**UNIVERSITATEA DIN BUCURESTI**
**FACULTATEA DE MATEMATICA ȘI INFORMATICA**
**CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI**

# PROIECT BAZE DE DATE

**Îndrumător proiect:**
**Silviu Laurențiu Vasile**

**Student:**
**Gheorghiță Elena**
**Raluca Lorena**
**Grupa: 264**

**AN UNIVERSITAR 2021-2022**

**UNIVERSITATEA DIN BUCURESTI**
**FACULTATEA DE MATEMATICA ȘI INFORMATICA**
**CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI**

# GESTIONAREA UNEI FARMACII

**Îndrumător proiect:**
**Silviu Laurențiu Vasile**

**Student:**
**Gheorghiță Elena**
**Raluca Lorena**
**Grupa: 264**

**AN UNIVERSITAR 2021-2022**

# Cuprins

# 1.Prezentarea modelului

O bază de date, uneori numită și bancă de date, este o colecţie organizată de informaţii, o modalitate de stocare a unor informații și date pe un suport extern, cu posibilitatea extinderii ușoare și a regăsirii rapide a acestora.

Tema proiectului este gestionarea unei farmacii. O bază de date corectă a unei farmacii poate să diminueze posibilitatea aparițiilor erorilor, care pot duce la probleme grave, de asemenea o bază de date fără erori reprezintă și o siguranță a medicamentelor cât și a clienților.

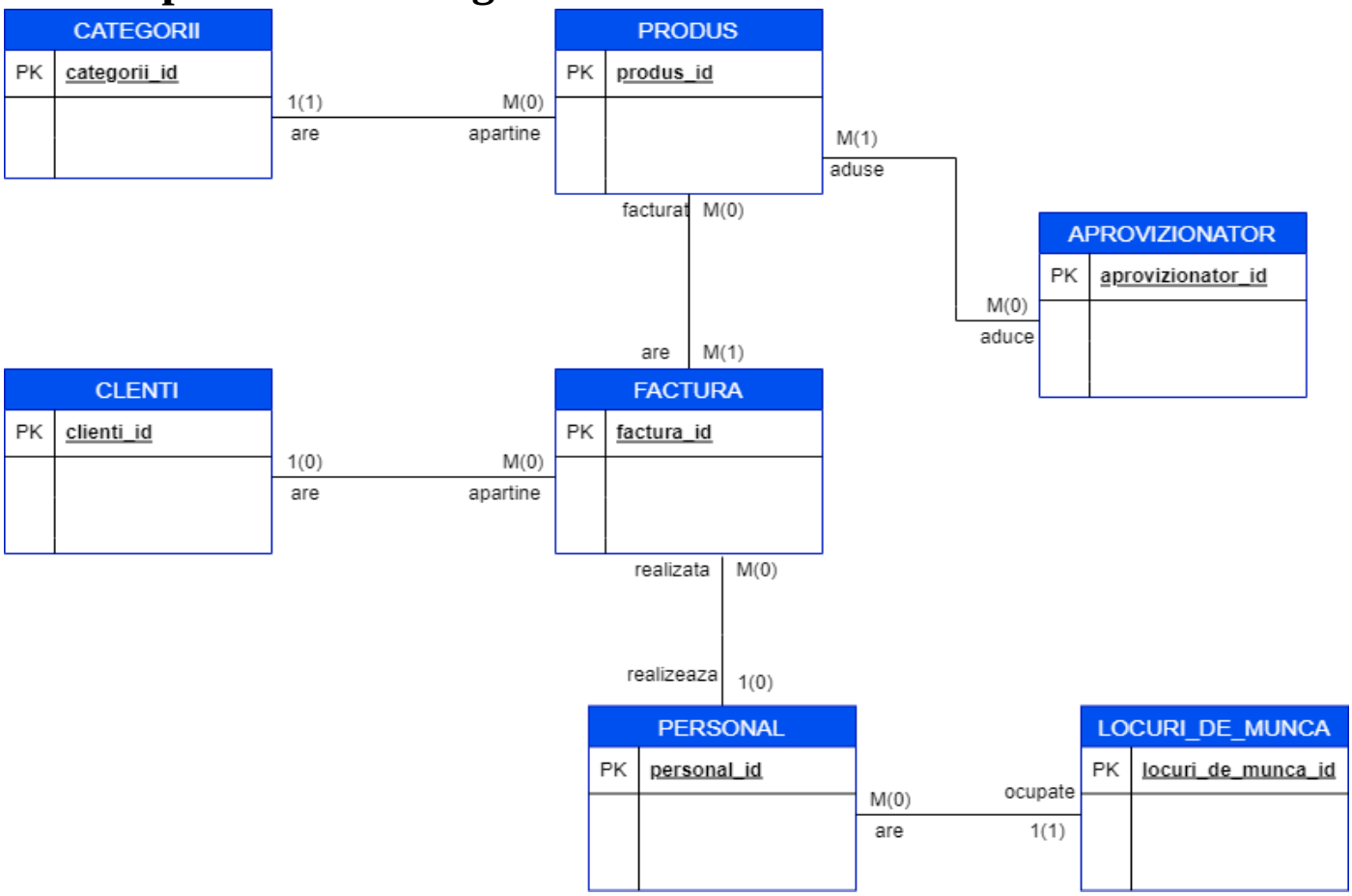
Am realizat o bază de date cu scopul de a gestiona produsele farmaceutice. Aceasta stochează date despre medicamente, clienți, angajați, furnizori și producători.

