

# Gestiunea unui Magazin de Muzică

Facultatea de Matematica si Informatica

Baze de Date

Grupa: 141

Rachieru Gheorghe Gabriel

## Contents

<b>1</b>	<b>Descrierea modelului real</b>	<b>2</b>
1.1	Utilitatea sistemului de baza de date . . . . .	2
<b>2</b>	<b>Prezentarea constrângerilor</b>	<b>2</b>
<b>3</b>	<b>Descrierea entităților</b>	<b>3</b>
<b>4</b>	<b>Descrierea relațiilor</b>	<b>3</b>
4.1	Relatiile de tip One-To-Many si Many-To-Many . . . . .	3
4.2	Relatia de tip 3 . . . . .	3
<b>5</b>	<b>Descrierea atributelor</b>	<b>4</b>
<b>6</b>	<b>Realizarea diagramei entitate-relație</b>	<b>10</b>
<b>7</b>	<b>Realizarea diagramei conceptuale</b>	<b>11</b>
<b>8</b>	<b>Scheme relaționale</b>	<b>11</b>
<b>9</b>	<b>Realizarea normalizarii FN1-FN3</b>	<b>12</b>
9.1	Forma normala 1 - FN1 . . . . .	12
9.2	Forma normala 2 - FN2 . . . . .	12
9.3	Forma normala 3 - FN3 . . . . .	13
<b>10</b>	<b>Crearea tabelelor in SQL si inserarea de date</b>	<b>14</b>
10.1	RECORD LABEL . . . . .	14
10.2	PRODUCT . . . . .	14
10.3	ALBUM . . . . .	15
10.4	MERCHANDISE . . . . .	15
10.5	GENRE . . . . .	15
10.6	ARTIST . . . . .	16
10.7	PROVIDER . . . . .	16
10.8	MEDIA FORMAT . . . . .	17
10.9	FUNCTION . . . . .	17
10.10	EMPLOYEE . . . . .	17
10.11	CUSTOMER . . . . .	18
10.12	STOCK . . . . .	19
10.13	PURCHASE . . . . .	19
10.14	SONG . . . . .	20
<b>11</b>	<b>Cereri SQL</b>	<b>22</b>
11.1	Cererea nr. 1 - Cererea din feedback . . . . .	22
11.2	Cererea nr. 2 - Cereri sincronizate cu 3 tabele . . . . .	23
11.3	Cererea nr. 3 - Subcereri nesincronizare in FROM . . . . .	24
11.4	Cererea nr.4 - Grupari cu subcereri in HAVING . . . . .	25
11.5	Cererea nr. 5 - NVL, DECODE, Sortari . . . . .	26
11.6	Cererea nr. 6 - Clauza WITH . . . . .	27
<b>12</b>	<b>Cereri de actualizare si suprimare a datelor</b>	<b>28</b>
12.1	Cererea nr.1 . . . . .	28
12.2	Cererea nr. 2 . . . . .	29
12.3	Cererea nr. 3 . . . . .	30

# 1 Descrierea modelului real

Pe zi ce trece, toate afacerile, activitățile și acțiunile iau amploare. 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, comenzile plasate de aceștia, produsele achiziționate de clienți, locația fizică a produselor în magazin, detalii despre angajații fiecărei 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 clienților.

# 2 Prezentarea constrângerilor

- Comenzile pot fi procesate între orele 08.00-22.00 de luni până sâmbătă.
- Clienții primesc confirmarea comenzii telefonic sau prin mail.
- Orice client care comandă de mai mult de 150 de lei nu plătește costul livrării.
- Un produs poate fi distribuit de maxim 5 furnizori.
- Prețului de bază al unui produs îi este adăugat "adaos" în funcție de formula media.
- O casa de discuri face mai multe albume sau niciunul.
- Un album are mai multe genuri muzicale si cel puțin unul.
- Un produs, fie el Album sau Merchandise, este asociat mai multor melodii si cel puțin una.
- Mai multi artisti pot inregistra o singura melodie.
- Un album este disponibil in mai multe formate media si cel puțin unul.
- Un produs apare in mai multe stocuri sau in niciunul.
- Un furnizor trimite mai multe stocuri, sau niciunul.
- Un Angajat are o singura functie si cel puțin una.
- Un Angajat se ocupa de o comanda sau de niciuna.
- Un Cumparator a efectuat mai multe comenzi, si cel puțin una.
- O achiziție are un singur produs, cantitatea acestuia, clientul si angajatul care se ocupa de comanda.
- Un angajat se ocupa de mai multe comenzi sau de niciuna.
- Un client are mai multe achizitii, sau niciuna, fiind inscris in baza de date fara a cumpara ceva anterior.
- Un produs poate aparea in mai multe achizitii, sau in niciuna.

### 3 Descrierea entităților

- Pentru fiecare casa de discuri se cunosc id-ul casei, numele casei si adresa
- Pentru fiecare produs de tip Merchandise se cunoaste id-ul, tipul, numele, si optional marimea.
- Pentru fiecare album se cunosc id-ul albumului, denumirea albumului, data aparitiei, lungimea și pretul de baza.
- 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	Recording Label-Album: One to Many	-
Exista in	Album-Media Format: One to Many	-
Are/Inregistreaza	Product-Artist: Many to Many	-
Distribuie	Provider-Product: Many to Many	-
Apartine	Product-Genre: One to Many	-
Are o	Employee-Function: One to Many	-

#### 4.2 Relatia de tip 3

**Relatia de tip 3** dintre entitatile **PRODUCT**, **CUSTOMER** si **EMPLOYEE** 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 procesează, 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 implicite	Observatii
id_product	NUMBER(3,0)	PK, FK	101, 102, 103	-	Identificator unic
album_name	VARCHAR2(100)	-	GNX, SOS, Swimming	-	Numele albumului ca text
length	CHAR(5)	-	45:30, 38:22, 72:15	-	Format MM:SS
id_record_label	NUMBER(3,0)	FK	1, 2, 3	-	Legat de casa de discuri

### 2. MERCHANDISE

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

### 3. PRODUCT

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori Im-plicite	Observatii
id_product	NUMBER(3,0)	PK	101, 201, 301	-	Cheia primară a produsului
price	NUMBER(6,2)	-	99.99, 149.99, 24.99	-	Preț de bază în RON
release_date	DATE	-	2024-02-05, 2023-11-12, 2022-08-30	SYSDATE	Data lansării produsului

### 4. GENRE

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori Im-plicite	Observatii
id_genre	NUMBER(3,0)	PK	1, 2, 3, 4	-	Identificator unic
genre_name	VARCHAR2(100)	-	Rock, Pop, Jazz, Metal, Hip-Hop	-	Numele genului muzical
id_album	NUMBER(3,0)	FK	101, 102, 103	-	Legat de albumul asociat

**5. ARTIST**

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

**6. MEDIA FORMAT**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_media_format	NUMBER(3,0)	PK	1, 2, 3, 4	-	Identificator unic
media_format_name	VARCHAR2(100)	-	Vinyl, CD, Digital, Cassette	-	Numele formatului media
id_product	NUMBER(3,0)	FK	101, 102, 103	-	Legat de produsul asociat
procent_added	NUMBER(3,0)	-	15, 5, 0, 10	0	Procentul adăugat la preț

**7. PROVIDER**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_provider	NUMBER(3,0)	PK	1, 2, 3	-	Identificator unic
provider_name	VARCHAR2(100)	-	Music Warehouse, Vinyl Plus, Global Distribution	-	Numele furnizorului
phone_number	CHAR(10)	-	0712345678, 0723456789, 0734567890	-	Format românesc
adresa	VARCHAR2(100)	-	Str. Muzicii 10, Bd. Distribuției 25, Alea Furnizorilor 5	-	Adresa furnizorului

**8. FUNCTION**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_function	NUMBER(3,0)	PK	1, 2, 3, 4	-	Identificator unic
function_name	VARCHAR2(100)	-	Manager, Sales Associate, Cashier, Inventory Specialist	-	Numele funcției
min_salary	NUMBER(6,2)	-	2500.00, 3000.00, 4000.00	-	Salariul minim în RON
max_salary	NUMBER(6,2)	-	5000.00, 7000.00, 10000.00	-	Salariul maxim în RON

**9. EMPLOYEE**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_employee	NUMBER(3,0)	PK	1, 2, 3, 4	-	Identificator unic
id_function	NUMBER(3,0)	FK	1, 2, 3, 4	-	Legat de funcția angajatului
last_name	VARCHAR2(100)	-	Ionescu, Popescu, Popa, Smith	-	Numele de familie
first_name	VARCHAR2(100)	-	Ion, Maria, Ana, John	-	Prenumele angajatului
min_salary	NUMBER(6,2)	-	2500.00, 3000.00, 4000.00	-	Salariul minim pentru funcție
max_salary	NUMBER(6,2)	-	5000.00, 7000.00, 10000.00	-	Salariul maxim pentru funcție
phone	CHAR(11)	-	0712345678, 0723456789, 0734567890	-	Format românesc
hire_date	DATE	-	2020-05-15, 2022-03-10, 2023-08-22	SYSDATE	Data angajării
salary	NUMBER(6,2)	-	3500.00, 4200.00, 6000.00	-	Salariul curent

