

Gestiunea unui Magazin de Muzică

Facultatea de Matematica si Informatica

Baze de Date

Grupa: 141

Rachieriu Gheorghe Gabriel

Contents

| | |
|---------------------------------------------------------------------|-----------|
| 1 Descrierea modelului real | 3 |
| 1.1 Utilitatea sistemului de baza de date | 3 |
| 2 Prezentarea constrângerilor | 3 |
| 3 Descrierea entităților | 3 |
| 4 Descrierea relațiilor | 4 |
| 4.1 Relatiile de tip One-To-Many si Many-To-Many | 4 |
| 4.2 Relatia de tip 3 | 4 |
| 5 Descrierea atributelor | 5 |
| 6 Realizarea diagramei entitate-relație | 11 |
| 7 Realizarea diagramei conceptuale | 12 |
| 8 Scheme relationale | 12 |
| 9 Realizarea normalizării FN1-FN3 | 13 |
| 9.1 Forma normală 1 - FN1 | 13 |
| 9.2 Forma normală 2 - FN2 | 13 |
| 9.3 Forma normală 3 - FN3 | 14 |
| 10 Crearea secentelor utilizate în inserarea înregistrarilor | 15 |
| 10.1 CASA DE DISCURI | 15 |
| 10.2 PRODUS | 15 |
| 10.3 GEN | 15 |
| 10.4 ARTIST | 15 |
| 10.5 FURNIZOR | 15 |
| 10.6 FORMAT MEDIA | 15 |
| 10.7 FUNCTIE | 15 |
| 10.8 ANGAJAT | 16 |
| 10.9 CUMPARATOR | 16 |
| 10.10STOC | 16 |
| 10.11ACHIZITIE | 16 |
| 10.12MELODIE | 16 |
| 11 Creearea tabelelor SQL și inserarea datelor | 16 |
| 11.1 CASA DE DISCURI | 16 |
| 11.2 PRODUS | 17 |
| 11.3 GEN | 17 |
| 11.4 ALBUM | 17 |
| 11.5 MERCHANDISE | 18 |
| 11.6 ARTIST | 18 |
| 11.7 FURNIZOR | 19 |
| 11.8 FORMAT MEDIA | 19 |
| 11.9 FUNCTIE | 20 |
| 11.10ANGAJAT | 20 |
| 11.11CUMPARATOR | 20 |
| 11.12STOC | 21 |
| 11.13ACHIZITIE | 21 |
| 11.14MELODIE | 22 |

| | |
|-----------------------------------------------------------------|-----------|
| 12 Cereri SQL | 24 |
| 12.1 Cererea nr. 1 - Cererea din feedback | 24 |
| 12.2 Cererea nr. 2 - Cereri sincronizate cu 3 tabele | 25 |
| 12.3 Cererea nr. 3 - Subcereri nesincronizare in FROM | 26 |
| 12.4 Cererea nr.4 - Grupari cu subcereri in HAVING | 27 |
| 12.5 Cererea nr. 5 - NVL, DECODE, Sortari | 28 |
| 12.6 Cererea nr. 6 - Clauza WITH | 29 |
| 13 Cereri de actualizare si suprimare a datelor | 30 |
| 13.1 Cererea nr.1 | 30 |
| 13.2 Cererea nr. 2 | 31 |
| 13.3 Cererea nr. 3 | 32 |

1 Descrierea modelului real

Pe zi ce trece, toate afacerile, activitățile și acțiunile iau ampoare. Același fenomen se poate observa și în ceea ce privește domeniul muzical. Produsele muzicale devin tot mai diverse și sunt prezentate în atât de multe forme, iar numărul de producători este într-o creștere continuă și, tocmai de aceea, crearea unei baze de date este necesară pentru reținerea acestor informații. Acest proiect a fost creat cu scopul de a gestiona toate datele necesare organizării activității unui magazin de muzică, numit "Rachi Records".

Totodată, am ales această temă deoarece vreau să îmi aprofundez cunoștințele în domeniul bazelor de date, cu precădere în sistemul de gestiune al bazelor de date Oracle. O bază de date bine structurată pentru "Rachi Records" ar permite gestionarea eficientă a stocului de muzică, urmărirea vânzărilor, analiza preferințelor clienților și automatizarea proceselor operaționale. Astfel, magazinul poate oferi o experiență mai bună clienților și poate crește eficiența afacerii în ansamblu.

1.1 Utilitatea sistemului de baza de date

Proiectul "Gestiunea unui magazin de muzică" este o bază de date care monitorizează vânzarea de produse din stocul unui magazin, acestea fiind achiziționate de la diferiți furnizori. Scopul principal al acesteia este de a furniza informații precise și actualizate.

Cu ajutorul bazei de date, sunt înregistrate următoarele informații: detalii despre clienți, comenzi plasate de aceștia, produsele achiziționate de clienți, locația fizică a produselor în magazin, detalii despre angajații fiecarei locații și informații despre furnizorii de produse. Astfel, această bază de date asigură o gestionare eficientă a activităților magazinului de muzică și contribuie la îmbunătățirea experienței clientilor.

2 Prezentarea constrângerilor

- Prețului de bază al unui produs îi este adaugat "adaos" în funcție de forma media.
- O casa de discuri face mai multe albume sau niciunul.
- Un album are mai multe genuri muzicale și cel puțin unul.
- Un produs, fie el Album sau Merchandise, este asociat mai multor melodii și cel puțin una.
- Mai mulți artiști pot înregistra o singura melodie.
- Un album este disponibil în mai multe formate media și cel puțin unul.
- Un produs apare în mai multe stocuri sau în niciunul.
- Un furnizor trimite mai multe stocuri, sau niciunul.
- Un Angajat are o singura funcție și cel puțin una.
- O comandă este realizată de cel puțin un angajat.
- Un Cumpărător a efectuat mai multe comenzi, și cel puțin una.
- O achiziție are un singur produs, cantitatea acestuia, clientul și angajatul care se ocupă de comandă.
- Un angajat se ocupă de mai multe comenzi sau de niciuna.
- Un client are mai multe achiziții, sau niciuna, fiind înscris în baza de date fără a cumpăra ceva anterior.
- Un produs poate apărea în mai multe achiziții, sau în niciuna.

3 Descrierea entităților

- Pentru fiecare casa de discuri se cunoște id-ul casei, numele casei și adresa
- Pentru fiecare produs de tip Merchandise se cunoaște id-ul, tipul, numele, și optional marimea.
- Pentru fiecare album se cunoște id-ul albumului, denumirea albumului, data aparției, lungimea și pretul de bază.

- Pentru fiecare furnizor se cunosc id-ul furnizorului si denumirea furnizorului
- Pentru fiecare format media se cunosc id-ul formatului și denumirea si adaosul.
- Pentru fiecare artist se cunoaste id-ul artistului, numărul, numele artistului, prenumele artistului si grupul muzical din care face parte.
- Pentru fiecare client se cunosc id-ul clientului, numele și prenumele, numărul de telefon, adresa si localitatea.
- Pentru fiecare angajat se cunosc id-ul angajatului, numele si prenumele, numărul de telefon, adresa, data angajarii si salariu.
- Pentru fiecare angajat se cunosc id-ul angajatului, numele si prenumele, numărul de telefon, adresa, data angajarii si salariu.
- Pentru fiecare functie se cunoaște id-ul functiei, denumirea, salariul minim si maxim.
- Pentru fiecare gen muzical se cunosc id-ul genului si numele.

4 Descrierea relațiilor

4.1 Relatiile de tip One-To-Many si Many-To-Many

| Relatie | Cardinalitate | Observatii |
|-------------------|--------------------------------------|------------|
| Produce | Casa De Discuri - Album: One to Many | - |
| Exista in | Album - Format Media: One to Many | - |
| Are/Inregistreaza | Produs - Artist: Many to Many | - |
| Distribuie | Furnizot - Produs: Many to Many | - |
| Apartine | Produs - Gen: One to Many | - |
| Are o | Angajat - Functie: One to Many | - |

4.2 Relatia de tip 3

Relatia de tip 3 dintre entitatatile **PRODUS**, **CUMPARATOR** si **ANGAJAT** definește procesul prin care un client cumpără produse, iar tranzacția este gestionată de un angajat. Fiecare achiziție este asociată cu un singur angajat, care o procezează, dar un angajat poate gestiona mai multe achiziții. De asemenea, un client poate face mai multe achiziții sau poate exista în sistem fără a fi cumpărat nimic. Această relație ajută la urmărirea istoricului de cumpărături și la gestionarea vânzărilor.