# 2.Regulile modelului

- O categorie poate avea mai multe produse
- Un produs poate aparține unei singure categorii
- Mai multe produse pot fi aduse de mai mulți aprovizionatori
- Mai mulți aprovizionatori pot aduce mai multe produse
- Mai multe produse se pot afla pe mai multe facturi
- O factură poate aparține unui singur client
- Un client poate avea mai multe facturi
- Un angajat poate realiza mai multe facturi
- O factură poate fi realizată de un singur angajat
- Un personal poate avea un singur loc de muncă
- Un loc de muncă poate avea mai mulți angajați

# 3.Diagrama Entitate-Relați

## 3.1.Reprezentarea diagramei




## 3.2.Descrierea entităților, atributelor, cheilor, relațiilor și a cardinalităților

### 3.2.1. Descrierea entităților, atributelor și a cheilor

#### *3.2.1.1. Tabelul CATEGORII*

- Tabelul CATEGORII cuprinde toate categoriile din care pot face parte produsele farmaceutice.
- Structura tabelului CATEGORII este:

| Cheie | Denumire | Tip | Descriere |
|---|---|---|---|

| | Atribut | | |
|---|---|---|---|
| PK | categorii_id | NUMBER | Identificatorul unei categorii |
| | categorie | VARCHAR2(30) | Clasificarea produselor farmaceutice (parafarmaceutice, suplimente alimentare, medicamente) |
| | denumire | VARCHAR2(30) | Clasificare produselor farmaceutice ( capsule, siropuri, drajeuri, etc.) |

### *3.2.1.2. Tabelul PRODUS*

- Tabelul PRODUS cuprinde toate produsele farmaceutice care se pot găsi într-o farmacie.
- Structura tabelului PRODUS este:

| Cheie | Denumire Atribut | Tip | Descriere |
|---|---|---|---|
| PK | produs_id | NUMBER(4) | Identificatorul unui produs |
| | denumire | VARCHAR2(30) | Denumirea produsului |
| | pret | NUMBER(4,2) | Prețul produselor |
| FK | categorii_id | NUMBER(4) | Face legătura cu tabelul CATEGORII |

### *3.2.1.3. Tabelul APROVIZIONATOR*

- Tabelul APROVIZIONATOR cuprinde toate datele despre aprovizionatori care aduc produse farmaceutice.
- Structura tabelului APROVIZIONATOR este:

| Cheie | Denumire Atribut | Tip | Descriere |
|---|---|---|---|
| PK | Aprovizionator_id | NUMBER(10) | Identificatorul unui aprovizionator |
| | tip | VARCHAR2(35) | Furnizor sau producător |
| | oras | VARCHAR2(35) | Adresă |
| | strada | NUMBER(35) | Adresă |
| | numar | NUMBER(5) | Adresă |
| | telefon | NUMBER(10) | Date de contact |
| | denumire | VARCHAR2(30) | Nume aprovizionator |

### 3.2.1.4. Tabelul FACTURA

- Tabelul FACTURA cuprinde informații despre factura primită de client la cumpărarea produselor farmaceutice.
- Structura tabelului FACTURA este:

| Cheie | Denumire Atribut | Tip | Descriere |
|---|---|---|---|
| PK | factura_id | NUMBER(4) | Identificatorul unei facturi |
| | valoare | NUMBER(4, 2) | Valoare pe care clientul trebuie să o achite |
| | data_fact | DATE | Data la care s-a realizat facturarea |
| FK | clienti_id | NUMBER(4) | Face legătura cu tabelul CLIENTI |
| FK | personal_id | NUMBER(4) | Face legătura cu tabelul PERSONAL |

### 3.2.1.5. Tabelul PERSONAL

- Tabelul PERSONAL cuprinde informații despre personalul care lucrează în farmacie.
- Structura tabelului PERSONAL este:

| Cheie | Denumire Atribut | Tip | Descriere |
|---|---|---|---|
| PK | personal_id | NUMBER(4) | Identificatorul unui angajat |
| | nume | VARCHAR2(30) | Numele angajatului |
| | prenume | VARCHAR2(40) | Prenumele angajatului |
| | telefon | NUMBER(10) | Date de contact |
| | strada | VARCHAR2(30) | Adresă |
| | numar | NUMBER(5) | Adresă |
| | localitate | VARCHAR2(30) | Adresă |
| | cod_postal | NUMBER(8) | Adresă |
| | salariu | NUMBER(6) | Suma lunară pe care o primește un angajat |
| FK | locuri_de_munca_id | NUMBER(4) | Face legătura cu tabelul LOCURI_DE_MUNCA |

### 3.2.1.6. Tabelul LOCURI_DE_MUNCA

- Tabelul LOCURI_DE_MUNCA cuprinde informații despre locurile de muncă dispuse de farmacie.
- Structura tabelului LOCURI_DE_MUNCA este:

| Cheie | Denumire Atribut | Tip | Descriere |
|---|---|---|---|
| PK | locuri_de_munca | NUMBER(4) | Identificatorul unui loc de muncă |
| | functie | VARCHAR2(35) | Funcția pe care un angajat o deține |
| | salariu_min | NUMBER(6) | Salariu minim pe care o funcție o are |
| | salariu_max | NUMBER(6) | Salariu maxim pe care o funcție o are |

#### 3.2.1.6. *Tabelul CLEINTI*

- Tabelul CLIENTI cuprinde informații despre clienți.
- Structura tabelului CLIENTI este:

| Cheie | Denumire Atribut | Tip | Descriere |
|---|---|---|---|
| PK | clienti_id | NUMBER | Identificatorul unui client |
| | nume | VARCHAR2(30) | Date personale |
| | prenume | VARCHAR2(40) | Date personale |
| | telefon | NUMBER | Date de contact |
| | strada | VARCHAR2(50) | Adresă |
| | numar | NUMBER | Adresă |
| | localitate | VARCHAR2(20) | Adresă |
| | cod_postal | NUMBER | Adresă |
| | asigurare | VARCHAR2(20) | Asigurarea de sănătate |

