

## **GESTIUNEA BĂNCILOR DIN ROMÂNIA**

Conținutul acestui referat se referă la proiectarea unui model de date ce furnizează informații despre băncile din România.

Voi prezenta modelul de date, constrângerile pe care acesta le respectă, și utilitatea reală a acestuia.

Acesta va gestiona informații legate de organizarea băncilor, ajutând la o mai bună funcționare a acestora. Fiecare bancă are (cel puțin) un sediu, la care lucrează mai mulți angajați. Fiecare sediu se află la o locație, iar fiecare locație aparține unui oraș. În fiecare sediu se poate afla un ATM, iar de pe fiecare ATM se pot face extrageri cu cardul. Fiecare card are un tip (e.g. VISA, MASTERCARD). Băncile pot colabora cu diferite magazine (în general supermarketuri), care pot accepta plata cu unul sau cu mai multe tipuri de card de la o anumită bancă. De asemenea, băncile pot colabora cu diferiți furnizori de utilități. Mai mult de atât, la fiecare bancă se pot deschide de către clienți credite și conturi, iar conturile au, în general, un card asociat.

Modelul de date respectă următoarele restricții de funcționare:

- Un cont poate fi gestionat de pe maxim un card.
- Am considerat, pentru ușurință, că ATM-urile băncilor se pot afla exclusiv în sedii.
- Pentru localizarea unui angajat s-a folosit o singură adresă de bază.
- Există o bancă (BCR) care are angajați ce lucrează de la distanță.

### **Entități**

Pentru modelul de date prezentat mai devreme m-am folosit de următoarele entități:

BANCA, SEDIU, LOCATIE, ORAS, ANGAJAT, ANGAJAT\_INTERN,  
ANGAJAT\_AUX, CREDIT, CONT, CLIENT, CARD, TIP\_CARD, MAGAZIN,  
FURNIZOR\_UTILITATI, ATM.

BANCA = instituția financiară principală, cheia primară a acestei entități este „id\_banca”.

SEDIU = clădire unde băncile își desfășoară activitatea, atributul „id\_sediu” este cheia primară, iar cheile externe sunt „id\_banca” și „id\_locatie”, referind cheile externe ale tabelelor „BANCA”, respective „LOCATIE”.

LOCATIE = colecție de date ce oferă informații despre localizarea unui anume sediu, cheia primară fiind „id\_locatie”, iar cheia externă – „id\_oras”, referindu-se la cheia primară a entității oraș.

ORAS = orașul unei anumite locații, având cheia primară „id\_oras”.

**ANGAJAT** = persoană fizică, angajată a unei bănci, având cheia primară “*id\_angajat*” și cheia externă “*id\_sediu*”, ce referă cheia primară a tabelului “**SEDIU**”.

**ANGAJAT\_INTERN** = persoană fizică făcând parte din personalul bancar, având cheia primară “*id\_angajat*”, care este în același timp cheie externă, referind cheia primară a tabelului “**ANGAJAT**”.

**ANGAJAT\_AUX** = persoană fizică făcând parte din personalul auxiliar (e.g. curățenie), având cheia primară “*id\_angajat*”, care este în același timp cheie externă, referind cheia primară a tabelului “**ANGAJAT**”.

**CLIENT** = client la o anumită bancă, ce deține conturi sau credite, având cheia primară “*id\_client*”.

**CREDIT** = credit la o anumită bancă deținut de un anumit client, are cheia primară “*id\_credit*”, și cheile externe “*id\_banca*” și “*id\_client*” ce referă cheile primare ale entităților “**BANCA**” și “**CLIENT**”.

**CONT** = cont la o anumită bancă deținut de un anumit client are cheia primară “*id\_credit*”, și cheile externe “*id\_banca*” și “*id\_client*” ce referă cheile primare ale entităților “**BANCA**” și “**CLIENT**”.

**CARD** = card asociat unui cont, atributul “*id\_card*” reprezintă cheia primară, iar attributele “*id\_cont*” și “*id\_tip*” – cheile externe ale entităților “**CONT**” respectiv “**TIP\_CARD**”.

**TIP\_CARD** = tipul unui anumit card (e.g. VISA, MASTERCARD, MAESTRO), având cheia primară “*id\_tip*”.

**MAGAZIN** = magazin, în general supermarket, unde clienții pot face plăti cu cardul; cheia primară este “*id\_magazin*”.

**FURNIZOR\_UTILITATI** = furnizor de utilități unde clienții anumitor bănci pot plăti facturile direct la bancă, având cheia primară “*id\_furnizor*”.

**ATM** = bancomat de unde clienții pot face extrageri cu cardul, având cheia primară “*id\_atm*” și cheia externă “*id\_sediu*” ce referă cheia primară a tabelului sediu.

## **Relații**

Voi prezenta în continuare relațiile modelului de date, dând o descriere completă a fiecăreia.

**BANCA\_colaboreaza\_cu\_FURNIZOR\_UTILITATI** = relație care leagă entitățile **BANCA** și **FURNIZOR\_UTILITATI**, reflectând legătura dintre acestea (ce furnizor utilități colaborează cu ce bancă). Ea are cardinalitatea minimă de 1:1 (am considerat că o bancă lucrează cu cel puțin un furnizor de utilități, dar și că un furnizor de utilități

colaborează cu cel puțin o bancă) și cardinalitatea maximă de m:n (o bancă poate lucra cu mai mulți furnizori de utilități și un furnizor de utilități poate colabora cu mai multe bănci). Datorită cardinalității maxime (many-to-many), în diagrama conceptuală apare un nou tabel, numit “FURNIZOR\_BANCA”, care are cheia primară compusă din atributele “id\_furnizor” și “id\_banca” ce referă cheile primare ale tabelelor “FURNIZOR\_UTILITATI” și “BANCA”.

BANCA\_are\_SEDIU = relație care leagă entitățile BANCA și SEDIU, reflectând legătura dintre acestea (ce sediu aparține, cărei banchi), având o cardinalitate minimă de 1:0 (o bancă are sedii, dar cel puțin zero, deoarece pot exista și bănci online, iar un sediu existent aparține de exact o bancă).

SEDIU\_are\_LOCATIE = relație care leagă entitățile SEDIU și LOCAȚIE, reflectând legătura dintre acestea, având cardinalitatea minimă și maximă de 1:1 (fiecare sediu se află într-o singură locație, iar la fiecare locație înregistrată se află exact un sediu).

LOCAȚIE\_se\_afla\_in\_ORAS = relație care leagă entitățile LOCAȚIE și ORAȘ, reflectând legătura dintre acestea (ce locație se află în ce oraș), având cardinalitatea minimă de 1:1 (orice locație se află într-un oraș și fiecare oraș înregistrat este al unei locații înregisterate) și cardinalitatea maximă de m:1 (într-un oraș se pot afla mai multe dintre locațiile înregisterate).

ANGAJAT\_lucreaza\_in\_SEDIU = relație care leagă entitățile ANGAJAT și SEDIU, reflectând legătura dintre acestea (ce angajat lucrează la ce sediu), având cardinalitatea minimă de 1:1 (orice angajat lucrează la un sediu și la orice sediu lucrează cel puțin un angajat) și maximă de m:1 (într-un sediu pot lucra mai mulți angajați).

ANGAJAT\_isa\_ANGAJAT\_INTERN = relație de subordine între tabelele ANGAJAT și ANGAJAT\_INTERN.

ANGAJAT\_isa\_ANGAJAT\_AUX = relație de subordine între tabelele ANGAJAT și ANGAJAT\_AUX.

SEDIU\_are\_ATM = relație care leagă entitățile SEDIU și ATM, reflectând legătura dintre acestea (ce ATM se află la ce sediu), având cardinalitatea minimă de 1:0 (într-un sediu se poate să nu se afle niciun ATM) și maximă de 1:m (într-un sediu se pot afla mai multe ATM-uri).

ATM\_extrageri\_CARD = relație care leagă entitățile ATM și CARD, reflectând legătura dintre acestea, (după ce card a fost extras de la ce ATM) având cardinalitatea minimă de 0:0 (pot exista ATM-uri după care nu a fost extrasă nicio sumă, și carduri după care nu s-a extras de la niciun bancomat) și maximă de m:n (de pe un ATM se pot face extrageri de pe mai multe carduri, și de pe un card se poate extrage de la mai multe ATM-uri). Cardinalitatea maximă de tip many-to-many dă naștere unui nou tabel în diagrama conceptuală, numit “EXTRAGERI”. Cheia primară a acestuia este

“id\_extragere”, iar cheile externe sunt “id\_atm” și “id\_card”, referind cheile primare ale tabelelor “id\_atm” respectiv “id\_card”.

CARD\_are\_TIP\_CARD = relație care leagă entitățile CARD și TIP\_CARD având cardinalitatea minimă de 1:1 (fiecare card are un tip și pentru fiecare tip existent există o înregistrare a unui card de acel tip).

TIP\_CARD\_BANCA\_MAGAZIN\_accepta = relație de tip 3 care leagă entitățile BANCA, TIP\_CARD și MAGAZIN reflectând legătura dintre acestea, desemnând ce magazine acceptă plata cu ce tip de card de la ce bancă. Aceasta dă naștere unui nou tabel în diagrama conceptuală, numit “ACCEPTEA”, ce are cheia primară compusă (“id\_tip”, “id\_banca”, “id\_magazin”) referind cheile primare ale tabelelor “TIP\_CARD”, “BANCA” respectiv “MAGAZIN”.

BANCA\_are\_CONT = relație care leagă entitățile BANCA și CONT reflectând legătura dintre acestea (ce cont a fost deschis la ce bancă) având o cardinalitate minimă de 1:1 (fiecare cont aparține de o bancă și la fiecare bancă avem măcar un cont) și maximă de 1:m (la o bancă pot fi deschise mai multe conturi, iar un cont aparține de o singură bancă).

CONT\_are\_CARD = relație care leagă entitățile CONT și CARD reflectând legătura dintre acestea (ce card aparține cărui cont) având o cardinalitate minimă de 1:0 (nu orice cont are asociat un card) și maximă de 1:1 (în general, orice cont are un card asociat).

BANCA\_are\_CREDIT = relație care leagă entitățile BANCA și CREDIT reflectând legătura dintre acestea (ce credit este la ce bancă) având cardinalitatea minimă de 1:0 (putem avea bănci la care nu există niciun credit deschis) și maximă de 1:m (la o bancă putem avea mai multe credite deschise, iar un credit aparține de o singură bancă).

CLIENT\_detine\_CONT = relație care leagă entitățile CLIENT și CONT, reflectând legătura dintre acestea (ce client detine ce cont), având cardinalitatea minimă de 1:1 (s-a considerat că orice client al unei bănci are cel puțin un cont și de asemenea un cont aparține unui singur client) și maximă de 1:M (un client poate avea mai multe conturi).

CLIENT\_are\_CREDIT = relație care leagă entitățile CLIENT și CREDIT, reflectând relația dintre acestea (ce client detine ce credit) având cardinalitatea minimă de 1:0 (un client poate să nu aibă niciun credit, dar un credit aparține unui singur client) și maximă de 1:m (un client poate detine mai multe credite).

## Atribute

Acum voi prezenta atributele conținute de către tabelele noastre. (în ordine alfabetică)

ACCEPTEA:

- id\_banca (variabilă de tip numeric de maxim 4 cifre)
- id\_magazin (variabilă de tip numeric de maxim 4 cifre)
- id\_tip (variabilă de tip numeric de maxim 4 cifre)

ANGAJAT:

- id\_angajat (variabilă de tip numeric de maxim 4 cifre)
- id\_sediu (variabilă de tip numeric de maxim 4 cifre)
- nume (variabilă de tip caracter de lungime maxim 20)
- prenume (variabilă de tip caracter de lungime maxim 20)
- telefon (variabilă de tip caracter de lungime maxim 20)
- data\_angajare (variabilă de tip dată calendaristică)
- salariu (variabilă de tip numeric de maxim 4 cifre)

ANGAJAT\_INTERN:

- id\_intern (variabilă de tip numeric de maxim 4 cifre)
- comision (variabilă de tip numeric cu maxim 2 cifre și cu 2 zecimale)

ANGAJAT\_AUX:

- id\_aux (variabilă de tip numeric de maxim 4 cifre)

ATM:

- id\_atm (variabilă de tip numeric de maxim 4 cifre)
- id\_sediu (variabilă de tip numeric de maxim 4 cifre)

BANCA:

- id\_banca (variabilă de tip numeric de maxim 4 cifre)
- denumire\_banca (variabilă de tip caracter de lungime maxim 20)

CARD:

- id\_card (variabilă de tip numeric de maxim 4 cifre)

- id\_tip (variabilă de tip numeric de maxim 4 cifre)
- id\_cont (variabilă de tip numeric de maxim 4 cifre)
- numar\_card (variabilă de tip caracter de lungime maxim 20)
- cvv (variabilă de tip caracter de lungime maxim 4)
- limita\_extragere (variabilă de tip numeric de maxim 4 cifre)
- data\_exp (variabilă de tip dată calendaristică)

#### CLIENT:

- id\_client (variabilă de tip numeric de maxim 4 cifre)
- nume (variabilă de tip caracter de lungime maxim 20)
- prenume (variabilă de tip caracter de lungime maxim 20)
- cnp (variabilă de tip caracter de lungime maxim 20)
- email (variabilă de tip caracter de lungime maxim 20)

#### CONT:

- id\_cont (variabilă de tip numeric de maxim 4 cifre)
- id\_banca (variabilă de tip numeric de maxim 4 cifre)
- id\_client (variabilă de tip numeric de maxim 4 cifre)
- sold (variabilă de tip numeric de maxim 4 cifre)
- moneda (variabilă de tip caracter de lungime maxim 20)

#### CREDIT:

- id\_credit (numar de 4 cifre)
- id\_banca (numar de 4 cifre)
- id\_client (numar de 4 cifre)
- moneda (variabilă de tip caracter de lungime maxim 5)
- suma (variabilă de tip numeric de maxim 7 cifre)
- dobânda (variabilă de tip numeric de maxim 2 cifre cu 2 zecimale)

#### EXTRAGERI:

- id\_extragere (variabilă de tip numeric de maxim 4 cifre)
- id\_card (variabilă de tip numeric de maxim 4 cifre)
- id\_atm (variabilă de tip numeric de maxim 4 cifre)
- suma (variabilă de tip numeric de maxim 4 cifre)

#### FURNIZOR\_BANCA:

- id\_furnizor (variabilă de tip numeric de maxim 4 cifre)
- id\_banca (variabilă de tip numeric de maxim 4 cifre)

#### FURNIZOR\_UTILITATI:

- id\_furnizor (variabilă de tip numeric de maxim 4 cifre)
- nume\_furnizor (variabilă de tip caracter de lungime maxim 20)

