

UNIVERSITATEA DIN BUCURESTI
FACULTATEA DE MATEMATICA SI INFORMATICA
CALCULATOARE SI TEHNOLOGIA INFORMATIEI

PROIECT BAZE DE DATE

PROFESOR COORDONATOR:
VASILE SILVIU-LAURENTIU

STUDENT:
CARAVAN MIHAI-COSMIN

BUCURESTI 2024

Baza de date a unui magazin second-hand
- ElectroDelicii -

PROFESOR COORDONATOR:
VASILE SILVIU-LAURENTIU

STUDENT:
CARAVAN MIHAI-COSMIN

BUCURESTI 2024

CUPRINS

Baza de date a unui magazin second-hand	1
Prezentarea modelului (din lumea reala) si a regulilor acestuia	4
Modelul din lumea reala si motivatia alegerii acestuia	4
Regulile modelului	4
Diagrama Entitate Relatie	5
Descrierea entitatilor, atributelor, cheilor, relatiilor si a cardinalitatilor	6
PRODUSE	6
MAGAZINE	6
PRETURI (TABEL INTERMEDIAR)	7
ANGAJATI	7
ISTORIC_JOBURI (TABEL INTERMEDIAR)	8
JOBURI	8
FURNIZORI	8
DONATII (TABEL INTERMEDIAR)	9
CLIENTI	9
TRANZACTII (TABEL INTERMEDIAR)	10
CATEGORII	10
PRODUSE_CATEGORII (TABEL INTERMEDIAR)	10
Diagrama Conceptuala	12
Descrierea constrangerilor de integritate	13
CATEGORII	13
FURNIZORI	13
PRODUSE	13
JOBURI	14
MAGAZINE	14
DONATII	14
PRETURI	15
CLIENTI	15
TRANZACTII	15
ANGAJATI	15

ISTORIC_JOBURI	16
PRODUSE_CATEGORII	16
Descrierea constrangerii ON DELETE	16
Schemele relationale	17
Script SQL	18

Prezentarea modelului (din lumea reala) si a regulilor acestuia

Modelul din lumea reala si motivatia alegerii acestuia

Tema aleasa de mine pentru proiectul la materia Baze de date este "Baza de date a unui lant de magazine second-hand", numind acesta magazin "ElectroDelicii", un joc de cuvinte ales la intamplare dupa cuvantul "Electronicii".

Am ales sa ma implic in proiectul unei baze de date pentru magazinul second-hand datorita pasiunii mele de lunga durata pentru vechituri si obiecte second-hand. Aceasta pasiune m-a dus la o descoperire a unicitatii si istoriei ascunse din spatele fiecarui obiect vechi. De-a lungul catorva ani, am dezvoltat o intelegera profunda a valorii obiectelor vechi si am dobandit abilitati practice in identificarea, evaluarea si restaurarea acestora.

Prin consolidarea acestor informatii intr-o baza de date bine structurata, intionez sa creez un instrument util pentru iubitorii de vechituri si colectionari. Astfel, magazinul "ElectroDelicii" va deveni nu doar un simplu spatiu de vanzare, ci si un hub cultural unde pasionatii vor putea explora si descoperi comori din trecut.

Regulile modelului

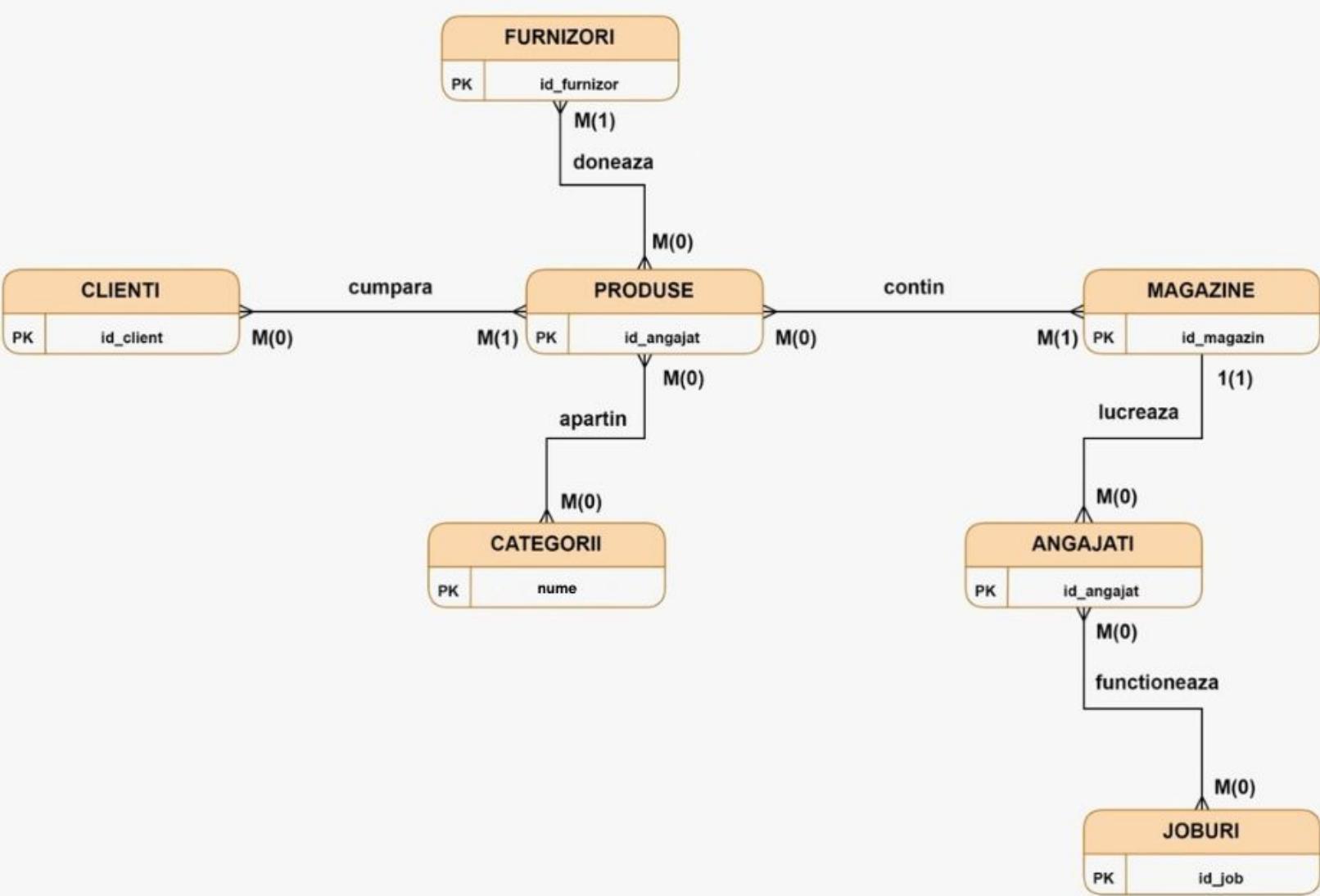
Consideram clienti doar persoanele care au cumparat cel putin un produs din lantul de magazine. De asemenea clientul nu poate returna un produs cumparat din magazin, contrar legii privind vanzarea produselor.