## 3.2.2 Descrierea relațiilor și a cardinalităților

#### 3.2.2.1. *CATEGORII-PRODUS*

**Relația:** În entitatea CATEEGORII sunt toate categoriile în care se pot încadra produsele farmaceutice.

**Cardinalități:**

- **Cardinalitate maximală**
  - Câte produse pot fi încadrate într-o categorie? => M
  - Câte categorii pot avea produse farmaceutice? => 1
- **Cardinalitate minimală**
  - Câte produse se pot afla într-o categorie? => 0
  - De câte categorii poate să aparțină un produs? =>1

### 3.2.2.2. PRODUS-APROVIZIONATOR

**Relația:** Entitatea PRODUSE conține toate produsele farmaceutice aduse de către aprovizionatori.

**Cardinalități:**

- **Cardinalitate maximală**
  - Câte produse pot aduce mai mulți aprovizionatori? => M
  - Câți aprovizionatori pot aduce mai multe produse farmaceutice?=>M
- **Cardinalitate minimală**
  - Câte produse pot aduce de aprovizionatori? => 1
  - De câți aprovizionatori pot fi aduse produse? => 0

### 3.2.2.3. PRODUS-FACTURA

**Relația:** Entitatea FACTURA conține detaliile despre vânzarea produsele farmaceutice.

**Cardinalități:**

- **Cardinalitate maximală**
  - Câte produse se pot afla pe mai multe facturi? => M
  - Câte facturi pot avea mai multe produse farmaceutice?=>M
- **Cardinalitate minimală**
  - Câte produse se pot afla pe facturi? => 0
  - Câte facturi pot avea produsele? => 1

### 3.2.2.4. FACTURA-CLEINTI

**Relația:** Entitatea CLIENTI conține detaliile despre clienții farmaciei.

**Cardinalități:**

- **Cardinalitate maximală**
  - Câte facturi pot aparține unui client? => M
  - Câte facturi poate avea un client? => 1
- **Cardinalitate minimală**
  - Câte facturi pot aparține unui client? => 0
  - Câte facturi poate avea un client? => 0

### *3.2.2.5. FACTURA-PERSONAL*

**Relația:** Entitatea personal conține detaliile despre personalul din cadrul farmaciei.

**Cardinalități:**

- **Cardinalitate maximală**
  - Câte facturi poate realiza un angajat? => M
  - Câte facturi pot fi realizate de un angajat? => 1
- **Cardinalitate minimală**
  - Câte facturi poate realiza personalul? => 0
  - Câte facturi se pot face de către personal? => 0

### *3.2.2.6. PERSONAL-LOCURI_DE_MUNCA*

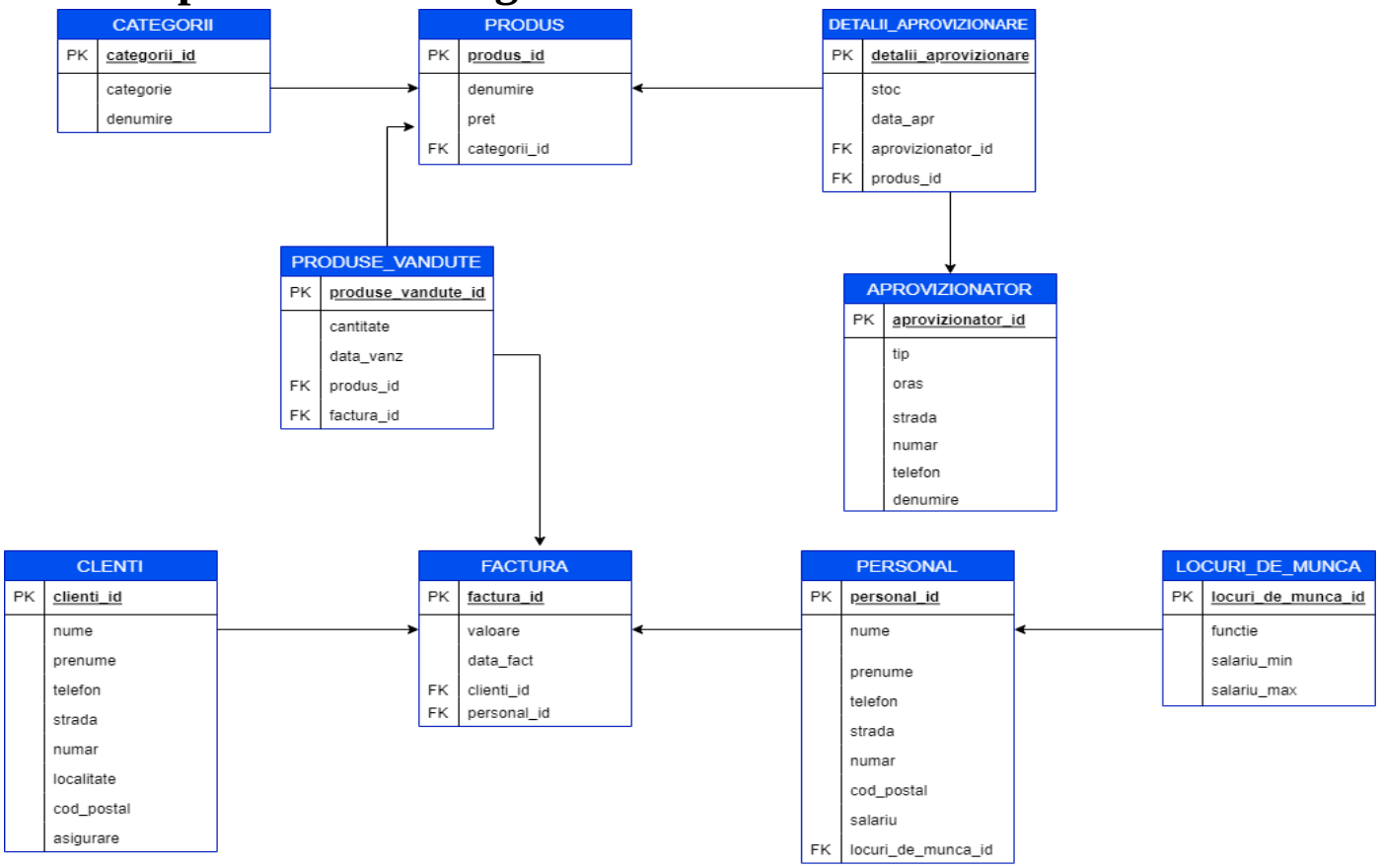
**Relația:** Entitatea LOCURI_DE_MUNCA conține detaliile despre locurile de muncă dispuse de farmacie.