#### LOCATIE:

- id\_locatie (variabilă de tip numeric de maxim 4 cifre)
- id\_oras (variabilă de tip numeric de maxim 4 cifre)
- cod\_postal (variabilă de tip caracter de lungime maxim 25)
- adresa\_strada (variabilă de tip caracter de lungime maxim 25)

#### MAGAZIN:

- id\_magazin (variabilă de tip numeric de maxim 4 cifre)
- nume\_magazin (variabilă de tip caracter de lungime maxim 20)

#### ORAS:

- id\_oras (variabilă de tip numeric de maxim 4 cifre)
- nume\_oras (variabilă de tip caracter de lungime maxim 20)

**SEDIU:**

- id\_sediu (variabilă de tip numeric de maxim 4 cifre)
- id\_banca (variabilă de tip numeric de maxim 4 cifre)
- id\_locatie (variabilă de tip numeric de maxim 4 cifre)
- telefon (variabilă de tip caracter de lungime maxim 20)

**TIP\_CARD:**

- id\_tip (variabilă de tip numeric de maxim 4 cifre)
- nume (variabilă de tip caracter de lungime maxim 20)

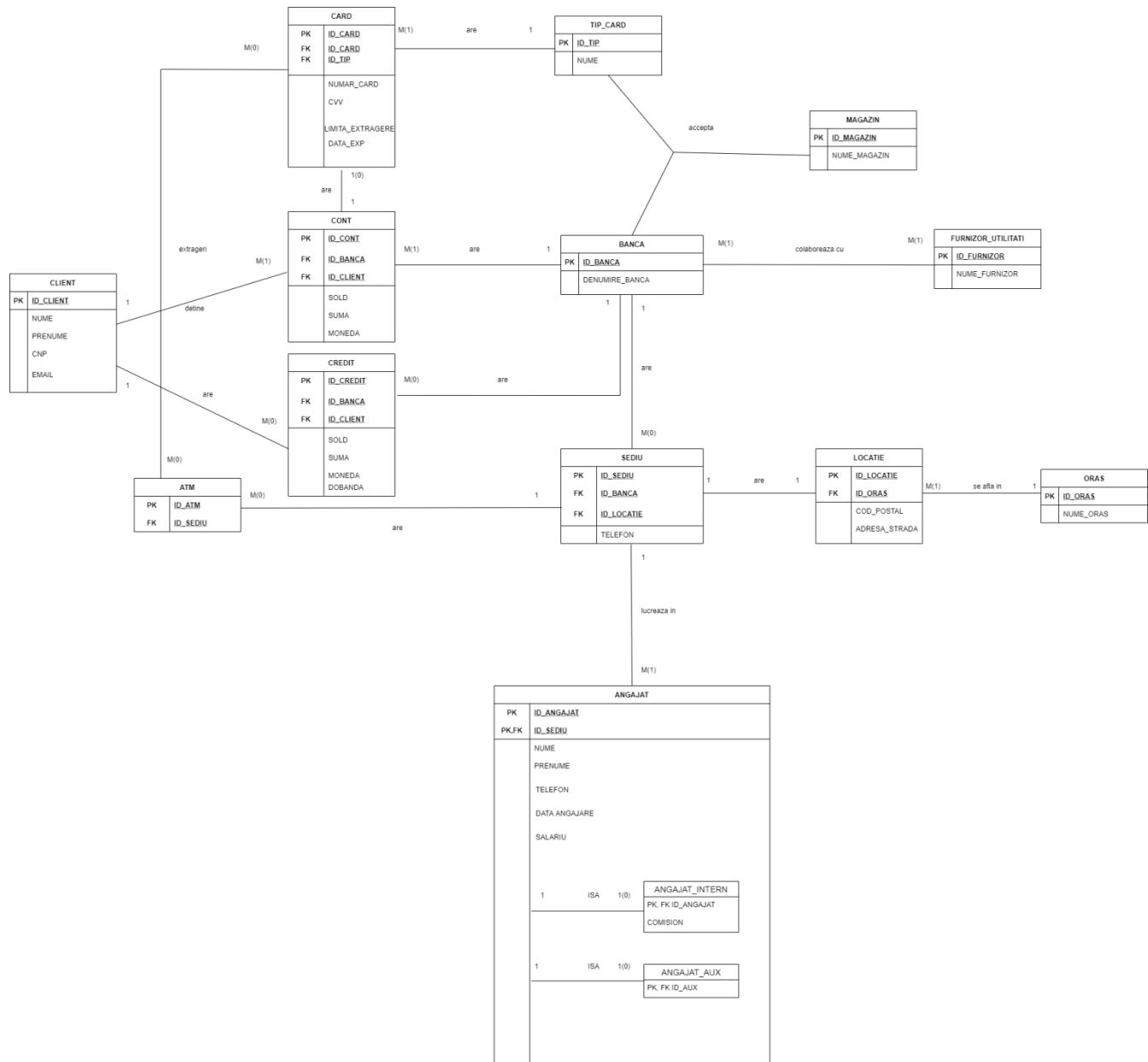
**SCHEMELE RELAȚIONALE CORESPUNZĂTOARE DIAGRAMEI CONCEPTUALE:**

- BANCA(#ID\_BANCA, DENUMIRE\_BANCA)
- CONT(#ID\_CONT, ID\_BANCA, ID\_CLIENT, SOLD, SUMA, MONEDA)
- CREDIT(#ID\_CREDIT, ID\_BANCA, ID\_CLIENT, SOLD, SUMA, MONEDA, DOBANDA)
- CLIENT(#ID\_CLIENT, NUME, PRENUME, CNP, EMAIL)
- CARD(#ID\_CARD, ID\_CONT, ID\_TIP, NUMAR\_CARD, CVV, LIMITA\_EXTRAGERE, DATA\_EXP)
- TIP\_CARD(#ID\_TIP, NUME)
- ACCEPTA(#ID\_TIP, #ID\_BANCA, #ID\_MAGAZIN)
- MAGAZIN(#ID\_MAGAZIN, NUME\_MAGAZIN)
- FURNIZOR\_UTILITATI(#ID\_FURNIZOR, NUME\_FURNIZOR)
- FURNIZOR\_BANCA(#ID\_BANCA, #ID\_FURNIZOR)
- SEDIU(#ID\_SEDIU, ID\_BANCA, ID\_LOCATIE, TELEFON)
- LOCATIE(#ID\_LOCATIE, ID\_ORAS, COD\_POSTAL, ADRESA\_STRADA)
- ORAS (#ID\_ORAS, NUME\_ORAS)

- ANGAJAT(#ID\_ANGAJAT, ID\_SEDIU, NUME, PRENUME, TELEFON, DATA\_ANGAJARE, SALARIU)
- ANGAJAT\_INTERN(#ID\_ANGAJAT, COMISION)
- ANGAJAT\_AUX(#ID\_AUX)
- ATM(#ID\_ATM, ID\_SEDIU)
- EXTRAGERI(#ID\_EXTRAGERE, ID\_ATM, ID\_CARD, SUMA)

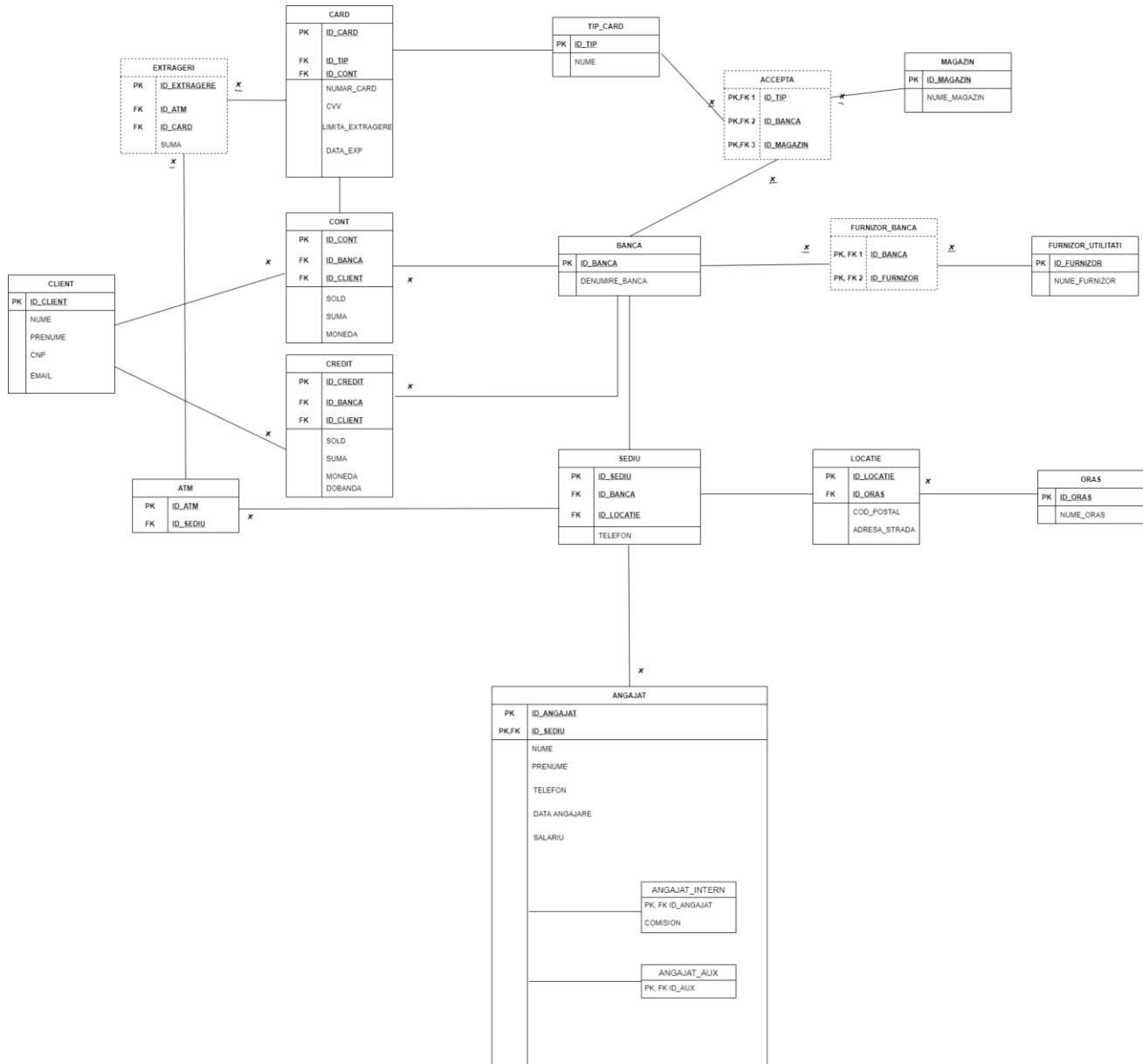
## EXERCIȚIUL 2

### Diagrama E/R



### EXERCIȚIUL 3

## Diagrama conceptuală



## EXERCIȚIUL 4 și EXERCIȚIUL 5

Voi prezenta mai jos codul pentru crearea și adăugarea înregistrărilor în fiecare dintre tabele.

### 1. BANCA

```
create table banca(
    id_banca number(4) constraint pk_id_banca primary key,
    denumire_banca varchar2(20) constraint null_den_banca not null,
    constraint unq_denumire_banca unique(denumire_banca)
);

select * from banca;

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (101.0, 'BCR');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (102.0, 'BRD');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (103.0, 'ING');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (104.0, 'BT');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (105.0, 'CEC');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (106.0, 'FB');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (107.0, 'ALPHA');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (108.0, 'RAIFEISSEN');
```

```

VALUES (106.0, 'FB');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (107.0, 'ALPHA');

INSERT INTO BANCA (ID_BANCA, DENUMIRE_BANCA)
VALUES (108.0, 'RAIFEISSEN');

select * from banca;

drop table furnizor_utilitati;
select * from oras;
create table furnizor_utilitati(

```

Query Result | All Rows Fetched: 8 in 0.027 seconds

	ID_BANCA	DENUMIRE_BANCA
1	101	BCR
2	102	BRD
3	103	ING
4	104	BT
5	105	CEC
6	106	FB
7	107	ALPHA
8	108	RAIFEISSEN

## 2. FURNIZOR\_UTILITATI

```

create table furnizor_utilitati(
    id_furnizor number(4) constraint pk_id_furnizor primary key,
    nume_furnizor varchar2(20) constraint null_nume_furnizor not null
constraint unq_nume_furniz unique
);

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (201.0, 'Orange');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (202.0, 'UPC');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (203.0, 'Vodafone');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (204.0, 'RCS RDS');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (205.0, 'ENGIE');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (206.0, 'Aquatim');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (207.0, 'CEZ');

```

```

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (208.0, 'Enel');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (209.0, 'Asirom');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (210.0, 'Generali');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (211.0, 'NN');

```

The screenshot shows the Oracle SQL Developer interface with the following details:

- Script Area:** Contains the SQL code for creating the table and inserting data.
- Script Output Tab:** Shows the results of the executed SQL statements.
- Query Result Tab:** Displays the final state of the FURNIZOR\_UTILITATI table.
- Table Data:** A grid showing the 11 rows inserted into the table.

```

select * from furnizor_utilitati;
create table furnizor_utilitati(
    id_furnizor number(4) constraint pk_id_furnizor primary key,
    nume_furnizor varchar2(20) constraint null_nume_furnizor not null constraint unq_nume_furniz unique
);

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (201.0, 'Orange');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (202.0, 'UPC');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)
VALUES (203.0, 'Vodafone');

INSERT INTO FURNIZOR_UTILITATI (ID_FURNIZOR, NUME_FURNIZOR)

```

ID_FURNIZOR	NUME_FURNIZOR
4	204 RCS RDS
5	205 ENGIE
6	206 Aquatim
7	207 CEZ
8	208 Enel
9	209 Asirom
10	210 Generali
11	211 NN

### **3. FURNIZOR\_BANCA**

```
create table furnizor_banca
(
    id_banca number(4) constraint fk_id_banca references banca(id_banca),
    id_furnizor number(4) constraint fk_id_furnizor references
furnizor_utilitati(id_furnizor),
    primary key(id_banca, id_furnizor)
);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (101.0, 201.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (101.0, 205.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (101.0, 206.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (101.0, 211.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (102.0, 202.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (102.0, 207.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (102.0, 206.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (102.0, 210.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (103.0, 201.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (103.0, 202.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (103.0, 208.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (103.0, 207.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (104.0, 202.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (104.0, 203.0);
```

```
INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (104.0, 209.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (105.0, 204.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (105.0, 205.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (105.0, 206.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (106.0, 201.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (106.0, 202.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (107.0, 201.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (107.0, 202.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (107.0, 203.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (107.0, 208.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (107.0, 209.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (108.0, 202.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (108.0, 203.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (108.0, 211.0);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (108.0, 204.0);
```

```

select * from furnizor_banca;
create table furnizor_banca
(
    id_banca number(4) constraint fk_id_banca references banca(id_banca),
    id_furnizor number(4) constraint fk_id_furnizor references furnizor_utilitati(id_furnizor),
    primary key(id_banca, id_furnizor)
);

INSERT INTO FURNIZOR_BANCA (ID_BANCA, ID_FURNIZOR)
VALUES (101.0, 201.0);