5 Descrierea atributelor

1. ALBUM

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori Impligate | Observatii |
|-----------------|---------------|--------------|---------------------|------------------|----------------------------|
| id_produs | NUMBER(3,0) | PK, FK | 101, 102, 103 | - | Identificator unic |
| nume_album | VARCHAR2(100) | - | GNX, SOS, Swimming | - | Numele albumului ca text |
| lungime | CHAR(5) | - | 45:30, 38:22, 72:15 | - | Format MM:SS |
| id_casa_discuri | NUMBER(3,0) | FK | 1, 2, 3 | - | Legat de casa de discuri |
| id_gen | NUMBER(3,0) | FK | 1, 2, 3 | - | Genul muzical al albumului |

2. MERCHANDISE

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori Impligate | Observatii |
|--------------|---------------|--------------|------------------------------|------------------|--------------------------------------|
| id_produs | NUMBER(3,0) | PK, FK | 201, 202, 203 | - | Identificator unic |
| tip_merch | VARCHAR2(100) | - | T-shirt, Hoodie, Poster, Cap | - | Tipul produsului |
| marime_merch | VARCHAR2(4) | - | S, M, L, XL, XXL | NULL | Obligatoriu doar pentru îmbrăcăminte |

3. PRODUS

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori Impligate | Observatii |
|--------------|-------------|--------------|------------------------------------|------------------|----------------------------|
| id_produs | NUMBER(3,0) | PK | 101, 201, 301 | - | Cheia primară a produsului |
| pret | NUMBER(6,2) | - | 99.99, 149.99, 24.99 | - | Pret de bază în RON |
| data_lansare | DATE | - | 2024-02-05, 2023-11-12, 2022-08-30 | SYSDATE | Data lansării produsului |

4. GEN

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori Impligate | Observatii |
|----------|---------------|--------------|---------------------------------|------------------|------------------------|
| id_gen | NUMBER(3,0) | PK | 1, 2, 3, 4 | - | Identificator unic |
| nume_gen | VARCHAR2(100) | - | Rock, Pop, Jazz, Metal, Hip-Hop | - | Numele genului muzical |

5. ARTIST

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Observatii |
|--------------|---------------|--------------|----------------------------------------|-------------------|-----------------------------------|
| id_artist | NUMBER(3,0) | PK | 401, 402, 403 | - | Identificator unic |
| prenume | VARCHAR2(100) | - | John, Maria, Alex, Laura | - | Prenumele artistului |
| nume_familie | VARCHAR2(100) | - | Doe, Smith, Johnson, Popescu | - | Numele de familie |
| trupa | VARCHAR2(100) | - | The Rockers, Metallica, Arctic Monkeys | NULL | Poate fi NULL pentru artiști solo |

6. FORMAT MEDIA

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Observatii |
|-------------------|---------------|--------------|------------------------------|-------------------|---------------------------|
| id_format_media | NUMBER(3,0) | PK | 1, 2, 3, 4 | - | Identificator unic |
| nume_format_media | VARCHAR2(100) | - | Vinyl, CD, Digital, Cassette | - | Numele formatului media |
| id_produs | NUMBER(3,0) | FK | 101, 102, 103 | - | Legat de produsul asociat |
| procent_adaugat | NUMBER(3,0) | - | 15, 5, 0, 10 | 0 | Procentul adăugat la preț |

7. FURNIZOR

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Observatii |
|---------------|---------------|--------------|------------------------------------------------------------|-------------------|---------------------|
| id_furnizor | NUMBER(3,0) | PK | 1, 2, 3 | - | Identificator unic |
| nume_furnizor | VARCHAR2(100) | - | Music Warehouse, Vinyl Plus, Global Distribution | - | Numele furnizorului |
| numar_telefon | CHAR(10) | - | 0712345678, 0723456789, 0734567890 | - | Format românesc |
| adresa | VARCHAR2(100) | - | Str. Muzicii 10, Bd. Distributiei 25, Aleea Furnizorilor 5 | - | Adresa furnizorului |

8. FUNCTIE

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Observatii |
|--------------|---------------|--------------|---------------------------------------------------------|-------------------|-----------------------|
| id_functie | NUMBER(3,0) | PK | 1, 2, 3, 4 | - | Identificator unic |
| nume_functie | VARCHAR2(100) | - | Manager, Sales Associate, Cashier, Inventory Specialist | - | Numele funcției |
| salariu_min | NUMBER(6,2) | - | 2500.00, 3000.00, 4000.00 | - | Salariul minim în RON |
| salariu_max | NUMBER(6,2) | - | 5000.00, 7000.00, 10000.00 | - | Salariul maxim în RON |

9. ANGAJAT

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Obersvatii |
|---------------|---------------|--------------|------------------------------------|-------------------|------------------------------|
| id_angajat | NUMBER(3,0) | PK | 1, 2, 3, 4 | - | Identificator unic |
| id_functie | NUMBER(3,0) | FK | 1, 2, 3, 4 | - | Legat de funcția angajatului |
| nume_familie | VARCHAR2(100) | - | Ionescu, Popescu, Popa, Smith | - | Numele de familie |
| prenume | VARCHAR2(100) | - | Ion, Maria, Ana, John | - | Prenumele angajatului |
| numar_telefon | CHAR(11) | - | 0712345678, 0723456789, 0734567890 | - | Format românesc |
| data_angajare | DATE | - | 2020-05-15, 2022-03-10, 2023-08-22 | SYSDATE | Data angajării |
| salariu | NUMBER(6,2) | - | 3500.00, 4200.00, 6000.00 | - | Salariul curent |

10. CUMPARATOR

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Obersvatiile |
|---------------|---------------|--------------|--------------------------------------------------------|-------------------|----------------------|
| id_cumparator | NUMBER(3,0) | PK | 1, 2, 3, 4 | - | Identificator unic |
| nume_familie | VARCHAR2(100) | - | Georgescu, Vasilescu, Brown, Johnson | - | Numele de familie |
| prenume | VARCHAR2(100) | - | Andrei, Elena, Robert, Sarah | - | Prenumele clientului |
| adresa | VARCHAR2(300) | - | Str. Primăverii 10, Bd. Tineretului 25, Main Street 42 | - | Adresa completă |
| oras | VARCHAR2(100) | - | Bucureşti, Cluj-Napoca, Iaşi, London, New York | - | Oraşul clientului |

11. CASA DE DISCURI

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Obersvatiile |
|-----------------|---------------|--------------|----------------------------------------------------------|-------------------|-------------------------|
| id_casa_discuri | NUMBER(3,0) | PK | 1, 2, 3, 4 | - | Identificator unic |
| nume_casa | VARCHAR2(100) | - | Universal Music, Sony Music, Warner Records | - | Numele casei de discuri |
| email | VARCHAR2(100) | - | contact@universal.com, info@sony.com, support@warner.com | - | Email de contact |

12. ACHIZITIE

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Obersvatiile |
|----------------|---------------|--------------|------------------------------------------|-------------------|------------------------------------|
| id_achizitie | NUMBER(3,0) | PK | 1001, 1002, 1003 | - | Identificator unic |
| id_cumparator | NUMBER(3,0) | FK | 1, 2, 3, 4 | - | Legat de clientul care cumpără |
| id_angajat | NUMBER(3,0) | FK | 1, 2, 3, 4 | - | Legat de angajatul care procesează |
| id_produs | NUMBER(3,0) | FK | 101, 201, 301 | - | Legat de produsul cumpărat |
| data_achizitie | DATE | - | 2024-03-15, 2024-02-28, 2024-01-10 | SYSDATE | Data achiziției |
| metoda_plata | VARCHAR2(100) | - | Card, Cash, PayPal, Transfer bancar | 'Card' | Metoda de plată |
| status | VARCHAR2(100) | - | Pending, Completed, Cancelled, Delivered | 'Pending' | Statusul comenzi |
| cantitate | NUMBER(3,0) | - | 1, 2, 3, 5, 10 | 1 | Cantitatea cumpărată |

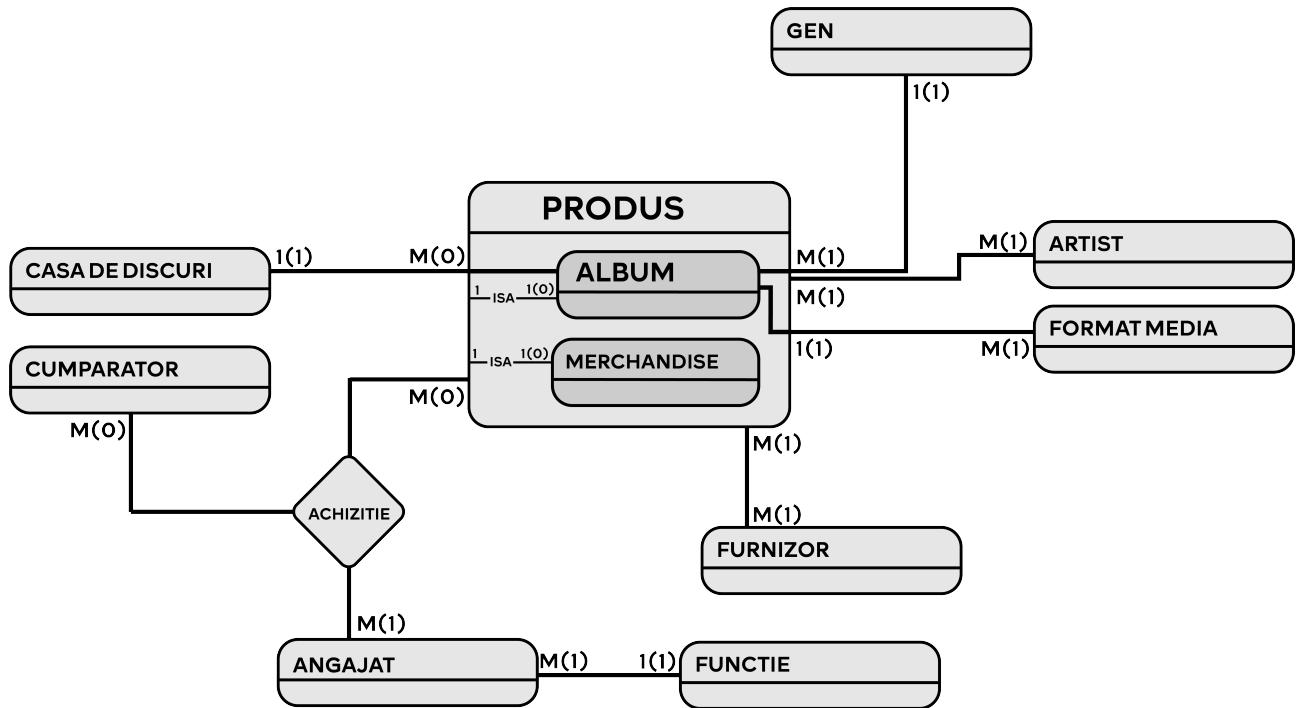
13. STOC

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Obersvatiile |
|-------------|-------------|--------------|-----------------|-------------------|--------------------------------|
| id_stoc | NUMBER(3,0) | PK | 501, 502, 503 | - | Identificator unic |
| id_produs | NUMBER(3,0) | FK | 101, 201, 301 | - | Legat de produsul în stoc |
| id_furnizor | NUMBER(3,0) | FK | 1, 2, 3 | - | Legat de furnizorul produsului |
| cantitate | NUMBER(3,0) | - | 10, 25, 50, 100 | 0 | Cantitatea disponibilă |