La deschiderea magazinului, acesta nu are angajati, dar ei se pot alatura echipei printr-un interview, si daca vor demisiona nu mai au voie sa se angajeze in alt magazin din lantul nostru, astfel istoricul va retine data la care angajatii actuali au venit sa lucreze in magazin sau doar o data de angajare si demisie pentru fiecare angajat care a plecat din echipa noasta. Angajatul poate ocupa mai multe posturi, astfel poate sa apara de 2 ori in istoricul joburilor.

Majoritatea produselor, provenind de la furnizori, sunt uzate sau chiar defecte. Există și excepții când ele sunt noi. Un produs poate fi cumpărat de mai mulți clienti, sau să ramane în magazin.

O categorie poate să cuprindă mai multe subcategorii care la rândul lor să cuprindă alte subcategorii. Am luat acest exemplu de la magazinele online.

Diagrama Entitate Relatie



Descrierea entitatilor, atributelor, cheilor, relatiilor si a cardinalitatilor

PRODUSE

Descriere entitate:

Produsul reprezinta bunul material care se va afla in magazin.

Descriere atribute:

nume - numele reprezentativ al produsului respectiv

stare - starea calitatii produsului

Chei:

id_produs (PK) - cheie primara pentru PRODUSE, un numar unic in functie de stocul magazinului cu produse

Relatii si cardinalitati:

Cu tabelul PRETURI - Un produs poate avea mai multe preturi in functie de magazinul in care se afla. M(1)

Cu tabelul TRANZACTII - Un produs poate fi in mai multe tranzactii sau niciuna. M(0)

Cu tabelul PRODUSE_CATEGORII - Un produs poate sa aiba mai multe categorii sau niciuna. M(0)

Cu tabelul DONATII - Un produs poate sa provina din mai multe donatii. M(1)

MAGAZINE

Despre entitate:

Locul unde clientii pot achizitiona produse.

Descriere atribute:

nume - numele care reprezinta magazinul

judet - judetul in care se afla magazinul

oras - orasul in care se afla magazinul

strada - strada la care se afla magazinul

numar - numarul la care se afla magazinul

cod_postal - codul postal al locatiei magazinului

Chei: **id_magazin (PK)** - cheie primara pentru MAGAZINE, id-ul unic si nenul care reprezinta magazinul

Relatii si cardinalitati:

Cu tabelul PRETURI - Un magazin poate avea mai multe preturi in functie de produsele disponibile, exista magazine fara produse. M(0)

Cu tabelul ANGAJATI - Un magazin poate sa aiba mai multi angajati sau niciunul. M(0)

PRETURI (TABEL INTERMEDIAR)

Descriere entitate:

Reprezinta suma de bani pe care clientul o da in functie de magazin si produs.

Descriere atribute:

suma - valoare in bani a produsului dintr-un anume magazin

Chei:

magazin (PK, FK) - cheie primara pentru tabelul PRETURI, magazinul in care se gaseste pretul, face legatura cu tabelul MAGAZINE

produs (PK, FK) - cheie primara pentru tabelul PRETURI, produsul cu acel pret, face legatura cu tabelul PRODUSE

Relatii si cardinalitati:

Cu tabelul PRODUSE - Un pret este atribuit obligatoriu unui produs. 1(1)

- Un produs poate avea mai multe preturi. M(1)

Cu tabelul MAGAZINE - Un pret al unui produs corespunde cu un magazin. 1(1)

- Un magazin poate sa nu aiba preturi daca nu are produse. M(0)

ANGAJATI

Despre entitate:

Persoana care lucreaza in cadrul magazinului pentru a indeplini anumite roluri si responsabilitati.

Descriere atribute:

nume - numele angajatului

prenume - prenumele angajatului

salariu - suma de bani pe care o primeste in schimbul serviciilor prestate

data_angajare - data cand angajatul a primit jobul

Chei:

id_angajat (PK) - cheie primara pentru tabelul ANGAJATI, un numar unic ce reprezinta angajatul

magazin (FK) - locul unde lucreaza angajatul, face legatura cu tabelul MAGAZINE

Relatii si cardinalitati:

Cu tabelul MAGAZINE - Un angajat poate lucra doar la un singur magazin. 1(1)

Cu tabelul ISTORIC_JOBURI - Un angajat poate sa aiba sau nu un job anterior. M(0)

ISTORIC_JOBURI (TABEL INTERMEDIAR)

Despre entitate:

Locul unde putem vedea joburile anterioare ale unui angajat.

Descriere atribute:

data_demisie - data la care angajatul a demisionat

nume - numele complet al angajatului

data_angajare - data la care angajatul a inceput sa lucreze

Chei:

angajat (PK, FK) - face legatura cu tabelul ANGAJATI

job (PK, FK) - face legatura cu tabelul JOBURI

Relatii si cardinalitati:

Cu tabelul ANGAJATI - Un angajat poate avea mai multe joburi anterioare sau niciunul. M(0)

- Istorul este corespunzator unui angajat. 1(1)

Cu tabelul JOBURI - Un job poate avea multi angajati anteriori. M(0)

- Istorul este corespunzator unui job. 1(1)

JOBURI

Despre entitate:

Pozitia pe care un angajat o ocupa.

Descriere atribute:

nume - numele jobului

salariu_minim - cel mai mic salariu de la job

salariu_maxim - cel mai mare salariu de la job

Chei:

id_job (PK) - cheie primara pentru MAGAZINE, id-ul unic si nenul care reprezinta jobul

Relatii si cardinalitati:

Cu tabelul ISTORIC_JOBURI - Un job poate sa nu sa aiba angajati cu un istoric. M(0)

FURNIZORI

Despre entitate:

Companiile care asigura magazinul cu produse.

Descriere atribute:

nume - numele care reprezinta furnizorul
judet - judetul in care se afla furnizorul
oras - orasul in care se afla furnizorul
strada - strada la care se afla furnizorul
numar - numarul la care se afla furnizorul
cod_postal - codul postal al locatiei furnizorul

Chei:

id_furnizor (PK) - cheie primara pentru tabelul FURNIZORI, codul numeric dat acestui furnizor prin care este identificat

Relatii si cardinalitati:

Cu tabelul DONATII - Un furnizor poate sau nu sa doneze produse magazinului. M(0)

DONATII (TABEL INTERMEDIAR)

Despre entitate:

Pozitia pe care un angajat o ocupa.