**10. CUSTOMER**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_customer	NUMBER(3,0)	PK	1, 2, 3, 4	-	Identificator unic
last_name	VARCHAR2(100)	-	Georgescu, Vasilescu, Brown, Johnson	-	Numele de familie
first_name	VARCHAR2(100)	-	Andrei, Elena, Robert, Sarah	-	Prenumele clientului
adresa	VARCHAR2(300)	-	Str. Primăverii 10, Bd. Tineretului 25, Main Street 42	-	Adresa completă
city	VARCHAR2(100)	-	București, Cluj-Napoca, Iași, London, New York	-	Orașul clientului

**11. RECORD LABEL**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_record_label	NUMBER(3,0)	PK	1, 2, 3, 4	-	Identificator unic
label_name	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. PURCHASE**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_purchase	NUMBER(3,0)	PK	1001, 1002, 1003	-	Identificator unic
id_customer	NUMBER(3,0)	FK	1, 2, 3, 4	-	Legat de clientul care cumpără
id_employee	NUMBER(3,0)	FK	1, 2, 3, 4	-	Legat de angajatul care procesează
id_product	NUMBER(3,0)	FK	101, 201, 301	-	Legat de produsul cumpărat
purchase_date	DATE	-	2024-03-15, 2024-02-28, 2024-01-10	SYSDATE	Data achiziției
payment_method	VARCHAR2(100)	-	Card, Cash, PayPal, Transfer bancar	'Card'	Metoda de plată
status	VARCHAR2(100)	-	Pending, Completed, Cancelled, Delivered	'Pending'	Statusul comenzii
quantity	NUMBER(3,0)	-	1, 2, 3, 5, 10	1	Cantitatea cumpărată

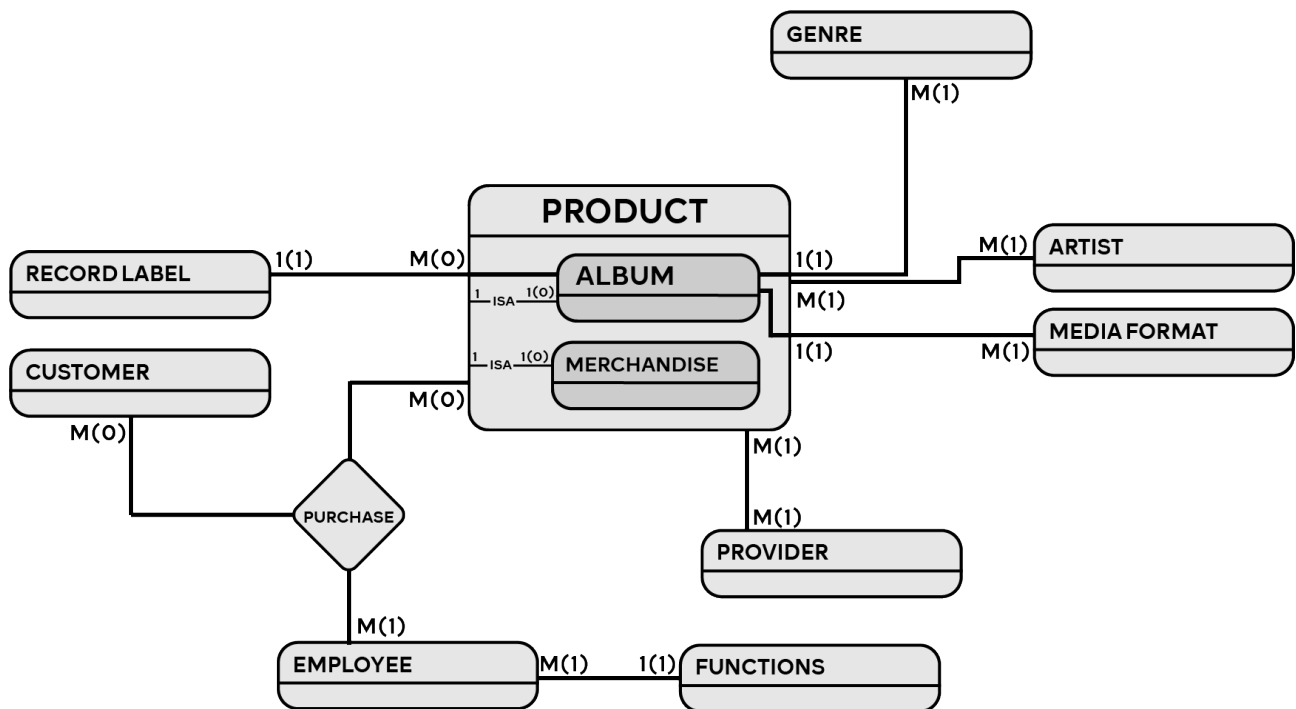
**13. STOCK**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori implicite	Observatii
id_stock	NUMBER(3,0)	PK	501, 502, 503	-	Identificator unic
id_product	NUMBER(3,0)	FK	101, 201, 301	-	Legat de produsul în stoc
id_provider	NUMBER(3,0)	FK	1, 2, 3	-	Legat de furnizorul produsului
quantity	NUMBER(3,0)	-	10, 25, 50, 100	0	Cantitatea disponibilă

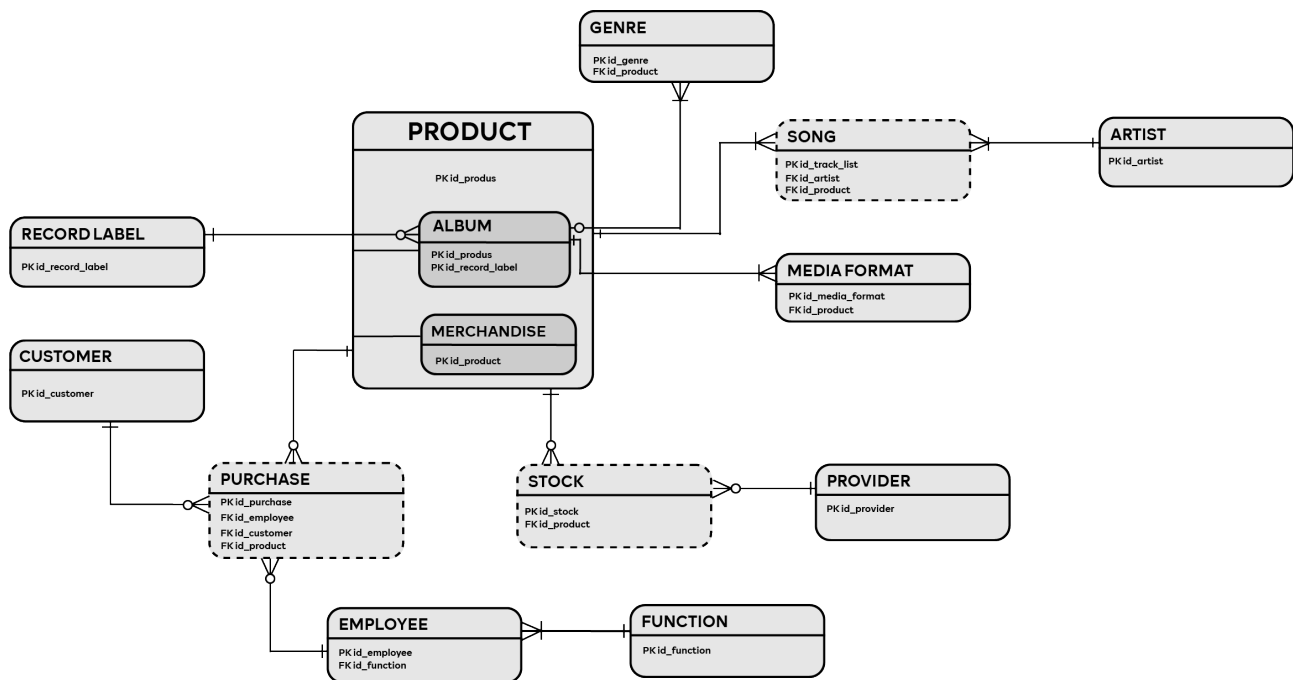
**14. SONG**

Atribut	Tip de Date	Constrangeri	Valori Posibile	Valori im-plicite	Obersvatii
id_song	NUMBER(3,0)	PK	601, 602, 603	-	Identificator unic
id_artist	NUMBER(3,0)	FK	401, 402, 403	-	Legat de artistul melodiei
id_product	NUMBER(3,0)	FK	101, 102, 103	-	Legat de produsul (albumul)
title	VARCHAR2(100)	-	Summer Vibes, Rock Forever, Soul of Mine	-	Titlul melodiei
length	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 relaționale