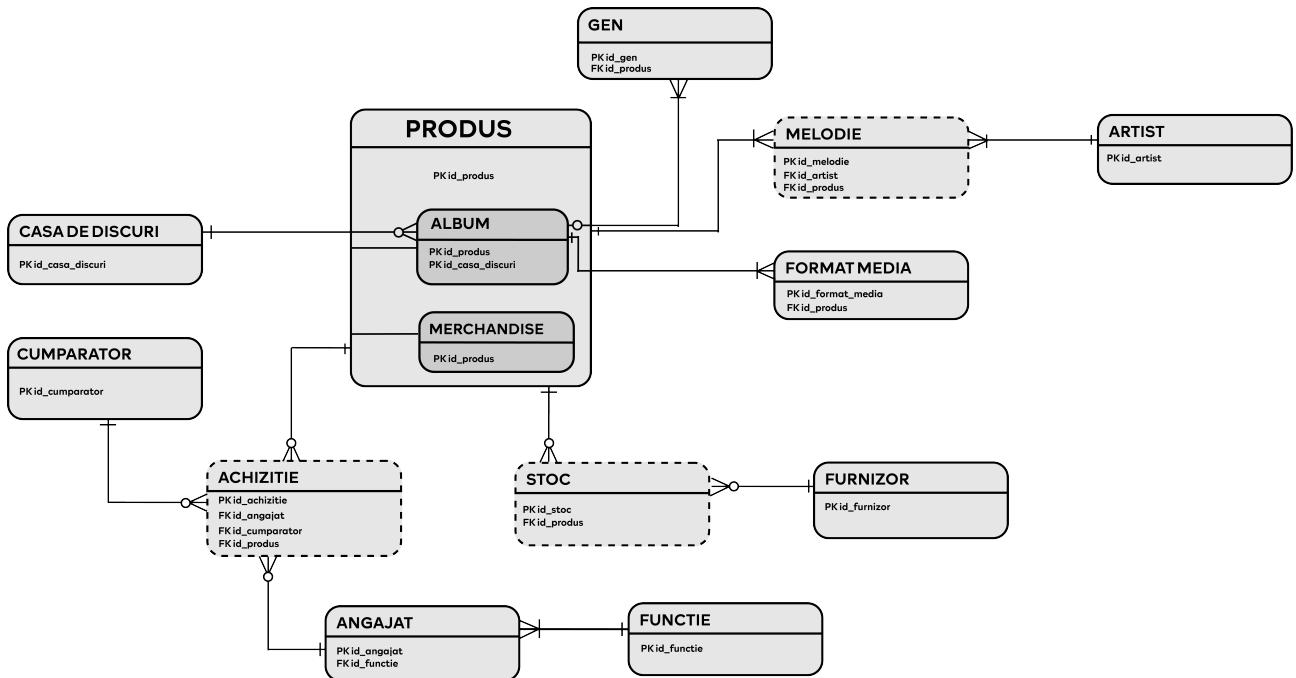
14. MELODIE

| Atribut | Tip de Date | Constrangeri | Valori Posibile | Valori im-plicite | Obersvatiile |
|------------|---------------|--------------|------------------------------------------|-------------------|-----------------------------|
| id_melodie | NUMBER(3,0) | PK | 601, 602, 603 | - | Identifier unic |
| id_artist | NUMBER(3,0) | FK | 401, 402, 403 | - | Legat de artistul melodiei |
| id_produs | NUMBER(3,0) | FK | 101, 102, 103 | - | Legat de produsul (albумul) |
| titlu | VARCHAR2(100) | - | Summer Vibes, Rock Forever, Soul of Mine | - | Titlul melodiei |
| lungime | VARCHAR2(100) | - | 3:45, 4:20, 2:55, 6:10 | - | Durata melodiei (MM:SS) |

6 Realizarea diagramei entitate-relație



7 Realizarea diagramei conceptuale



8 Scheme relationale

Schemele relaționale corespunzătoare diagramei conceptuale sunt următoarele:

ALBUM(id_produs#, nume_album, lungime, id_casa_discuri, id_gen)

MERCHANDISE(id_produs#, tip_merch, marime_merch)

PRODUS(id_produs#, pret, data_lansare)

GEN(id_gen#, nume_gen)

ARTIST(id_artist#, prenume, nume_familie, trupa)

FORMAT_MEDIA(id_format_media#, nume_format_media, id_produs, procent_adaugat)

FURNIZOR(id_furnizor#, nume_furnizor, numar_telefon, adresa)

FUNCTIE(id_functie#, nume_functie, salariu_min, salariu_max)

ANGAJAT(id_angajat#, id_functie, nume_familie, prenume, numar_telefon, data_angajare, salariu)

CUMPARATOR(id_cumparator#, nume_familie, prenume, adresa, oras)

CASA_DE_DISCURI(id_casa_discuri#, nume_casa, email)

ACHIZITIE(id_achizitie#, id_cumparator, id_angajat, id_produs, data_achizitie, metoda_plata, status, cantitate)

STOC(id_stoc#, id_produs, id_furnizor, cantitate)

MELODIE(id_melodie#, id_artist, id_produs, titlu, lungime)

9 Realizarea normalizării FN1-FN3

9.1 Forma normală 1 - FN1

Forma normală 1 (FN1) impune ca toate valorile stocate într-un câmp al unei baze de date să fie atomice, adică să conțină o singură valoare, nu liste sau grupuri de date. Totodată, fiecare rând din tabel trebuie să poată fi identificat în mod unic, prin intermediul unei chei primare.

În modelul implementat, cerințele acestei forme normale sunt respectate: nu există câmpuri care conțin mai multe valori, iar fiecare înregistrare este identificabilă în mod clar cu ajutorul unei chei primare.

Pentru a ilustra această etapă de normalizare, se poate analiza următorul exemplu:

| ARTIST | MELODIE |
|---------|---------------|
| Artist1 | ML1, ML2, ML3 |
| Artist2 | ML2, ML4 |
| Artist3 | ML3 |

Table 1: Exemplu non-FN1.

| ARTIST | MELODIE |
|---------|---------|
| Artist1 | ML1 |
| Artist1 | ML2 |
| Artist1 | ML3 |
| Artist2 | ML2 |
| Artist2 | ML4 |
| Artist3 | ML3 |

Table 2: Exemplu FN1.

9.2 Forma normală 2 - FN2

O relație se află în a doua formă normală (FN2) doar dacă este deja în forma normală 1 (FN1) și dacă fiecare atribut care nu face parte din cheia primară depinde de întreaga cheie, nu doar de o parte a acesteia.

Mai exact, FN2 presupune ca toate atributele non-cheie dintr-o tabelă să fie dependente funcțional de cheia primară în întregime, nu parțial.

În cazul modelului implementat, sunt îndeplinite condițiile formei normale a două: relațiile sunt în FN1, iar toate atributurile care nu fac parte din cheile primare sunt dependente în mod complet de acestea.

Pentru a ilustra procesul de normalizare, vom analiza exemplul următor:

| id_artist# | prenume | id_produs# | pret |
|------------|----------|------------|-------|
| A1 | Kendrick | P1 | 6.99 |
| A2 | West | P1 | 6.99 |
| A1 | Kendrick | P2 | 15.99 |
| A3 | Miller | P3 | 10.49 |

Table 3: Exemplu non-FN2.

Observăm că avem următoarele dependențe:

$\{id_artist\} \rightarrow \{prenume\}$. id_artist determină funcțional numele.
 $\{id_produs\} \rightarrow \{pret\}$

| id_artist# | id_produs# | pret |
|------------|------------|-------|
| A1 | P1 | 6.99 |
| A2 | P1 | 6.99 |
| A1 | P2 | 15.99 |
| A3 | P3 | 10.49 |

Table 4: Exemplu FN2.

| id_artist# | prenume |
|------------|----------|
| A1 | Kendrick |
| A2 | West |
| A3 | Miller |

Table 5: Exemplu FN2.

9.3 Forma normală 3 - FN3

Asemănător, o entitate se găsește în a treia formă(FN3) normală dacă și numai dacă se găsește în a doua formă normală(FN2) și în plus niciun atribut care nu este parte a UID-ului nu depinde de un alt atribut non-UID. Cu alte cuvinte, nu se acceptă dependențe tranzitive, adică un atribut să depindă de UID în mod indirect.

FN3 cere ca orice atribut non-cheie să depindă doar de cheia primară, de întreaga cheie și exclusiv de aceasta, fără intermedieri prin alte atrbute.

În cadrul modelului implementat, se respectă toate cerințele formei normale a treia: toate atrbutele care nu sunt chei sunt dependente direct de cheia primară.

Pentru a ilustra această normalizare, vom analiza exemplul următor:

| id_artist# | id_produs# | pret |
|------------|------------|-------|
| A1 | P1 | 6.99 |
| A2 | P1 | 6.99 |
| A1 | P2 | 15.99 |
| A3 | P3 | 10.49 |

Table 6: Exemplu non-FN3.

Pentru a aduce relația R în forma normală 3 (FN3), se elimină dependențele funcționale tranzitive, astfel încât toate atrbutele non-cheie să depindă direct și exclusiv de cheia primară.