Descriere atribute:

data - data cand a fost facuta donatia

Chei:

furnizor (FK) - face legatura cu tabelul FURNIZORI
produs (FK) - face legatura cu tabelul PRODUSE
id_donatie (PK) - cheia primara a tabelului DONATII

Relatii si cardinalitati:

Ew'cdgnwn'HWTP K' QTK'/"Un furnizor poate sau nu doneze produse."M(0)
" /"Donatia"are obligatoriu un furnizor03*3+
Ew'cdgnwn'RTQF WUG /"Un produs provine doar din donatii.0O *1+
- Donatia are obligatoriu un produs. 1(1)

CLIENTI

Despre entitate:

Persoana care achizitioneaza produse dintr-un magazin.

Descriere atribute:

nume - numele clientului
prenume - prenumele clientului
nr_telefon - numarul de telefon al clientului

Chei:

id_client (PK) - cheie primara pentru tabelul CLIENTI, codul numeric pentru a identifica clientul

Relatii si cardinalitati:

Cu tabelul TRANZACTII - Un client poate avea una sau mai multe tranzactii. M(1)

TRANZACTII (TABEL INTERMEDIAR)

Despre entitate:

Schimbul de bunuri intre client si magazin.

Descrierea atributelor:

metoda_plata - modul in care a platit clientul

data - momentul in care a fost facut tranzactia

Chei:

id_tranzactie (PK) - cheie primara pentru TRANZACTII, id-ul unic si nenul care reprezinta tranzactia

client (FK) - face legatura cu tabelul CLIENTI

produs (FK) - face legatura cu tabelul PRODUSE

Relatii cu cardinalitatii:

Cu tabelul CLIENTI - Un client trebuie sa cumpere minim produs. M(1)

- Tranzactia are obligatoriu un client. 1(1)

Cu tabelul PRODUSE - Un produs poate sa nu fie vandut. M(0)

- Tranzactia are obligatoriu un produs. 1(1)

CATEGORII

Despre entitate:

Clasificarea produselor in functie de caracteristici comune.

Descrierea atributelor:

Chei:

nume (PK) - cheie primara pentru tabelul CATEGORII, un nume unic ce reprezinta categoria

parinte (FK) - in cazul in care o categorie face parte din alta categorie, face legatura cu tabelul CATEGORII

Relatii si cardinalitatii:

Cu tabelul PRODUSE_CATEGORII - O categorie poate sa contine mai multe produse sau niciunul. M(0)

PRODUSE_CATEGORII (TABEL INTERMEDIAR)

Despre entitate:

Retine produsele si categoriile din care fac parte la data la care s-au adaugat.

Descrierea atributelor:

data - momentul cand un produs a fost atribuit unei categorii

Chei:

produs(PK, FK) - cheie primara pentru tabelul PRODUSE_CATEGORII, face legatura cu tabelul PRODUSE

categorie(PK, FK) - cheie primara pentru tabelul PRODUSE_CATEGORII, face legatura cu tabelul CATEGORII

Relatii si cardinalitati:

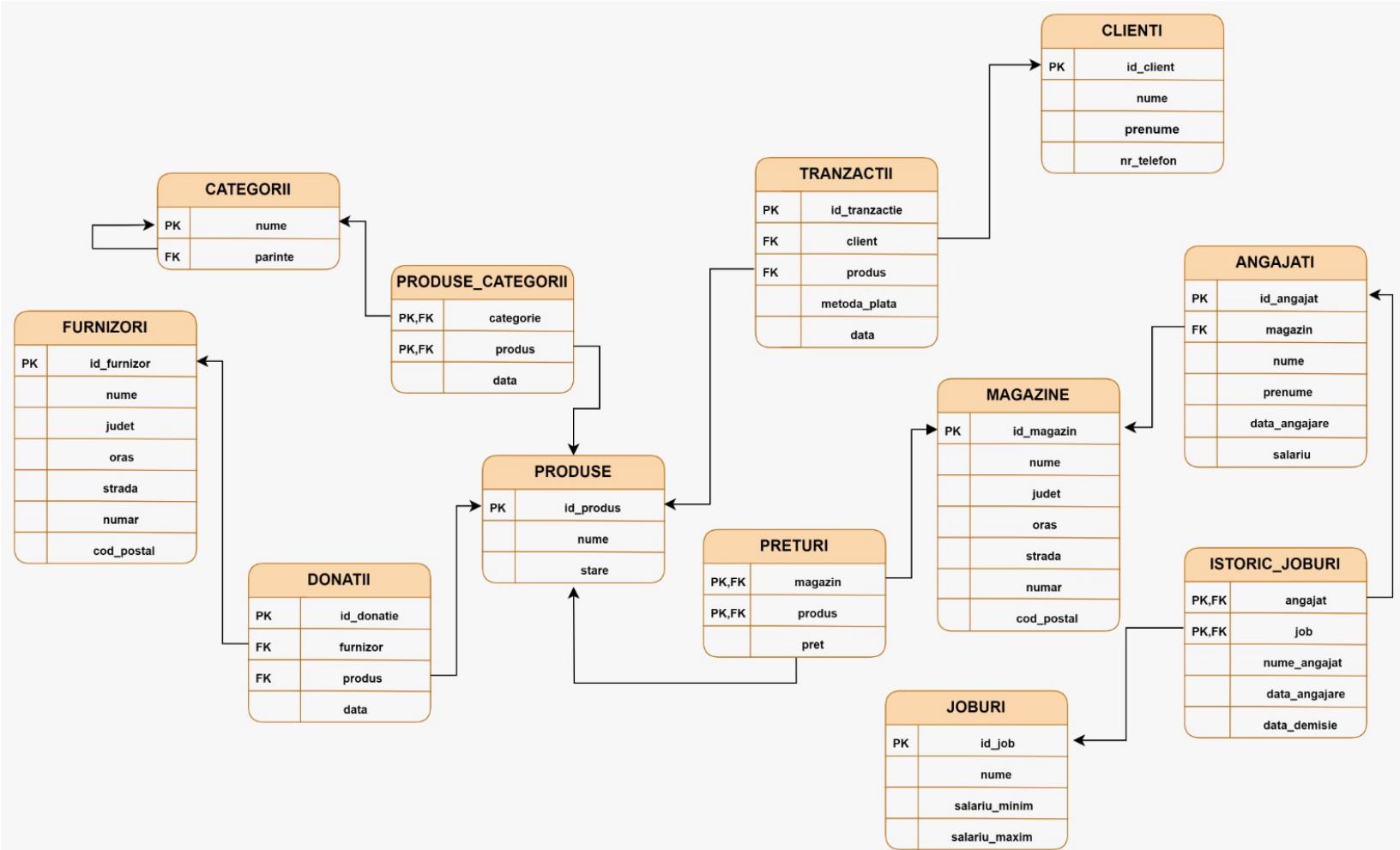
Cu tabelul PRODUSE - Un produs poate avea mai multe categorii. M(0)