Schemele relaționale corespunzătoare diagramei conceptuale sunt următoarele:

ALBUM(id\_product#, length, id\_record\_label)

MERCHANDISE(id\_product#, merch\_type, merch\_size)

PRODUCT(id\_product#, price, release\_date)

GENRE(id\_genre#, genre\_name, id\_album)

ARTIST(id\_artist#, first\_name, last\_name, band)

MEDIA\_FORMAT(id\_media\_format#, media\_format\_name, id\_product, procent\_added)

PROVIDER(id\_provider#, provider\_name, phone\_number, adress)

FUNCTION(id\_function#, function\_name, min\_salary, max\_salary)

EMPLOYEE(id\_employee#, id\_function, last\_name, first\_name, phone, hire\_date, salary)

CUSTOMER(id\_customer#, last\_name, first\_name, adress, city)

RECORD LABEL(id\_record\_label#, label\_name, email)

PURCHASE(id\_purchase#, id\_customer, id\_employee, id\_product, purchase\_date, payment\_method, status, quantity)

STOCK(id\_stock#, id\_product, id\_provider, quantity)

SONG(id\_song#, id\_artist, id\_product, title, length)

## 9 Realizarea normalizării FN1-FN3

### 9.1 Forma normala 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	SONG
Artist1	SG1, SG2, SG3
Artist2	SG2, SG4
Artist3	SG3

Table 1: Exemplu non-FN1.

ARTIST	SONG
Artist1	SG1
Artist1	SG2
Artist1	SG3
Artist2	SG2
Artist2	SG4
Artist3	SG3

Table 2: Exemplu FN1.

### 9.2 Forma normala 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 doua: relațiile sunt în FN1, iar toate atributele 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#	first_name	id_product#	price
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 ca avem următoarele dependente:

{id\_artist#} - first\_name. id\_artist determina functional numele.

{id\_product#} - price

id_artist#	id_product#	price
A1	P1	6.99
A2	P1	6.99
A1	P2	15.99
A3	P3	10.49

Table 4: Exemplu FN2.

id_artist#	first_name
A1	Kendrick
A2	West
A3	Miller

Table 5: Exemplu FN2.

### 9.3 Forma normala 3 - FN3

Asemănător, o relație se află în forma normală 3 (FN3) doar dacă este deja în forma normală 2 (FN2) și fiecare atribut care nu face parte din cheia primară este dependent direct de cheia primară, fără să existe dependențe tranzitive.

Cu alte cuvinte, FN3 cere ca orice atribut non-cheie să depindă doar de cheia primară, de întreaga cheie și exclusiv de aceasta, fără intermediari prin alte atribute.

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

Pentru a ilustra această normalizare, vom analiza exemplul următor:

id_artist#	id_product#	price
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 atributele non-cheie să depindă direct și exclusiv de cheia primară.

R1(id\_product#, price)

R2(id\_artist#, id\_product#)

id_product#	price
P1	6.99
P2	15.99
P3	10.49

Table 7: Exemplu FN3.

id_artist#	id_product#
A1	P1
A2	P1
A1	P2
A3	P3

Table 8: Exemplu FN3.

## 10 Crearea tabelelor in SQL si inserarea de date

### 10.1 RECORD LABEL

```
CREATE TABLE RECORD_LABEL (  
    id_record_label NUMBER(3,0) CONSTRAINT pk_record_label PRIMARY KEY,  
    label_name VARCHAR2(100),  
    email VARCHAR2(100)  
);  
  
CREATE SEQUENCE SEQ_RECORD_LABEL  
    INCREMENT BY 1  
    START WITH 1  
    MAXVALUE 1000  
    NOCYCLE;  
  
INSERT INTO RECORD_LABEL VALUES (SEQ_RECORD_LABEL.NEXTVAL, 'Universal Music',  
'contact@universalmusic.com');  
INSERT INTO RECORD_LABEL VALUES (SEQ_RECORD_LABEL.NEXTVAL, 'Sony Music',  
'info@sonymusic.com');  
INSERT INTO RECORD_LABEL VALUES (SEQ_RECORD_LABEL.NEXTVAL, 'Republic Records',  
'office@republicrecords.com');  
INSERT INTO RECORD_LABEL VALUES (SEQ_RECORD_LABEL.NEXTVAL, 'Atlantic Records',  
'info@atlanticrecords.com');  
INSERT INTO RECORD_LABEL VALUES (SEQ_RECORD_LABEL.NEXTVAL, 'Electrecord',  
'contact@electrecord.ro');  
  
COMMIT;
```

### 10.2 PRODUCT

```
CREATE TABLE PRODUCT (  
    id_product NUMBER(3,0) CONSTRAINT pk_product PRIMARY KEY,  
    price NUMBER(6,2),  
    release_date DATE DEFAULT SYSDATE  
);  
  
CREATE SEQUENCE SEQ_PRODUCT  
    INCREMENT BY 1  
    START WITH 1  
    MAXVALUE 1000  
    NOCYCLE;  
  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 49.99, TO_DATE('15/03/2022', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 49.99, TO_DATE('15/03/2022', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 39.99, TO_DATE('22/05/2022', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 29.99, TO_DATE('10/07/2022', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 19.99, TO_DATE('30/09/2022', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 59.99, TO_DATE('12/12/2022', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 34.99, TO_DATE('25/01/2023', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 44.99, TO_DATE('18/03/2023', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 24.99, TO_DATE('05/05/2023', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 64.99, TO_DATE('28/06/2023', 'DD/MM/YYYY'));  
INSERT INTO PRODUCT VALUES (SEQ_PRODUCT.NEXTVAL, 54.99, TO_DATE('15/08/2023', 'DD/MM/YYYY'));  
  
COMMIT;
```

### 10.3 ALBUM

```
CREATE TABLE ALBUM (  
    id_product NUMBER(3,0) CONSTRAINT pk_album PRIMARY KEY,  
    album_name VARCHAR2(100), -- Adăugat numele albumului conform documentației  
    length CHAR(5), -- Modificat tipul de date conform documentației  
    id_record_label NUMBER(3,0),  
    CONSTRAINT fk_album_product FOREIGN KEY (id_product) REFERENCES PRODUCT(id_product),  
    CONSTRAINT fk_album_label FOREIGN KEY (id_record_label) REFERENCES RECORD_LABEL(id_record_label)  
);  
  
INSERT INTO ALBUM VALUES (1, 'Swimming', '49:07', 1);  
INSERT INTO ALBUM VALUES (2, 'GNX', '42:11', 2);  
INSERT INTO ALBUM VALUES (3, 'Short n Sweet', '42:19', 3);  
INSERT INTO ALBUM VALUES (4, 'HUMBLE', '43:00', 4);  
INSERT INTO ALBUM VALUES (5, 'folklore', '45:30', 5);  
INSERT INTO ALBUM VALUES (6, 'the album', '30:45', 5);  
  
COMMIT;
```

### 10.4 MERCHANDISE

```
CREATE TABLE MERCHANDISE (  
    id_product NUMBER(3,0) CONSTRAINT pk_merchandise PRIMARY KEY,  
    merch_type VARCHAR2(100),  
    merch_size VARCHAR2(4),  
    CONSTRAINT fk_merchandise_product FOREIGN KEY (id_product) REFERENCES PRODUCT(id_product)  
);  
  
INSERT INTO MERCHANDISE VALUES (6, 'T-Shirt', 'M');  
INSERT INTO MERCHANDISE VALUES (7, 'Poster', NULL);  
INSERT INTO MERCHANDISE VALUES (8, 'Cap', 'S');  
INSERT INTO MERCHANDISE VALUES (9, 'Mug', NULL);  
INSERT INTO MERCHANDISE VALUES (10, 'Hoodie', 'XL');  
  
COMMIT;
```

### 10.5 GENRE

```
CREATE TABLE GENRE (  
    id_genre NUMBER(3,0) CONSTRAINT pk_genre PRIMARY KEY,  
    genre_name VARCHAR2(100),  
    id_album NUMBER(3,0),  
    CONSTRAINT fk_genre_album FOREIGN KEY (id_album) REFERENCES ALBUM(id_product)  
);  
  
CREATE SEQUENCE SEQ_GENRE  
    INCREMENT BY 1  
    START WITH 1  
    MAXVALUE 1000  
    NOCYCLE;  
  
INSERT INTO GENRE VALUES (SEQ_GENRE.NEXTVAL, 'RNB', 1);  
INSERT INTO GENRE VALUES (SEQ_GENRE.NEXTVAL, 'Rap', 2);  
INSERT INTO GENRE VALUES (SEQ_GENRE.NEXTVAL, 'Pop', 3);  
INSERT INTO GENRE VALUES (SEQ_GENRE.NEXTVAL, 'Rap-RNB', 4);  
INSERT INTO GENRE VALUES (SEQ_GENRE.NEXTVAL, 'Folk', 5);  
  
COMMIT;
```