R1(id_produs#, pret)

R2(id_artist#, id_produs#)

| id_produs# | pret |
|------------|-------|
| P1 | 6.99 |
| P2 | 15.99 |
| P3 | 10.49 |

Table 7: Exemplu FN3.

| id_artist# | id_produs# |
|------------|------------|
| A1 | P1 |
| A2 | P1 |
| A1 | P2 |
| A3 | P3 |

Table 8: Exemplu FN3.

10 Crearea seventelor utilizate in inserarea inregistrarilor

10.1 CASA DE DISCURI

```
CREATE SEQUENCE SEQ_CASA_DE_DISCURI
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.2 PRODUS

```
CREATE SEQUENCE SEQ_PRODUS
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.3 GEN

```
CREATE SEQUENCE SEQ_GEN
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.4 ARTIST

```
CREATE SEQUENCE SEQ_ARTIST
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.5 FURNIZOR

```
CREATE SEQUENCE SEQ_FURNIZOR
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.6 FORMAT MEDIA

```
CREATE SEQUENCE SEQ_FORMAT_MEDIA
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.7 FUNCTIE

```
CREATE SEQUENCE SEQ_FUNCTIE
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.8 ANGAJAT

```
CREATE SEQUENCE SEQ_ANGAJAT
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.9 CUMPARATOR

```
CREATE SEQUENCE SEQ_CUMPARATOR
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.10 STOC

```
CREATE SEQUENCE SEQ_STOC
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.11 ACHIZITIE

```
CREATE SEQUENCE SEQ_ACHIZITIE
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

10.12 MELODIE

```
CREATE SEQUENCE SEQ_MELODIE
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;
```

11 Creearea tabelelor SQL si inserarea datelor

11.1 CASA DE DISCURI

```
CREATE TABLE CASA_DE_DISCURI (
    id_casa_discuri NUMBER(3,0) CONSTRAINT pk_casa_de_discuri PRIMARY KEY,
    nume_casa VARCHAR2(100),
    email VARCHAR2(100)
);

INSERT INTO CASA_DE_DISCURI VALUES (SEQ_CASA_DE_DISCURI.NEXTVAL, 'Universal Music',
'contact@universalmusic.com');
INSERT INTO CASA_DE_DISCURI VALUES (SEQ_CASA_DE_DISCURI.NEXTVAL, 'Sony Music',
'info@sonymusic.com');
INSERT INTO CASA_DE_DISCURI VALUES (SEQ_CASA_DE_DISCURI.NEXTVAL, 'Republic Records',
'office@republicrecords.com');
INSERT INTO CASA_DE_DISCURI VALUES (SEQ_CASA_DE_DISCURI.NEXTVAL, 'Atlantic Records',
'info@atlanticrecords.com');
INSERT INTO CASA_DE_DISCURI VALUES (SEQ_CASA_DE_DISCURI.NEXTVAL, 'Electrecord',
```

```
'contact@electrecord.ro');
INSERT INTO CASA_DE_DISCURI VALUES (SEQ_CASA_DE_DISCURI.NEXTVAL, 'Def Jam Recordings',
'contact@defjam.com');
INSERT INTO CASA_DE_DISCURI VALUES (SEQ_CASA_DE_DISCURI.NEXTVAL, 'Interscope Records',
'info@interscope.com');

COMMIT;
```

11.2 PRODUS

```
CREATE TABLE PRODUS (
    id_produs NUMBER(3,0) CONSTRAINT pk_produs PRIMARY KEY,
    pret NUMBER(6,2),
    data_lansare DATE DEFAULT SYSDATE
);

INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 49.99, TO_DATE('15/03/2022', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 49.99, TO_DATE('15/03/2022', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 39.99, TO_DATE('22/05/2022', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 29.99, TO_DATE('10/07/2022', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 19.99, TO_DATE('30/09/2022', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 59.99, TO_DATE('12/12/2022', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 34.99, TO_DATE('25/01/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 44.99, TO_DATE('18/03/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 24.99, TO_DATE('05/05/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 64.99, TO_DATE('28/06/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 54.99, TO_DATE('15/08/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 52.99, TO_DATE('15/09/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 47.99, TO_DATE('25/10/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 22.99, TO_DATE('08/11/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 37.99, TO_DATE('01/12/2023', 'DD/MM/YYYY'));
INSERT INTO PRODUS VALUES (SEQ_PRODUS.NEXTVAL, 42.99, TO_DATE('15/01/2024', 'DD/MM/YYYY'));

COMMIT;
```

11.3 GEN

```
CREATE TABLE GEN (
    id_gen NUMBER(3,0) CONSTRAINT pk_gen PRIMARY KEY,
    nume_gen VARCHAR2(100)
);

INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'RnB');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Rap');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Pop');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Rap-RNB');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Folk');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Hip-Hop');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Alternative');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Indie');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Electronic');
INSERT INTO GEN VALUES (SEQ_GEN.NEXTVAL, 'Alternative Pop');

COMMIT;
```

11.4 ALBUM

```
CREATE TABLE ALBUM (
    id_produs NUMBER(3,0) CONSTRAINT pk_album PRIMARY KEY,
```

```

nume_album VARCHAR2(100),
lungime CHAR(5),
id_casa_discuri NUMBER(3,0),
id_gen NUMBER(3,0),
CONSTRAINT fk_album_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs),
CONSTRAINT fk_album_casa FOREIGN KEY (id_casa_discuri) REFERENCES CASA_DE_DISCURI(id_casa_discuri),
CONSTRAINT fk_album_gen FOREIGN KEY (id_gen) REFERENCES GEN(id_gen)
);

INSERT INTO ALBUM VALUES (1, 'Swimming', '49:07', 1, 1);
INSERT INTO ALBUM VALUES (2, 'GNX', '42:11', 2, 2);
INSERT INTO ALBUM VALUES (3, 'Short n Sweet', '42:19', 3, 3);
INSERT INTO ALBUM VALUES (4, 'HUMBLE', '43:00', 4, 4);
INSERT INTO ALBUM VALUES (5, 'folklore', '45:30', 5, 5);
INSERT INTO ALBUM VALUES (11, 'the album', '30:45', 1, 8);
INSERT INTO ALBUM VALUES (12, 'Blonde', '60:08', 3, 7);
INSERT INTO ALBUM VALUES (13, 'DAMN.', '54:54', 4, 2);
INSERT INTO ALBUM VALUES (15, 'After Hours', '56:19', 6, 9);
INSERT INTO ALBUM VALUES (16, 'Happier Than Ever', '56:15', 7, 10);

COMMIT;

```

11.5 MERCANDISE

```

CREATE TABLE MERCANDISE (
    id_produs NUMBER(3,0) CONSTRAINT pk_merchandise PRIMARY KEY,
    tip_merch VARCHAR2(100),
    marime_merch VARCHAR2(4),
    CONSTRAINT fk_merchandise_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs)
);

INSERT INTO MERCANDISE VALUES (6, 'T-Shirt', 'M');
INSERT INTO MERCANDISE VALUES (7, 'Poster', NULL);
INSERT INTO MERCANDISE VALUES (8, 'Cap', 'S');
INSERT INTO MERCANDISE VALUES (9, 'Mug', NULL);
INSERT INTO MERCANDISE VALUES (10, 'Hoodie', 'XL');
INSERT INTO MERCANDISE VALUES (14, 'Vinyl Stand', NULL);

COMMIT;

```

11.6 ARTIST

```

CREATE TABLE ARTIST (
    id_artist NUMBER(3,0) CONSTRAINT pk_artist PRIMARY KEY,
    prenume VARCHAR2(100),
    nume_familie VARCHAR2(100),
    trupa VARCHAR2(100)
);

INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'Mac', 'Miller', NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'Kendrick', 'Lamar', NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'Sabrina', 'Carpenter', NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'Taylor', 'Swift', NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'boygenius', 'boygenius', 'band');
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'Rihanna', NULL, NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'Frank', 'Ocean', NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'Billie', 'Eilish', NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'The Weeknd', NULL, NULL);
INSERT INTO ARTIST VALUES (SEQ_ARTIST.NEXTVAL, 'SZA', NULL, NULL);

```

COMMIT;

11.7 FURNIZOR

```
CREATE TABLE FURNIZOR (
    id_furnizor NUMBER(3,0) CONSTRAINT pk_furnizor PRIMARY KEY,
    nume_furnizor VARCHAR2(100),
    numar_telefon CHAR(10),
    adresa VARCHAR2(100)
);

INSERT INTO FURNIZOR VALUES (SEQ_FURNIZOR.NEXTVAL, 'Music Distribution SRL', '07456123890',
'Str. Muzicii nr. 15, Bucureşti');
INSERT INTO FURNIZOR VALUES (SEQ_FURNIZOR.NEXTVAL, 'Vinyl Factory', '07345678912',
'Calea Victoriei nr. 78, Bucureşti');
INSERT INTO FURNIZOR VALUES (SEQ_FURNIZOR.NEXTVAL, 'Merch Production', '07234567891',
'Bd. Unirii nr. 45, Bucureşti');
INSERT INTO FURNIZOR VALUES (SEQ_FURNIZOR.NEXTVAL, 'Global Logistics', '07123456789',
'Str. Industriilor nr. 12, Cluj-Napoca');
INSERT INTO FURNIZOR VALUES (SEQ_FURNIZOR.NEXTVAL, 'Record Distributor', '07987654321',
'Str. Republicii nr. 36, Timişoara');
INSERT INTO FURNIZOR VALUES (SEQ_FURNIZOR.NEXTVAL, 'Audio Tech SRL', '07765432109',
'Str. Aviației nr. 24, Bucureşti');
INSERT INTO FURNIZOR VALUES (SEQ_FURNIZOR.NEXTVAL, 'Music Warehouse', '07654321098',
'Calea Floreasca nr. 55, Bucureşti');
```

COMMIT;

11.8 FORMAT MEDIA