- Produsul trebuie sa fie grupat dupa o categorie in acest tabel. 1(1)

Cu tabelul CATEGORII - O categorie poate avea mai multe produse. M(1)

- O categorie trebuie sa aiba un produs. 1(1)

Diagrama Conceptuala



Descrierea constrangerilor de integritate:

CATEGORII

nume **VARCHAR2(20) - PRIMARY KEY**, numele categoriei de maxim 20 caractere
CHECK(LENGTH(nume) >= 2)
parinte **VARCHAR(20) - FOREIGN KEY** face legatura cu tabelul CATEGORII
CHECK(LENGTH(parinte) >= 2)

FURNIZORI

id_furnizor **NUMBER(2) - PRIMARY KEY**, id-ul furnizorului **CHECK (id_furnizor >= 0)**
nume **VARCHAR2(20) - NOT NULL**, vreau ca furnizorul sa aiba un nume
CHECK (LENGTH(nume) >= 2)
judet **VARCHAR2(20) - NOT NULL**, vreau ca furnizorul sa aiba un judet
CHECK (LENGTH(judet) >= 2)
oras **VARCHAR2(20) - NOT NULL**, vreau ca furnizorul sa aiba un oras
CHECK (LENGTH(oras) >= 4)
strada **VARCHAR2(20) - NOT NULL**, vreau ca furnizorul sa aiba un strada
CHECK (LENGTH(strada) >= 4)
numar **VARCHAR2(10) - NOT NULL**, furnizorul va avea un numar si am pus atributul
VARCHAR2 pentru a avea de exemplu '3B'
cod_postal **VARCHAR2(6) - NOT NULL**, codul_postal al furnizorului este format doar din
cifre si are lungimea de 6 caractere **CHECK (REGEXP_LIKE(cod_postal, '^[0-9]{6}\$'))**
CONSTRAINT uq_locatie UNIQUE (nume, judet, oras, strada, numar, cod_postal) -
furnizorul se poate afla la o singura locatie cu un singur nume

PRODUSE

id_produs **NUMBER(3) - PRIMARY KEY**, id-ul produsului **CHECK (id_furnizor >= 0)**
nume **VARCHAR2(20) - NOT NULL**, numele produsului
stare **VARCHAR2(10) - DEFAULT 'Utilizat'**, orice produs de la un furnizor care nu are o
stare specificata este utilizat **CHECK (stare IN('Defect', 'Utilizat', 'Nou'))**
CONSTRAINT uq_produs UNIQUE (nume, stare) - pentru nu a adauga produse similare
numele si starea si unice

JOBURI

id_job **NUMBER(2) - PRIMARY KEY**, id-ul jobului **CHECK (id_job >= 0)**

nume **VARCHAR2(20) - UNIQUE NOT NULL**, numele jobului este unic

salariu_minim **NUMBER(6) - NOT NULL**, salariul trebuie sa fie minim 2000

CHECK (salariu_minim >= 2000)

salariu_maxim **NUMBER(6) - NOT NULL**

CONSTRAINT chk_salariu_maxim CHECK (salariu_maxim > salariu_minim) -

salariul maxim trebuie sa fie mai mare ca cel minim

MAGAZINE

id_magazin **NUMBER(2) - PRIMARY KEY**, id-ul magazinului **CHECK (id_magazin >= 0)**

nume **VARCHAR2(30) - DEFAULT 'ElectroDelicii'**, numele magazinului contine 'ElectroDelicii' **CHECK (REGEXP_LIKE(ume, 'ElectroDelicii', 'i'))**

judet **VARCHAR2(20) - NOT NULL**, vreau ca magazinul sa aiba un judet **CHECK (LENGTH(judet) >= 2)**

oras **VARCHAR2(20) - NOT NULL**, vreau ca magazinul sa aiba un oras **CHECK (LENGTH(oras) >= 4)**

strada **VARCHAR2(20) - NOT NULL**, vreau ca magazinul sa aiba un strada **CHECK (LENGTH(strada) >= 4)**

numar **VARCHAR2(10) - NOT NULL**, magazinul va avea un numar si am pus atributul VARCHAR2 pentru a avea, de exemplu '3C'

cod_postal **VARCHAR2(6) - NOT NULL**, codul_postal al magazinului este format doar din cifre si are lungimea de 6 caractere **CHECK (REGEXP_LIKE(cod_postal, '^[0-9]{6}\$'))**

CONSTRAINT uq_magazin UNIQUE (judet, oras, strada, numar, cod_postal) - magazinul se poate afla la o singura locatie

DONATII

id_donatie **NUMBER(4) - PRIMARY KEY**, id-ul donatiei **CHECK (id_donatie >= 0)**

furnizor **NUMBER(2) - FOREIGN KEY**, face legatura cu tabelul FURNIZORI

produs **NUMBER(2) - FOREIGN KEY**, face legatura cu tabelul PRODUSE

data **DATE - DEFAULT CURRENT_TIMESTAMP**, vreau sa retin timpul cand a fost facuta o donatie, iar daca nu este introdusa atunci ea este egala cu timpul curent

PRETURI

magazin **NUMBER(2) - FOREIGN KEY**, face legatura cu tabelul MAGAZINE
produs **NUMBER(2) - FOREIGN KEY**, face legatura cu tabelul PRODUSE
suma **NUMBER(7, 2)**, pretul unui produs difera de la magazin la magazin
CHECK (suma >= 1)

CONSTRAINT pk_preturi **PRIMARY KEY (magazin, produs)**, cheia primara acestui tabel este compusa din id-ul magazinului si id-ul produsului

CLIENTI

id_client **NUMBER(4) - PRIMARY KEY**, id-ul clientului **CHECK (id_client >= 0)**

nume **VARCHAR2(20) - NOT NULL**, numele clientului

prenume **VARCHAR2(20) - NOT NULL**, prenumele clientului

nr_telefon **VARCHAR2(10) - NOT NULL**, numarul de telefon al clientului, numarul il verific sa fie cu prefix de Romania **CHECK (REGEXP_LIKE(nr_telefon, '^[0-9]{10}\$') AND SUBSTR(nr_telefon, 1, 2) = '07')**

CONSTRAINT uq_clienti **UNIQUE (nume, prenume, nr_telefon)**, vom salva doar clientii cu nume, prenume si numar de telefon unice