```

Script Output x Query Result x

SQL | All Rows Fetched: 29 in 0.008 seconds

ID_BANCA	ID_FURNIZOR
22	107
23	107
24	107
25	107
26	108
27	108
28	108
29	108
	202
	203
	208
	209
	202
	203
	204
	211

## 4. ORAS

```

create table oras(
    id_oras number(4) constraint pk_id_oras primary key,
    nume_oras varchar2(20) constraint nume_oras_null not null constraint
    unq_nume_oras unique
);

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (301.0, 'Bucuresti');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (302.0, 'Ploiesti');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (303.0, 'Pitesti');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (304.0, 'Slatina');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (305.0, 'Alba Iulia');

```

```

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (306.0, 'Craiova');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (307.0, 'Targu Jiu');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (308.0, 'Cluj Napoca');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (309.0, 'Timisoara');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (310.0, 'Suceava');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (311.0, 'Brasov');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (312.0, 'Satu Mare');

```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor containing the following SQL script:

```

select * from oras;
create table oras(
  id_oras number(4) constraint pk_id_oras primary key,
  nume_oras varchar2(20) constraint nume_oras_null not null constraint unq_nume_oras unique
);

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (301.0, 'Bucuresti');

INSERT INTO ORAS (ID_ORAS, NUME_ORAS)
VALUES (302.0, 'Ploiesti');

```

In the bottom-right pane, there is a results grid titled "Query Result" showing the data inserted into the table:

ID_ORAS	NUME_ORAS
1	301 Bucuresti
2	302 Ploiesti
3	303 Pitesti
4	304 Slatina
5	305 Alba Iulia
6	306 Craiova
7	307 Targu Jiu
8	308 Cluj Napoca

The status bar at the bottom indicates "All Rows Fetched: 12 in 0 seconds".

## 5. LOCATIE

```
create table locatie(
    id_locatie number(4) constraint pk_id_loc primary key,
    id_oras number(4) constraint fk_locatie_oras references oras(id_oras)
constraint null_oras_locatie not null,
    cod_postal varchar2(25) constraint null_cp not null,
    adresa_strada varchar2(25) constraint null_strada not null,
    constraint unq_cod_postal unique(cod_postal),
    constraint unq_adresa_strada unique(adresa_strada)
);

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (401.0, 301.0, '200001', 'Revolutiei 25');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (402.0, 302, '1002118', 'Eroilor 43');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (403.0, 303.0, '54431', 'Dezrobirii 12');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (404.0, 304.0, '118931', 'Calea Bucuresti 11');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (405.0, 305.0, '121210', 'Nanterre 89');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (406.0, 306.0, '251005', 'Mierlei 27');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (407.0, 301.0, '31200', 'Cocorului 12');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (408.0, 301.0, '6252', 'Crinului 14');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (409.0, 301.0, '14365', 'Cernele 128');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (410.0, 306.0, '514541', 'Brestei 4');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (411.0, 307.0, '608987', 'Alunilor 25');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (412.0, 308.0, '142641', 'Caracal 13');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (413.0, 308.0, '614152', 'Albinelor 88');
```

```

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (414.0, 309.0, '123515', 'Fulger 13');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (415.0, 310.0, '532112', 'Lacramioarei 13');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (416.0, 310.0, '151325', 'Becher 12');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (417.0, 311.0, '12531', 'Cantonului 22');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (418.0, 310.0, '9890', 'Plopului 12');

INSERT INTO LOCATIE (ID_LOCATIE, ID_ORAS, COD_POSTAL, ADRESA_STRADA)
VALUES (419.0, 312.0, '7089', 'Paringului 80');

```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a script editor containing the following SQL code:

```

from oras;

select * from locatie;
create table locatie(
    id_locatie number(4) constraint pk_id_loc primary key,
    id_oras number(4) constraint fk_locatie_oras references oras(id_oras) constraint null_oras_locatie not null,
    cod_postal varchar2(25) constraint null_cp not null,
    adresa_strada varchar2(25) constraint null_strada not null,
);

```

In the bottom-right pane, there is a query result grid titled "Query Result". It displays 19 rows of data from the locatie table:

	ID_LOCATIE	ID_ORAS	COD_POSTAL	ADRESA_STRADA
1	401	301 200001	Revolutiei 25	
2	402	302 1002118	Eroilor 43	
3	403	303 54431	Dezrobirii 12	
4	404	304 118931	Calea Bucuresti 11	
5	405	305 121210	Nanterre 89	
6	406	306 251005	Mierlei 27	
7	407	301 31200	Cocorului 12	
8	408	301 6252	Crinului 14	
9	409	301 14365	Cernele 128	
10	410	306 514541	Breste 4	
11	411	307 500007	Mihai Viteazul 25	

Below the grid, a status bar indicates "All Rows Fetched: 19 in 0 seconds".

## 6. SEDIU

```
create table sediu(
id_sediu number(4) constraint pk_id_sediu primary key,
id_banca number(4) constraint fk_sediu_banca_id references banca(id_banca)
constraint null_id_banca_sediu not null,
id_locatie number(4) constraint fk_locatie_sediu_id references
locatie(id_locatie) constraint null_sediu_locatie not null,
telefon varchar2(30) constraint telef_null not null,
constraint unq_tel unique(telefon)
);

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2000.0, 101.0, 401.0, '0289713190');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2001.0, 108.0, 402.0, '0341892133');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2002.0, 107.0, 403.0, '0251411335');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2003.0, 101.0, 404.0, '0231433299');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2004.0, 105.0, 405.0, '0261781230');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2005.0, 103.0, 406.0, '0231890765');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2006.0, 102.0, 407.0, '0387900099');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2007.0, 102.0, 408.0, '0341998000');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2008.0, 103.0, 409.0, '0288901239');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2009.0, 105.0, 410.0, '0309809801');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2010.0, 104.0, 411.0, '0309993012');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2011.0, 106.0, 412.0, '0234130981');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2012.0, 106.0, 413.0, '0298892009');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2013.0, 107.0, 414.0, '0231132909');
```

```
INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2014.0, 105.0, 415.0, '0245123019');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2015.0, 108.0, 416.0, '0289003902');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2016.0, 104.0, 417.0, '0287778992');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2017.0, 102.0, 418.0, '0287990003');

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2018.0, 103.0, 419.0, '0349000213');
```

```
select * from sediu;
create table sediu(
    id_sedu number(4) constraint pk_id_sedu primary key,
    id_banca number(4) constraint fk_sedu_banca_id references banca(id_banca) constraint null_id_banca_sedu not null,
    id_locatie number(4) constraint fk_locatie_sedu_id references locatie(id_locatie) constraint null_sedu_locatie not null,
    telefon varchar2(30) constraint telef_null not null,
    constraint unq_tel unique(telefon)
);

INSERT INTO SEDIU (ID_SEDIU, ID_BANCA, ID_LOCATIE, TELEFON)
VALUES (2000.0, 101.0, 401.0, '0289713190');
```

Script Output x Query Result x

SQL | All Rows Fetched: 19 in 0.005 seconds

	ID_SEDIU	ID_BANCA	ID_LOCATIE	TELEFON
1	2000	101	401	0289713190
2	2001	108	402	0341892133
3	2002	107	403	0251411335
4	2003	101	404	0231433299
5	2004	105	405	0261781230
6	2005	103	406	0231890765
7	2006	102	407	0387900099
8	2007	102	408	0341998000
9	2008	103	409	0288901239
10	2009	105	410	0309809801
11	2010	104	411	0306663012
12	2011	106	412	0306663013
13	2012	107	413	0306663014
14	2013	108	414	0306663015
15	2014	109	415	0306663016
16	2015	110	416	0306663017
17	2016	111	417	0306663018
18	2017	112	418	0306663019
19	2018	113	419	0306663020

## 7. MAGAZIN

```
create table magazin(
    id_magazin number(4) constraint pk_id_magazin primary key,
    nume_magazin varchar2(20) constraint unq_nume_magazin unique constraint
null_nume_magazin not null
);
INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (501.0, 'Auchan');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (502.0, 'Real');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (503.0, 'Kaufland');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (504.0, 'LIDL');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (505.0, 'Penny');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (506.0, 'Profi');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (507.0, 'Selgros');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (508.0, 'Metro');
```

```
select * from magazin;
```

```
create table magazin(
    id_magazin number(4) constraint pk_id_magazin primary key,
    nume_magazin varchar2(20) constraint unq_nume_magazin unique constraint
null_nume_magazin not null
);
INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (501.0, 'Auchan');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (502.0, 'Real');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (503.0, 'Kaufland');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (504.0, 'LIDL');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (505.0, 'Penny');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (506.0, 'Profi');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (507.0, 'Selgros');

INSERT INTO MAGAZIN (ID_MAGAZIN, NUME_MAGAZIN)
VALUES (508.0, 'Metro');
```

Script Output x | Query Result x | All Rows Fetched: 8 in 0 seconds

ID_MAGAZIN	NUME_MAGAZIN
1	Auchan
2	Real
3	Kaufland
4	LIDL
5	Penny
6	Profi
7	Selgros
8	Metro

## 8. CONT

```
create table cont(
    id_cont number(4) constraint pk_id_cont primary key,
    id_banca number(4) constraint fk_cont_banca references banca(id_banca),
    id_client number(4) constraint fk_cont_client references client(id_client),
    sold number(4) not null,
    moneda varchar(5) not null
);

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (601.0, 101.0, 1.0, 1200.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (602.0, 101.0, 1.0, 1300.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (603.0, 102.0, 2.0, 2500.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (604.0, 103.0, 2.0, 3000.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (605.0, 104.0, 3.0, 2000.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (606.0, 105.0, 4.0, 0.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (607.0, 106.0, 5.0, 0.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (608.0, 107.0, 6.0, 100.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (609.0, 108.0, 7.0, 2000.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (610.0, 101.0, 8.0, 6000.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (611.0, 102.0, 9.0, 3000.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (612.0, 103.0, 10.0, 2000.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (613.0, 104.0, 11.0, 8000.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (614.0, 105.0, 12.0, 1200.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (615.0, 106.0, 13.0, 3200.0, 'USD');
```

```
INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (616.0, 107.0, 14.0, 6500.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (617.0, 108.0, 15.0, 2450.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (618.0, 101.0, 16.0, 1230.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (619.0, 102.0, 17.0, 6520.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (620.0, 103.0, 18.0, 2000.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (621.0, 104.0, 19.0, 3000.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (622.0, 105.0, 20.0, 4000.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (623.0, 106.0, 21.0, 1500.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (624.0, 107.0, 22.0, 6000.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (625.0, 108.0, 23.0, 2500.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (626.0, 101.0, 24.0, 3000.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (627.0, 102.0, 25.0, 1500.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (628.0, 103.0, 26.0, 2500.0, 'EUR');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (629.0, 104.0, 27.0, 3000.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (630.0, 105.0, 28.0, 8990.0, 'USD');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (631.0, 106.0, 29.0, 6500.0, 'RON');

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)
VALUES (632.0, 107.0, 30.0, 1200.0, 'EUR');
```

```

select * from cont;
create table cont(
    id_cont number(4) constraint pk_id_cont primary key,
    id_banca number(4) constraint fk_cont_banca references banca(id_banca),
    id_client number(4) constraint fk_cont_client references client(id_client),
    sold number(4) not null,
    moneda varchar(5) not null
);

INSERT INTO CONT (ID_CONT, ID_BANCA, ID_CLIENT, SOLD, MONEDA)

```

Script Output x Query Result x

SQL | All Rows Fetched: 32 in 0.006 seconds

ID_CONT	ID_BANCA	ID_CLIENT	SOLD	MONEDA
1	601	101	1	1200 EUR
2	602	101	1	1300 RON
3	603	102	2	2500 USD
4	604	103	2	3000 EUR
5	605	104	3	2000 RON
6	606	105	4	0 USD
7	607	106	5	0 EUR
8	608	107	6	100 RON
9	609	108	7	2000 USD
10	610	101	8	6000 EUR
11	611	102	9	2000 RON

## 9. TIP\_CARD

```

create table tip_card(
    id_tip number(4) constraint pk_tip_card primary key,
    nume varchar2(20) constraint null_nume_tip not null
);

INSERT INTO TIP_CARD (ID_TIP, NUME)
VALUES (701.0, 'VISA');

```

```
INSERT INTO TIP_CARD (ID_TIP, NUME)
VALUES (702.0, 'MASTERCARD');

INSERT INTO TIP_CARD (ID_TIP, NUME)
VALUES (703.0, 'MAESTRO');

INSERT INTO TIP_CARD (ID_TIP, NUME)
VALUES (704.0, 'EUROCARD');

INSERT INTO TIP_CARD (ID_TIP, NUME)
VALUES (705.0, 'AMERICAN EXPRESS');
```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor containing the following SQL statements:

```
select *
from tip_card;

create table tip_card(
    id_tip number(4) constraint pk_tip_card primary key,
    nume varchar2(20) constraint null_nume_tip not null
);

TINSERT INTO TIP CARD (ID_TIP, NUME)
```

In the bottom-right pane, there is a "Query Result" tab showing the data from the table:

ID_TIP	NUME
1	701 VISA
2	702 MASTERCARD
3	703 MAESTRO
4	704 EUROCARD
5	705 AMERICAN EXPRESS

## 10. CARD

```
create table card(
    id_card number(4) constraint pk_card primary key,
    id_cont number(4) constraint fk_cont_card references cont(id_cont),
    numar_card varchar2(40) constraint null_numar_card not null constraint
unq_numar_card unique,
    cvv varchar2(4) constraint null_cvv not null,
    limita_extragere number(4),
    data_exp date constraint null_data not null
);

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (801.0, 601.0, '4401578900219789', '123', 2000.0, to_date('17-Jun-
2023'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (802.0, 602.0, '4401578900219790', '124', NULL, to_date('21-Sep-
2022'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (803.0, 603.0, '4378004578901234', '125', 1000.0, to_date('13-Jan-
2022'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (804.0, 604.0, '5401578900219789', '126', 4000.0, to_date('03-Jan-
2021'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (805.0, 605.0, '5401578900219790', '127', NULL, to_date('21-May-
2024'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (806.0, 606.0, '5378004578901234', '128', NULL, to_date('25-Jun-
2022'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (807.0, 607.0, '6401578900219789', '129', 2000.0, to_date('05-Feb-
2025'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (808.0, 608.0, '6401578900219790', '130', NULL, to_date('07-Feb-
2022'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
```

```

VALUES (809.0, 609.0, '6378004578901234', '131', 1500.0, to_date('17-Aug-2024'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (810.0, 611.0, '6401578989219789', '132', NULL, to_date('16-Aug-2024'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (811.0, 610.0, '6401578923219790', '133', NULL, to_date('28-Sep-2027'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (812.0, 612.0, '6378004577801234', '134', 4000.0, to_date('30-Sep-2027'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (813.0, 613.0, '7401578900219790', '135', NULL, to_date('07-Mar-2023'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (814.0, 614.0, '7378004578901234', '136', NULL, to_date('07-Dec-2023'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (815.0, 615.0, '7401578900219789', '137', NULL, to_date('07-Dec-2026'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (816.0, 616.0, '7401578900279790', '138', 3500.0, to_date('18-May-2025'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (817.0, 617.0, '7378004578921234', '139', NULL, to_date('24-Dec-2024'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (818.0, 618.0, '7401578900239789', '140', NULL, to_date('24-Jul-2025'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (819.0, 619.0, '7401578900249790', '141', NULL, to_date('15-Nov-2023'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (820.0, 620.0, '8378004578901234', '142', NULL, to_date('10-Aug-2022'));

```

```

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE,
DATA_EXP)
VALUES (821.0, 632.0, '6378904578901234', '123', NULL, to_date('18-Aug-
2022'));

VALUES (818.0, 618.0, '7401578900239789', '140', NULL, to_date('24-Jul-2025'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (819.0, 619.0, '7401578900249790', '141', NULL, to_date('15-Nov-2023'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (820.0, 620.0, '8378004578901234', '142', NULL, to_date('10-Aug-2022'));

INSERT INTO CARD (ID_CARD, ID_CONT, NUMAR_CARD, CVV, LIMITA_EXTRAGERE, DATA_EXP)
VALUES (821.0, 632.0, '6378904578901234', '123', NULL, to_date('18-Aug-2022'));

SELECT * FROM CARD;
commit;

select *

```

Script Output x Query Result x

SQL | All Rows Fetched: 21 in 0 seconds

	ID_CARD	ID_CONT	NUMAR_CARD	CVV	LIMITA_EXTRAGERE	DATA_EXP
1	801	601	4401578900219789	123		2000 17-JUN-23
2	802	602	4401578900219790	124	(null)	21-SEP-22
3	803	603	4378004578901234	125		1000 13-JAN-22
4	804	604	5401578900219789	126		4000 03-JAN-21
5	805	605	5401578900219790	127	(null)	21-MAY-24
6	806	606	5378004578901234	128	(null)	25-JUN-22
7	807	607	6401578900219789	129		2000 05-FEB-25
8	808	608	6401578900219790	130	(null)	07-FEB-22
9	809	609	6378004578901234	131		1500 17-AUG-24
10	810	611	6401578989219789	132	(null)	16-AUG-24
11	811	610	6401578923219790	133	(null)	28-SEP-27
12	812	612	6378004577801234	134		4000 30-SEP-27
13	813	613	7401578900219790	135	(null)	07-MAR-23
14	814	614	7378004578901234	136	(null)	07-DEC-23
15	815	615	7401578900219789	137	(null)	07-DEC-26

## 11. ACCEPTA

```
create table accepta(
    id_banca number(4) constraint fk_3_banca references banca(id_banca),
    id_magazin number(4) constraint fk_3_magazin references
magazin(id_magazin),
    id_tip number(4) constraint fk_3_tip references tip_card(id_tip),
    primary key(id_banca, id_magazin, id_tip)
);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (101.0, 502.0, 701.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (101.0, 503.0, 702.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (102.0, 502.0, 705.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (102.0, 501.0, 703.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (103.0, 504.0, 704.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (104.0, 502.0, 705.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (104.0, 503.0, 702.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (104.0, 501.0, 703.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (105.0, 501.0, 705.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (105.0, 508.0, 701.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (106.0, 507.0, 702.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (107.0, 506.0, 703.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (107.0, 506.0, 702.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (108.0, 507.0, 701.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (108.0, 508.0, 702.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
```

```

VALUES (108.0, 505.0, 704.0);

INSERT INTO ACCEPTA (ID_BANCA, ID_MAGAZIN, ID_TIP)
VALUES (108.0, 505.0, 703.0);

```

```

commit;

select *
from accepta;

create table client(

```

Script Output x Query Result x

All Rows Fetched: 17 in 0 seconds

	ID_BANCA	ID_MAGAZIN	ID_TIP
1	101	502	701
2	101	503	702
3	102	501	703
4	102	502	705
5	103	504	704
6	104	501	703
7	104	502	705
8	104	503	702
9	105	501	705
10	105	508	701
11	106	507	702
12	107	506	702
13	107	506	703
14	108	505	703
15	108	505	704

## 12. CLIENT

```

create table client(
    id_client number(4) constraint pk_client_id primary key,
    nume varchar2(20) constraint null_nume_client not null,
    prenume varchar2(20) constraint null_prenume_client not null,
    cnp varchar2(20) constraint null_cnp_client not null,
    email varchar2(40) constraint null_email_client not null,
    constraint unq_cnp unique(cnp),
    constraint unq_email unique(email)
);

```

```
INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (1.0, 'Ionescu', 'Marius', '1991113438211', 'marius2007@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (2.0, 'Popescu', 'Andrei', '1980213810923',
'andreipopescu@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (3.0, 'Oporanu', 'Alexandru', '1713841329092',
'alexoporanull@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (4.0, 'Ivan', 'Codrut', '1049873217712', 'codrut2009@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (5.0, 'Micu', 'Andreea', '2071439827893', 'micuandreea@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (6.0, 'Tutescu', 'Ana', '2432718091232', 'anaT@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (7.0, 'Radu', 'Ana Maria', '2709909809801', 'raduana12@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (8.0, 'Munteanu', 'Ionela', '2841791201913', 'IonelaM@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (9.0, 'Gabroveanu', 'Sofia', '2971923870104', 'sofiaG@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (10.0, 'Stefanescu', 'Maria', '2144323323422',
'stefanescuM@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (11.0, 'Iordache', 'Stefan', '1132809800999', 'stefan19@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (12.0, 'Tenea', 'Mariana', '2987438579232', 'mariana11@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (13.0, 'Iovanovici', 'Vlad', '1432452342338',
'vlad_iovanovici@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (14.0, 'Georgescu', 'Patric', '1986101012315',
'patric_georgescu@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (15.0, 'Petrescu', 'Ion', '1781318068695', 'ionPetrescu@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (16.0, 'Popescu', 'Mihai', '1502719549202 ', 'MihaiP@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (17.0, 'Protopopescu', 'Razvan', '1695862148007 ',
'razvan11@yahoo.com');
```

```
INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (18.0, 'Minea', 'Ionela', '2480876207694 ',
'ionela_ionela@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (19.0, 'Balseanu', 'Anca', '2778303607726 ', 'ancaB@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (20.0, 'Dinca', 'Victoria', '2394450567250', 'dincaV@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (21.0, 'Stan', 'Petre', '2413213242101', 'petre_Stan@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (22.0, 'Nica', 'Sorin', '1714776352527', 'sorin_nica@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (23.0, 'Staicu', 'Eduard', '1332022119913 ', 'edi_staicu@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (24.0, 'Stancu', 'Mihai', '2165702939000 ', 'mihaiStancu@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (25.0, 'Diaconu', 'Alexandru', '1859712404689',
'diaconuAlex@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (26.0, 'Bran', 'Marian', '1202453478865', 'mari_bran@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (27.0, 'Doica', 'Teodor', '1420727594908', 'Tedi_Doica@yahoo.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (28.0, 'Boboc', 'Silviu', '1476386347216',
'silviu_boboc@outlook.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (29.0, 'Cuza', 'Claudia', '2488819347327',
'claudia_claudia@gmail.com');

INSERT INTO CLIENT (ID_CLIENT, NUME, PRENUME, CNP, EMAIL)
VALUES (30.0, 'Munteanu', 'Cornel', '1905469662525',
'cornel_munteanu@gmail.com');
```

```

    COMMIT;

    select *
    from client;
    commit;

    create table ATM(
        id_atm number(4) constraint pk_atm primary key,
        id_sediu number(4) constraint fk_sediu_atm references sediu(id_sediu)
    );

```

Script Output | Query Result | All Rows Fetched: 30 in 0 seconds

ID_CLIENT	NUME	PRENUME	CNP	EMAIL
1	Ionescu	Marius	1991113438211	marius2007@yahoo.com
2	Popescu	Andrei	1980213810923	andreipopescu@yahoo.com
3	Oporanu	Alexandru	1713841329092	alexoporanull@gmail.com
4	Ivan	Codrut	1049873217712	codrut2009@gmail.com
5	Micu	Andreea	2071439827893	micuandreea@gmail.com
6	Tutescu	Ana	2432718091232	anaT@gmail.com
7	Radu	Ana Maria	2709909809801	raduanal2@yahoo.com
8	Munteanu	Ionela	2841791201913	IonelaM@yahoo.com
9	Gabroveanu	Sofia	2971923870104	sofiaG@yahoo.com
10	Stefanescu	Maria	2144323323422	stefanescuM@yahoo.com
11	Iordache	Stefan	1132809800999	stefan19@gmail.com
12	Tenea	Mariana	2987438579232	marianall@gmail.com
13	Iovanovici	Vlad	1432452342338	vlad_iovanovici@gmail.com
14	Georgescu	Patric	1986101012315	patric_georgescu@yahoo.com
15	Petrescu	Ion	1781318068695	ionPetrescu@yahoo.com

## 13. ATM

```

create table ATM(
    id_atm number(4) constraint pk_atm primary key,
    id_sediu number(4) constraint fk_sediu_atm references sediu(id_sediu)
);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9001.0, 2000.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9002.0, 2002.0);

```

```
INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9003.0, 2002.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9004.0, 2002.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9005.0, 2003.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9006.0, 2005.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9007.0, 2004.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9008.0, 2005.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9009.0, 2005.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9010.0, 2006.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9011.0, 2007.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9012.0, 2008.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9013.0, 2009.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9014.0, 2010.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9015.0, 2011.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9016.0, 2012.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9017.0, 2013.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9018.0, 2014.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9019.0, 2015.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9020.0, 2012.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9021.0, 2013.0);
```

```

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9022.0, 2014.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9023.0, 2015.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9024.0, 2016.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9025.0, 2017.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9026.0, 2012.0);

```

The screenshot shows a MySQL Workbench environment. The top pane contains the SQL script with the last two insert statements removed. The bottom pane shows the 'Query Result' tab with the output of the executed script.

```

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9025.0, 2017.0);

INSERT INTO ATM (ID_ATM, ID_SEDIU)
VALUES (9026.0, 2012.0);

commit;

select *
from atm;

```

	ID_ATM	ID_SEDIU
1	9001	2000
2	9002	2002
3	9003	2002
4	9004	2002
5	9005	2003
6	9006	2005
7	9007	2004
8	9008	2005
9	9009	2005
10	9010	2006
11	9011	2007
12	9012	2008
13	9013	2009
14	9014	2010
15	9015	2011

## 14. EXTRAGERI

```
create table extrageri(
    id_extragere number(4) constraint pk_extragere primary key,
    id_card number(4) constraint fk_ccard_extrage references card(id_card),
    id_atm number(4) constraint fk_extrage_atm references atm(id_atm),
    suma number(4) constraint suma_atm_null not null
);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1301.0, 801.0, 9001.0, 100.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1302.0, 802.0, 9002.0, 1100.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1303.0, 803.0, 9003.0, 423.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1304.0, 804.0, 9004.0, 3423.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1305.0, 805.0, 9005.0, 756.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1306.0, 806.0, 9006.0, 543.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1307.0, 807.0, 9007.0, 1350.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1308.0, 807.0, 9008.0, 49.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1309.0, 810.0, 9008.0, 992.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1310.0, 811.0, 9008.0, 890.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1311.0, 812.0, 9013.0, 1150.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1312.0, 813.0, 9014.0, 999.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1313.0, 810.0, 9016.0, 55.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1314.0, 820.0, 9022.0, 159.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1315.0, 805.0, 9023.0, 123.0);
```

```
INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1316.0, 820.0, 9022.0, 754.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1317.0, 817.0, 9024.0, 543.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1318.0, 814.0, 9019.0, 325.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1319.0, 812.0, 9020.0, 974.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1320.0, 818.0, 9021.0, 87.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1321.0, 818.0, 9021.0, 1000.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1322.0, 819.0, 9019.0, 1200.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1323.0, 820.0, 9014.0, 1000.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1324.0, 808.0, 9008.0, 900.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1325.0, 803.0, 9013.0, 1000.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1326.0, 805.0, 9026.0, 400.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1327.0, 813.0, 9025.0, 400.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1328.0, 820.0, 9022.0, 540.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1329.0, 812.0, 9020.0, 1500.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1330.0, 810.0, 9011.0, 360.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1331.0, 809.0, 9010.0, 850.0);

INSERT INTO EXTRAGERI (ID_EXTRAGERE, ID_CARD, ID_ATM, SUMA)
VALUES (1332.0, 801.0, 9007.0, 1150.0);
```

```

SELECT * FROM EXTRAGERI;
create table extrageri(
    id_extragere number(4) constraint pk_extragere primary key,
    id_card number(4) constraint fk_ccard_extrage references card(id_card),
    id_atm number(4) constraint fk_extrage_atm references atm(id_atm),
    suma number(4) constraint suma_atm_null not null
);

```

Script Output | Query Result | All Rows Fetched: 32 in 0.006 seconds

ID_EXTRAGERE	ID_CARD	ID_ATM	SUMA
1	1301	801	9001 100
2	1302	802	9002 1100
3	1303	803	9003 423
4	1304	804	9004 3423
5	1305	805	9005 756
6	1306	806	9006 543
7	1307	807	9007 1350
8	1308	807	9008 49
9	1309	810	9008 992
10	1310	811	9008 890
11	1311	812	9013 1150
12	1312	813	9014 999
13	1313	810	9016 55
14	1314	820	9022 159
15	1315	805	9023 123