## 10.6 ARTIST

```
CREATE TABLE ARTIST (  
    id_artist NUMBER(3,0) CONSTRAINT pk_artist PRIMARY KEY,  
    first_name VARCHAR2(100),  
    last_name VARCHAR2(100),  
    band VARCHAR2(100)  
);  
  
CREATE SEQUENCE SEQ_ARTIST  
    INCREMENT BY 1  
    START WITH 1  
    MAXVALUE 1000  
    NOCYCLE;  
  
INSERT INTO ARTIST VALUES (1, 'Mac', 'Miller', NULL);  
INSERT INTO ARTIST VALUES (2, 'Kendrick', 'Lamar', NULL);  
INSERT INTO ARTIST VALUES (3, 'Sabrina', 'Carpenter', NULL);  
INSERT INTO ARTIST VALUES (4, 'Taylor', 'Swift', NULL);  
INSERT INTO ARTIST VALUES (5, 'boygenius', 'boygenius', 'band');  
INSERT INTO ARTIST VALUES (6, 'Rihanna', NULL, NULL);  
INSERT INTO ARTIST VALUES (7, 'Frank', 'Ocean', NULL);  
INSERT INTO ARTIST VALUES (8, 'Billie', 'Eilish', NULL);  
INSERT INTO ARTIST VALUES (9, 'The Weeknd', NULL, NULL);  
INSERT INTO ARTIST VALUES (10, 'SZA', NULL, NULL);  
  
COMMIT;
```

## 10.7 PROVIDER

```
CREATE TABLE PROVIDER (  
    id_provider NUMBER(3,0) CONSTRAINT pk_provider PRIMARY KEY,  
    provider_name VARCHAR2(100),  
    phone_number CHAR(11),  
    address VARCHAR2(100)  
);  
  
CREATE SEQUENCE SEQ_PROVIDER  
    INCREMENT BY 1  
    START WITH 1  
    MAXVALUE 1000  
    NOCYCLE;  
  
INSERT INTO PROVIDER VALUES (SEQ_PROVIDER.NEXTVAL, 'Music Distribution SRL', '07456123890',  
'Str. Muzicii nr. 15, București');  
INSERT INTO PROVIDER VALUES (SEQ_PROVIDER.NEXTVAL, 'Vinyl Factory', '07345678912',  
'Calea Victoriei nr. 78, București');  
INSERT INTO PROVIDER VALUES (SEQ_PROVIDER.NEXTVAL, 'Merch Production', '07234567891',  
'Bd. Unirii nr. 45, București');  
INSERT INTO PROVIDER VALUES (SEQ_PROVIDER.NEXTVAL, 'Global Logistics', '07123456789',  
'Str. Industriilor nr. 12, Cluj-Napoca');  
INSERT INTO PROVIDER VALUES (SEQ_PROVIDER.NEXTVAL, 'Record Distributor', '07987654321',  
'Str. Republicii nr. 36, Timișoara');  
INSERT INTO PROVIDER VALUES (SEQ_PROVIDER.NEXTVAL, 'Audio Tech SRL', '07765432109',  
'Str. Aviatiei nr. 24, București');  
INSERT INTO PROVIDER VALUES (SEQ_PROVIDER.NEXTVAL, 'Music Warehouse', '07654321098',  
'Calea Floreasca nr. 55, București');  
  
COMMIT;
```

## 10.8 MEDIA FORMAT

```
CREATE TABLE MEDIA_FORMAT (  
    id_media_format NUMBER(3,0) CONSTRAINT pk_media_format PRIMARY KEY,  
    media_format_name VARCHAR2(100),  
    id_product NUMBER(3,0),  
    procent_added NUMBER(3,0) DEFAULT 0,  
    CONSTRAINT fk_media_format_product FOREIGN KEY (id_product) REFERENCES PRODUCT(id_product)  
);
```

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

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

```
COMMIT;
```

## 10.9 FUNCTION

```
CREATE TABLE FUNCTION (  
    id_function NUMBER(3,0) CONSTRAINT pk_function PRIMARY KEY,  
    function_name VARCHAR2(100),  
    min_salary NUMBER(6,2),  
    max_salary NUMBER(6,2)  
);
```

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

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

```
COMMIT;
```

## 10.10 EMPLOYEE

```
CREATE TABLE EMPLOYEE (  
    id_employee NUMBER(3,0) CONSTRAINT pk_employee PRIMARY KEY,  
    id_function NUMBER(3,0),  
    last_name VARCHAR2(100),  
    first_name VARCHAR2(100),  
    phone CHAR(11),  
    hire_date DATE DEFAULT SYSDATE,  
    salary NUMBER(6,2),
```

```
CONSTRAINT fk_employee_function FOREIGN KEY (id_function) REFERENCES FUNCTION(id_function)
);

CREATE SEQUENCE SEQ_EMPLOYEE
  INCREMENT BY 1
  START WITH 1
  MAXVALUE 1000
  NOCYCLE;

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

COMMIT;
```

## 10.11 CUSTOMER

```
CREATE TABLE CUSTOMER (
  id_customer NUMBER(3,0) CONSTRAINT pk_customer PRIMARY KEY,
  last_name VARCHAR2(100),
  first_name VARCHAR2(100),
  address VARCHAR2(300),
  city VARCHAR2(100)
);

CREATE SEQUENCE SEQ_CUSTOMER
  INCREMENT BY 1
  START WITH 1
  MAXVALUE 1000
  NOCYCLE;

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

```
COMMIT;
```

## 10.12 STOCK

```
CREATE TABLE STOCK (  
    id_stock NUMBER(3,0) CONSTRAINT pk_stock PRIMARY KEY,  
    id_product NUMBER(3,0),  
    id_provider NUMBER(3,0),  
    quantity NUMBER(3,0) DEFAULT 0,  
    CONSTRAINT fk_stock_product FOREIGN KEY (id_product) REFERENCES PRODUCT(id_product),  
    CONSTRAINT fk_stock_provider FOREIGN KEY (id_provider) REFERENCES PROVIDER(id_provider)  
);
```

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

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

```
COMMIT;
```

## 10.13 PURCHASE

```
CREATE TABLE PURCHASE (  
    id_purchase NUMBER(3,0) CONSTRAINT pk_purchase PRIMARY KEY,  
    id_customer NUMBER(3,0),  
    id_employee NUMBER(3,0),  
    id_product NUMBER(3,0),  
    purchase_date DATE DEFAULT SYSDATE,  
    payment_method VARCHAR2(100) DEFAULT 'Card',  
    status VARCHAR2(100) DEFAULT 'Pending',  
    quantity NUMBER(3,0) DEFAULT 1,  
    CONSTRAINT fk_purchase_customer FOREIGN KEY (id_customer) REFERENCES CUSTOMER(id_customer),  
    CONSTRAINT fk_purchase_employee FOREIGN KEY (id_employee) REFERENCES EMPLOYEE(id_employee),  
    CONSTRAINT fk_purchase_product FOREIGN KEY (id_product) REFERENCES PRODUCT(id_product)  
);
```

```
CREATE SEQUENCE SEQ_PURCHASE  
    INCREMENT BY 1  
    START WITH 1  
    MAXVALUE 1000
```

```

NOCYCLE;

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

COMMIT;

```

## 10.14 SONG

```

CREATE TABLE SONG (
    id_song NUMBER(3,0) CONSTRAINT pk_song PRIMARY KEY,
    id_artist NUMBER(3,0),
    id_product NUMBER(3,0),
    title VARCHAR2(100),
    length VARCHAR2(100),
    CONSTRAINT fk_song_artist FOREIGN KEY (id_artist) REFERENCES ARTIST(id_artist),
    CONSTRAINT fk_song_product FOREIGN KEY (id_product) REFERENCES PRODUCT(id_product)
);

```

```

CREATE SEQUENCE SEQ_SONG
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 1000
    NOCYCLE;

```

```

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Come Back to Earth', '2:41');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Hurt Feelings', '4:05');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'What''s the Use?', '4:48');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Perfecto', '3:35');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Self Care', '5:45');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Wings', '4:10');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Ladders', '4:47');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Small Worlds', '4:31');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Conversation Pt. 1', '3:30');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Dunno', '3:57');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'Jet Fuel', '5:45');

```

```
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, '2009', '5:47');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 1, 1, 'So It Goes', '5:12');

-- Inserare in album cu id_product = 2
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'waccded out murals', '5:17');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'squabble up', '2:37');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 10, 2, 'luther (with sza)', '2:57');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'luther (with sza)', '2:57'); --adaugat de 2 ori dar c
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'man at the garden', '3:53');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'hey now (feat. dody6)', '3:37');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'reincarnated', '4:35');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'tv off (feat. hefty gunplay)', '3:40');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'dodger blue (feat. wallie the...)', '2:11');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'peekaboo (feat. azchike)', '2:35');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 2, 'heart pt. 6', '4:52');