TRANZACTII

id_tranzactie **NUMBER(5) - PRIMARY KEY**, id-ul tranzactiei **CHECK (id_tranzactie >= 0)**

client **NUMBER(4) - FOREIGN KEY**, face legatura cu tabelul CLIENTI

produs **NUMBER(3) - FOREIGN KEY**, face legatura cu tabelul PRODUSE

metoda_plata **VARCHAR2(5) - DEFAULT 'Cash'**, magazinul accepta doar 2 forme de plata
CHECK (metoda_plata IN ('Cash', 'Card'))

data **DATE - DEFAULT CURRENT_TIMESTAMP**, vreau sa retin timpul cand a fost facuta o tranzactie, iar daca nu este introdusa atunci ea este egala cu timpul curent

ANGAJATI

id_angajat **NUMBER(3) - PRIMARY KEY**, id-ul angajatului **CHECK (id_angajat >= 0)**

magazin **NUMBER(2) - FOREIGN KEY**, face legatura cu tabelul MAGAZINE

nume **VARCHAR2(20) - NOT NULL**, numele angajatului

prenume **VARCHAR2(20) - NOT NULL**, prenumele angajatului

data_angajare **DATE - DEFAULT CURRENT_TIMESTAMP**, vreau sa retin timpul cand angajatul a primit jobul, iar daca nu este introdusa atunci ea este egala cu timpul curent

salariu **NUMBER(6) - NOT NULL**, salariul unui angajat **CHECK (salariu >= 2000)**

CONSTRAINT uq_angajat **UNIQUE (nume, prenume)**, angajatii au nume unice pentru a putea retine in in tabelul ISTORIC_JOBURI angajatii care nu mai lucreaza intr-un magazin

ISTORIC_JOBURI

data_angajare **DATE - DEFAULT CURRENT_TIMESTAMP NOT NULL**
angajat **NUMBER(4) - FOREIGN KEY**, face legatura cu tabelul ANGAJATI
nume_angajat **VARCHAR2(40) - NOT NULL**, numele complet al angajatului
job **NUMBER(2) - FOREIGN KEY**, face legatura cu tabelul JOBURI

data_demisie **DATE**

CONSTRAINT chk_data_demisie **CHECK ((data_demisie IS NULL) OR (data_demisie > data_angajare))**

CONSTRAINT pk_istoric **PRIMARY KEY (angajat, job)**, cheia primara acestui tabel este compusa din jobul si id-ul angajatului.

PRODUSE_CATEGORII

produs **NUMBER(3) - FOREIGN KEY**, face legatura cu tabelul PRODUSE
categorie **VARCHAR2(20) - FOREIGN KEY**, face legatura cu tabelul CATEGORII

data **DATE - DEFAULT CURRENT-TIMESTAMP**

CONSTRAINT pk_produse_categorii **PRIMARY KEY (produs, categorie)**, cheia primara acestui tabel este compusa din produs si numele categoriei.

Descrierea constrangerii ON DELETE

Am pus constrangeri de tipul **ON DELETE** pentru fiecare foreign key pentru a defini comportamentul in situatiile respective.

DONATII

CONSTRAINT fk_donatii_furnizor **FOREIGN KEY (furnizor) REFERENCES** furnizori(id_furnizor) **ON DELETE CASCADE** - atunci cand stergem un furnizor donatiile facute de el vor disparea din tabelul DONATII

CONSTRAINT fk_donatii_produs **FOREIGN KEY (produs) REFERENCES** produse(id_produs) **ON DELETE CASCADE** - atunci cand vom sterge un produs el va disparea din lista donatiilor

PRETURI

CONSTRAINT fk_preturi_magazin **FOREIGN KEY (magazin) REFERENCES** magazine(id_magazin) **ON DELETE CASCADE** - atunci cand un magazin se inchide lista cu preturi a produselor din acel magazin se vor sterge

CONSTRAINT fk_preturi_produs **FOREIGN KEY (produs) REFERENCES** produse(id_produs) **ON DELETE CASCADE** - cand un produs este sters pretul lui va fi sters

TRANZACTII

CONSTRAINT fk_tranzactii_client **FOREIGN KEY (client) REFERENCES** clienti(id_client) **ON DELETE CASCADE** - cand un client este sters atunci acesta va disparea si din lista de tranzactii

CONSTRAINT fk_tranzactii_produs **FOREIGN KEY (produs) REFERENCES** produse(id_produs) **ON DELETE CASCADE** - daca un produs nu mai exista atunci el va disparea din lista tranzactiilor

ANGAJATI

CONSTRAINT fk_angajati_magazin **FOREIGN KEY**(magazin) **REFERENCES** magazine(id_magazin) **ON DELETE CASCADE** - daca un magazin se inchide atunci si toti angajatii la acel magazin vor disparea

ISTORIC_JOBURI

CONSTRAINT fk_istoric_joburi_job **FOREIGN KEY** (job) **REFERENCES** joburi(id_job) **ON DELETE CASCADE** - daca un job dispare atunci si angajatii anteriori la el nu vor mai avea un istoric

CONSTRAINT fk_istoric_joburi_angajat **FOREIGN KEY** (angajat) **REFERENCES** angajati(id_angajat) **ON DELETE SET NULL** - vom pune NULL atunci cand stergem angajat din baza de date

PRODUSE_CATEGORII

CONSTRAINT fk_intermediar_produse **FOREIGN KEY** (produs) **REFERENCES** produse(id_produs) **ON DELETE CASCADE** - cand un produs este sters categoria din care facea parte acesta va disparea din aceasta entitate

CONSTRAINT fk_intermediar_categorie **FOREIGN KEY** (categorie) **REFERENCES** categorii(nume) **ON DELETE CASCADE** - cand o categorie este stearsa atunci toate produsele grupate in ea vor disparea din acest tabel

SCHEMELE RELATIONALE

PRODUSE (id_produs#, nume, stare);

MAGAZINE (id_magazin#, nume, judet, oras, strada, numar, cod_postal);

PRETURI (magazin#, produs#, pret);

ANGAJATI (id_angajat#, magazin#, nume, prenume, data_angajare, salariu);

ISTORIC_JOBURI (angajat#, job#, nume_angajat, data_angajare, data_demisie);

JOBURI (id_job#, nume, salariu_minim, salariu_maxim);

FURNIZORI (id_furnizor#, nume, judet, oras, strada, numar, cod_postal);

DONATII (id_donatie#, furnizor#, produs#, data);

CLIENTI (id_client#, nume, prenume, nr_telefon0;

TRANZACTII (id_tranzactie#, client#, produs#, metoda_plata, data);

CATEGORII (nume#, parinte#);

PRODUSE_CATEGORII (categorie#, produs#, data).