## 15. ANGAJAT

```

create table angajat (
id_angajat number(4) constraint pk_id_angajat primary key,
id_sediu number(4) constraint null_sediu_angajat not null constraint
fk_angajat_sediu references sediu(id_sediu),
nume varchar2(20) constraint null_nume_angajat not null,
prenume varchar2(20) constraint null_prenume_angajat not null,
telefon varchar2(20) constraint null_telefon_angajat not null,
data_angajare date constraint null_data_ang not null,
salariu number(4) constraint null_salariu not null
);

alter table angajat drop constraint null_sediu_angajat;

alter table angajat
add constraint unq_telefon_angajat unique(telefon);

```

```
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9001.0, 2000.0, 'Ionescu', 'Maria', '0711841648', to_date('17-Jun-  
2017'), 5000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9002.0, 2000.0, 'Popescu', 'Ioana', '0789633254', to_date('21-Sep-  
2019'), 2500.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9003.0, 2001.0, 'Ion', 'Ionela', '0744456230', to_date('13-Jan-  
2013'), 6000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9004.0, 2001.0, 'Matache', 'Ionela', '0799955621', to_date('03-Jan-  
2010'), 2600.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9005.0, 2001.0, 'Dobrescu', 'Gheorghe', '0784630127', to_date('21-  
May-2011'), 9000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9006.0, 2002.0, 'Marinescu', 'Aron', '0763860205', to_date('25-Jun-  
2017'), 8000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9007.0, 2002.0, 'Cazacu', 'Livia', '0790772119', to_date('05-Feb-  
2018'), 2800.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9008.0, 2003.0, 'Trincu', 'Teodor', '0708320840', to_date('07-Feb-  
2019'), 7000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9009.0, 2003.0, 'Benea', 'Mircea', '0700427748', to_date('17-Aug-  
2014'), 2400.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9010.0, 2004.0, 'Negoescu', 'Mariana', '0779666379', to_date('16-Aug-  
2014'), 9000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9011.0, 2004.0, 'Pascu', 'George', '0762334958', to_date('28-Sep-  
2017'), 3300.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)
```

```

VALUES (9012.0, 2005.0, 'Stoian', 'Nicolae', '0799261040', to_date('30-Sep-2017'), 9500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9013.0, 2005.0, 'Popescu', 'Cosmin', '0745753345', to_date('07-Mar-2018'), 3500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9014.0, 2006.0, 'Protopopescu', 'Darius', '0721418156 ', to_date('07-Dec-2015'), 9900.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9015.0, 2006.0, 'Mihaescu', 'Mihai', '0769394244', to_date('07-Dec-2014'), 3800.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9016.0, 2007.0, 'Mirea', 'Nicoleta', '0717451606', to_date('18-May-2015'), 5500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9017.0, 2007.0, 'Davidescu', 'Mihai', '0774646382', to_date('24-Dec-2017'), 4300.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9018.0, 2008.0, 'Ciovica', 'Ovidiu', '0748973035', to_date('24-Jul-2015'), 8500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9019.0, 2008.0, 'Vatavu', 'Cristian', '0797889640', to_date('15-Nov-2018'), 3300.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9020.0, 2009.0, 'Mihailescu', 'Nicolae', '0764453097', to_date('10-Aug-2019'), 6500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9021.0, 2009.0, 'Gherorghe', 'Marin', '0792154170', to_date('18-Jul-2016'), 5300.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9022.0, 2010.0, 'Nicolaescu', 'David', '0796467908', to_date('10-Apr-2012'), 7500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9023.0, 2010.0, 'Draghici', 'Marian', '0776267794', to_date('01-May-2015'), 3500.0);

```

```
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9024.0, 2010.0, 'Dragulescu', 'Andrei', '0721323744', to_date('10-  
Oct-2014'), 8000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9025.0, 2011.0, 'Nicolescu', 'Andrei', '0714996417', to_date('16-Nov-  
2019'), 9500.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9026.0, 2011.0, 'Petrescu', 'Mihai', '0742663596', to_date('16-Jul-  
2017'), 2400.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9027.0, 2012.0, 'Andrei', 'Alexandru', '0775174209', to_date('28-Sep-  
2018'), 9000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9028.0, 2012.0, 'Tudor', 'Robert', '0758259884', to_date('14-Jan-  
2019'), 4200.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9029.0, 2013.0, 'Avram', 'Larisa', '0791285339 ', to_date('08-Mar-  
2010'), 4500.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9030.0, 2013.0, 'Popesciu', 'Madalin', '0769991336', to_date('20-Aug-  
2017'), 5000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9031.0, 2014.0, 'Mutu', 'Mariana', '0721292865', to_date('03-Jan-  
2010'), 5200.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9032.0, 2014.0, 'Beldea', 'Ionut', '0716930843 ', to_date('21-May-  
2011'), 4200.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9033.0, 2015.0, 'Piciu', 'Rares', '0721681396 ', to_date('25-Jun-  
2017'), 6000.0);  
  
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,  
DATA_ANGAJARE, SALARIU)  
VALUES (9034.0, 2015.0, 'Bran', 'Dan', '0790408877', to_date('05-Feb-2018'),  
6700.0);
```

```

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,
DATA_ANGAJARE, SALARIU)
VALUES (9035.0, 2016.0, 'Banica', 'Mirela', '0766699106 ', to_date('07-Feb-2019'), 8500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,
DATA_ANGAJARE, SALARIU)
VALUES (9036.0, 2016.0, 'Banescu', 'Vasile', '0759936576', to_date('17-Aug-2014'), 4400.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,
DATA_ANGAJARE, SALARIU)
VALUES (9037.0, 2017.0, 'Popa', 'Gabriela', '0785331159', to_date('16-Aug-2014'), 9000.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,
DATA_ANGAJARE, SALARIU)
VALUES (9038.0, 2017.0, 'Costache', 'Lucretia', '0731072290', to_date('28-Sep-2017'), 3300.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,
DATA_ANGAJARE, SALARIU)
VALUES (9039.0, 2018.0, 'Ungureanu', 'Patrick', '0741524878', to_date('30-Sep-2017'), 9000.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,
DATA_ANGAJARE, SALARIU)
VALUES (9040.0, 2018.0, 'Ganea', 'George', '0703881323', to_date('07-Mar-2018'), 3500.0);

INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON,
DATA_ANGAJARE, SALARIU)
VALUES (9041.0, NULL, 'Minulescu', 'Marin', '0740230001', to_date('04-Mar-2018'), 2500.0);

```

The screenshot shows a MySQL command-line interface with two tabs: "Script Output" and "Query Result".

**Script Output:**

```

VALUES (9040.0, 2018.0, 'Ganea', 'George', '0703881323', to_date('07-Mar-2018'), 3500.0);
INSERT INTO ANGAJAT (ID_ANGAJAT, ID_SEDIU, NUME, PRENUME, TELEFON, DATA_ANGAJARE, SALARIU)
VALUES (9041.0, NULL, 'Minulescu', 'Marin', '0740230001', to_date('04-Mar-2018'), 2500.0);

SELECT * FROM ANGAJAT;
commit;

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9001.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9003.0, NULL);

```

**Query Result:**

ID_ANGAJAT	ID_SEDIU	NUME	PRENUME	TELEFON	DATA_ANGAJARE	SALARIU	
1	9001	2000	Ionescu	Maria	0711841648	17-JUN-17	5000
2	9002	2000	Popescu	Ioana	0789633254	21-SEP-19	2500
3	9003	2001	Ion	Ionela	0744456230	13-JAN-13	6000
4	9004	2001	Matache	Ionela	0799955621	03-JAN-10	2600
5	9005	2001	Dobrescu	Gheorghe	0784630127	21-MAY-11	9000
6	9006	2002	Marinescu	Aron	0763860205	25-JUN-17	8000
7	9007	2002	Cazacu	Livia	0790772119	05-FEB-18	2800
8	9008	2003	Trincu	Teodor	0708320840	07-FEB-19	7000
9	9009	2003	Benea	Mircea	0700427748	17-AUG-14	2400
10	9010	2004	Negoescu	Mariana	0779666379	16-AUG-14	9000
11	9011	2004	Pascu	George	0762334958	28-SEP-17	3300
12	9012	2005	Stoian	Nicolae	0799261040	30-SEP-17	9500
13	9013	2005	Popescu	Cosmin	0745753345	07-MAR-18	3500
14	9014	2006	Protopopescu	Darius	0721418156	07-DEC-15	9900
15	9015	2006	Mihăescu	Mihai	0769394244	07-DEC-14	3800

## 16. ANGAJAT\_INTERN

```
CREATE TABLE angajat_intern(
id_intern number(4) constraint fk_intern_angajat REFERENCES
angajat(id_angajat),
comision number(2,2)
);

alter table angajat_intern
add constraint pk_id_intern primary key(id_intern);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9001.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9003.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9005.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9006.0, 0.12);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9008.0, 0.13);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9010.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9012.0, 0.05);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9014.0, 0.25);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9016.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9018.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9020.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
```

```

VALUES (9022.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9024.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9025.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9027.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9029.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9031.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9033.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9035.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9037.0, NULL);

INSERT INTO angajat_intern (ID_intern, COMISION)
VALUES (9039.0, NULL);

```

The screenshot shows a SQL query editor with the following details:

- Script Area:** Contains the SQL code provided above.
- Output Area:** Shows the results of the executed queries. It includes a table with columns `ID_INTERN` and `COMISION`, containing 13 rows of data.
- Status Bar:** Displays "All Rows Fetched: 21 in 0 seconds".

	ID_INTERN	COMISION
1	9001	(null)
2	9003	(null)
3	9005	(null)
4	9006	0.12
5	9008	0.13
6	9010	(null)
7	9012	0.05
8	9014	0.25
9	9016	(null)
10	9018	(null)
11	9020	(null)
12	9022	(null)
13	9024	(null)

## 17. ANGAJAT\_AUX

```
create table angajat_aux(
id_aux number(4) constraint fk_angajat references angajat(id_angajat)
);

alter table angajat_aux
add constraint pk_id_aux primary key(id_aux);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9004.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9007.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9009.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9011.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9013.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9015.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9017.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9019.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9021.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9023.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9026.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9028.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9030.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9032.0);
```

```
INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9034.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9036.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9038.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9040.0);
```

The screenshot shows the Oracle SQL Developer interface. The top half contains a script editor window with the following SQL code:

```
INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9038.0);

INSERT INTO ANGAJAT_AUX (ID_AUX)
VALUES (9040.0);

SELECT * FROM ANGAJAT_AUX;

select * from angajat_aux;
```

The bottom half shows the "Query Result" tab of the results window, which displays the following table:

ID_AUX
1 9002
2 9004
3 9007
4 9009
5 9011
6 9013
7 9015
8 9017
9 9019
10 9021
11 9023
12 9026
13 9028

The status bar at the bottom of the results window indicates: "All Rows Fetched: 19 in 0 seconds".

## 18. CREDIT

```
create table credit(
    id_credit number(4) constraint pk_credit primary key,
    id_banca number(4) constraint fk_credit_banca references banca(id_banca)
constraint null_credit not null,
    id_client number(4) constraint fk_client_credit references
client(id_client) constraint null_credit_client not null,
    moneda varchar2(5) constraint null_mcredit not null,
    suma number(7) constraint null_suma_credit not null,
    dobanda number(2,2)
);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1601.0, 101.0, 1.0, 2200.0, 'EUR', 0.05);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1602.0, 101.0, 1.0, 2400.0, 'RON', 0.02);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1603.0, 102.0, 2.0, 3000.0, 'USD', 0.03);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1604.0, 103.0, 2.0, 4000.0, 'EUR', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1605.0, 104.0, 3.0, 10000.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1606.0, 105.0, 4.0, 90000.0, 'USD', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1607.0, 106.0, 5.0, 5000.0, 'EUR', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1608.0, 107.0, 6.0, 4000.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1609.0, 108.0, 7.0, 4500.0, 'USD', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1610.0, 101.0, 8.0, 3500.0, 'EUR', 0.04);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1611.0, 102.0, 9.0, 9900.0, 'RON', 0.06);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1612.0, 103.0, 10.0, 10000.0, 'USD', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1613.0, 104.0, 11.0, 150000.0, 'EUR', NULL);
```

```
INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1614.0, 105.0, 12.0, 33000.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1615.0, 106.0, 13.0, 8000.0, 'USD', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1616.0, 107.0, 14.0, 4000.0, 'EUR', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1617.0, 108.0, 15.0, 4500.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1618.0, 101.0, 16.0, 9000.0, 'USD', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1619.0, 102.0, 17.0, 5000.0, 'EUR', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1620.0, 103.0, 18.0, 9000.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1621.0, 104.0, 19.0, 5000.0, 'USD', 0.06);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1622.0, 105.0, 20.0, 35000.0, 'EUR', 0.07);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1623.0, 106.0, 21.0, 15000.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1624.0, 107.0, 22.0, 2400.0, 'USD', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1625.0, 108.0, 23.0, 90000.0, 'EUR', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1626.0, 101.0, 24.0, 15000.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1627.0, 102.0, 25.0, 40000.0, 'USD', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1628.0, 103.0, 26.0, 1000.0, 'EUR', 0.04);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1629.0, 104.0, 27.0, 4000.0, 'RON', NULL);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1630.0, 105.0, 28.0, 3000.0, 'USD', 0.03);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1631.0, 106.0, 29.0, 450000.0, 'RON', 0.07);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1632.0, 107.0, 30.0, 9000.0, 'EUR', 0.09);
```

```

VALUES (1631.0, 106.0, 29.0, 450000.0, 'RON', 0.07);

INSERT INTO CREDIT (ID_CREDIT, ID_BANCA, ID_CLIENT, SUMA, MONEDA, DOBANDA)
VALUES (1632.0, 107.0, 30.0, 9000.0, 'EUR', 0.09);

select *
from credit;

select *

```

Script Output x Query Result x

SQL | All Rows Fetched: 31 in 0.005 seconds

	ID_CREDIT	ID_BANCA	ID_CLIENT	MONEDA	SUMA	DOBANDA
1	1601	101	1	EUR	2200	0.05
2	1602	101	1	RON	2400	0.02
3	1603	102	2	USD	3000	0.03
4	1605	104	3	RON	10000	(null)
5	1606	105	4	USD	90000	(null)
6	1607	106	5	EUR	5000	(null)
7	1608	107	6	RON	4000	(null)
8	1609	108	7	USD	4500	(null)
9	1610	101	8	EUR	3500	0.04
10	1611	102	9	RON	9900	0.06
11	1612	103	10	USD	10000	(null)
12	1613	104	11	EUR	150000	(null)
13	1614	105	12	RON	33000	(null)
14	1615	106	13	USD	8000	(null)
15	1616	107	14	EUR	4000	(null)