**Cardinalități:**

- **Cardinalitate maximală**
  - Câți angajați pot ocupa o funcție? => M
  - Câte locuri de muncă poate avea un angajat? => 1
- **Cardinalitate minimală**
  - Câți angajați pot deține un loc de muncă? => 0
  - Câte locui poate ocupa personalul ? => 1

# 4. Diagrama Conceptuală

## 4.1. Reprezentarea diagramei




## 4.2. Descrierea constrângerilor de integritate

### 4.2.1. Tabelul CLIENTI

- **Constrângeri de tipul PRIMARY KEY**
  - **clienti_id**, atributul este cheie primară pentru tabelul CLIENTI

- **Constrângeri de tipul NOT NULL**
  - **nume**, pentru atributul nume
    - Clinții trebuie să aibă neapărat un nume.
  - **prenume**, pentru atributul preunume
    - Clinții trebuie să aibă neapărat un prenume.
  - **telefon**, pentru atributul telefon
    - Clinții trebuie să aibă neapărat un număr de telefon.

- **strada**, pentru atributul strada
  - Clinții trebuie să aibă o adresă, strada.
- **numar**, pentru atributul numar
  - Clinții trebuie să aibă un număr de locuință, numar.
- **localitate**, pentru atributul localitate
  - Clinții trebuie să aibă o localitate.
- **cod_postal**, pentru atributul cod_postal
  - Clinții trebuie să aibă neapărat un cod poștal.
- **asigurare**, pentru atributul asigurare
  - Clinții trebuie să arate dacă au sau nu asigurare.

### 4.2.2. Tabelul PERSONAL

- **Constrângeri de tipul PRIMARY KEY**
  - **personal_id**, cheie primară
    - Atributul este cheie primară pentru tabelul PERSONAL

- **Constrângeri de tipul FOREIGN KEY**
  - **loc_de_munca_id**, cheie străină
    - Face legătura cu tabelul LOC_DE_MUNCA.
    - Determină ce loc de muncă ocupă fiecare.

- **Constrângeri de tipul NOT NULL**
  - **nume**, pentru atributul nume
    - Personalul trebuie să aibă neapărat un nume.
  - **prenume,** pentru atributul preunume
    - Personalul trebuie să aibă neapărat un prenume.
  - **telefon**, pentru atributul telefon
    - Personalul trebuie să aibă neapărat un număr de telefon.
  - **strada**, pentru atributul strada
    - Personalul trebuie să aibă o adresă, strada.
  - **numar**, pentru atributul numar
    - Personalul trebuie să aibă un număr de locuință, numar.
  - **localitate**, pentru atributul localitate

- Personalul trebuie să aibă o localitate.

- **cod_postal**, pentru atributul cod_postal
  - Personalul trebuie să aibă neapărat un cod poștal.
- **salariu**, pentru atributul cod_postal
  - Personalul trebuie să aibă neapărat un salariu.

- **Constrângeri de tipul UNIQUE**
  - **telefon_unique**, pentru atributul telefon
    - Personalul trebuie să aibă obligatoriu propriul număr de telefon.
- **Constrângeri de tipul CHECK**
  - **telefon**, pentru atributul telefon
    - Prin funcția LENGTH, se verifică dacă numărul de telefon are 10 numere.

### *4.2.3. Tabelul LOC_DE_MUNCA*

- **Constrângeri de tipul PRIMARY KEY**
  - **loc_de_muna_id**, cheie primară
    - Atributul este cheie primară în tabelul LOC_DE_MUNCA

- **Constrângeri de tipul NOT NULL**
  - **functie**, pentru atributul functie
    - Locul de muncă trebuie să aibă obligatoriu o funcție.
  - **salariu_min**, pentru atributul salariu_min
    - Salariu minim pe care un loc de muncă îl poate avea.
  - **salariu_max**, pentru atributul salariu_max
    - Salariu maxim pe care un loc de muncă îl poate avea.

### 4.2.4. Tabelul FACTURA

- **Constrângeri de tipul PRIMARY KEY**
  - **factura_id**, cheie primară
    - Atributul este cheia primară în tabelul FACTURA

- **Constrângeri de tipul FOREIGN KEY**
  - **clienti_id**, pentru atributul clienti_id
    - Face legătura cu tabelul CLIENTI.
    - Determină date despre posibilul cumpărător.
  - **personal_id**, pentru atributul personal_id
    - Face legătura cu tabelul PERSONAL.
    - Determină date despre fiecare angajat.

- **Constrângeri de tipul NOT NULL**
  - **valoare**, pentru atributul valoare
    - Fiecare factură trebuie să aibă o valoare, chiar și nulă.
  - **data_fact**, pentru atributul data_fact
    - Fiecare factură trebuie să aibă o dată la care a fost realizată.

### 4.2.5. Tabelul PRODUSE_VANDUTE

- **Constrângeri de tipul PRIMARY KEY**
  - **produse_vandute_id**, cheie primară
    - Atributul este cheie primară în tabelul PRODUSE_VANDUTE

- **Constrângeri de tipul FOREIGN KEY**
  - **factura_id**, cheie externă
    - Face legătura cu tabelul FACTURA.
    - Determină facturile prin care s-au realizat vânzări.
  - **produs_id**, pentru atributul produs_id
    - Face legătura cu tabelul PRODUS.