SCRIPT SQL

Tabelul CATEGORII

Crearea tabelului:

```
CREATE TABLE categorii
( parinte VARCHAR2(20) DEFAULT NULL
, CONSTRAINT chk_parinte_lungime CHECK(LENGTH(parinte) >= 2)
, CONSTRAINT fk_categorii_nume FOREIGN KEY(parinte) REFERENCES
  categorii(nume) ON DELETE CASCADE
, nume VARCHAR2(20)
, CONSTRAINT pk_categorii PRIMARY KEY(nume)
, CONSTRAINT chk_categorie_lungime CHECK(LENGTH(nume) >= 2));
```

Inserarea datelor in tabel:

```
INSERT INTO categorii(nume, parinte) VALUES('Electronice', NULL);
INSERT INTO categorii(nume, parinte) VALUES('Desktop', 'Electronice');
INSERT INTO categorii(nume, parinte) VALUES('Tablete', 'Electronice');
INSERT INTO categorii(nume, parinte) VALUES('PC', 'Desktop');
INSERT INTO categorii(nume, parinte) VALUES('Monitoare', 'Desktop');
INSERT INTO categorii(nume, parinte) VALUES('Laptopuri', 'Desktop');
INSERT INTO categorii(nume, parinte) VALUES('Componente PC', 'PC');
INSERT INTO categorii(nume, parinte) VALUES('mini PC', 'PC');
INSERT INTO categorii(nume, parinte) VALUES('Aspiratoare', 'Electronice');
INSERT INTO categorii(nume, parinte) VALUES('Periferice', 'Desktop');
INSERT INTO categorii(nume, parinte) VALUES('Mouse-uri', 'Periferice');
INSERT INTO categorii(nume, parinte) VALUES('Tastaturi', 'Periferice');
INSERT INTO categorii(nume, parinte) VALUES('Casti', 'Periferice');
```

Tabelul PRODUSE_CATEGORII

Crearea tabelului:

```
CREATE TABLE produse_categorii
( produs NUMBER(3)
, CONSTRAINT fk_intermediar_produse FOREIGN KEY (produs) REFERENCES produse(id_produs) ON DELETE CASCADE
, categorie VARCHAR(20)
, CONSTRAINT pk_produse_categorii PRIMARY KEY (produs, categorie)
, CONSTRAINT fk_intermediar_categorie FOREIGN KEY (categorie) REFERENCES categorii(nume) ON DELETE CASCADE
, data DATE DEFAULT CURRENT_TIMESTAMP
);
```

Inserarea datelor in tabel:

```
INSERT INTO produse_categorii(produs, categorie, data) VALUES(0,'Laptopuri', TO_DATE('2024-01-01', 'YYYY-MM- DD'));
INSERT INTO produse_categorii(produs, categorie, data) VALUES(0,'Electronice', TO_DATE('2024-01-01', 'YYYY-MM- DD'));
INSERT INTO produse_categorii(produs, categorie, data) VALUES(1,'Desktop', TO_DATE('2024-01-01', 'YYYY-MM- DD'));
INSERT INTO produse_categorii(produs, categorie, data) VALUES(2,'Tablete', TO_DATE('2024-01-01', 'YYYY-MM- DD'));
INSERT INTO produse_categorii(produs, categorie, data) VALUES(3,'Laptopuri', TO_DATE('2024-01-01', 'YYYY-MM- DD'));
INSERT INTO produse_categorii(produs, categorie, data) VALUES(4,'Aspiratoare', TO_DATE('2024-01-01', 'YYYY-MM- DD'));
```

Tabelul FURNIZORI

Crearea tabelului:

```
CREATE TABLE furnizori
( id_furnizor NUMBER(2) CONSTRAINT pk_furnizori PRIMARY KEY CHECK (id_furnizor >= 0)
, nume VARCHAR2(20) NOT NULL CHECK (LENGTH(nume) >= 2)
, judet VARCHAR2(20) NOT NULL CHECK (LENGTH(judet) >= 2)
, oras VARCHAR2(20) NOT NULL CHECK (LENGTH(oras) >= 4)
, strada VARCHAR2(20) NOT NULL CHECK (LENGTH(strada) >= 4)
, numar VARCHAR2(10) NOT NULL
, cod_postal VARCHAR2(6) NOT NULL CHECK (REGEXP_LIKE(cod_postal, '^[0-9]{6}$'))
, CONSTRAINT uq_locatie UNIQUE (nume, judet, oras, strada, numar, cod_postal)
);
```

Inserarea datelor in tabel:

```
INSERT INTO furnizori(id_furnizor, nume, judet, oras, strada, numar, cod_postal) VALUES (0, 'HP', 'Iasi', 'Iasi', 'Palat', '7A', '700011');
INSERT INTO furnizori(id_furnizor, nume, judet, oras, strada, numar, cod_postal) VALUES (1, 'Dell', 'Vrancea', 'Focsani', 'Unirii', '25', '620125');
INSERT INTO furnizori(id_furnizor, nume, judet, oras, strada, numar, cod_postal) VALUES (2, 'Lenovo', 'Cluj', 'Cluj-Napoca', 'Memorandumului', '15', '400001');
INSERT INTO furnizori(id_furnizor, nume, judet, oras, strada, numar, cod_postal) VALUES (3, 'Acer', 'Timis', 'Timisoara', 'Libertatii', '7', '300105');
INSERT INTO furnizori(id_furnizor, nume, judet, oras, strada, numar, cod_postal) VALUES (4, 'Philips', 'Arges', 'Recea', 'Postei', '14', '173341');
```

Tabelul PRODUSE

Crearea tabelului:

```
CREATE TABLE produse
( id_produs NUMBER(3) CONSTRAINT pk_furnizori PRIMARY KEY CHECK (id_furnizor >= 0)
, nume VARCHAR2(20) NOT NULL
, stare VARCHAR2(10) DEFAULT 'Utilizat', CHECK (stare IN('Defect', 'Utilizat', 'Nou'))
, CONSTRAINT uq_produs UNIQUE (nume, stare)
);
```

Inserarea datelor in tabel:

```
INSERT INTO produse(id_produs, nume, stare) VALUES(0, 'Laptop cu SSD', 'Nou');
INSERT INTO produse(id_produs, nume, stare) VALUES(1, 'PC i7', 'Utilizat');
INSERT INTO produse(id_produs, nume, stare) VALUES(2, 'Tableta Android', 'Defect');
INSERT INTO produse(id_produs, nume, stare) VALUES(3, 'Gaming Laptop RGB', 'Nou');
INSERT INTO produse(id_produs, nume, stare) VALUES(4, 'Aspirator Cyclone', 'Utilizat');
```