## EXERCIȚIUL 6

Afișați sediile care se află într-un anumit oraș (transmis ca parametru prin nume) și pentru fiecare dintre sedii lista angajaților care lucrează acolo.

```
set serveroutput on;
CREATE OR REPLACE PROCEDURE sediiAngajati (numeOras oras.nume_oras % TYPE)
AS
    type tabel_angajati is table of angajat%rowtype index by pls_integer;
    angajati tabel_angajati;
    type tabel_idSedii is varray(55) of sediu.id_sediu % TYPE;
    id_sedii tabel_idSedii;
    idOras oras.id_oras % TYPE;
    idLocatie locatie.id_locatie % TYPE;
BEGIN

    select id_oras into idOras
    from oras
    where nume_oras = numeOras;

    select s.id_sediu bulk collect into id_sedii
    from sediu s, locatie l
        where
            s.id_locatie = l.id_locatie
            and
            l.id_oras = idOras;
    dbms_output.put('In orașul ' || numeOras || ' se află sediile cu
urmatoarele id-uri: ');
    FOR i in id_sedii.first..id_sedii.last - 1 loop
        dbms_output.put(id_sedii(i) || ', ');
    end loop;
    dbms_output.put_line(id_sedii(id_sedii.last) || '. ');

    FOR i in id_sedii.first..id_sedii.last loop

        select * bulk collect into angajati
        from angajat
        where id_sediul = id_sedii(i);

        IF angajati.count = 0 then
            dbms_output.put_line('În sediul cu id-ul ' || id_sedii(i) || ' nu
lucrează niciun angajat.');
        ELSIF angajati.count = 1 then
            dbms_output.put_line('În sediul cu id-ul ' || id_sedii(i) || '
lucrează un singur angajat: ' || angajati(1).nume || ' ' ||
angajati(1).prenume || '. ');
        ELSE
            dbms_output.put('Sunt ' || angajati.count || ' angajati care lucrează
în sediul cu id-ul ' || id_sedii(i) || ': ');

            for i in angajati.first..(angajati.last - 2) loop
                dbms_output.put(angajati(i).nume || ' ' || angajati(i).prenume || '
', );
            end loop;
        END IF;
    end loop;
END;
```

```

        dbms_output.put(angajati(angajati.count - 1).nume || ' ' ||
angajati(angajati.count - 1).prenume);
        dbms_output.put(' si ');
        dbms_output.put(angajati(angajati.count).nume || ' ' ||
angajati(angajati.count).prenume);
        dbms_output.put_line('.');
    END IF;
END LOOP;
EXCEPTION WHEN NO_DATA_FOUND then
    dbms_output.put_line('Bancile din Romania nu au sedii in orasul ' ||
numeOras || '!');
END;
/

```

```

BEGIN
    sediiAngajati('Craiova');
END;
/

```

```

IF angajati.count = 0 then
    dbms_output.put_line('In sediul cu id-ul ' || id_sedii(i) || ' nu lucreaza niciun angajat.');
ELSIF angajati.count = 1 then
    dbms_output.put_line('In sediul cu id-ul ' || id_sedii(i) || ' lucreaza un singur angajat: ' || angajati(1).nume || ' ' || angajati(1).prenume || '.');
ELSE
    dbms_output.put('Sunt ' || angajati.count || ' angajati care lucreaza in sediul cu id-ul ' || id_sedii(i) || ':');

    for i in angajati.first..(angajati.last - 2) loop
        dbms_output.put(angajati(i).nume || ' ' || angajati(i).prenume || ', ');
    end loop;

    dbms_output.put(angajati(angajati.count - 1).nume || ' ' || angajati(angajati.count - 1).prenume);
    dbms_output.put(' si ');
    dbms_output.put(angajati(angajati.count).nume || ' ' || angajati(angajati.count).prenume);
    dbms_output.put_line('.');
END IF;
END LOOP;
EXCEPTION WHEN NO_DATA_FOUND then
    dbms_output.put_line('Bancile din Romania nu au sedii in orasul ' || numeOras || '!');
END;
/

```

```

BEGIN
    sediiAngajati('Craiova');
END;
/

```

Script Output X | Query Result X

Task completed in 0.001 seconds

anonymous block completed

In orasul Craiova se afla sediile cu urmatoarele id-uri: 2005, 2009.

Sunt 2 angajati care lucreaza in sediul cu id-ul 2005: Stoian Nicolae si Popescu Cosmin.

Sunt 2 angajati care lucreaza in sediul cu id-ul 2009: Mihalescu Nicolae si Gherorghe Marin.

## EXERCIȚIUL 7

Afişați numele și prenumele angajaților care lucrează la băncile care sunt partenere cu un anumit magazin (transmis ca parametru prin nume), adică la cele la care se pot face plăti cu cardul la supermarket-ul respectiv.

```
CREATE OR REPLACE PROCEDURE angajatiBanciPartenere(numeMagazin
magazin.nume_magazin % TYPE)
AS
    nume angajat.nume % type;
    prenume angajat.prenume % type;
    cnt number(3) := 1;
    CURSOR c IS
        SELECT a.nume, a.prenume
            FROM angajat a
        WHERE a.id_sediu IN (
            SELECT id_sediu
                FROM sediu s, banca b, accepta ac, magazin m
                WHERE s.id_banca = b.id_banca
                AND b.id_banca = ac.id_banca
                AND ac.id_magazin = m.id_magazin
                AND lower(m.nume_magazin) = lower(numeMagazin)
        );
BEGIN
    OPEN c;
    dbms_output.put_line('Angajatii bancilor partenere cu magazinul ' ||
    numeMagazin || ' sunt: ');
    LOOP
        FETCH c INTO nume, prenume;
        EXIT WHEN c % NOTFOUND;
        dbms_output.put_line('    ' || cnt || '. ' || nume || ' ' ||
    prenume);
        cnt := cnt + 1;
    END LOOP;
    CLOSE c;
    dbms_output.put_line('');
END;
/
BEGIN
    angajatiBanciPartenere('Auchan');
END;
/
```

```
        from sediu s, banca b, accepta ac, magazin m
        where s.id_banca = b.id_banca
        and b.id_banca = ac.id_banca
        and ac.id_magazin = m.id_magazin
        and lower(m.numere_magazin) = lower(numemagazin)
    );
BEGIN

    OPEN c;
    dbms_output.put_line('Angajatii bancilor partenere cu magazinul ' || numemagazin || ' sunt: ');
    LOOP
        FETCH c into nume, prenume;
        EXIT WHEN c % NOTFOUND;
        dbms_output.put_line(' ' || cnt || '.', ' || nume || ' || prenume);
        cnt := cnt + 1;
    END LOOP;
    CLOSE c;
    dbms_output.put_line('');
END;
/

BEGIN
    angajatiBanciPartenere('Auchan');
END;
/
```

Script Output | Query Result | Task completed in 0.022 seconds

```
anonymous block completed
Angajatii bancilor partenere cu magazinul Auchan sunt:
1. Negoescu Mariana
2. Pascu George
3. Protopopescu Darius
4. Mihaescu Mihai
5. Mirea Nicoleta
6. Davidescu Mihai
7. Mihailescu Nicolae
8. Chiriacu Manu
```

## EXERCIȚIUL 8

Afişați orașul în care se află sediul unde lucrează un angajat (transmis ca parametru prin numele de familie).

Tratați următoarele exceptii:

- angajatul lucrează remote (deci nu lucrează la un sediu)
- nu există niciun angajat cu numele introdus
- există mai mulți angajați cu același nume de familie
- alte exceptii

```
CREATE OR REPLACE FUNCTION orasSediu(numeAngajat Angajat.nume % TYPE) RETURN
VARCHAR2
IS
    REMOTE_WORKER_EXCEPTION EXCEPTION;
    numeOras oras.nume_oras % type;
    idSediu sediu.id_sediu % type;
BEGIN
    SELECT a.id_sediu INTO idSediu
        FROM angajat a
       WHERE LOWER(a.nume) = LOWER(numeAngajat);

    IF idSediu IS NULL THEN
        RAISE REMOTE_WORKER_EXCEPTION;
    END IF;

    SELECT o.nume_oras INTO numeOras
        FROM sediu s, locatie l, oras o
       WHERE
            s.id_sediu = idSediu
        AND
            s.id_locatie = l.id_locatie
        AND
            l.id_oras = o.id_oras;

    RETURN numeOras;

EXCEPTION
    WHEN REMOTE_WORKER_EXCEPTION THEN
        DBMS_OUTPUT.PUT_LINE('Angajatul ' || numeAngajat || ' lucreaza
remote.');
        RETURN 'err';
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu numele introdus.');
        RETURN 'err';
    WHEN TOO_MANY_ROWS THEN
        dbms_output.put_line('Sunt mai multi angajati cu acelasi nume de
familie!');
```

```

        RETURN 'err';
WHEN OTHERS THEN
    dbms_output.put_line('Alta exceptie!');
    RETURN 'err';
END;
/

begin
    if((orasSediu('Stoian')) != 'err') then
        dbms_output.put_line('Angajatul cu numele de familie Stoian lucreaza la
un sediu din ' || orasSediu('Stoian') || '.');
    end if;
end;
/

begin
    if((orasSediu('Stoican')) != 'err') then
        dbms_output.put_line(orasSediu('Stoican'));
    end if;
end;
/

begin
    if((orasSediu('Popescu')) != 'err') then
        dbms_output.put_line(orasSediu('Popescu'));
    end if;
end;
/

begin
    if((orasSediu('Minulescu')) != 'err') then
        dbms_output.put_line(orasSediu('Minulescu'));
    end if;
end;
/

```

```

l.id_oras = o.id_oras;

RETURN numeOras;

EXCEPTION
    WHEN REMOTE_WORKER_EXCEPTION THEN
        DBMS_OUTPUT.PUT_LINE('Angajatul ' || numeAngajat || ' lucreaza remote.');
        RETURN 'err';
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu numele introdus.');
        RETURN 'err';
    WHEN TOO_MANY_ROWS THEN
        dbms_output.put_line('Sunt mai multi angajati cu acelasi nume de familie!');
        RETURN 'err';
    WHEN OTHERS THEN
        dbms_output.put_line('Alta exceptie!');
        RETURN 'err';
END;
/

begin
    if((orasSediul('Stoian')) != 'err') then
        dbms_output.put_line('Angajatul cu numele de familie Stoian lucreaza la un sediu din ' || orasSediul('Stoian') || '.');
    end if;
end;
/
begin

```

Script Output | Query Result | Task completed in 0.003 seconds

anonymous block completed  
Angajatul cu numele de familie Stoian lucreaza la un sediu din Craiova.

```

WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu numele introdus.');
    RETURN 'err';
WHEN TOO_MANY_ROWS THEN
    dbms_output.put_line('Sunt mai multi angajati cu acelasi nume de familie!');
    RETURN 'err';
WHEN OTHERS THEN
    dbms_output.put_line('Alta exceptie!');
    RETURN 'err';
END;
/

begin
    if((orasSediul('Stoian')) != 'err') then
        dbms_output.put_line('Angajatul cu numele de familie Stoian lucreaza la un sediu din ' || orasSediul('Stoian') || '.');
    end if;
end;
/
begin
    if((orasSediul('Stoican')) != 'err') then
        dbms_output.put_line(orasSediul('Stoican'));
    end if;
end;
/
begin
    if((orasSediul('Popescu')) != 'err') then
        dbms_output.put_line(orasSediul('Popescu'));
    end if;
end;

```

Script Output | Query Result | Task completed in 0.001 seconds

anonymous block completed  
Nu exista niciun angajat cu numele introdus.

```

begin
  if((orasSedi('Stoian')) != 'err') then
    dbms_output.put_line('Angajatul cu numele de familie Stoian lucreaza la un sediu din ' || orasSedi('Stoian') || '.');
  end if;
end;
/

begin
  if((orasSedi('Stoican')) != 'err') then
    dbms_output.put_line(orasSedi('Stoican'));
  end if;
end;
/

begin
  if((orasSedi('Popescu')) != 'err') then
    dbms_output.put_line(orasSedi('Popescu'));
  end if;
end;
/

begin
  if((orasSedi('Minulescu')) != 'err') then
    dbms_output.put_line(orasSedi('Minulescu'));
  end if;
end;
/

```

Script Output | Query Result | Task completed in 0.001 seconds  
X anonymous block completed  
 Sunt mai multi angajati cu acelasi nume de familie!

```

begin
  if((orasSedi('Stoian')) != 'err') then
    dbms_output.put_line('Angajatul cu numele de familie Stoian lucreaza la un sediu din ' || orasSedi('Stoian') || '.');
  end if;
end;
/

begin
  if((orasSedi('Stoican')) != 'err') then
    dbms_output.put_line(orasSedi('Stoican'));
  end if;
end;
/

begin
  if((orasSedi('Popescu')) != 'err') then
    dbms_output.put_line(orasSedi('Popescu'));
  end if;
end;
/

begin
  if((orasSedi('Minulescu')) != 'err') then
    dbms_output.put_line(orasSedi('Minulescu'));
  end if;
end;
/

```

Script Output | Query Result | Task completed in 0.001 seconds  
X anonymous block completed  
 Angajatul Minulescu lucreaza remote.

## EXERCIȚIUL 9

Dat fiind cvv – ul unui card, determinați din ce orașe a făcut extrageri proprietarul acestuia.