- Determină date despre fiecare produs vândute

- **Constrângeri de tipul NOT NULL**
  - **cantitate**, pentru atributul cantitate
    - Trebuie să se cunoască cantitatea de produse vândute.
  - **data_vanz**, pentru atributul data_vanz
    - Fiecare zi încheiată trebuie să fie trecută.

### 4.2.6. Tabelul PRODUS

- **Constrângeri de tipul PRIMARY KEY**
  - **produs_id**, cheie primară
    - Atributul este cheie primară în tabelul PRODUS

- **Constrângeri de tipul FOREIGN KEY**
  - **categorie_id**, cheie externă
    - Face legătura cu tabelul CATEGORIE.
    - Determină categoriile din care fac parte produsele.

- **Constrângeri de tipul NOT NULL**
  - **denumire**, pentru atributul denumire
    - Fiecare produs trebuie să aibă o denumire.
  - **pret**, pentru atributul pret
    - Fiecare produs trebuie să aibă un pret.

### 4.2.7. Tabelul CATEGORII

- **Constrângeri de tipul PRIMARY KEY**
  - **categorii_id**, cheie primară
    - Atributul este cheie primară în tabelul CATEGORII

- **Constrângeri de tipul NOT NULL**
  - **categorie**, pentru atributul categorie
    - Fiecare produs trebuie să se încadreze într-o categorie, produse parafarmaceutice, suplimente alimentare, medicamente.
  - **denumire**, pentru atributul denumire
    - Fiecare produs trebuie să aibă o denumire, capsule, siropuri, drajeuri, etc.

### 4.2.8. Tabelul DETALII_APROVIZIONARE

- **Constrângeri de tipul PRIMARY KEY**
  - **detalii_aprovizionare_id**, cheie primară
    - Atributul este cheie primară în tabelul DETALII_APROVIZIONARE

- **Constrângeri de tipul FOREIGN KEY**
  - **aprovizionator_id**, cheie externă
    - Face legătura cu tabelul APROVIZIONATOR.
    - Determină detalii despre fiecare aprovizionator.

- **Constrângeri de tipul NOT NULL**
  - **data_apr**, pentru atributul data_apr
    - Fiecare aprovizionare trebuie să fie înregistrată.
  - **stoc**, pentru atributul stoc
    - La fiecare aprovizionare trebuie trecută cantitatea de produse.

### 4.2.9. Tabelul APROVIZIONATORI

- **Constrângeri de tipul PRIMARY KEY**
  - aprovizionatori_id, cheie primară
    - Atributul este cheie primară în tabelul APROVIZIONATORI
- **Constrângeri de tipul NOT NULL**

- **tip**, pentru atributul tip
  - Fiecare aprovizionator trebuie să fie furnizor sau producător.
- **oras**, pentru atributul oras
  - Fiecare aprovizionator trebuie să apărțină de un oraș.
- **strada**, pentru atributul strada
  - Fiecare aprovizionator trebuie să apărțină de o stradă.
- **numar**, pentru atributul numar
  - Fiecare aprovizionator trebuie să aibă un număr.
- **telefon**, pentru atributul telefon
  - Fiecare aprovizionator trebuie să aibă un număr de telefon.
- **denumire**, pentru atributul denumire
  - Fiecare aprovizionator trebuie să aibă o denumire.

- **Constrângeri de tipul CHECK**
  - **telefon**, pentru atributul telefon
    - Prin funcția LENGTH, se verifică dacă numărul de telefon are 10 numere.

! S-a folosit CHECK(LENGTH(telefon)=9) pentru a putea insera linii cu numere de telefon de 10 cifre începâd cu 0.

## 4.3. Schemele relaționale

### 4.3.1. Schemele relaționale

* Schemele relaționale atașate diagramei conceptuale:

- CLIENTI(PK- clienti_id, nume, prenume, telefon, strada, numar, localitate, cod_postal, asigurare);
- FACTURA(PK- factura_id, valoare, data, FK- clienti_id, FK- personal_id);

- PERSONAL(PK- personal_id, nume, prenume, telefon, strada, numar, localitate, cod_postal, salariu, FK- locuri_de_munca_id);
- LOCURI_DE_MUNCA(PK- locuri_de_munca_id, salariu_min, salariu_max);
- PRODUSE_VANDUTE(PK- produse_vandute_id, cantitate, KF- produs_id, FK- factura_id);
- PRODUS(PK- produs_id, denumire, FK- categorii_id, pret);
- CATEGORII(PK- categorii_id, categorie, denumire);
- DETALII_APROVIZIONARE(PK- detalii_aprovizionare_id, data, FK- aprovizionator_id, FK- produs_id);
- APROVIZIONATOR(PK- aprovizionator_id, tip, oras, strada, numar, telefon, denumire).

### 4.3.2. Descrierea constângerilor ON DELETE

* Există 8 costângeri de tip ON DELETE, câte una pentru fiecare costângere de tip FOREIGN KEY:

## ON DELETE SET NULL

- clienti_id, **ON DELETE SET NULL**
  - clienti_id este FK pentru tabelul FACTURA
    - Dacă clientul este șters, facutra va rămâne în baza de date.

- personal_id **ON DELETE SET NULL**
  - personal_id este FK pentru tabelul FACTURA
    - Dacă personalul este șters, factura va rămâne în baza de date.

- produs_id **ON DELETE SET NULL**
  - produs_id este FK pentru tabelul PRODUSE_VANDUTE
    - Dacă produsul este șters, acesta nu va afecta produsele vândute.

- factura_id **ON DELETE SET NULL**
  - factura_id este FK pentru tabelul PRODUSE_VANDUTE
    - Dacă factura este stearsă, nu va afecta produsele vândute.

- categorii_id **ON DELETE SET NULL**
  - categorii_id este FK pentru tabelul PRDOUS
    - Dacă cateoriile sunt șterse, produsele nu vor fi afectate.