```
CREATE TABLE FORMAT_MEDIA (
    id_format_media NUMBER(3,0) CONSTRAINT pk_format_media PRIMARY KEY,
    nume_format_media VARCHAR2(100),
    id_produs NUMBER(3,0),
    procent_adaugat NUMBER(3,0) DEFAULT 0,
    CONSTRAINT fk_format_media_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs)
);

INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'CD', 1, 10);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Vinyl', 1, 25);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Cassette', 1, 25);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Digital', 1, 25);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Vinyl', 2, 20);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Digital', 3, 0);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'CD', 3, 0);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Cassette', 4, 5);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Digital', 5, 0);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Vinyl', 5, 25);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Cassette', 5, 10);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Digital', 11, 0);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Digital', 12, 0);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Vinyl', 12, 25);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Cassette', 13, 10);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Digital', 15, 0);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'CD', 15, 15);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Digital', 16, 0);
INSERT INTO FORMAT_MEDIA VALUES (SEQ_FORMAT_MEDIA.NEXTVAL, 'Vinyl 7"', 16, 20);
```

COMMIT;

11.9 FUNCTIE

```
CREATE TABLE FUNCTIE (
    id_functie NUMBER(3,0) CONSTRAINT pk_functie PRIMARY KEY,
    nume_functie VARCHAR2(100),
    salariu_min NUMBER(6,2),
    salariu_max NUMBER(6,2)
);

INSERT INTO FUNCTIE VALUES (SEQ_FUNCTIE.NEXTVAL, 'Manager', 5000, 8000);
INSERT INTO FUNCTIE VALUES (SEQ_FUNCTIE.NEXTVAL, 'Sales Representative', 3000, 5000);
INSERT INTO FUNCTIE VALUES (SEQ_FUNCTIE.NEXTVAL, 'Cashier', 2500, 3500);
INSERT INTO FUNCTIE VALUES (SEQ_FUNCTIE.NEXTVAL, 'Marketing Specialist', 4000, 6000);
INSERT INTO FUNCTIE VALUES (SEQ_FUNCTIE.NEXTVAL, 'Store Assistant', 2000, 3000);
INSERT INTO FUNCTIE VALUES (SEQ_FUNCTIE.NEXTVAL, 'IT Support', 3500, 5500);
INSERT INTO FUNCTIE VALUES (SEQ_FUNCTIE.NEXTVAL, 'Inventory Manager', 4500, 6500);
```

COMMIT;

11.10 ANGAJAT

```
CREATE TABLE ANGAJAT (
    id_angajat NUMBER(3,0) CONSTRAINT pk_angajat PRIMARY KEY,
    id_functie NUMBER(3,0),
    nume_familie VARCHAR2(100),
    prenume VARCHAR2(100),
    numar_telefon CHAR(10),
    data_angajare DATE DEFAULT SYSDATE,
    salariu NUMBER(6,2),
    CONSTRAINT fk_angajat_functie FOREIGN KEY (id_functie) REFERENCES FUNCTIE(id_functie)
);

INSERT INTO ANGAJAT VALUES (SEQ_ANGAJAT.NEXTVAL, 1, 'Popescu', 'Maria', '07123456789',
TO_DATE('15/01/2020', 'DD/MM/YYYY'), 6500);
INSERT INTO ANGAJAT VALUES (SEQ_ANGAJAT.NEXTVAL, 2, 'Ionescu', 'Andrei', '07234567890',
TO_DATE('10/03/2020', 'DD/MM/YYYY'), 4000);
INSERT INTO ANGAJAT VALUES (SEQ_ANGAJAT.NEXTVAL, 3, 'Popa', 'Elena', '07345678901',
TO_DATE('22/05/2021', 'DD/MM/YYYY'), 3000);
INSERT INTO ANGAJAT VALUES (SEQ_ANGAJAT.NEXTVAL, 4, 'Georgescu', 'Mihai', '07456789012',
TO_DATE('17/08/2021', 'DD/MM/YYYY'), 5000);
INSERT INTO ANGAJAT VALUES (SEQ_ANGAJAT.NEXTVAL, 5, 'Stanciu', 'Cristina', '07567890123',
TO_DATE('05/12/2021', 'DD/MM/YYYY'), 2500);
INSERT INTO ANGAJAT VALUES (SEQ_ANGAJAT.NEXTVAL, 6, 'Dumitrache', 'Alexandru', '07678901234',
TO_DATE('10/02/2022', 'DD/MM/YYYY'), 4500);
INSERT INTO ANGAJAT VALUES (SEQ_ANGAJAT.NEXTVAL, 7, 'Marinescu', 'Diana', '07789012345',
TO_DATE('18/04/2022', 'DD/MM/YYYY'), 5500);
```

COMMIT;

11.11 CUMPARATOR

```
CREATE TABLE CUMPARATOR (
    id_cumparator NUMBER(3,0) CONSTRAINT pk_cumparator PRIMARY KEY,
    nume_familie VARCHAR2(100),
    prenume VARCHAR2(100),
    adresa VARCHAR2(300),
    oras VARCHAR2(100)
```

```

);

INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Dumitrescu', 'Alexandru',
'Str. Libertății nr. 10, București', 'București');
INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Radu', 'Ioana',
'Str. Unirii nr. 25, Cluj-Napoca', 'Cluj-Napoca');
INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Stoica', 'Gabriel',
'Bd. Independenței nr. 15, Iași', 'Iași');
INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Munteanu', 'Diana',
'Str. Mihai Viteazul nr. 8, Timișoara', 'Timișoara');
INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Dinu', 'Bogdan',
'Str. Primăverii nr. 12, Brașov', 'Brașov');
INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Vasile', 'Andreea',
'Str. Florilor nr. 5, Constanța', 'Constanța');
INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Preda', 'Robert',
'Str. Tudor Vladimirescu nr. 18, Craiova', 'Craiova');
INSERT INTO CUMPARATOR VALUES (SEQ_CUMPARATOR.NEXTVAL, 'Manole', 'Cristina',
'Str. Mioritei nr. 7, Sibiu', 'Sibiu');

COMMIT;

```

11.12 STOC

```

CREATE TABLE STOC (
    id_stoc NUMBER(3,0) CONSTRAINT pk_stoc PRIMARY KEY,
    id_produs NUMBER(3,0),
    id_furnizor NUMBER(3,0),
    cantitate NUMBER(3,0) DEFAULT 0,
    CONSTRAINT fk_stoc_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs),
    CONSTRAINT fk_stoc_furnizor FOREIGN KEY (id_furnizor) REFERENCES FURNIZOR(id_furnizor)
);

INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 1, 1, 50);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 1, 3, 50);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 2, 2, 30);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 2, 7, 30);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 2, 5, 70);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 3, 1, 100);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 4, 2, 20);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 5, 1, 40);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 6, 3, 75);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 7, 3, 60);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 8, 3, 90);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 9, 3, 45);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 10, 3, 30);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 11, 1, 25);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 12, 2, 15);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 13, 1, 35);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 14, 3, 50);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 15, 2, 40);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 15, 4, 60);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 15, 5, 70);
INSERT INTO STOC VALUES (SEQ_STOC.NEXTVAL, 16, 1, 30);

COMMIT;

```

11.13 ACHIZITIE

```
CREATE TABLE ACHIZITIE (
```

```

id_achizitie NUMBER(3,0) CONSTRAINT pk_achizitie PRIMARY KEY,
id_cumparator NUMBER(3,0),
id_angajat NUMBER(3,0),
id_produs NUMBER(3,0),
data_achizitie DATE DEFAULT SYSDATE,
metoda_plata VARCHAR2(100) DEFAULT 'Card',
status VARCHAR2(100) DEFAULT 'Pending',
cantitate NUMBER(3,0) DEFAULT 1,
CONSTRAINT fk_achizitie_cumparator FOREIGN KEY (id_cumparator) REFERENCES CUMPARATOR(id_cumparator)
CONSTRAINT fk_achizitie_angajat FOREIGN KEY (id_angajat) REFERENCES ANGAJAT(id_angajat),
CONSTRAINT fk_achizitie_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs)
);

INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 1, 2, 1, TO_DATE('20/04/2023', 'DD/MM/YYYY'), 'Card', 'Completed', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 2, 2, 3, TO_DATE('25/04/2023', 'DD/MM/YYYY'), 'Cash', 'Completed', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 3, 3, 6, TO_DATE('30/04/2023', 'DD/MM/YYYY'), 'Card', 'Completed', 2);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 4, 2, 2, TO_DATE('05/05/2023', 'DD/MM/YYYY'), 'Card', 'Pending', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 5, 3, 8, TO_DATE('10/05/2023', 'DD/MM/YYYY'), 'Cash', 'Pending', 3);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 1, 2, 5, TO_DATE('15/05/2023', 'DD/MM/YYYY'), 'Card', 'Completed', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 2, 3, 10, TO_DATE('20/05/2023', 'DD/MM/YYYY'), 'Cash', 'Cancelled', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 3, 2, 4, TO_DATE('25/05/2023', 'DD/MM/YYYY'), 'Card', 'Completed', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 6, 4, 12, TO_DATE('02/06/2023', 'DD/MM/YYYY'), 'Card', 'Completed', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 7, 5, 13, TO_DATE('10/06/2023', 'DD/MM/YYYY'), 'Cash', 'Completed', 2);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 8, 2, 15, TO_DATE('18/06/2023', 'DD/MM/YYYY'), 'Card', 'Pending', 1);
INSERT INTO ACHIZITIE VALUES (SEQ_ACHIZITIE.NEXTVAL, 4, 3, 11, TO_DATE('25/06/2023', 'DD/MM/YYYY'), 'Cash', 'Completed', 1);

