru vizualizarea rapida a informatiilor asociate cu un anumit camp de interes.

*EXEMPLE:*

- ASOCIEREA TUTUROR DATELOR DIN BAZA DE DATE

select \*

from album alb

full outer join casa\_discuri cd on alb.fk\_casa\_discuri = cd.pk\_casa\_discuri

full outer join tip\_album ta on alb.fk\_tip\_album = ta.pk\_tip\_album

full outer join inchiriere i on alb.pk\_album = i.fk\_album

full outer join client c on i.fk\_client=c.pk\_client

full outer join vanzare v on alb.pk\_album = v.fk\_album

full outer join melodie m on alb.pk\_album = m.fk\_album

full outer join artist artm on m.fk\_artist = artm.pk\_artist

full outer join artist arta on alb.fk\_artist = arta.pk\_artist

full outer join tara tm on artm.fk\_tara = tm.pk\_tara

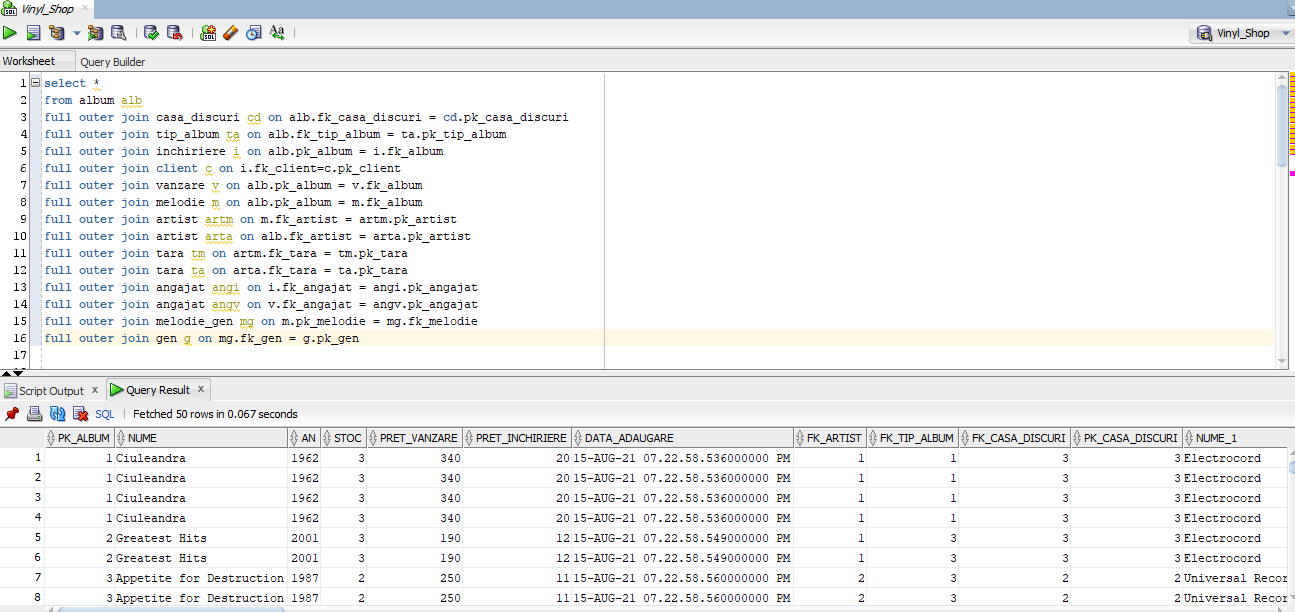
full outer join tara ta on arta.fk\_tara = ta.pk\_tara

full outer join angajat angi on i.fk\_angajat = angi.pk\_angajat

full outer join angajat angv on v.fk\_angajat = angv.pk\_angajat

full outer join melodie\_gen mg on m.pk\_melodie = mg.fk\_melodie

full outer join gen g on mg.fk\_gen = g.pk\_gen



- AFISAREA CLIENTILOR SI A TITLURILOR ALBUMELOR INCHIRIATE, CAT SI A DATELOR CAND AU INCHIRIAT

select

c.nume,

c.prenume,

c.telefon,

LISTAGG(alb.nume,' , ') WITHIN GROUP ( ORDER BY c.nume, c.prenume ),

LISTAGG(TRUNC(i.data\_inchiriere),' , ') WITHIN GROUP ( ORDER BY c.nume, c.prenume )

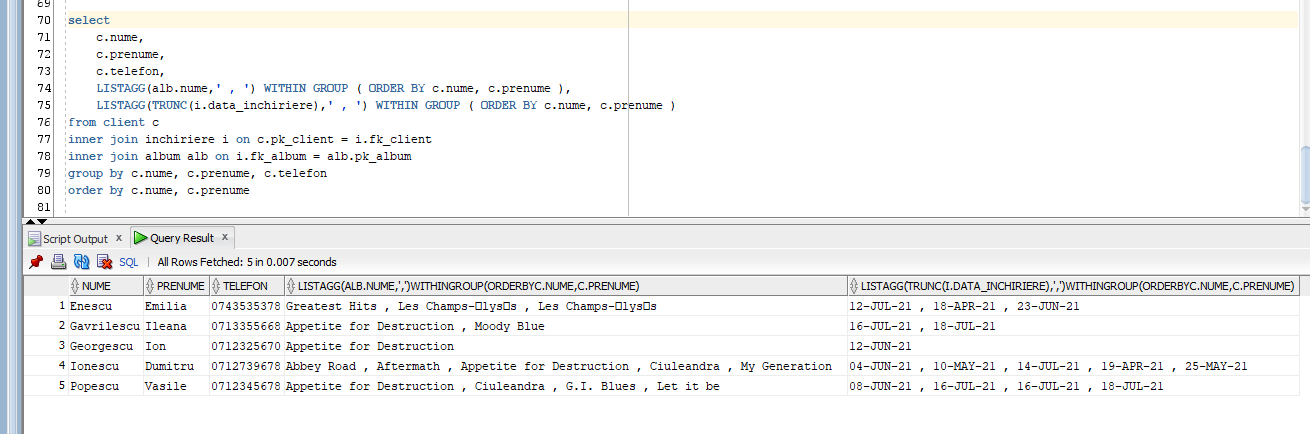
from client c

inner join inchiriere i on c.pk\_client = i.fk\_client

inner join album alb on i.fk\_album = alb.pk\_album

group by c.nume, c.prenume, c.telefon

order by c.nume, c.prenume



- AFISAREA TUTUROR ALBUMELOR SI A MELODIILOR COMPONENTE

select

art.nume,

alb.nume,

alb.an,

cd.nume,

ta.nume,

LISTAGG( CONCAT(CONCAT(m.nume,'/'),

CONCAT(m.durata,' sec.')),

' , ')

WITHIN GROUP ( ORDER BY art.nume )

from album alb

inner join artist art on alb.fk\_artist = art.pk\_artist

inner join casa\_discuri cd on alb.fk\_casa\_discuri = cd.pk\_casa\_discuri

inner join tip\_album ta on alb.fk\_tip\_album = ta.pk\_tip\_album

inner join melodie m on alb.pk\_album = m.fk\_album

group by art.nume,alb.nume,alb.an,cd.nume,ta.nume

order by art.nume asc