-- Inserare in album cu id_product = 3
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Taste', '4:54');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Please Please Please', '5:57');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Good Graces', '3:05');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Sharpest Tool', '3:38');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Coincidence', '2:44');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Bed Chem', '2:51');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Espresso', '2:55');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Dumb n Poetic', '2:13');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Slim Pickins', '2:32');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Juno', '3:43');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 3, 3, 'Lie to girls', '3:22');

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'BLOOD.', '1:58');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'DNA', '3:05');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'YAH.', '2:40');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'ELEMENT.', '3:28');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'FEEL.', '3:34');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'LOYALTY.', '3:47');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 6, 4, 'LOYALTY.', '3:47');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'PRIDE.', '4:35');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'HUMBLE', '6:22');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'LUST.', '5:07');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'LOVE.', '3:33');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'XXX.', '4:14');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'FEAR.', '7:40');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'GOD.', '4:08');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 4, 'DUCKWORTH.', '4:08');

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'cardigan', '3:59');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'august', '4:21');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'exile (featuring Bon Iver)', '4:45');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'my tears ricochet', '4:15');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'mirrorball', '3:29');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'seven', '3:28');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'this is me trying', '3:15');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'illicit affairs', '3:10');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'betty', '4:54');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'peace', '5:23');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 5, 'hoax', '3:40');

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 11, 'two', '3:12');
```

```
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 11, 'three', '2:57');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 11, 'four', '3:22');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 11, 'five', '4:01');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 11, 'six', '3:44');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 11, 'seven', '3:18');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 4, 11, 'eight', '2:55');

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 7, 12, 'Pink + White', '3:04');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 7, 12, 'Solo', '4:17');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 7, 12, 'Self Control', '4:09');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 7, 12, 'Nights', '5:07');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 7, 12, 'White Ferrari', '4:08');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 7, 12, 'Godspeed', '2:57');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 7, 12, 'Futura Free', '9:24');

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 13, 'BLOOD.', '1:58');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 13, 'DNA.', '3:05');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 13, 'ELEMENT.', '3:28');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 13, 'FEEL.', '3:34');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 13, 'HUMBLE.', '2:57');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 13, 'XXX.', '4:14');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 2, 13, 'FEAR.', '7:40');

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'After Hours', '6:01');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'Heartless', '3:18');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'In Your Eyes', '3:57');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'Too Late', '3:59');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'Hardest To Love', '3:31');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'Scared To Live', '3:11');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'Snowchild', '4:07');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'Escape From LA', '5:55');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 9, 15, 'Faith', '4:43');

INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'Therefore I Am', '2:54');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'Your Power', '4:05');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'Lost Cause', '3:32');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'my future', '3:30');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'Oxytocin', '3:30');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'GOLDWING', '2:31');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'Billie Bossa Nova', '3:16');
INSERT INTO SONG VALUES (SEQ_SONG.NEXTVAL, 8, 16, 'Getting Older', '4:04');
```

```
COMMIT;
```

## 11 Cereri SQL

### 11.1 Cererea nr. 1 - Cererea din feedback

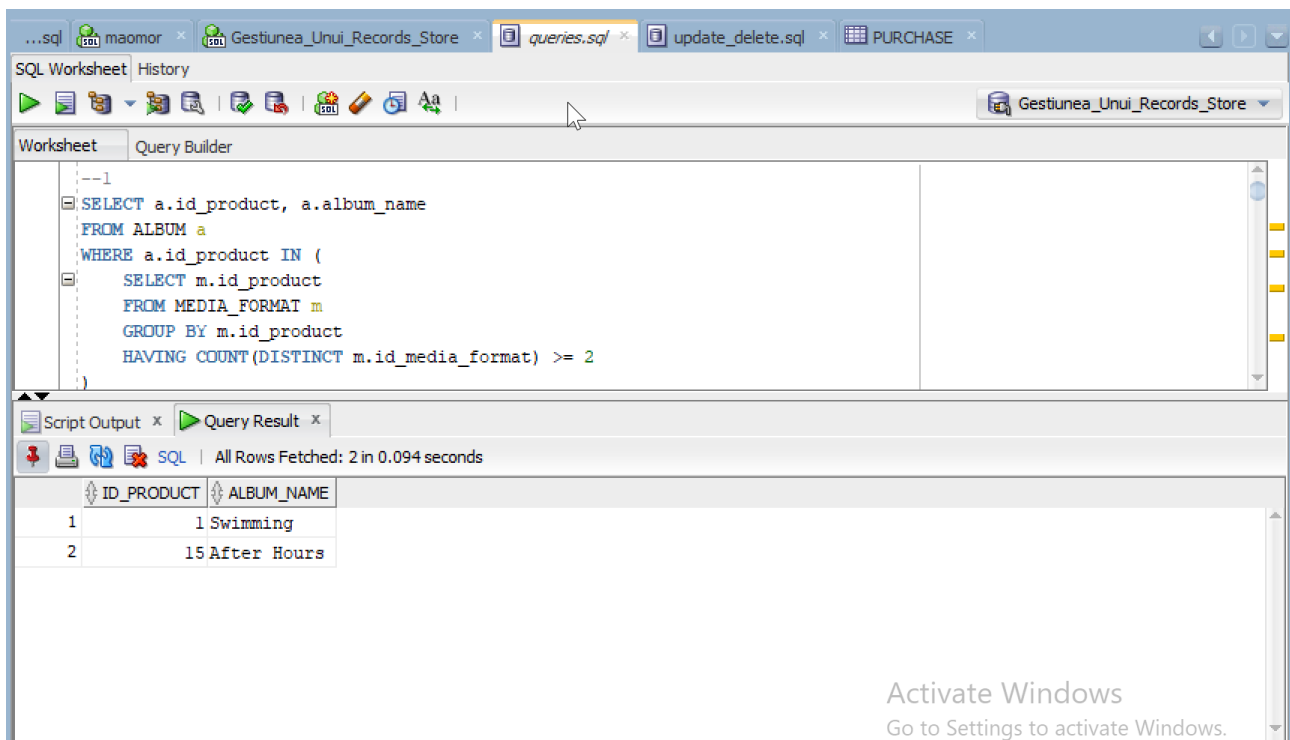
Care sunt albumele care se pot achizitiona in cel putin doua formate diferite, si care sunt aduse de cel putin doi furnizori.

```
SELECT a.id_product, a.album_name
FROM ALBUM a
WHERE a.id_product IN (
    SELECT m.id_product
```

```

        FROM MEDIA_FORMAT m
        GROUP BY m.id_product
        HAVING COUNT(DISTINCT m.id_media_format) >= 2
    )
    AND a.id_product IN (
        SELECT s.id_product
        FROM STOCK s
        GROUP BY s.id_product
        HAVING COUNT(DISTINCT s.id_provider) >= 2
    );

```



## 11.2 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.

```

SELECT DISTINCT a.album_name, p.price, r.label_name
FROM ALBUM a
JOIN PRODUCT p ON a.id_product = p.id_product
JOIN RECORD_LABEL r ON a.id_record_label = r.id_record_label
JOIN SONG s ON s.id_product = a.id_product
JOIN ARTIST ar ON s.id_artist = ar.id_artist
WHERE (ar.first_name LIKE 'K%' OR ar.last_name LIKE 'M%')
    AND p.price > (
        SELECT AVG(price)
        FROM PRODUCT
        WHERE id_product IN (SELECT id_product FROM ALBUM)
    );

```



The screenshot shows an SQL Worksheet window with the following components:

- Worksheet Tab:** Contains the following SQL query:
 