COMMIT;

```

11.14 MELODIE

```

CREATE TABLE MELODIE (
    id_melodie NUMBER(3,0) CONSTRAINT pk_melodie PRIMARY KEY,
    id_artist NUMBER(3,0),
    id_produs NUMBER(3,0),
    titlu VARCHAR2(100),
    lungime VARCHAR2(100),
    CONSTRAINT fk_melodie_artist FOREIGN KEY (id_artist) REFERENCES ARTIST(id_artist),
    CONSTRAINT fk_melodie_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs)
);

INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Come Back to Earth', '2:41');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Hurt Feelings', '4:05');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'What's the Use?', '4:48');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Perfecto', '3:35');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Self Care', '5:45');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Wings', '4:10');

```

```

INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Ladders', '4:47');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Small Worlds', '4:31');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Conversation Pt. 1', '3:30');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Dunno', '3:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'Jet Fuel', '5:45');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, '2009', '5:47');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 1, 1, 'So It Goes', '5:12');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'wacked out murals', '5:17');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'squabble up', '2:37');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 10, 2, 'luther (with sza)', '2:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'luther (with sza)', '2:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'man at the garden', '3:53');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'hey now (feat. dody6)', '3:37');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'reincarnated', '4:35');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'tv off (feat. hefty gunplay)', '3:40');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'dodger blue (feat. wallie the...)', '2:11');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'peekaboo (feat. azchike)', '2:35');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 2, 'heart pt. 6', '4:52');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Taste', '4:54');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Please Please Please', '5:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Good Graces', '3:05');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Sharpest Tool', '3:38');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Coincidence', '2:44');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Bed Chem', '2:51');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Espresso', '2:55');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Dumb n Poetic', '2:13');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Slim Pickins', '2:32');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Juno', '3:43');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 3, 3, 'Lie to girls', '3:22');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'BLOOD.', '1:58');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'DNA', '3:05');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'YAH.', '2:40');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'ELEMENT.', '3:28');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'FEEL.', '3:34');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'LOYALTY.', '3:47');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 6, 4, 'LOYALTY.', '3:47');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'PRIDE.', '4:35');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'HUMBLE', '6:22');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'LUST.', '5:07');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'LOVE.', '3:33');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'XXX.', '4:14');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'FEAR.', '7:40');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'GOD.', '4:08');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 4, 'DUCKWORTH.', '4:08');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'cardigan', '3:59');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'august', '4:21');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'exile (featuring Bon Iver)', '4:45');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'my tears ricochet', '4:15');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'mirrorball', '3:29');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'seven', '3:28');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'this is me trying', '3:15');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'illicit affairs', '3:10');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'betty', '4:54');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'peace', '5:23');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 4, 5, 'hoax', '3:40');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 5, 11, 'two', '3:12');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 5, 11, 'three', '2:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 5, 11, 'four', '3:22');

```

```

INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 5, 11, 'five', '4:01');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 5, 11, 'six', '3:44');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 5, 11, 'seven', '3:18');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 5, 11, 'eight', '2:55');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 7, 12, 'Pink + White', '3:04');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 7, 12, 'Solo', '4:17');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 7, 12, 'Self Control', '4:09');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 7, 12, 'Nights', '5:07');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 7, 12, 'White Ferrari', '4:08');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 7, 12, 'Godspeed', '2:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 7, 12, 'Futura Free', '9:24');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 13, 'BLOOD.', '1:58');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 13, 'DNA.', '3:05');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 13, 'ELEMENT.', '3:28');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 13, 'FEEL.', '3:34');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 13, 'HUMBLE.', '2:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 13, 'XXX.', '4:14');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 2, 13, 'FEAR.', '7:40');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'After Hours', '6:01');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'Heartless', '3:18');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'In Your Eyes', '3:57');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'Too Late', '3:59');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'Hardest To Love', '3:31');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'Scared To Live', '3:11');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'Snowchild', '4:07');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'Escape From LA', '5:55');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 9, 15, 'Faith', '4:43');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'Therefore I Am', '2:54');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'Your Power', '4:05');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'Lost Cause', '3:32');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'my future', '3:30');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'Oxytocin', '3:30');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'GOLDWING', '2:31');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'Billie Bossa Nova', '3:16');
INSERT INTO MELODIE VALUES (SEQ_MELODIE.NEXTVAL, 8, 16, 'Getting Older', '4:04');

COMMIT;

```

12 Cereri SQL

12.1 Cererea nr. 1 - Cererea din feedback

Care sunt albumele care se pot achizitiona in cel putin doua formate diferite, si care sunt aduse de cel putin doi furnizori. In aceasta cerere am folosit 2 subinterrogari. Prima returneaza albumele ce se gasesc in cel putin 2 formaturi media, grupandu-le dupa id_product si numarand cate randuri distincte are fiecare. A doua functioneaza asemanator, dar cu datele din tabela STOCK.

```

SELECT a.id_produs, a.nume_album
FROM ALBUM a
WHERE a.id_produs IN (
    SELECT m.id_produs
    FROM FORMAT_MEDIA m
    GROUP BY m.id_produs
    HAVING COUNT(DISTINCT m.id_format_media) >= 2
)
AND a.id_produs IN (
    SELECT s.id_produs
    FROM STOC s

```

```

        GROUP BY s.id_produs
        HAVING COUNT(DISTINCT s.id_furnizor) >= 2
);

```

The screenshot shows a SQL worksheet interface with multiple tabs at the top: 'Gestiunea_Unui_Records_Store', 'clear.sql', 'create_tables.sql', 'insert_in_tables.sql', 'queries.sql' (which is the active tab), and 'update_delete.sql'. Below the tabs is a toolbar with icons for running, saving, and zooming. The main area is divided into two panes: 'Worksheet' and 'Query Builder'. The 'Worksheet' pane contains the following SQL code:

```

--1
SELECT a.id_produs, a.numere_album
FROM ALBUM a
WHERE a.id_produs IN (
    SELECT m.id_produs
    FROM FORMAT_MEDIA m
    GROUP BY m.id_produs
    HAVING COUNT(DISTINCT m.id_format_media) >= 2
)
AND a.id_produs IN (
    SELECT s.id_produs
    FROM STOC s
    GROUP BY s.id_produs
    HAVING COUNT(DISTINCT s.id_furnizor) >= 2
);

```

The 'Query Result' pane below shows the output of the query:

| ID_PRODUS | NUMERE_ALBUM |
|-----------|----------------|
| 1 | 1 Swimming |
| 2 | 15 After Hours |

12.2 Cererea nr. 2 - Cereri sincronizate cu 3 tabele

Afisati numele albumului, pretul si casa de discuri a artistilor al caror nume incepe cu "K" sau al caror nume de familie incepe cu M si au un pret mai mare decat pretul mediu al tuturor albumelor din baza de date. Aceasta cerere foloseste JOIN pe 4 tabele, pentru a accesa toate datele necesare in enunt, iar subinterrogarea pentru medie fara de care query-ul nu poate continua face cererea sincronizata.

```

SELECT DISTINCT a.numere_album, p.pret, r.numere_casa
FROM ALBUM a
JOIN PRODUS p ON a.id_produs = p.id_produs
JOIN CASA_DE_DISCURI r ON a.id_casa_discuri = r.id_casa_discuri
JOIN MELODIE mel ON mel.id_produs = a.id_produs
JOIN ARTIST ar ON mel.id_artist = ar.id_artist
WHERE (ar.prenume LIKE 'K%' OR ar.numere_familie LIKE 'M%')
    AND p.pret > (
        SELECT AVG(pret)
        FROM PRODUS
        WHERE id_produs IN (SELECT id_produs FROM ALBUM)
    );

```

The screenshot shows a SQL worksheet interface with multiple tabs at the top: 'Gestiunea_Unui_Records_Store', 'dear.sql', 'create_tables.sql', 'insert_in_tables.sql', 'queries.sql', and 'update_delete.sql'. The 'Worksheet' tab is selected. The main area contains the following SQL code:

```
--2
SELECT DISTINCT a.numere_album, p.pret, r.numere_casa
FROM ALBUM a
JOIN PRODUS p ON a.id_produs = p.id_produs
JOIN CASA_DE_DISCURI r ON a.id_casa_discuri = r.id_casa_discuri
JOIN MELODIE mel ON mel.id_produs = a.id_produs
JOIN ARTIST ar ON mel.id_artist = ar.id_artist
WHERE (ar.prenume LIKE 'K%' OR ar.nume_familie LIKE 'M%')
AND p.pret > (
    SELECT AVG(pret)
    FROM PRODUS
    WHERE id_produs IN (SELECT id_produs FROM ALBUM)
);
```

Below the code, the 'Query Result' tab is selected, showing the following table:

| NUME_ALBUM | PRET | NUME_CASA |
|------------|-------|------------------|
| 1 Swimming | 49.99 | Universal Music |
| 2 GNX | 49.99 | Sony Music |
| 3 DAMN. | 47.99 | Atlantic Records |

12.3 Cererea nr. 3 - Subcereri nesincronizare in FROM

Afisati numele formatului media, numarul de albume care se gasesc in respectivul format media, si media pretului tuturor albumelor din respectivul format media dupa adaugarea procentului la pretul de baza.