### ON DELETE CASCADE

- locuri_de_munca_id **ON DELETE CASCADE**
  - locuri_de_munca_id este FK pentru tabelul PERSONAL
    - Daca un loc de muncă este șters personalul se va șterge și el.
- aprovizionator_id **ON DELETE CASCADE**
  - aprovizionator_id este FK pentru tabelul DETALII_APROVIZIONATORI
    - Dacă un aprovizionator este șters, va afecta detaliile și se vor șterge și ele.

- produs_id **ON DELETE CASCADE**
  - produs_id este FK pentru tabelul DETALII_APROVIZIONARE
    - Dacă un produs se șterge acesta va afecta detaliile aprovizionatorilor.

# 5. Scriptul SQL

## 5.1. Introducere

SQL este limbaj de interogare structurat, limbaj de programare specific pentru manipularea datelor în sistemele de manipulare a bazelor de date relaționale , iar la origine este un limbaj bazat pe

algebra relațională. Acesta are ca scop inserarea datelor, interogații, actualizare și ștergere, modificarea și crearea schemelor, precum și controlul accesului la date. A devenit un standard în domeniu, fiind cel mai popular limbaj utilizat pentru crearea, modificarea, regăsirea și manipularea datelor de către Sistemele de Gestiune a Bazelor de Date relaționale. Pe lângă versiunile standardizate ale limbajului, există o mulțime de dialecte și variante, unele proprietare, fiind specifice anumitor Sistemele de Gestiune a Bazelor de Date și de asemenea conținând extensii pentru a suporta Sistemele de Baze de Date obiectuale.

Scriptul SQL, salvat cu numele scriptSQL_GHEORGHITA_ELENA_RALUCA_LORENA_G264.sql, a fost scris folosind Oracle SQL Developer.

## 5.2. Creearea tebelelor

Tabelel au fost create cu ajutorul instrucțiunii CREATE TABLE nume_tabel( ); care face parte din setul de instrucțiuni (Data Definition Language) ale SQL.

### 5.2.1. Crearea tabelului CATEGORII

```
CREATE TABLE CATEGORII(
categorii_id NUMBER(4) PRIMARY KEY,
categorie VARCHAR2(30) NOT NULL,
denumire VARCHAR2(30) NOT NULL);
CREATE SEQUENCE PK_APROVIZIONATOR
START WITH 1 INCREMENT BY 1;
```

### 5.2.2. Crearea tabelului CLIENTI

```
CREATE TABLE CLIENTI(
```

```
clienti_id NUMBER(4) PRIMARY KEY,
nume VARCHAR2(30)NOT NULL,
prenume VARCHAR2(30)NOT NULL,
telefon NUMBER(10) CHECK(LENGTH(telefon)=9)
NOT NULL,
strada VARCHAR2(30)NOT NULL,
numar NUMBER(5)NOT NULL,
localitate VARCHAR2(30)NOT NULL,
cod_postal NUMBER(8)NOT NULL,
asigurare VARCHAR2(5)NOT NULL);
CREATE SEQUENCE PK_APROVIZIONATOR
START WITH 1 INCREMENT BY 1;
```

### 5.2.3. Crearea tabelului LOCURI_DE_MUNCA

```
CREATE TABLE LOCURI_DE_MUNCA (
locuri_de_munca_id NUMBER(4) PRIMARY KEY,
functie VARCHAR2(35) NOT NULL,
 salariu_min NUMBER(6)NOT NULL,
salariu_max NUMBER(6)NOT NULL);
CREATE SEQUENCE PK_APROVIZIONATOR
START WITH 1 INCREMENT BY 1;
```

### 5.2.4. Crearea tabelului PERSONAL

```
CREATE TABLE PERSONAL(
```

```
personal_id NUMBER(4) PRIMARY KEY,
nume VARCHAR2(30)NOT NULL,
prenume VARCHAR2(40)NOT NULL,
telefon NUMBER(10) CHECK(LENGTH(telefon)=9)
NOT NULL,
CONSTRAINT telefon_unique UNIQUE (telefon),
strada VARCHAR2(30)NOT NULL,
numar NUMBER(5)NOT NULL,
localitate VARCHAR2(30)NOT NULL,
cod_postal NUMBER(8)NOT NULL,
salariu NUMBER(6)NOT NULL,
locuri_de_munca_id NUMBER(4),
CONSTRAINT FK_LOCURI_DE_MUNCA FOREIGN
KEY(locuri_de_munca_id) REFERENCES
LOCURI_DE_MUNCA(locuri_de_munca_id) ON
DELETE CASCADE);
CREATE SEQUENCE PK_APROVIZIONATOR
START WITH 1 INCREMENT BY 1;
```

### 5.2.5. Crearea tabelului FACTURA

```
CREATE TABLE FACTURA(
factura_id NUMBER(4) PRIMARY KEY,
valoarea NUMBER(4,2)NOT NULL,
data_fact DATE NOT NULL,
```

```
clienti_id NUMBER(4),
personal_id NUMBER (4),
CONSTRAINT FK_CLIENTI FOREIGN KEY(clienti_id) REFERENCES CLIENTI(clienti_id) ON DELETE SET NULL,
CONSTRAINT FK_PERSONAL FOREIGN KEY(personal_id) REFERENCES PERSONAL(personal_id) ON DELETE SET NULL);
CREATE SEQUENCE PK_APROVIZIONATOR START WITH 1 INCREMENT BY 1;
```

### 5.2.6. Crearea tabelului PRODUS

```
CREATE TABLE PRODUS (
produs_id NUMBER(4) PRIMARY KEY,
denumire VARCHAR2(50) NOT NULL,
pret NUMBER(4,2) NOT NULL,
categorii_id NUMBER(4) NOT NULL,
CONSTRAINT FK_CATEGORII FOREIGN KEY (categorii_id) REFERENCES CATEGORII(categorii_id) ON DELETE SET NULL);
CREATE SEQUENCE PK_APROVIZIONATOR START WITH 1 INCREMENT BY 1;
```