```
--2
SELECT DISTINCT a.album_name, p.price, r.label_name
FROM ALBUM a
JOIN PRODUCT p ON a.id_product = p.id_product
JOIN RECORD_LABEL r ON a.id_record_label = r.id_record_label
JOIN SONG s ON s.id_product = a.id_product
JOIN ARTIST ar ON s.id_artist = ar.id_artist
WHERE (ar.first_name LIKE 'K%' OR ar.last_name LIKE 'M%')
AND p.price > (
```
- Script Output Tab:** Shows the execution status: "All Rows Fetched: 3 in 0.096 seconds".
- Query Result 1 Tab:** Displays the results in a table:
 

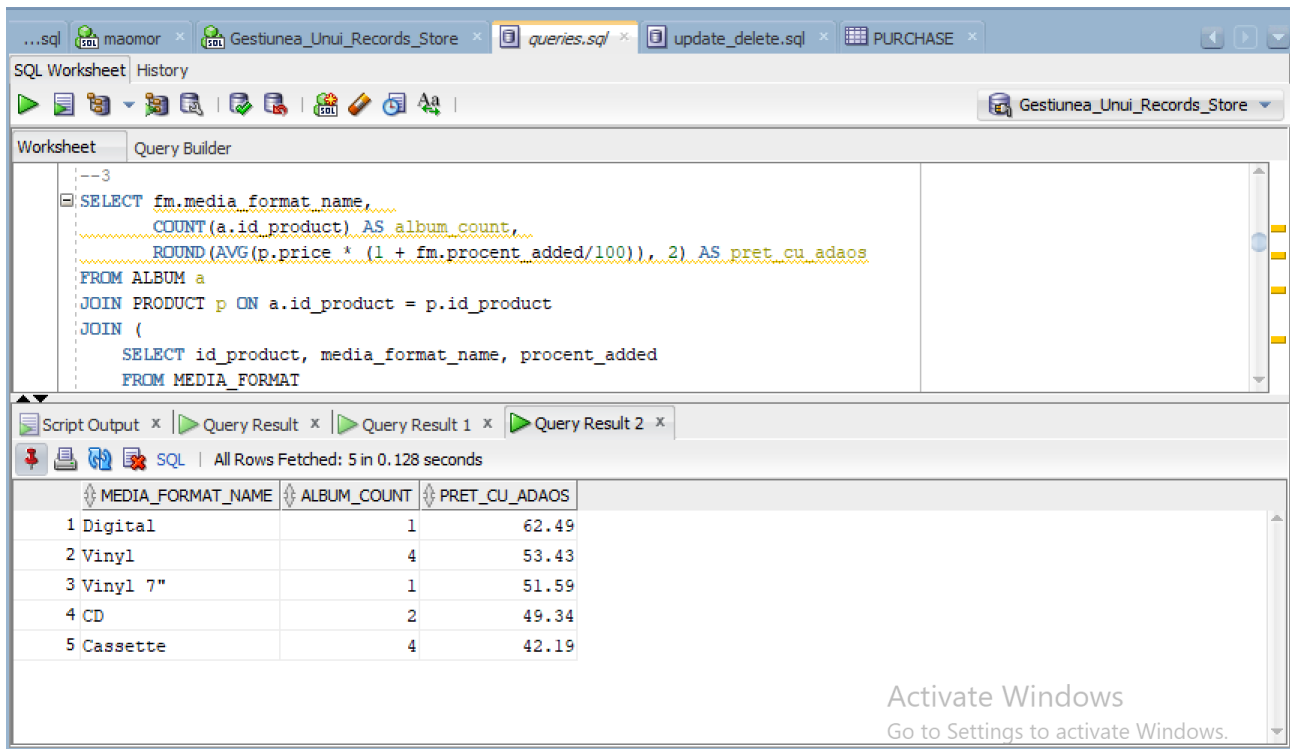
	ALBUM_NAME	PRICE	LABEL_NAME
1	Swimming	49.99	Universal Music
2	GNX	49.99	Sony Music
3	DAMN.	47.99	Atlantic Records

An "Activate Windows" watermark is visible in the bottom right corner of the application window.

### 11.3 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.media_format_name,
       COUNT(a.id_product) AS album_count,
       ROUND(AVG(p.price * (1 + fm.procent_added/100)), 2) AS pret_cu_adaos
FROM ALBUM a
JOIN PRODUCT p ON a.id_product = p.id_product
JOIN (
    SELECT id_product, media_format_name, procent_added
    FROM MEDIA_FORMAT
    WHERE procent_added > 0
) fm ON fm.id_product = a.id_product
GROUP BY fm.media_format_name
ORDER BY pret_cu_adaos DESC;
```



The screenshot shows a SQL Worksheet interface with a query and its results. The query is as follows:

```
--3
SELECT fm.media_format_name,
       COUNT(a.id_product) AS album_count,
       ROUND(AVG(p.price * (1 + fm.procent_added/100)), 2) AS pret_cu_adaos
FROM ALBUM a
JOIN PRODUCT p ON a.id_product = p.id_product
JOIN (
  SELECT id_product, media_format_name, procent_added
  FROM MEDIA_FORMAT
```

The results are displayed in a table with the following columns: MEDIA\_FORMAT\_NAME, ALBUM\_COUNT, and PRET\_CU\_ADAOS. The data is as follows:

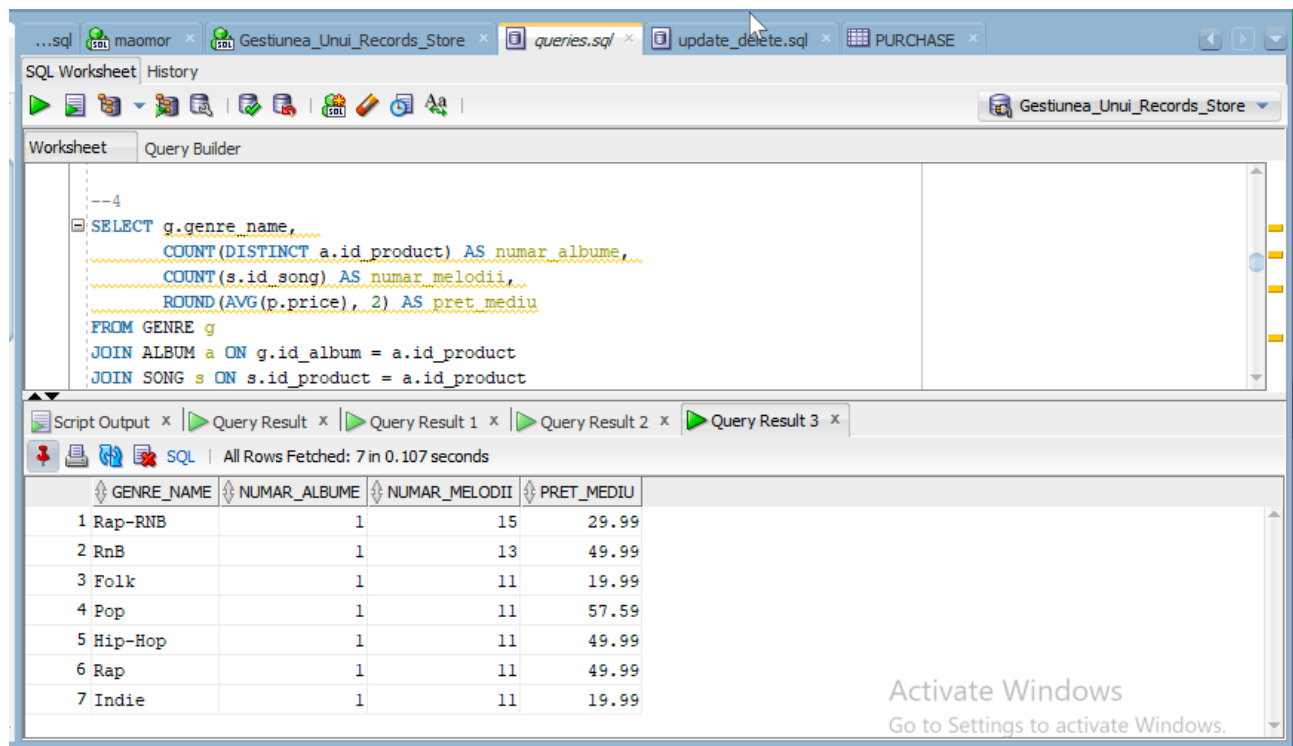
MEDIA_FORMAT_NAME	ALBUM_COUNT	PRET_CU_ADAOS
1 Digital	1	62.49
2 Vinyl	4	53.43
3 Vinyl 7"	1	51.59
4 CD	2	49.34
5 Cassette	4	42.19

At the bottom right, there is a watermark that says "Activate Windows Go to Settings to activate Windows."

#### 11.4 Cererea nr.4 - Grupari cu subcereri in HAVING

Afisiati pentru fiecare gen numarul de albume, numarul total de melodii si pretul mediu al albumelor din respectivul gen muzical.

```
SELECT g.genre_name,
       COUNT(DISTINCT a.id_product) AS numar_albume,
       COUNT(s.id_song) AS numar_melodii,
       ROUND(AVG(p.price), 2) AS pret_mediu
FROM GENRE g
JOIN ALBUM a ON g.id_album = a.id_product
JOIN SONG s ON s.id_product = a.id_product
JOIN PRODUCT p ON a.id_product = p.id_product
GROUP BY g.genre_name
HAVING COUNT(s.id_song) > (
  SELECT AVG(numar_melodii)
  FROM (SELECT COUNT(*) AS numar_melodii FROM SONG GROUP BY id_product)
)
ORDER BY numar_melodii DESC;
```



The screenshot shows a SQL Worksheet interface with a query and its results. The query is as follows:

```
--4
SELECT g.genre_name,
       COUNT(DISTINCT a.id_product) AS numar_albume,
       COUNT(s.id_song) AS numar_melodii,
       ROUND(AVG(p.price), 2) AS pret_mediu
FROM GENRE g
JOIN ALBUM a ON g.id_album = a.id_product
JOIN SONG s ON s.id_product = a.id_product
```

The results are displayed in a table with the following columns: GENRE\_NAME, NUMAR\_ALBUME, NUMAR\_MELODII, and PRET\_MEDIU. The data is as follows:

GENRE_NAME	NUMAR_ALBUME	NUMAR_MELODII	PRET_MEDIU
1 Rap-RNB	1	15	29.99
2 RnB	1	13	49.99
3 Folk	1	11	19.99
4 Pop	1	11	57.59
5 Hip-Hop	1	11	49.99
6 Rap	1	11	49.99
7 Indie	1	11	19.99

An "Activate Windows" watermark is visible in the bottom right corner of the screenshot.