Tabelul JOBURI

Crearea tabelului:

```
CREATE TABLE joburi
( id_job NUMBER(2)
, CONSTRAINT pk_id_job PRIMARY KEY (id_job)
, CONSTRAINT chk_id_job CHECK (id_job >= 0)
, nume VARCHAR2(20) UNIQUE NOT NULL
, salariu_minim NUMBER(6) NOT NULL CHECK (salariu_minim >= 2000)
, salariu_maxim NUMBER(6) NOT NULL
, CONSTRAINT chk_salariu_maxim CHECK (salariu_maxim > salariu_minim)
);
```

Inserarea datelor in tabel:

```
INSERT INTO joburi(id_job, nume, salariu_minim, salariu_maxim) VALUES(0, 'Comercial', 2400, 3000);
INSERT INTO joburi(id_job, nume, salariu_minim, salariu_maxim) VALUES(1, 'Curatenie', 3000, 3500);
INSERT INTO joburi(id_job, nume, salariu_minim, salariu_maxim) VALUES(2, 'Manager', 2800, 4000);
```

Tabelul MAGAZINE

Crearea tabelului:

```
CREATE TABLE magazine
( id_magazin NUMBER(2),
, CONSTRAINT pk_id_magazin PRIMARY KEY (id_magazin)
, CONSTRAINT chk_id_magazin CHECK (id_magazin >= 0)
, nume VARCHAR2(30) DEFAULT 'ElectroDelicii' CHECK (REGEXP_LIKE(nume, 'ElectroDelicii', 'i'))
, judet VARCHAR2(20) NOT NULL CHECK (LENGTH(judet) >= 2)
, oras VARCHAR2(20) NOT NULL CHECK (LENGTH(oras) >= 4)
, strada VARCHAR2(20) NOT NULL CHECK (LENGTH(strada) >= 4)
, numar VARCHAR2(10) NOT NULL
, cod_postal VARCHAR2(6) NOT NULL CHECK (REGEXP_LIKE(cod_postal, '^[0-9]{6}$'))
, CONSTRAINT uq_magazin UNIQUE (judet, oras, strada, numar, cod_postal)
);
```

Inserarea datelor in tabel:

```
INSERT INTO magazine(id_magazin, judet, oras, strada, numar, cod_postal) VALUES (0, 'Arges', 'Pitesti',
'Papucesti', '3B', '000321');
INSERT INTO magazine(id_magazin, judet, oras, strada, numar, cod_postal) VALUES (1, 'Valcea', 'Brezoi',
'Ciresului', '21', '173582');
INSERT INTO magazine(id_magazin, judet, oras, strada, numar, cod_postal) VALUES (2, 'Cluj', 'Cluj-Napoca',
'Tisa', '4A', '400071');
```

Tabelul DONATII

Crearea tabelului:

```
CREATE TABLE donatii
( id_donatie NUMBER(4) CONSTRAINT pk_donatii PRIMARY KEY CHECK (id_donatie >= 0)
, furnizor NUMBER(2)
, produs NUMBER(2)
, data DATE DEFAULT CURRENT_TIMESTAMP
, CONSTRAINT fk_donatii_furnizor FOREIGN KEY (furnizor) REFERENCES furnizori(id_furnizor) ON DELETE CASCADE
, CONSTRAINT fk_donatii_produs FOREIGN KEY (produs) REFERENCES produse(id_produs) ON DELETE CASCADE
);
```

Inserarea datelor in tabel:

```
INSERT INTO donatii(id_donatie, furnizor, produs, data) VALUES (0, 0, 0, TO_DATE('2023-06-12', 'YYYY-MM-DD'));
INSERT INTO donatii(id_donatie, furnizor, produs, data) VALUES (1, 0, 3, TO_DATE('2023-02-1', 'YYYY-MM-DD'));
INSERT INTO donatii(id_donatie, furnizor, produs, data) VALUES (2, 1, 1, TO_DATE('2021-04-8', 'YYYY-MM-DD'));
INSERT INTO donatii(id_donatie, furnizor, produs, data) VALUES (3, 2, 0, TO_DATE('2022-03-1', 'YYYY-MM-DD'));
INSERT INTO donatii(id_donatie, furnizor, produs, data) VALUES (4, 2, 3, TO_DATE('2022-04-26', 'YYYY-MM-DD'));
INSERT INTO donatii(id_donatie, furnizor, produs, data) VALUES (5, 4, 4, TO_DATE('2023-11-4', 'YYYY-MM-DD'));
INSERT INTO donatii(id_donatie, furnizor, produs) VALUES (6, 4, 2);
INSERT INTO donatii(id_donatie, furnizor, produs) VALUES (7, 4, 2);
INSERT INTO donatii(id_donatie, furnizor, produs, data) VALUES (8, 3, 2, TO_DATE('2020-01-19', 'YYYY-MM-DD'));
```

Tabelul PRETURI

Crearea tabelului:

```
CREATE TABLE preturi
( magazin NUMBER(2)
, produs NUMBER(2)
, suma NUMBER(7, 2) CHECK (suma >= 1)
, CONSTRAINT pk_preturi PRIMARY KEY (magazin, produs)
, CONSTRAINT fk_preturi_magazin FOREIGN KEY (magazin) REFERENCES magazine(id_magazin) ON DELETE CASCADE
, CONSTRAINT fk_preturi_produs FOREIGN KEY (produs) REFERENCES produse(id_produs) ON DELETE CASCADE
);
```

Inserarea datelor in tabel:

```
INSERT INTO preturi(magazin, produs, suma) VALUES (0, 0, 2100);
INSERT INTO preturi(magazin, produs, suma) VALUES (1, 0, 1800);
INSERT INTO preturi(magazin, produs, suma) VALUES (1, 1, 780);
INSERT INTO preturi(magazin, produs, suma) VALUES (1, 2, 210);
INSERT INTO preturi(magazin, produs, suma) VALUES (0, 2, 165.5);
INSERT INTO preturi(magazin, produs, suma) VALUES (0, 3, 3100);
INSERT INTO preturi(magazin, produs, suma) VALUES (0, 4, 320);
INSERT INTO preturi(magazin, produs, suma) VALUES (1, 4, 240);
```

Tabelul CLIENTI

Crearea tabelului:

```
CREATE TABLE clienti
( id_client NUMBER(4)
, CONSTRAINT pk_clienti PRIMARY KEY(id_client)
, CONSTRAINT chk_clienti CHECK (id_client >= 0)
, nume VARCHAR2(20) NOT NULL
, prenume VARCHAR2(20) NOT NULL
, nr_telefon VARCHAR2(10) NOT NULL CHECK (REGEXP_LIKE(nr_telefon, '^[0-9]{10}$') AND SUBSTR(nr_telefon, 1, 2) = '07')
, CONSTRAINT uq_clienti UNIQUE (nume, prenume, nr_telefon)
);
```