Tratați următoarele exceptii:

- sunt mai multe carduri cu același cvv (eveniment uzuial în lumea reală, dar în baza mea de date, fiind relativ puține înregistrări comparativ cu un sistem real de gestionare a băncilor, este o excepție)
- nu există niciun card care să aibă acel cvv
- cvv-ul este invalid (adică este în afara intervalului [100, 999])
- alte excepții

```
CREATE OR REPLACE PROCEDURE locatiiExtrageri(cvv_ card.cvv % type)
AS
    idCard card.id_card % type;
    type tabel_orase is table of oras.numere_oras % type index by pls_integer;
    orase tabel_orase;
    cvv_invalid EXCEPTION;
BEGIN

    IF (cvv_ - 99) * (cvv_ - 1000) >= 0 then
        RAISE cvv_invalid;
    END IF;

    select id_card into idCard
    from card
    where cvv = cvv_;
    select distinct o.numere_oras bulk collect into orase
        from atm a, sediul s, locatie l, oras o, extrageri e
    where
        e.id_card = idCard
        and
        e.id_atm = a.id_atm
        and
        a.id_sediul = s.id_sediul
        and
        s.id_locatie = l.id_locatie
        and
        l.id_oras = o.id_oras;

    IF orase.count = 0 then
        dbms_output.put_line('De pe cardul cu cvv-ul egal cu ' || cvv_ || ' nu s-a facut nicio extragere.');
    ELSIF orase.count = 1 then
        dbms_output.put_line('De pe cardul cu cvv-ul egal cu ' || cvv_ || ' s-au facut extrageri doar din ' || orase(1) || '.');
    ELSE
        dbms_output.put('De pe cardul cu cvv-ul egal cu ' || cvv_ || ' s-au facut extrageri in următoarele orase: ');
        for i in orase.first..(orase.last - 2) loop
```

```

        dbms_output.put(orase(i) || ', ');
    end loop;
    dbms_output.put(orase(orase.count - 1));
    dbms_output.put(' si ');
    dbms_output.put(orase(orase.count));
    dbms_output.put_line('.');

END IF;

EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Sunt mai multe carduri cu acest cvv.');
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun card cu cvv-ul introdus.');
    WHEN cvv_invalid THEN
        dbms_output.put_line('CVV-ul introdus este invalid, acesta trebuie sa
aiba 3 cifre!');
    WHEN OTHERS THEN
        dbms_output.put_line('Alta eroare!');
END;
/

begin
    locatiiExtrageri(123);
end;
/

begin
    locatiiExtrageri(142);
end;
/

begin
    locatiiExtrageri(763);
end;
/

begin
    locatiiExtrageri(1000);
end;

```

```

        dbms_output.put(' si ');
        dbms_output.put(orase.orase.count);
        dbms_output.put_line('.');
    END IF;

    EXCEPTION
        WHEN TOO_MANY_ROWS THEN
            DBMS_OUTPUT.PUT_LINE('Sunt mai multe carduri cu acest cvv.');
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Nu exista niciun card cu cvv-ul introdus.');
        WHEN cvv_invalid THEN
            dbms_output.put_line('CVV-ul introdus este invalid, acesta trebuie sa aiba 3 cifre!');
        WHEN OTHERS THEN
            dbms_output.put_line('Alta eroare!');
    END;
    /

begin
    locatiiExtrageri(142);
end;
/

```

Script Output | Query Result | Task completed in 0.003 seconds  
 anonymous block completed  
 De pe cardul cu cvv-ul egal cu 142 s-au facut extrageri in urmatoarele orase: Suceava si Targu Jiu.

```

        dbms_output.put(' si ');
        dbms_output.put(orase.orase.count);
        dbms_output.put_line('.');
    END IF;

    EXCEPTION
        WHEN TOO_MANY_ROWS THEN
            DBMS_OUTPUT.PUT_LINE('Sunt mai multe carduri cu acest cvv.');
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Nu exista niciun card cu cvv-ul introdus.');
        WHEN cvv_invalid THEN
            dbms_output.put_line('CVV-ul introdus este invalid, acesta trebuie sa aiba 3 cifre!');
        WHEN OTHERS THEN
            dbms_output.put_line('Alta eroare!');
    END;
    /

begin
    locatiiExtrageri(123);
end;
/

```

Script Output | Query Result | Task completed in 0 seconds  
 anonymous block completed  
 Sunt mai multe carduri cu acest cvv.

```
        dbms_output.put_line('CVV-ul introdus este invalid, acesta trebuie sa aiba 3 cifre!');
        WHEN OTHERS THEN
            dbms_output.put_line('Alta eroare!');
    END;
/

begin
    locatiiExtrageri(142);
end;
/

begin
    locatiiExtrageri(123);
end;
/

begin
    locatiiExtrageri(763);
end;
/

begin
    locatiiExtrageri(1000);
```

Script Output | Query Result | Task completed in 0 seconds

anonymous block completed  
Nu exista niciun card cu cvv-ul introdus.

```
END;
/

begin
    locatiiExtrageri(142);
end;
/

begin
    locatiiExtrageri(123);
end;
/

begin
    locatiiExtrageri(763);
end;
/

begin
    locatiiExtrageri(1000);
end;
/
```

Script Output | Query Result | Task completed in 0.007 seconds

anonymous block completed  
CVV-ul introdus este invalid, acesta trebuie sa aiba 3 cifre!

## EXERCIȚIUL 10

Am creat un TRIGGER care nu permite operații de tipul INSERT, UPDATE sau DELETE în zilele care sunt libere prin lege. Am ales să particularizez doar pentru anul 2022, întrucât unele dintre acestea nu au date fixe. De asemenea, am ales să fac operația de DELETE pe tabelul "ANGAJAT\_AUX", întrucât acesta nu este referit de niciun alt tabel prin cheie externă, aşa că cu siguranță orice eroare cauzată de ștergerea unor linii din el este cauzată de TRIGGER. Pentru a testa dacă TRIGGER-ul meu este într-adevăr funcțional, am introdus și data în care am rezolvat eu acest exercițiu (5. 01.2022), chiar dacă nu este zi liberă prin lege.

```
create or replace trigger zileLibere
  before insert or update or delete on angajat_aux
BEGIN
  IF
    (TO_CHAR(SYSDATE, 'DD/MM/YY') = '01/01/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '02/01/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '24/01/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '22/04/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '24/04/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '25/04/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '01/05/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '01/06/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '12/06/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '13/06/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '15/08/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '30/11/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '01/12/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '25/12/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '26/12/22' OR
     TO_CHAR(SYSDATE, 'DD/MM/YY') = '05/01/22'
    ) THEN
      RAISE_APPLICATION_ERROR(-20000, 'Azi este zi libera!');
    END IF;
END ;
/
```

```

TO_CHAR(SYSDATE, 'DD/MM/YY') = '24/04/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '25/04/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '01/05/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '01/06/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '12/06/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '13/06/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '15/08/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '30/11/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '01/12/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '25/12/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '26/12/22' OR
TO_CHAR(SYSDATE, 'DD/MM/YY') = '05/01/22'
) THEN
RAISE_APPLICATION_ERROR(-20000, 'Azi este zi libera!');
END IF;
END;
/
delete from angajat_aux where id_aux = 9002;

```

Script Output    Query Result

| Task completed in 0.016 seconds

---

~~DELETE FROM angajat\_aux WHERE id\_aux = 9002~~

Error report -

SQL Error: ORA-20000: Azi este zi libera!

ORA-06512: at "ALEX.ZILELIBERE", line 20

ORA-04088: error during execution of trigger 'ALEX.ZILELIBERE'

20000. 00000 - "%s"

\*Cause: The stored procedure 'raise\_application\_error'  
was called which causes this error to be generated.

\*Action: Correct the problem as described in the error message or contact  
the application administrator or DBA for more information.

## EXERCIȚIU 11

Am creat un TRIGGER care nu permite adăugarea unui nou salariat cu un salariu minim mai mic decât salariul minim net pe economie, sau modificarea sub acest prag a venitului lunar al unui angajat existent. Salariul minim net pe economie este în prezent de 1524 de lei. În capturile de ecran de mai jos putem vedea cum nu este posibil să actualizăm salariu angajatului cu ID-ul = 9001 la 1523 de lei, dar la 1525 este.

```
CREATE OR REPLACE TRIGGER salariuMinim
  BEFORE UPDATE OR INSERT OF salariu ON angajat
  FOR EACH ROW
  WHEN (NEW.salariu < 1524)
BEGIN
  RAISE_APPLICATION_ERROR(-20001, 'Salariu minim net este de 1524 de
lei!');
END;
/

update angajat set salariu = 1523 where id_angajat = 9001;
update angajat set salariu = 1525 where id_angajat = 9001;
```

The screenshot shows the Oracle SQL Developer interface. The top pane displays the PL/SQL code for creating a trigger named 'salariuMinim' that prevents updates or inserts of salaries below 1524. The bottom pane shows the 'Script Output' tab with the executed SQL statements and their results. The second update statement fails with an ORA-20001 error because the salary is set to 1523, which is below the minimum allowed value of 1524.

```
CREATE OR REPLACE TRIGGER salariuMinim
  BEFORE UPDATE OR INSERT OF salariu ON angajat
  FOR EACH ROW
  WHEN (NEW.salariu < 1524)
BEGIN
  RAISE_APPLICATION_ERROR(-20001, 'Salariu minim net este de 1524 de
lei!');
END;
/

update angajat set salariu = 1523 where id_angajat = 9001;
update angajat set salariu = 1525 where id_angajat = 9001;
```

Script Output | Query Result | Task completed in 0.017 seconds

```
Error starting at line : 1,777 in command -
update angajat set salariu = 1523 where id_angajat = 9001
Error report -
SQL Error: ORA-20001: Salariu minim net este de 1524 de lei!
ORA-06512: at "ALEX.SALARIUMINIM", line 2
ORA-04088: error during execution of trigger 'ALEX.SALARIUMINIM'
```

```
CREATE OR REPLACE TRIGGER salariuMinim
  BEFORE UPDATE OR INSERT OF salariu ON angajat
  FOR EACH ROW
  WHEN (NEW.salariu < 1524)
BEGIN
  RAISE_APPLICATION_ERROR(-20001, 'Salariu minim net este de 1524 de lei!');
END;
/

update angajat set salariu = 1523 where id_angajat = 9001;
update angajat set salariu = 1525 where id_angajat = 9001;
```

Script Output    Query Result

| Task completed in 0 seconds

```
Error starting at line : 1,777 in command -
update angajat set salariu = 1523 where id_angajat = 9001
Error report -
SQL Error: ORA-20001: Salariu minim net este de 1524 de lei!
ORA-06512: at "ALEX.SALARIUMINIM", line 2
ORA-04088: error during execution of trigger 'ALEX.SALARIUMINIM'

1 rows updated.
```

## EXERCIȚIU 12

Pentru a ușura munca administratorilor bazei de date, am creat un trigger care salveaza într-un tabel informații despre schimbările cauzate de operațiile (de tip LDD) CREATE, DROP sau ALTER efectuate asupra bazei de date. De asemenea, aceștia doresc ca astfel de operații să fie efectuate numai de luni până vineri.

```
CREATE TABLE informatii(
    utilizator VARCHAR2(30),
    nume_bd VARCHAR2(50),
    eveniment VARCHAR2(20),
    nume_object VARCHAR2(30),
    data DATE
);

CREATE OR REPLACE TRIGGER fillInfo
    BEFORE CREATE OR DROP OR ALTER ON SCHEMA
BEGIN
    IF TO_CHAR (sysdate, 'DY', 'NLS_DATE_LANGUAGE=ENGLISH') IN ('SAT', 'SUN')
    THEN
        RAISE_APPLICATION_ERROR(-20028, 'Operatiile CREATE, DROP si ALTER pot
        fi efectuate doar in timpul saptamanii!');
    END IF;
    INSERT INTO INFORMATII VALUES (
        SYS.LOGIN_USER,
        SYS.DATABASE_NAME,
        SYS.SYSEVENT,
        SYS.DICTIONARY_OBJ_NAME,
        SYSDATE);
END;
/

create table dummy1(a number);
drop table dummy1;
create table dummy2(b string);
drop table dummy2;
```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor containing the SQL script provided above. In the bottom-right pane, there is a results grid showing the data from the 'informatii' table. The results are:

	UTILIZATOR	NUME_BD	EVENIMENT	NUME_OBIECT	DATA
1	ALEX	XE	CREATE	DUMMY2	06-JAN-22
2	ALEX	XE	CREATE	DUMMY1	06-JAN-22
3	ALEX	XE	DROP	DUMMY1	06-JAN-22
4	ALEX	XE	DROP	DUMMY2	06-JAN-22

## EXERCIȚIU 13

La acest exercițiu am creat un pachet care conține toate obiectele definite în cadrul proiectului.

```
CREATE OR REPLACE PACKAGE proiect_alex AS
    PROCEDURE sediiAngajati (numeOras oras.nume_oras % TYPE);
    PROCEDURE angajatiBanciPartenere (numeMagazin magazin.nume_magazin % TYPE);
    FUNCTION orasSediu (numeAngajat angajat.nume % TYPE) RETURN VARCHAR2;
    PROCEDURE locatiiExtrageri (cvv_ card.cvv % type);
END proiect_alex;
/

CREATE OR REPLACE PACKAGE BODY proiect_alex AS

    PROCEDURE sediiAngajati (numeOras oras.nume_oras % TYPE)
        AS
        type tabel_angajati is table of angajat%rowtype index by pls_integer;
        angajati tabel_angajati;
        type tabel_idSedii is varray(55) of sediu.id_sediu % TYPE;
        id_sedii tabel_idSedii;
        idOras oras.id_oras % TYPE;
        idLocatie locatie.id_locatie % TYPE;
    BEGIN

        select id_oras into idOras
        from oras
        where nume_oras = numeOras;

        select s.id_sediu bulk collect into id_sedii
        from sediu s, locatie l
            where
                s.id_locatie = l.id_locatie
                and
                l.id_oras = idOras;
        dbms_output.put('In orasul ' || numeOras || ' se afla sediile cu
urmatoarele id-uri: ');
        FOR i IN id_sedii.first..id_sedii.last - 1 loop
            dbms_output.put(id_sedii(i) || ', ');
        end loop;
        dbms_output.put_line(id_sedii(id_sedii.last) || '. ');
        FOR i IN id_sedii.first..id_sedii.last loop

            select * bulk collect into angajati
            from angajat
            where id_sediu = id_sedii(i);

            IF angajati.count = 0 then
                dbms_output.put_line('In sediul cu id-ul ' || id_sedii(i) || ' nu
lucreaza niciun angajat.');
            ELSIF angajati.count = 1 then
```

```

        dbms_output.put_line('In sediul cu id-ul ' || id_sedii(i) || '
lucreaza un singur angajat: ' || angajati(1).nume || ' ' ||
angajati(1).prenume || '.');
    ELSE
        dbms_output.put('Sunt ' || angajati.count || ' angajati care
lucreaza in sediul cu id-ul ' || id_sedii(i) || ': ');
        for i in angajati.first..(angajati.last - 2) loop
            dbms_output.put(angajati(i).nume || ' ' || angajati(i).prenume || '
', );
        end loop;

        dbms_output.put(angajati(angajati.count - 1).nume || ' ' || '
angajati(angajati.count - 1).prenume);
        dbms_output.put(' si ');
        dbms_output.put(angajati(angajati.count).nume || ' ' || '
angajati(angajati.count).prenume);
        dbms_output.put_line('.');
    END IF;
END LOOP;
EXCEPTION WHEN NO_DATA_FOUND then
    dbms_output.put_line('Bancile din Romania nu au sedii in orasul ' || '
numeOras || '!');
END sediiAngajati;