### 11.5 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. Sortati descrescator in functie de numarul de achizitii, in caz de egalitate, sortati crescator dupa numarul de bani cheltuiti.

```
SELECT
  c.last_name || ', ' || c.first_name AS customer_name,
  c.city,
  COUNT(p.id_purchase) AS purchase_count,
  SUM(p.quantity) AS total_items,
  NVL(SUM(p.quantity * pur.price), 0) AS total_spent,
  NVL(MAX(TO_CHAR(p.purchase_date, 'YYYY-MM-DD')), 'Never purchased') AS last_purchase,
  DECODE(COUNT(DISTINCT a.id_product), 0, 'New Here',
         CASE WHEN COUNT(DISTINCT a.id_product) > 2 THEN 'Music lLover'
              ELSE 'Casual listener' END) AS customer_type
FROM CUSTOMER c
LEFT JOIN PURCHASE p ON c.id_customer = p.id_customer
LEFT JOIN PRODUCT pur ON p.id_product = pur.id_product
LEFT JOIN ALBUM a ON p.id_product = a.id_product
GROUP BY c.last_name, c.first_name, c.city
ORDER BY purchase_count DESC, total_spent ASC;
```

The screenshot shows a SQL Worksheet window with a query and its results. The query is as follows:

```

ORDER BY numar_melodii DESC;

--5
SELECT
  c.last_name || ', ' || c.first_name AS customer_name,
  c.city,
  COUNT(p.id_purchase) AS purchase_count,
  SUM(p.quantity) AS total_items,
  NVL(SUM(p.quantity * pur.price), 0) AS total_spent,

```

The results are displayed in a table with the following columns: CUSTOMER\_NAME, CITY, PURCHASE\_COUNT, TOTAL\_ITEMS, TOTAL\_SPENT, LAST\_PURCHASE, and CUSTOMER\_TYPE. The data is as follows:

CUSTOMER_NAME	CITY	PURCHASE_COUNT	TOTAL_ITEMS	TOTAL_SPENT	LAST_PURCHASE	CUSTOMER_TYPE
1 Dumitrescu, Alexandru	București	2	2	69.98	2023-05-15	Casual Listener
2 Munteanu, Diana	Timișoara	2	2	104.98	2023-06-25	Casual Listener
3 Radu, Ioana	Cluj-Napoca	2	2	122.58	2023-05-20	Casual Listener
4 Stoica, Gabriel	Iași	2	3	149.97	2023-05-25	Casual Listener
5 Manole, Cristina	Sibiu	1	1	37.99	2023-06-18	Casual Listener
6 Vasile, Andreea	Constanța	1	1	52.99	2023-06-02	Casual Listener
7 Preda, Robert	Craiova	1	2	95.98	2023-06-10	Casual Listener
8 Dinu, Bogdan	Brasov	1	3	134.97	2023-05-10	New Here

## 11.6 Cererea nr. 6 - Clauza WITH

Afișați numele complet al angajaților 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). Afișați și salariul total primit de la firma de la angajarea acestuia. Ordonați descrescător după valoarea vânzărilor și limitați rezultatul la primii 15 angajați.

```

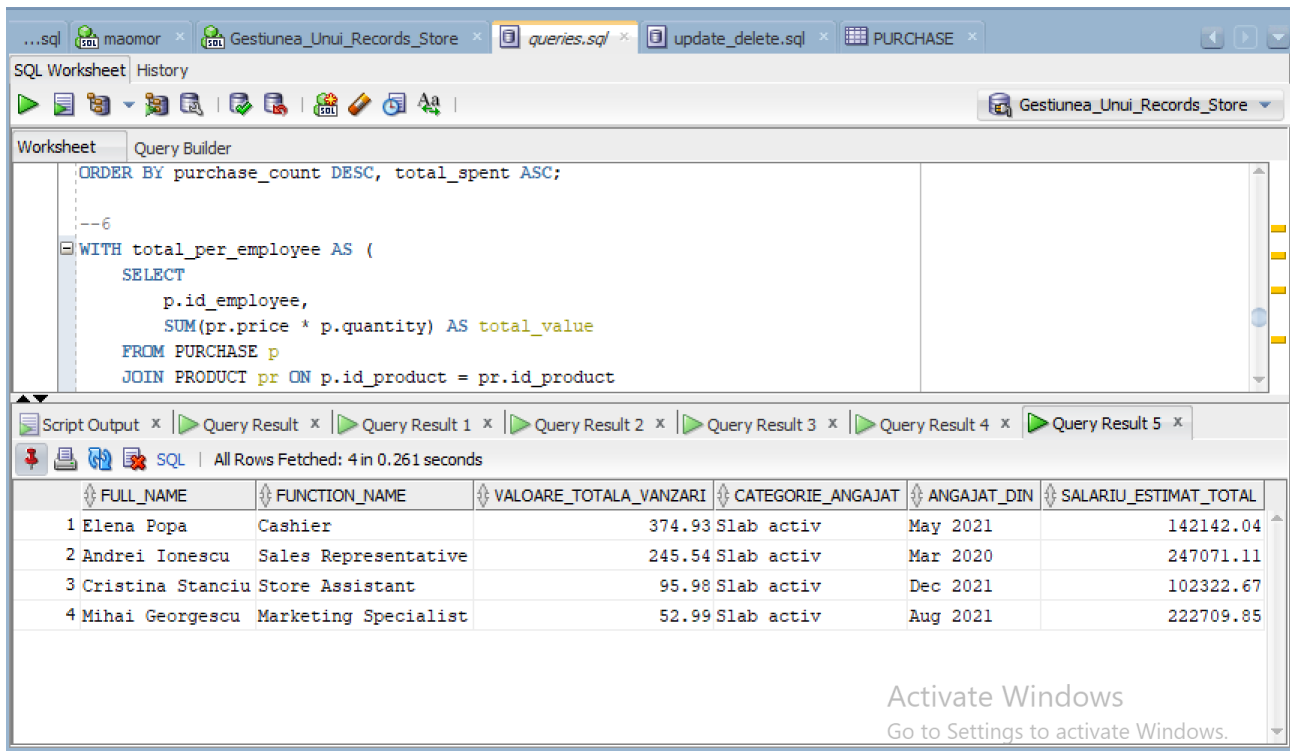
WITH total_per_employee AS (
  SELECT
    p.id_employee,
    SUM(pr.price * p.quantity) AS total_value
  FROM PURCHASE p
  JOIN PRODUCT pr ON p.id_product = pr.id_product
  GROUP BY p.id_employee
)
SELECT *
FROM (
  SELECT
    e.first_name || ' ' || NVL(e.last_name, 'anonim') AS full_name,
    f.function_name,
    ROUND(t.total_value, 2) AS valoare_totala_vanzari,
    CASE
      WHEN t.total_value > 1000 THEN 'Performant'
      WHEN t.total_value > 500 THEN 'Activ'
      ELSE 'Slab activ'
    END AS categorie_angajat,
    TO_CHAR(e.hire_date, 'Mon YYYY') AS angajat_din,
    ROUND(e.salary * MONTHS_BETWEEN(SYSDATE, e.hire_date), 2) AS salariu_estimat_total
  FROM EMPLOYEE e
  JOIN FUNCTION f ON e.id_function = f.id_function
  JOIN total_per_employee t ON e.id_employee = t.id_employee
  WHERE EXISTS (
    SELECT 1

```

```

        FROM PURCHASE p
        WHERE p.id_employee = e.id_employee
    )
    ORDER BY t.total_value DESC
)
WHERE ROWNUM <= 15;

```



The screenshot shows an SQL Worksheet window with the following tabs: ...sql, maamor, Gestiunea\_Unui\_Records\_Store, queries.sql, update\_delete.sql, and PURCHASE. The 'queries.sql' tab is active, displaying a query in the 'Query Builder' section. The query is as follows:

```

ORDER BY purchase_count DESC, total_spent ASC;

--6
WITH total_per_employee AS (
    SELECT
        p.id_employee,
        SUM(pr.price * p.quantity) AS total_value
    FROM PURCHASE p
    JOIN PRODUCT pr ON p.id_product = pr.id_product

```

Below the query, the 'Script Output' tab shows the results of the query. The output is a table with 6 columns: FULL\_NAME, FUNCTION\_NAME, VALOARE\_TOTALA\_VANZARI, CATEGORIE\_ANGAJAT, ANGAJAT\_DIN, and SALARIU\_ESTIMAT\_TOTAL. The table contains 4 rows of data:

	FULL_NAME	FUNCTION_NAME	VALOARE_TOTALA_VANZARI	CATEGORIE_ANGAJAT	ANGAJAT_DIN	SALARIU_ESTIMAT_TOTAL
1	Elena Popa	Cashier	374.93	Slab activ	May 2021	142142.04
2	Andrei Ionescu	Sales Representative	245.54	Slab activ	Mar 2020	247071.11
3	Cristina Stanciu	Store Assistant	95.98	Slab activ	Dec 2021	102322.67
4	Mihai Georgescu	Marketing Specialist	52.99	Slab activ	Aug 2021	222709.85

At the bottom of the window, there is a message: 'Activate Windows. Go to Settings to activate Windows.'