Inserarea datelor in tabel:

```
INSERT INTO clienti(id_client, nume, prenume, nr_telefon) VALUES (0, 'Popescu', 'Vasilica', '0711123456');
INSERT INTO clienti(id_client, nume, prenume, nr_telefon) VALUES (1, 'Vasilescu', 'Florină', '0766123456');
INSERT INTO clienti(id_client, nume, prenume, nr_telefon) VALUES (2, 'Popa', 'Maria', '0722123456');
INSERT INTO clienti(id_client, nume, prenume, nr_telefon) VALUES (3, 'Radu', 'Marin', '0733123456');
```

Tabelul TRANZACTII

Crearea tabelului:

```
CREATE TABLE tranzactii
( id_tranzactie NUMBER(4)
, CONSTRAINT pk_tranzactii PRIMARY KEY(id_tranzactie)
, CONSTRAINT chk_tranzactii CHECK (id_tranzactie >= 0)
, client NUMBER(4)
, CONSTRAINT fk_tranzactii_client FOREIGN KEY (client) REFERENCES clienti(id_client) ON DELETE CASCADE
, produs NUMBER(3)
, CONSTRAINT fk_tranzactii_produs FOREIGN KEY (produs) REFERENCES produse(id_produs) ON DELETE CASCADE
, metoda_plata VARCHAR2(5) DEFAULT 'Cash', CHECK (metoda_plata IN ('Cash', 'Card'))
, data DATE DEFAULT CURRENT_TIMESTAMP
);
```

Inserarea datelor in tabel:

```
INSERT INTO tranzactii(id_tranzactie, client, produs, metoda_plata, data) VALUES (0, 0, 1, 'Cash', TO_DATE('2023-01-19', 'YYYY-MM-DD'));
INSERT INTO tranzactii(id_tranzactie, client, produs, metoda_plata, data) VALUES (1, 0, 1, 'Card', TO_DATE('2023-01-19', 'YYYY-MM-DD'));
INSERT INTO tranzactii(id_tranzactie, client, produs, metoda_plata, data) VALUES (2, 1, 3, 'Cash', TO_DATE('2022-08-19', 'YYYY-MM-DD'));
INSERT INTO tranzactii(id_tranzactie, client, produs, metoda_plata, data) VALUES (3, 2, 4, 'Cash', TO_DATE('2023-11-10', 'YYYY-MM-DD'));
INSERT INTO tranzactii(id_tranzactie, client, produs, metoda_plata, data) VALUES (4, 3, 3, 'Cash', TO_DATE('2023-12-8', 'YYYY-MM-DD'));
INSERT INTO tranzactii(id_tranzactie, client, produs, metoda_plata, data) VALUES (5, 3, 2, 'Card', TO_DATE('2022-12-8', 'YYYY-MM-DD'));
```

Tabelul ANGAJATI

Crearea tabelului:

```
CREATE TABLE angajati
( id_angajat NUMBER(3)
, CONSTRAINT pk_angajat PRIMARY KEY(id_angajat)
, CONSTRAINT chk_angajati CHECK (id_angajat >=0)
, magazin NUMBER(2)
, CONSTRAINT fk_angajati_magazin FOREIGN KEY(magazin) REFERENCES magazine(id_magazin) ON DELETE CASCADE
, nume VARCHAR2(20) NOT NULL
, prenume VARCHAR2(20) NOT NULL
, data_angajare DATE DEFAULT CURRENT_TIMESTAMP
, salariu NUMBER(6) NOT NULL CHECK (salariu >= 2000)
, CONSTRAINT uq_angajat UNIQUE (nume, prenume)
);
```

Inserarea datelor in tabel:

```
INSERT INTO angajati(id_angajat, magazin, nume, prenume, salariu) VALUES (0, 0, 'Popescu', 'Ion', 2900);
INSERT INTO angajati(id_angajat, magazin, nume, prenume, data_angajare, salariu) VALUES (1, 0, 'Elmo', 'Marian', TO_DATE('2020-11-19', 'YYYY-MM-DD'), 3100);
INSERT INTO angajati(id_angajat, magazin, nume, prenume, data_angajare, salariu) VALUES (2, 0, 'Saif', 'Alexandra', TO_DATE('2022-9-11', 'YYYY-MM-DD'), 2700);
INSERT INTO angajati(id_angajat, magazin, nume, prenume, salariu) VALUES (3, 1, 'Ion', 'Popescu', 3000);
INSERT INTO angajati(id_angajat, magazin, nume, prenume, data_angajare, salariu) VALUES (4, 1, 'Sarbus', 'Alin', TO_DATE('2023-11-8', 'YYYY-MM-DD'), 3000);
```

Tabelul ISTORIC_JOBURI

Crearea tabelului:

```
CREATE TABLE istoric_joburi
( data_angajare DATE DEFAULT CURRENT_TIMESTAMP NOT NULL
, angajat NUMBER(4)
, CONSTRAINT pk_istoric PRIMARY KEY (angajat, job)
, nume_angajat VARCHAR2(40)NOT NULL
, CONSTRAINT fk_istoric_joburi_angajat FOREIGN KEY (angajat) REFERENCES angajati(id_angajat) ON DELETE SET NULL
, job NUMBER(2)
, CONSTRAINT fk_istoric_joburi_job FOREIGN KEY (job) REFERENCES joburi(id_job) ON DELETE CASCADE
, data_demisie DATE
, CONSTRAINT chk_data_demisie CHECK ((data_demisie IS NULL) OR (data_demisie > data_angajare))
);
```

Inserarea datelor in tabel:

```
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job) VALUES (TO_DATE('2020-11-19', 'YYYY-MM-DD'), 1, 'Elmo Marian', 0);
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job) VALUES (TO_DATE('2020-11-19', 'YYYY-MM-DD'), 1, 'Elmo Marian', 1);
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job) VALUES (TO_DATE('2022-9-11', 'YYYY-MM-DD'), 2, 'Saif Alexandra', 0);
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job) VALUES (TO_DATE('2023-11-8', 'YYYY-MM-DD'), 4, 'Sarbus Alin', 2);
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job) VALUES (TO_DATE('2024-01-6', 'YYYY-MM-DD'), 0, 'Popescu Ion', 2);
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job) VALUES (TO_DATE('2024-01-6', 'YYYY-MM-DD'), 3, 'Ion Popescu', 0);
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job) VALUES (TO_DATE('2024-01-6', 'YYYY-MM-DD'), 3, 'Ion Popescu', 1);
INSERT INTO istoric_joburi(data_angajare, angajat, nume_angajat, job, data_demisie) VALUES (TO_DATE('2022-9-11', 'YYYY-MM-DD'), 5, 'Mihai Cosmin', 2, TO_DATE('2023-11-18', 'YYYY-MM-DD'));
```