```
SELECT fm.nume_format_media,
       COUNT(a.id_produs) AS numar_albume,
       ROUND(AVG(p.pret * (1 + fm.procent_adaugat/100)), 2) AS pret_cu_adao
FROM ALBUM a
JOIN PRODUS p ON a.id_produs = p.id_produs
JOIN (
    SELECT id_produs, nume_format_media, procent_adaugat
    FROM FORMAT_MEDIA
    WHERE procent_adaugat > 0
) fm ON fm.id_produs = a.id_produs
GROUP BY fm.nume_format_media
ORDER BY pret_cu_adao DESC;
```

The screenshot shows a SQL Server Management Studio window. The top bar has tabs for 'Gestiunea_Unui_Records_Store' and several other scripts like 'dear.sql', 'create_tables.sql', etc. The main area is a 'Worksheet' tab with a 'Query Builder' interface. The query itself is:

```
--3
SELECT fm.nume_format_media,
       COUNT(a.id_produs) AS numar_albume,
       ROUND(AVG(p.pret * (1 + fm.procent_adaugat/100)), 2) AS pret_cu_adao
FROM ALBUM a
JOIN PRODUS p ON a.id_produs = p.id_produs
JOIN (
    SELECT id_produs, nume_format_media, procent_adaugat
    FROM FORMAT_MEDIA
    WHERE procent_adaugat > 0
) fm ON fm.id_produs = a.id_produs
GROUP BY fm.nume_format_media
ORDER BY pret_cu_adao DESC;
```

Below the code, the 'Query Result' tab is open, showing the following data:

| NUME_FORMAT_MEDIA | NUMAR_ALBUME | PRET CU ADAOS |
|-------------------|--------------|---------------|
| 1 Digital | 1 | 62.49 |
| 2 Vinyl | 4 | 53.43 |
| 3 Vinyl 7" | 1 | 51.59 |
| 4 CD | 2 | 49.34 |
| 5 Cassette | 4 | 42.19 |

12.4 Cererea nr.4 - Grupari cu subcereri in HAVING

Afisati genurile muzicale care au **numarul de melodii mai mare decat media numarului de melodii per album**. Pentru fiecare gen muzical afisat se cere si **numarul de albume distinste**, **numarul total de melodii si pretul mediu al albumelor pentru fiecare gen**. Rezultatele vor fi afisate descrescator dupa numarul de melodii.

Am folosit **ROUND(AVG(p.price), 2)** pentru a afisa media pretului rotunjita la 2 zecimale. In ultimele randuri, folosim o subcerere care intoarce o tabela cu numarul de albume de pe fiecare album. Acestei noi tabele ii aplicam functia **AVG(numar_melodii)** pentru a afla numarul mediu de melodii al albumelor. Acest rezultat este comparat in clauza **HAVING** cu **COUNT(s.id_song)**, conform cerintei.

```
SELECT g.nume_gen,
       COUNT(DISTINCT a.id_produs) AS numar_albume,
       COUNT(mel.id_melodie) AS numar_melodii,
       ROUND(AVG(p.pret), 2) AS pret_mediul
FROM GEN g
JOIN ALBUM a ON g.id_gen = a.id_gen
JOIN MELODIE mel ON mel.id_produs = a.id_produs
JOIN PRODUS p ON a.id_produs = p.id_produs
GROUP BY g.nume_gen
HAVING COUNT(mel.id_melodie) > (
    SELECT AVG(numar_melodii_per_album)
    FROM (
        SELECT COUNT(id_melodie) AS numar_melodii_per_album
        FROM MELODIE
        GROUP BY id_produs
    )
)
ORDER BY numar_melodii DESC;
```

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, there are tabs for 'Gestiunea_Unui_Records_Store', 'dear.sql', 'create_tables.sql', 'insert_in_tables.sql', 'queries.sql' (which is currently selected), and 'update_delete.sql'. Below the tabs is a toolbar with icons for running queries, saving, and zooming.

The main area has two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab contains the following SQL query:

```

--4
SELECT g.num_gen,
       COUNT(DISTINCT a.id_produs) AS numar_albume,
       COUNT(mel.id_melodie) AS numar_melodii,
       ROUND(AVG(p.pret), 2) AS pret_mediul
FROM GEN g
JOIN ALBUM a ON g.id_gen = a.id_gen |
JOIN MELODIE mel ON mel.id_produs = a.id_produs
JOIN PRODUS p ON a.id_produs = p.id_produs
GROUP BY g.num_gen
HAVING COUNT(mel.id_melodie) > (
    SELECT AVG(numar_melodii) OVER()
)

```

Below the query is a note: 'SQL | All Rows Fetched: 5 in 0.271 seconds'.

The 'Script Output' tab displays the results of the query in a grid format:

| NUM_GEN | NUMAR_ALBUME | NUMAR_MELODII | PRET_MEDIU |
|-----------|--------------|---------------|------------|
| 1 Rap | 2 | 18 | 49.21 |
| 2 Rap-RNB | 1 | 15 | 29.99 |
| 3 RnB | 1 | 13 | 49.99 |
| 4 Pop | 1 | 11 | 39.99 |
| 5 Folk | 1 | 11 | 19.99 |

12.5 Cererea nr. 5 - NVL, DECODE, Sortari

Afisati profilul unui cumparator, care cuprinde numele complet, orasul acestuia, numarul de achizitii, numarul de produse cumparate, suma totala cheltuita, data ultimei achizitii, si catalogatii in functie de numarul de albume cumparate(Nu merchandise!) astfel: 0: New Here, mai mare de 3: Music Lover, restul: Casual Listener. Soratati descrescator in functie de numarul de achizitii, in caz de egalitate, sortati crescator dupa numarul de bani cheltuiti.

Pentru a crea un profil ce contine numele complet pe o coloana, am concatenat `last_name` cu `first_name` folosindu-ma de "|||". Vrem sa calculam suma totala cheltuita pe site, iar pentru a evita cazurile NULL, folosim `NVL(SUM(p.quantity * pur.price), 0)`, care inlocuieste NULL cu 0. Asemanator, in cazul in care cumparatorul nu are nicio achizitie, `NVL(MAX(TO_CHAR(p.purchase_date, 'YYYY-MM-DD')), 'Never purchased')`, inlocuieste o data eronata cu "Never purchased". `DECODE` ne ajuta sa asociem un "customer_type" cumparatorului in functie de numarul de albume cumparate. Gruparea se face pe fiecare client, dupa cele 3 coloane care ne intereseaza din customer, (nume + prenume + oras), ca sa putem calcula totaluri individuale. Conform enuntului, sortarea se face dupa 2 criterii, folosind doi parametrii in clauza `ORDER BY`.

```

SELECT
    c.nume_familie || ', ' || c.prenume AS nume_client,
    c.oras,
    COUNT(ach.id_achizitie) AS numar_achizitii,
    SUM(ach.cantitate) AS total_articole,
    NVL(SUM(ach.cantitate * prod.pret), 0) AS total_cheltuit,
    NVL(MAX(TO_CHAR(ach.data_achizitie, 'YYYY-MM-DD')), 'Nicio achizitie') AS ultima_achizitie,
    DECODE(COUNT(DISTINCT alb.id_produs), 0, 'Client nou',
           CASE WHEN COUNT(DISTINCT alb.id_produs) > 2 THEN 'Meloman'
                 ELSE 'Ascultator ocazional' END) AS tip_client
FROM CUMPARATOR c
LEFT JOIN ACHIZITIE ach ON c.id_cumparator = ach.id_cumparator
LEFT JOIN PRODUS prod ON ach.id_produs = prod.id_produs

```

```
LEFT JOIN ALBUM alb ON prod.id_produs = alb.id_produs
GROUP BY c.nume_familie, c.prenume, c.oras
ORDER BY numar_achizitii DESC, total_cheltuit ASC;
```

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, there are tabs for 'Gestiunea_Unui_Records_Store', 'clear.sql', 'create_tables.sql', 'insert_in_tables.sql', 'queries.sql' (which is currently selected), and 'update_delete.sql'. Below the tabs is a toolbar with various icons. The main area has two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab contains the following SQL code:

```
--5
SELECT
    c.nume_familie || ' ' || c.prenume AS nume_client,
    c.oras,
    COUNT(ach.id_achizitie) AS numar_achizitii,
    SUM(ach.cantitate) AS total_articole,
    NVL(SUM(ach.cantitate * prod.pret), 0) AS total_cheltuit,
    NVL(MAX(TO_CHAR(ach.data_achizitie, 'YYYY-MM-DD')), 'Nicio achizitie') AS ultima_achizitie,
    DECODE(COUNT(DISTINCT alb.id_produs), 0, 'Client nou',
           CASE WHEN COUNT(DISTINCT alb.id_produs) > 2 THEN 'Meloman'
                 ELSE 'Asculatator ocazional' END) AS tip_client
FROM CUMPARATOR c
LEFT JOIN ACHIZITIE ach ON c.id_cumparator = ach.id_cumparator
LEFT JOIN PRODUS prod ON ach.id_produs = prod.id_produs
```

Below the code, the 'Script Output' tab is active, showing the results of the query:

| NUME_CLIENT | ORAS | NUMAR_ACHIZITII | TOTAL_ARTICOLE | TOTAL_CHELTUIT | ULTIMA_ACHIZITIE | TIPI_CLIENT |
|-------------------------|-------------|-----------------|----------------|----------------|------------------|-----------------------|
| 1 Dumitrescu, Alexandru | București | 2 | 2 | 69.98 | 2023-05-15 | Asculatator ocazional |
| 2 Radu, Ioana | Cluj-Napoca | 2 | 2 | 104.98 | 2023-05-20 | Asculatator ocazional |
| 3 Munteanu, Diana | Timișoara | 2 | 2 | 104.98 | 2023-06-25 | Asculatator ocazional |
| 4 Stoica, Gabriel | Iași | 2 | 3 | 149.97 | 2023-05-25 | Asculatator ocazional |
| 5 Manole, Cristina | Sibiu | 1 | 1 | 37.99 | 2023-06-18 | Asculatator ocazional |
| 6 Vasile, Andreea | Constanta | 1 | 1 | 52.99 | 2023-06-02 | Asculatator ocazional |
| 7 Preda, Robert | Craiova | 1 | 2 | 95.98 | 2023-06-10 | Asculatator ocazional |
| 8 Dinu, Bogdan | Brașov | 1 | 3 | 134.97 | 2023-05-10 | Client nou |

12.6 Cererea nr. 6 - Clauza WITH

Afișați numele complet al angajatilor care au procesat comenzi, denumirea funcției lor, valoarea totală a comenzilor procesate, categoria de activitate în funcție de valoarea vânzărilor și luna în care au fost angajați(Mai mare de 500: Activ, Mai mare de 1000: Performant, altfel: Slab activ). Afisati si salariul total primit de la firma de la angajarea acestuia. Ordonati descrescător după valoarea vânzărilor și limitati rezultatul la primii 15 angajați.

Clauza **WITH** ne realizeaza un tabel temporar cu valoarea totala a produselor din comenziile aflate sub evidenta fiecarui angajat pentru a fi accesate anterior. Salariul angajatului de cand este angajat in firma este calculat cu diferite functii de calcul cu date precum **MONTHS_BETWEEN**(calculeaza numarul de luni dintre 2 date) si **SYSDATE**(returneaza data actuala la runtime). Subcererea din **WHERE** verifica daca angajatul luat in calcul, s-a ocupat de cel putin o comanda.