## 12 Cereri de actualizare si suprimare a datelor

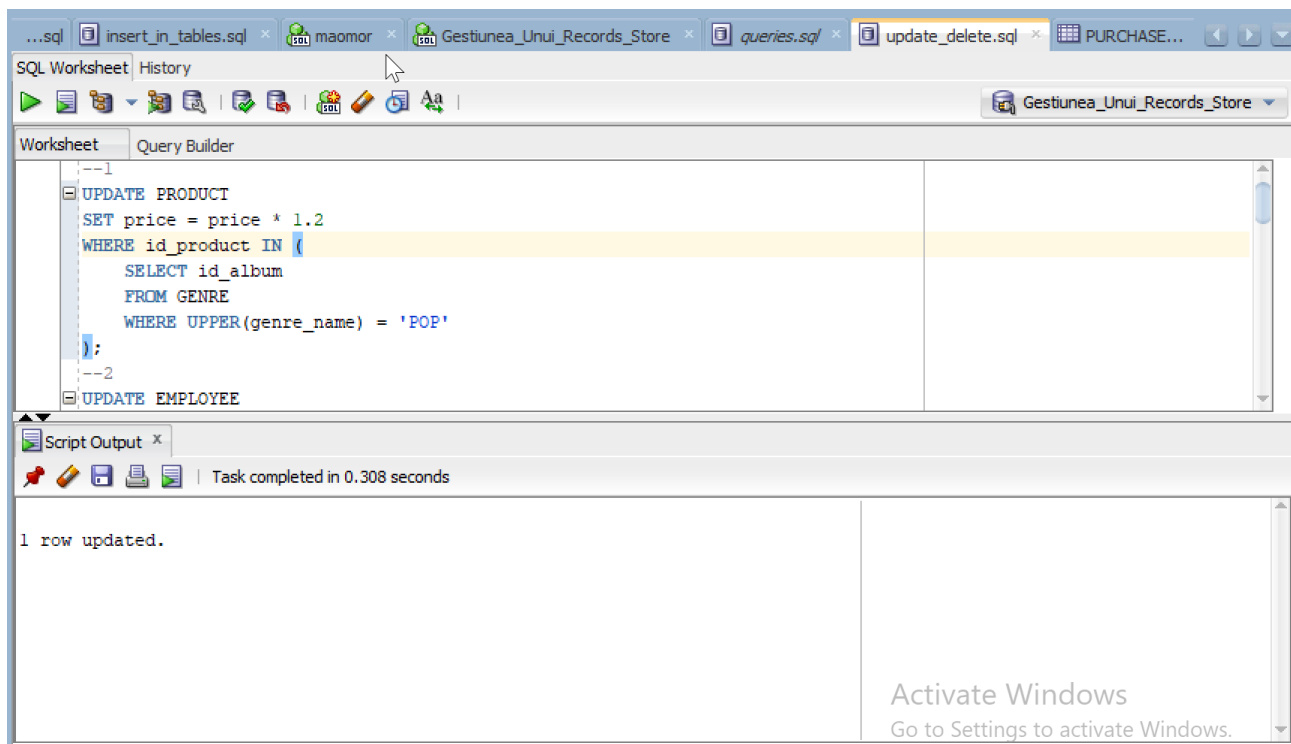
### 12.1 Cererea nr.1

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

```

UPDATE PRODUCT
SET price = price * 1.2
WHERE id_product IN (
    SELECT id_album
    FROM GENRE
    WHERE UPPER(genre_name) = 'POP'
);

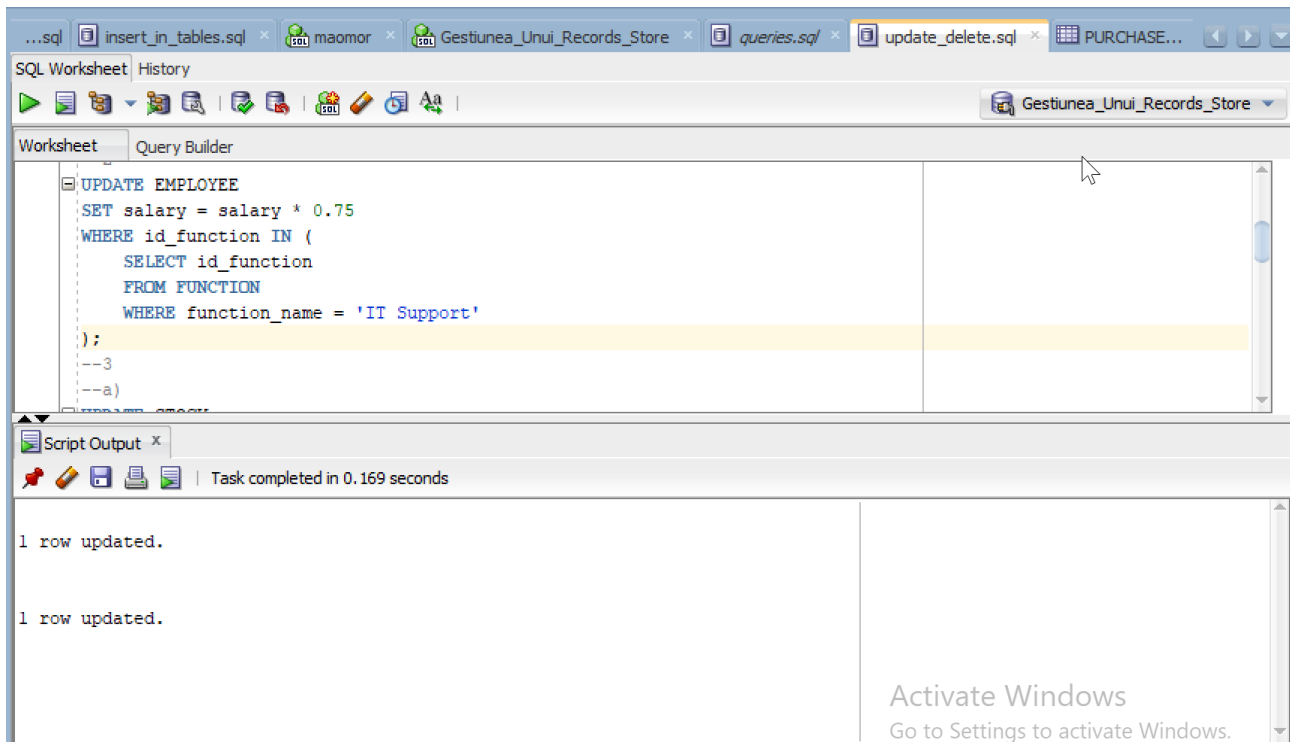
```



## 12.2 Cererea nr. 2

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

```
UPDATE EMPLOYEE
SET salary = salary * 0.75
WHERE id_function IN (
    SELECT id_function
    FROM FUNCTION
    WHERE function_name = 'IT Support'
);
```



### 12.3 Cererea nr. 3

Sa se reactualizeze stocurile de produse in functie comenzile prezente pentru comenzile 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.

```
--3
--a)
UPDATE STOCK s
SET s.quantity = s.quantity - (
  SELECT SUM(p.quantity)
  FROM PURCHASE p
  WHERE p.status = 'Pending'
  AND p.id_product = s.id_product
)
WHERE EXISTS (
  SELECT 1
  FROM PURCHASE p
  WHERE p.status = 'Pending'
  AND p.id_product = s.id_product
)
AND s.id_stock = (
  SELECT MIN(s2.id_stock)
  FROM STOCK s2
  WHERE s2.id_product = s.id_product
);

UPDATE PURCHASE
SET status = 'Completed'
WHERE status = 'Pending';
--b)
DELETE FROM STOCK
WHERE quantity <= 0;
```