### 5.2.7. Crearea tabelului APROVIZIONATOR

```
CREATE TABLE APROVIZIONATOR (
```

```
 aprovizionator_id NUMBER(10) PRIMARY KEY,
 tip VARCHAR2(35) NOT NULL,
 oras VARCHAR2(35) NOT NULL,
 strada VARCHAR2(35) NOT NULL,
 telefon NUMBER(10) CHECK(LENGTH(telefon)=9)
NOT NULL,
 denumire VARCHAR2(30));
 CREATE SEQUENCE PK_APROVIZIONATOR
START WITH 1 INCREMENT BY 1;
```

### 5.2.8. Crearea tabelului PRODUSE_VANDUTE

```
CREATE TABLE PRODUSE_VANDUTE(
produse_vandute_id NUMBER(4) PRIMARY KEY,
cantitatea VARCHAR2(3) NOT NULL,
data_vanz DATE NOT NULL,
produs_id NUMBER(4),
factura_id NUMBER(4),
CONSTRAINT FK_PRODUS FOREIGN
KEY(produs_id) REFERENCES PRODUS(produs_id)
ON DELETE SET NULL,
CONSTRAINT FK_FACTURA FOREIGN
KEY(factura_id) REFERENCES FACTURA(factura_id)
ON DELETE SET NULL);
CREATE SEQUENCE PK_APROVIZIONATOR
START WITH 1 INCREMENT BY 1;
```

### 5.2.9. Crearea tabelului DETALII_APROVIZONARE

```
CREATE TABLE DETALII_APROVIZONARE(
detalii_aprovizionare_id NUMBER(4) PRIMARY KEY,
stoc NUMBER(3) NOT NULL,
data_apr DATE NOT NULL,
aprovizionator_id NUMBER(4),
produs_id NUMBER(4),
CONSTRAINT FK_APROVIZIONATOR FOREIGN KEY(aprovizionator_id) REFERENCES APROVIZIONATOR(aprovizionator_id) ON DELETE CASCADE,
CONSTRAINT FK_PRODUS_D_A FOREIGN KEY(produs_id) REFERENCES PRODUS(produs_id) ON DELETE CASCADE);
CREATE SEQUENCE PK_APROVIZIONATOR START WITH 1 INCREMENT BY 1;
```

## 5.3. Introducerea datelor în tabele

### 5.3.1. Introducerea datelor în tabelul CATEGORII

```
INSERT INTO  CATEGORII VALUES (PK_CATEGORII.NEXTVAL,'Suplimente Alimentare','Drajeuri');
INSERT INTO  CATEGORII VALUES (PK_CATEGORII.NEXTVAL,'Suplimente Alimentare','Capsule');
INSERT INTO  CATEGORII VALUES (PK_CATEGORII.NEXTVAL,'Medicamente ','Capsule');
INSERT INTO  CATEGORII VALUES (PK_CATEGORII.NEXTVAL,'Medicamente ','Siropuri');
```

```
INSERT INTO  CATEGORII VALUES
(PK_CATEGORII.NEXTVAL,'Medicamente ','Supozitoare');

INSERT INTO  CATEGORII VALUES
(PK_CATEGORII.NEXTVAL,'Medicamente ','Drajeuri');

INSERT INTO  CATEGORII VALUES
(PK_CATEGORII.NEXTVAL,'Parafarmaceutice','Compresesterile');

INSERT INTO  CATEGORII VALUES
(PK_CATEGORII.NEXTVAL,'Parafarmaceutice','Leucoplast');
```

## 5.3. 2. Introducerea datelor în tabelul CLIENTI

```
INSERT INTO CLIENTI VALUES
(PK_CLIENTI.NEXTVAL,'Dovleac','Denisa',0799240687,'Aninoasa',4,'Iancu Jianu',237220,'DA');

INSERT INTO CLIENTI VALUES
(PK_CLIENTI.NEXTVAL,'Petrica','Dorina',0799240675,'Florilor',342,'Horezu',437220,'NU');

INSERT INTO CLIENTI VALUES
(PK_CLIENTI.NEXTVAL,'Gheorghita','Marian',0731621356,'Primaverii',280,'Barbu Sirbei',237210,'DA');

INSERT INTO CLIENTI VALUES
(PK_CLIENTI.NEXTVAL,'Serban','Maria',0712045687,'1Mai',20,'Soparlita',342220,'DA');

INSERT INTO CLIENTI VALUES
(PK_CLIENTI.NEXTVAL,'Bodac','Elena',0799227392,'Stanca',75,'Cisnadie',234330,'DA');
```

### 5.3.3. Introducerea datelor în tabelul  LOCURI_DE_MUNCA

```
INSERT INTO LOCURI_DE_MUNCA VALUES
(PK_LOCURI_DE_MUNCA.NEXTVAL,'Asistenta',1929,3534);

INSERT INTO LOCURI_DE_MUNCA VALUES
(PK_LOCURI_DE_MUNCA.NEXTVAL,'Farmacist',4388,5528);
```

```
INSERT INTO LOCURI_DE_MUNCA VALUES
(PK_LOCURI_DE_MUNCA.NEXTVAL,'Curatator',1000,1500);
```