```
WITH total_vanzari_per_angajat AS (
    SELECT
        ach.id_angajat,
        SUM(prod.pret * ach.cantitate) AS valoare_totala
    FROM ACHIZITIE ach
    JOIN PRODUS prod ON ach.id_produs = prod.id_produs
    GROUP BY ach.id_angajat
)
SELECT *
FROM (
    SELECT
        ang.prenume || ' ' || NVL(ang.nume_familie, 'anonim') AS nume_complet,
        f.nume_functie,
```

```

ROUND(tva.valoare_totala, 2) AS valoare_totala_vanzari,
CASE
    WHEN tva.valoare_totala > 1000 THEN 'Performant'
    WHEN tva.valoare_totala > 500 THEN 'Activ'
    ELSE 'Slab activ'
END AS categorie_angajat,
TO_CHAR(ang.data_angajare, 'Mon YYYY', 'NLS_DATE_LANGUAGE=Romanian') AS angajat_din_luna,
ROUND(ang.salariu * MONTHS_BETWEEN(SYSDATE, ang.data_angajare), 2) AS salariu_estimat_total
FROM ANGAJAT ang
JOIN FUNCTIE f ON ang.id_functie = f.id_functie
JOIN total_vanzari_per_angajat tva ON ang.id_angajat = tva.id_angajat
WHERE EXISTS (
    SELECT 1
    FROM ACHIZITIE ach_check
    WHERE ach_check.id_angajat = ang.id_angajat
)
ORDER BY tva.valoare_totala DESC
)
WHERE ROWNUM <= 15;

```

The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, there are tabs for 'Gestiunea_Unui_Records_Store' and several other scripts like 'clear.sql', 'create_tables.sql', etc. Below the tabs is a toolbar with icons for running queries, saving, and zooming.

The main area is divided into two panes. The left pane is a 'Worksheet' where the SQL query is typed. The right pane is a 'Query Builder' which visualizes the query structure. The query itself is as follows:

```

LEFT JOIN ACHIZITIE ach ON c.id_cumparator = ach.id_cumparator
LEFT JOIN PRODUS prod ON ach.id_produs = prod.id_produs
LEFT JOIN ALBUM alb ON prod.id_produs = alb.id_produs
GROUP BY c.nume_familie, c.prenume, c.oras
ORDER BY numar_achizitii DESC, total_cheltuit ASC;
---6
WITH total_vanzari_per_angajat AS (
    SELECT
        ach.id_angajat,
        SUM(prod.pret * ach.cantitate) AS valoare_totala
    FROM ACHIZITIE ach
    JOIN PRODUS prod ON ach.id_produs = prod.id_produs
    GROUP BY ach.id_angajat
)

```

Below the worksheet, there are tabs for 'Script Output', 'Query Result', 'Query Result 1', and 'Query Result 2'. The 'Query Result' tab is active, showing the following data:

| | NUME_COMPLET | NUME_FUNCTIE | VALOARE_TOTALA_VANZARI | CATEGORIE_ANGAJAT | ANGAJAT_DIN_LUNA | SALARIU_ESTIMAT_TOTAL |
|---|------------------|----------------------|------------------------|-------------------|------------------|-----------------------|
| 1 | Elena Popa | Cashier | 374.93 | Slab activ | Mai 2021 | 143497.98 |
| 2 | Andrei Ionescu | Sales Representative | 227.94 | Slab activ | Mar 2020 | 248879.02 |
| 3 | Cristina Stanciu | Store Assistant | 95.98 | Slab activ | Dec 2021 | 103452.61 |
| 4 | Mihai Georgescu | Marketing Specialist | 52.99 | Slab activ | Aug 2021 | 224969.74 |

13 Cereri de actualizare si suprimare a datelor

13.1 Cererea nr.1

Sa se mareasca pretul tuturor albumelor care au unul dintre genurile muzicale "Pop" cu 20%.

```

UPDATE PRODUS
SET pret = pret * 1.2
WHERE id_produs IN (
    SELECT alb.id_produs

```

```

FROM ALBUM alb
JOIN GEN g ON alb.id_gen = g.id_gen
WHERE UPPER(g.numar_gen) = 'POP'
);

```

The screenshot shows a SQL worksheet window with the following content:

```

--1
UPDATE PRODUS
SET pret = pret * 1.2
WHERE id_produs IN (
    SELECT alb.id_produs
    FROM ALBUM alb
    JOIN GEN g ON alb.id_gen = g.id_gen
    WHERE UPPER(g.numar_gen) = 'POP'
);
--2
UPDATE ANGAJAT
SET salariu = salariu * 0.75
WHERE id_functie IN (
    SELECT id_functie
    FROM FUNCTIE
    WHERE numar_functie = 'IT Support'
);

```

Script Output

1 row updated.

13.2 Cererea nr. 2

Sa se reduca salariul angajatilor din departamentul de "IT Support" cu 20%.

```

UPDATE ANGAJAT
SET salariu = salariu * 0.75
WHERE id_functie IN (
    SELECT id_functie
    FROM FUNCTIE
    WHERE numar_functie = 'IT Support'
);

```

```

...ge Gestiunea_Unui_Records_Store dear.sql create_tables.sql insert_in_tables.sql queries.sql update_delete.sql
SQL Worksheet History
Worksheet Query Builder
JOIN GEN g ON alb.id_gen = g.id_gen
WHERE UPPER(g.numere_gen) = 'POP'
);
--2
UPDATE ANGAJAT
SET salariu = salariu * 0.75
WHERE id_functie IN (
    SELECT id_functie
    FROM FUNCTIE
    WHERE nume_functie = 'IT Support'
);
--3
--a)
UPDATE STOC s

```

Script Output | Task completed in 0.117 seconds

```

1 row updated.

1 row updated.

```

13.3 Cererea nr. 3

Sa se reactualizeze stocurile de produse in functie comenziile prezente in 'Pending', urmand ca statusul acestora sa fie schimbat in 'Completed'. In cazul in care un stoc are 0 produse(Este epuizat), stergeti definitiv stocul respectiv.

Subcererea din **SET** ne calculeaza numarul de produse vandute aflate doar in comenziile ce asteapta sa fie trimise ("Pending") si actualizeaza doar un singur stoc de acest produs.In a treia subcerere, folosind **SELECT MIN(s2.id_stoc)** si metoda in care sunt adaugate date in tabele cu sechete, ne asiguram ca scadem cantitatea din cel mai vechi stoc de acel produs, pentru a respecta principiul **first in first out**. Ulterior, toate comenziile "Pending" sunt transformate in "Completed", iar in cazul in care un stoc este epuizat complet si nu mai este nevoie de el, este sters cu total din tabela **STOCK**.

```

--a)
UPDATE STOC s
SET s.cantitate = s.cantitate - (
    SELECT SUM(ach.cantitate)
    FROM ACHIZITIE ach
    WHERE ach.status = 'Pending'
    AND ach.id_produs = s.id_produs
)
WHERE EXISTS (
    SELECT 1
    FROM ACHIZITIE ach
    WHERE ach.status = 'Pending'
    AND ach.id_produs = s.id_produs
)
AND s.id_stoc = (
    SELECT MIN(s2.id_stoc)
    FROM STOC s2
    WHERE s2.id_produs = s.id_produs
)

```

```
);  
UPDATE ACHIZITIE  
SET status = 'Completed'  
WHERE status = 'Pending';  
--b)  
DELETE FROM STOC  
WHERE cantitate <= 0;
```

The screenshot shows a SQL Worksheet interface with two main panes: 'Worksheet' and 'Script Output'.

Worksheet: Displays the following SQL code:

```
--3  
--a)  
UPDATE STOC s  
SET s.cantitate = s.cantitate - (  
    SELECT SUM(ach.cantitate)  
    FROM ACHIZITIE ach  
    WHERE ach.status = 'Pending'  
    AND ach.id_produs = s.id_produs  
)  
WHERE EXISTS (  
    SELECT 1  
    FROM ACHIZITIE ach  
    WHERE ach.status = 'Pending'  
    AND ach.id_produs = s.id_produs
```

Script Output: Displays the results of the executed query:

```
1 row updated.  
  
0 rows updated.  
  
0 rows updated.  
  
0 rows deleted.
```