```

```

PROCEDURE angajatiBanciPartenere(numeMagazin magazin.nume_magazin % TYPE)
AS
nume angajat.nume % type;
prenume angajat.prenume % type;
cnt number(3) := 1;
CURSOR c IS
    select a.nume, a.prenume
        from angajat a
    where a.id_sediul in (
        select id_sediul
            from sediul s, banca b, accepta ac, magazin m
            where s.id_banca = b.id_banca
            and b.id_banca = ac.id_banca
            and ac.id_magazin = m.id_magazin
            and lower(m.nume_magazin) = lower(numeMagazin)
        );
BEGIN

    OPEN c;
    dbms_output.put_line('Angajatii bancilor partenere cu magazinul ' || '
numeMagazin || ' sunt: ');
    LOOP
        FETCH c into nume, prenume;
        EXIT WHEN c % NOTFOUND;
        dbms_output.put_line(' ' || cnt || '. ' || nume || ' ' || '
prenume);
        cnt := cnt + 1;
    END LOOP;
    CLOSE c;

```

```

        dbms_output.put_line('');
END angajatiBanciPartenere;

function orasSediu(numeAngajat angajat.nume % TYPE) RETURN VARCHAR2
IS
    REMOTE_WORKER_EXCEPTION EXCEPTION;
    numeOras oras.nume_oras % type;
    idSediu sediu.id_sediu % type;
BEGIN
    select a.id_sediu into idSediu
        from angajat a
     where lower(a.nume) = lower(numeAngajat);

    IF idSediu is NULL then
        RAISE REMOTE_WORKER_EXCEPTION;
    END IF;

    select o.nume_oras into numeOras
        from sediu s, locatie l, oras o
     where
        s.id_sediu = idSediu
        and
        s.id_locatie = l.id_locatie
        and
        l.id_oras = o.id_oras;

    RETURN numeOras;

EXCEPTION
    WHEN REMOTE_WORKER_EXCEPTION THEN
        DBMS_OUTPUT.PUT_LINE('Angajatul ' || numeAngajat || ' lucreaza
remote.');
        RETURN 'err';
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun angajat cu numele
introdus.');
        RETURN 'err';
    WHEN TOO_MANY_ROWS THEN
        dbms_output.put_line('Sunt mai multi angajati cu acelasi nume de
familie!');
        RETURN 'err';
    WHEN OTHERS THEN
        dbms_output.put_line('Alta exceptie!');
        RETURN 'err';
END orasSediu;

PROCEDURE locatiiExtrageri(cvv_ card.cvv % type)
AS
    idCard card.id_card % type;

```

```

        type tabel_orase is table of oras.nume_oras % type index by
pls_integer;
    orase tabel_orase;
    cvv_invalid EXCEPTION;
BEGIN

    IF (cvv_ - 99) * (cvv_ - 1000) >= 0 then
        RAISE cvv_invalid;
    END IF;

    select id_card into idCard
    from card
    where cvv = cvv_;
    select distinct o.nume_oras bulk collect into orase
        from atm a, sediu s, locatie l, oras o, extrageri e
    where
        e.id_card = idCard
    and
        e.id_atm = a.id_atm
    and
        a.id_sediu = s.id_sediu
    and
        s.id_locatie = l.id_locatie
    and
        l.id_oras = o.id_oras;

    IF orase.count = 0 then
        dbms_output.put_line('De pe cardul cu cvv-ul egal cu ' || cvv_ ||
' nu s-a facut nicio extragere.');
    ELSIF orase.count = 1 then
        dbms_output.put_line('De pe cardul cu cvv-ul egal cu ' || cvv_ ||
' s-a facut extrageri doar din ' || orase(1) || '.');
    ELSE
        dbms_output.put('De pe cardul cu cvv-ul egal cu ' || cvv_ || ' s-
au facut extrageri in urmatoarele orase: ');
        for i in orase.first..(orase.last - 2) loop
            dbms_output.put(orase(i) || ', ');
        end loop;
        dbms_output.put(orase(orase.count - 1));
        dbms_output.put(' si ');
        dbms_output.put(orase(orase.count));
        dbms_output.put_line('.');
    END IF;

EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Sunt mai multe carduri cu acest cvv.');
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista niciun card cu cvv-ul
introdus.');
    WHEN cvv_invalid THEN
        dbms_output.put_line('CVV-ul introdus este invalid, acesta
trebuie sa aiba 3 cifre!');

    WHEN OTHERS THEN
        dbms_output.put_line('Alta eroare!');
END locatiextrageri;

```

```
END ;
/
```

```
CREATE or REPLACE PACKAGE proiect_alex AS
    PROCEDURE sediiAngajati (numeOras oras.nume_oras % TYPE);
    PROCEDURE angajatiBanciPartenere(numemagazin magazin.nume_magazin % TYPE);
    FUNCTION orasSediui(umeAngajat angajat.nume % TYPE) RETURN VARCHAR2;
    PROCEDURE locatiiExtrageri(cvv_ card.cvv % type);
END proiect_alex;
/
CREATE OR REPLACE PACKAGE BODY proiect_alex AS
    PROCEDURE sediiAngajati (numeOras oras.nume_oras % TYPE)
        AS
            type tabel_angajati is table of angajat%rowtype index by pls_integer;
```

Script Output x | Query Result x | Query Result 1 x

✖️ 🖍️ 📁 📄 | Task completed in 0.007 seconds

PACKAGE PROIECT\_ALEX compiled

```
CREATE OR REPLACE PACKAGE BODY proiect_alex AS
    PROCEDURE sediiAngajati (numeOras oras.nume_oras % TYPE)
        AS
            type tabel_angajati is table of angajat%rowtype index by pls_integer;
            angajati tabel_angajati;
            type tabel_idSedii is varray(55) of sediu.id_sediui % TYPE;
            id_sedii tabel_idSedii;
            idOras oras.id_oras % TYPE;
            idLocatie locatie.id_locatie % TYPE;
        BEGIN
```

Script Output x | Query Result x | Query Result 1 x

✖️ 🖍️ 📁 📄 | Task completed in 0.008 seconds

PACKAGE PROIECT\_ALEX compiled

PACKAGE BODY PROIECT\_ALEX compiled

## EXERCIȚIU 14

La exercițiul 14 am creat un pachet care conține câteva tipuri de date complexe și obiecte care facilitează lucrul cu baza de date.

Acestea sunt:

- funcția salariuAngajatiBanca care returnează totalul salariilor angajaților băncii primite ca input
- funcția getNumeAngajat care returnează numele unui angajat căruia i se transmite ID-ul ca parametru
- funcția getDenumireBanca – returnează numele unei bănci pentru care ID-ul este transmis ca input
- procedura angajatiBanca – afișează lista angajaților unei bănci căreia i se transmite numele ca input
- procedura extrageriATM care afișează detalii despre extragerile făcute de la un anumit ATM primit ca input prin ID
- tipul cardSuma, util în implementarea procedurii anterior menționate care conține informații despre extragere
- tipul tabel\_detalii, tabel care conține elemente de tip cardSuma
- tipul listaSalariati care conține elemente de tipul angajat % ROWTYPE

```
CREATE OR REPLACE PACKAGE BODY pachet_utilitar AS
  FUNCTION getNumeAngajat (id_angajat.id_angajat % TYPE) RETURN VARCHAR2
  IS nume_angajat angajat.nume % TYPE;
  BEGIN
    SELECT nume INTO nume_angajat
      FROM angajat
     WHERE id_angajat = id_;
    RETURN nume_angajat;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR (-20031, 'Nu exista
niciun angajat cu acest ID!');
  END getNumeAngajat;

  FUNCTION getDenumireBanca (id_banca.id_banca % TYPE) RETURN VARCHAR2
  IS denumire_banca angajat.nume % TYPE;
  BEGIN
    SELECT denumire_banca INTO denumire_banca
      FROM banca
     WHERE id_banca = id_;
    RETURN denumire_banca;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN RAISE_APPLICATION_ERROR (-20030, 'Nu exista
nicio banca cu acest ID!');
  END getDenumireBanca;
```

```

PROCEDURE angajatiBanca(nume_banca banca.denumire_banca % TYPE)
AS
BEGIN
    IF nume_banca = 'BCR' THEN
        SELECT DISTINCT A.id_angajat, A.id_sediu, A.nume, A.prenume,
A.telefon, A.data_angajare, A.salariu BULK COLLECT INTO lista_angajati
        FROM angajat A, sediu s, banca b
        WHERE
            (A.id_sediu = s.id_sediu
            AND
            s.id_banca = b.id_banca
            AND
            b.denumire_banca = nume_banca)
            OR A.id_sediu IS NULL;
    ELSE
        SELECT A.id_angajat, A.id_sediu, A.nume, A.prenume, A.telefon,
A.data_angajare, A.salariu BULK COLLECT INTO lista_angajati
        FROM angajat A, sediu s, banca b
        WHERE
            A.id_sediu = s.id_sediu
            AND
            s.id_banca = b.id_banca
            AND
            b.denumire_banca = nume_banca;
    END IF;

    dbms_output.put('La banca ' || nume_banca || ' lucreaza urmatorii
angajati: ');

    FOR i IN lista_angajati.FIRST..(lista_angajati.LAST - 2) loop
        dbms_output.put(lista_angajati(i).nume || ' ' ||
lista_angajati(i).prenume || ', ');
    END loop;

    dbms_output.put(lista_angajati(lista_angajati.count - 1).nume || ' '
|| lista_angajati(lista_angajati.count - 1).prenume);
    dbms_output.put(' si ');
    dbms_output.put(lista_angajati(lista_angajati.count).nume || ' ' ||
lista_angajati(lista_angajati.count).prenume);
    dbms_output.put_line('.');
END angajatiBanca;

FUNCTION salariuAngajatiBanca(nume_banca banca.denumire_banca % TYPE)
RETURN NUMBER
IS salariuTotal NUMBER;
BEGIN
    IF nume_banca = 'BCR' THEN
        SELECT DISTINCT A.id_angajat, A.id_sediu, A.nume, A.prenume,
A.telefon, A.data_angajare, A.salariu BULK COLLECT INTO lista_angajati
        FROM angajat A, sediu s, banca b
        WHERE
            (A.id_sediu = s.id_sediu
            AND

```

```

        s.id_banca = b.id_banca
        AND
        b.denumire_banca = nume_banca)
        OR A.id_sediu IS NULL;
    ELSE
        SELECT A.id_angajat, A.id_sediu, A.nume, A.prenume, A.telefon,
A.data_angajare, A.salariu BULK COLLECT INTO lista_angajati
        FROM angajat A, sediu s, banca b
        WHERE
        A.id_sediu = s.id_sediu
        AND
        s.id_banca = b.id_banca
        AND
        b.denumire_banca = nume_banca;
    END IF;
    salariuTotal := 0;

    FOR i IN lista_angajati.FIRST..lista_angajati.LAST loop
        salariuTotal := salariuTotal + lista_angajati(i).salariu;
    END loop;
    RETURN salariuTotal;
END salariuAngajatiBanca;

PROCEDURE extrageriATM(id_atm atm.id_atm % TYPE)
AS
cnt number;
BEGIN
SELECT e.id_card, e.suma BULK COLLECT INTO tabel_card_suma
FROM extrageri e
WHERE id_atm = id_atm;
cnt := 0;
dbms_output.put_line('Detaliile extragerilor facute de pe ATM-ul cu ID-'-
ul: ' || id_atm || ': ');
FOR i IN tabel_card_suma.FIRST .. tabel_card_suma.LAST loop
    cnt := cnt + 1;
    dbms_output.put_line('    ' || cnt || '. ' || 'ID CARD: ' || 
tabel_card_suma(i).id_card || ' ' || 'SUMA: ' || tabel_card_suma(i).suma);
end loop;
END extrageriATM;

END;
/

begin
    dbms_output.put_line(pachet_utilitar.getNumAngajat(9006));
end;
/

begin
    dbms_output.put_line(pachet_utilitar.getDenumireBanca(106));
end;
/

begin

```

```
pachet_utilitar.angajatiBanca('FB');
end;
/
```

```
select * from atm;

CREATE or REPLACE PACKAGE pachet_utilitar AS
    FUNCTION salariuAngajatiBanca(nume_banca_banca.denumire_banca % type) return number;
    FUNCTION getNumeAngajat (id_angajat.id_angajat % TYPE) return varchar2;
    FUNCTION getDenumireBanca (id_banca.id_banca % TYPE) return varchar2;
    PROCEDURE angajatiBanca(nume_banca_banca.denumire_banca % type);
    PROCEDURE extrageriATM(id_atm atm.id_atm % type);
    TYPE listaSalariati is table of angajat % rowtype index by pls_integer;
    TYPE cardSuma IS RECORD(
        id_card card.id_card % type,
        suma number
    );
    TYPE tabel_detalii is varray(55) of cardSuma;
    lista_angajati listaSalariati;
    tabel_card_suma tabel_detalii;
END pachet_utilitar;
/


CREATE OR REPLACE PACKAGE BODY pachet_utilitar AS
    FUNCTION getNumeAngajat (id_angajat.id_angajat % TYPE) RETURN VARCHAR2
    IS nume_angajat angajat.nume % TYPE;
    BEGIN
        SELECT nume INTO nume_angajat
```

Script Output x | Query Result x  
| Task completed in 0.008 seconds

```
PACKAGE PACHET_UTILITAR compiled
PACKAGE BODY PACHET_UTILITAR compiled
```

```
    end loop;

  END extrageriATM;

END;
/

begin
  dbms_output.put_line(pachet_utilitar.getNumAngajat(9006));
end;
/
```

Script Output    Query Result

Task completed in 0 seconds

```
anonymous block completed
Marinescu
```

```
/
begin
  dbms_output.put_line(pachet_utilitar.getDenumireBanca(106));
end;
/
```

```
set serveroutput on;
begin
  pachet_utilitar.angajatiBanca('FB');
end;
```

Script Output    Query Result

Task completed in 0 seconds

```
anonymous block completed
FB
```

```
begin
    pachet_utilitar.angajatiBanca('FB');
end;
/
begin
    dbms_output.put_line(pachet_utilitar.salariuAngajatiBanca('FB'));

```

Script Output x Query Result x | Task completed in 0 seconds

anonymous block completed  
25100

anonymous block completed  
La banca FB lucreaza urmatorii angajati: Nicolescu Andrei, Petrescu Mihai, Andrei Alexandru si Tudor Robert.

```
begin
    dbms_output.put_line(pachet_utilitar.salariuAngajatiBanca('FB'));
end;
/
begin
    pachet_utilitar.extrageriATM(9008);
end;
```

Script Output x Query Result x | Task completed in 0 seconds

anonymous block completed  
25100

```
begin
    pachet_utilitar.extrageriATM(9008);
end;
/
```

Script Output x Query Result x

| Task completed in 0 seconds

```
anonymous block completed
Detaliile extragerilor facute de pe ATM-ul cu ID-ul: 9008:
1. ID CARD: 807 SUMA: 49
2. ID CARD: 810 SUMA: 992
3. ID CARD: 811 SUMA: 890
4. ID CARD: 808 SUMA: 900
```