### 5.3.4. Introducerea datelor în tabelul PERSONAL

```
INSERT INTO PERSONAL
VALUES(PK_PERSONAL.NEXTVAL,'Cretu','Claudia',0764288264,'Mologesti',6,'Branovat',212456,1929,1);

INSERT INTO PERSONAL
VALUES(PK_PERSONAL.NEXTVAL,'Chirtibus','Ionela',0722388264,'Mihai Viteazu',23,'Bulzesti',127896,3534,1);

INSERT INTO PERSONAL
VALUES(PK_PERSONAL.NEXTVAL,'Bidiric','Nicoleta',0782501286,'Unirea',86,'Diculesti',127857,3534,1);

INSERT INTO PERSONAL
VALUES(PK_PERSONAL.NEXTVAL,'Ghita','Alexandru',0799240367,'Revolutiei',13,'Bals',556432,4388,2);

INSERT INTO PERSONAL
VALUES(PK_PERSONAL.NEXTVAL,'Petrescu','Daniel',0762011933,'Ciresilor',1,'Corbeni',442155,5528,2);

INSERT INTO PERSONAL
VALUES(PK_PERSONAL.NEXTVAL,'Chiriac','Maria',0793672917,'Sabin Balasea',10,'Dobriceni',773921,1500,3);

INSERT INTO PERSONAL
VALUES(PK_PERSONAL.NEXTVAL,'Mita','Ana',0742671917,'Fulgerului',71,'Preotesti',342001,1000,3);
```

### 5.3.5. Introducerea datelor în tabelul FACTURA

```
INSERT INTO FACTURA
VALUES(PK_FACTURA.NEXTVAL,47.50,TO_DATE('11-01-2022','DD-MM-YYYY'),1,2);

INSERT INTO FACTURA
VALUES(PK_FACTURA.NEXTVAL,37.5,TO_DATE('11-01-2022','DD-MM-YYYY'),2,1);
```

```
INSERT INTO FACTURA
VALUES(PK_FACTURA.NEXTVAL,91.5,TO_DATE('15-01-2022','DD-MM-YYYY'),3,2);

INSERT INTO FACTURA
VALUES(PK_FACTURA.NEXTVAL,20,TO_DATE('28-01-2022','DD-MM-YYYY'),4,2);

INSERT INTO FACTURA
VALUES(PK_FACTURA.NEXTVAL,40,TO_DATE('21-01-2022','DD-MM-YYYY'),5,1);
```

### 5.3.6. Introducerea datelor în tabelul PRODUS

```
INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'DULCOLAX',11.50,5);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'KETOMAG',10,5);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'STODAL',26,4);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'EURESPAL',30,4);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'REMOSTABIL',5,3);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'SPAVERIN',15,3);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'HERBIN',9,6);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'TUMMY TOX',90,1);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'ACNE OUT',12,1);

INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'OCTACARE',4.50,8);
```

```
INSERT INTO PRODUS
VALUES(PK_PRODUS.NEXTVAL,'CARA',30,7);
```

### 5.3.7. Introducerea datelor în tabelul APROVIZIONATOR

```
INSERT INTO APROVIZIONATOR
VALUES(PK_APROVIZIONATOR.NEXTVAL,'Producator','Craiova','Marului',120,0767540123,'FARMA CLASS INDUSTRY SRL');

INSERT INTO APROVIZIONATOR
VALUES(PK_APROVIZIONATOR.NEXTVAL,'Producator','Bucuresti','Pericle Papahagi',230,0769530687,'MICROSIN SRL');

INSERT INTO APROVIZIONATOR
VALUES(PK_APROVIZIONATOR.NEXTVAL,'Furnizor','Focsani','Mihai Viteazu',24,0785385107,'FARMMIZ');

INSERT INTO APROVIZIONATOR
VALUES(PK_APROVIZIONATOR.NEXTVAL,'Furnizor','Bals','Soarelui',36,0776831197,'DEMI FARM');
```

### 5.3.8. Introducerea datelor în tabelul PRODUSE_VANDUTE

```
NSERT INTO PRODUSE_VANDUTE
VALUES(PK_PRODUSE_VANDUTE.NEXTVAL,4,TO_DATE('28-01-2022','DD-MM-YYYY'),5,4);

INSERT INTO PRODUSE_VANDUTE
VALUES(PK_PRODUSE_VANDUTE.NEXTVAL,8,TO_DATE('11-01-2022','DD-MM-YYYY'),10,2);

INSERT INTO PRODUSE_VANDUTE
VALUES(PK_PRODUSE_VANDUTE.NEXTVAL,3,TO_DATE('15-01-2022','DD-MM-YYYY'),4,3);

INSERT INTO PRODUSE_VANDUTE
VALUES(PK_PRODUSE_VANDUTE.NEXTVAL,4,TO_DATE('21-01-2022','DD-MM-YYYY'),2,5);
```

```
INSERT INTO PRODUSE_VANDUTE
VALUES(PK_PRODUSE_VANDUTE.NEXTVAL,4,TO_DATE('1
1-01-2022','DD-MM-YYYY'),1,1);
```

### 5.3.9. Introducerea datelor în tabelul DETALII_APROVIZIONARE

```
INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE.NEXTVAL,50,TO_DATE('09-
01-2022','DD-MM-YYYY'),1,1);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE.NEXTVAL,70,TO_DATE('19-
03-2021','DD-MM-YYYY'),4,2);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,44,
TO_DATE(('09-07-2021', 'DD-MM-YYYY'),2,3);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,89,
TO_DATE(('28-05-2021', 'DD-MM-YYYY'),3,4);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,35,
TO_DATE(('14-08-2021', 'DD-MM-YYYY'),3,5);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,80,
TO_DATE(('01-01-2022', 'DD-MM-YYYY'),4,6);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,60,
TO_DATE(('22-06-2021', 'DD-MM-YYYY'),1,7);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,67,
TO_DATE(('10-01-2021', 'DD-MM-YYYY'),1,8);

INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,95,
TO_DATE(('06-01-2022', 'DD-MM-YYYY'),2,9);
```

```
INSERT INTO DETALII_APROVIZONARE VALUES
(PK_DETALII_APROVIZONARE. NEXTVAL,55
TO_DATE(('15-01-2022', 'DD-MM-YYYY'),3,10);

INSERT INTO DETALII_APROVIZONARE
VALUES(PK_DETALII_APROVIZONARE.
NEXTVAL,75,TO_DATE('14-12-2021', 'DD-MM-YYYY'),4,11);
```