MINISTERUL EDUCAŢIEI ŞI CERCETĂRII ŞTIINŢIFICE

UNIVERSITATEA PETROL – GAZE DIN PLOIEŞTI

# FACULTATEA LITERE ŞI ŞTIINŢE

DEPARTAMENTUL INFORMATICĂ, TEHNOLOGIA INFORMAŢIEI, MATEMATICĂ ŞI FIZICĂ

SPECIALIZAREA INFORMATICĂ

CURSURI DE ZI

## **PROIECT LA DISCIPLINA** **BAZE DE DATE**

TEMA: Animale si Pasari - Fundatie tip Adapost, care Intermediaza si Adoptii

|  |  |
| --- | --- |
| Titular disciplină:  Vladoiu Monica |  |
|  | Student:  Manea Robert Constantin |

PLOIEŞTI

2021

## **CUPRINS**

1. DATE GENERALE
2. Formularea problemei
3. Resurse folosite
4. ETAPELE DEZVOLTARII BAZEI DE DATE
5. Crearea Diagramei Entitate-Legatura
6. Structura diagramei
7. Implementarea Bazei de date
8. Creare legaturi in baza de date
9. Set de interogari care sa exemplifice urmatoarele operatiile ale algebrei relationale(Reuniune, Diferenta, Selectie, Proiectie, )
10. CONCLUZII
11. BIBLIOGRAFIE
12. Date Generale
13. Formularea problemei

Baza de date create de mine ajuta Asociatia animale fericite sa gestioneze mai usor stocul de animale pe care il au. Sa gestioneze mai lista de client care vor sa adopte un animal pe care asociatia il detine in unul din adaposturile pe care le detine la mai multe adrese. Sa gestioneze usor lista de donator care vor sa ajute Asociatia cu mai multe tipuri de donatii.

1. Resurse folosite:

O bază de date este o colecție organizată de informații sau de date structurate, stocate electronic într-un computer. O bază de date este controlată, de regulă, de un sistem de management al bazelor de date (DBSM). Cumulat, datele, DBMS și aplicațiile asociate reprezintă un sistem de baze de date, denumit prescurtat bază de date.

Datele din cele mai obișnuite tipuri de baze de date sunt distribuite de regulă pe linii și coloane, în diferite tabele, pentru eficientizarea procesării și interogării datelor. Datele pot fi accesate, gestionate, modificate, actualizate, controlate și organizate cu ușurință. Majoritatea bazelor de date utilizează un limbaj structurat de interogare (SQL) pentru scrierea și interogarea datelor.

SQL este un limbaj de programare utilizat de aproape toate bazele de date relaționale, pentru interogarea, gestionarea și definirea datelor, precum și pentru controlul oferirii accesului. SQL a fost dezvoltat mai întâi de IBM în anii 1970, cu ajutorul esențial al companiei Oracle, ceea ce a condus la implementarea standardului SQL ANSI, ulterior SQL extinzându-se de la companii precum IBM, Oracle și Microsoft. Deși SQL este în continuare utilizat la scară largă, apar în continuare noi limbaje de programare.

De la apariția lor, la începutul anilor 1960, bazele de date au evoluat semnificativ. Bazele de date pentru navigare, precum cele cu o structură ierarhică, (care se bazau pe un model arborescent și permiteau numai interacțiuni de acest tip) și bazele de date în rețea (un model mai flexibil, care permite interacțiuni mai complexe), au constituit sistemele inițiale de stocare și gestionare a datelor. Deși erau simple, aceste sisteme nu erau flexibile. În anii 1980, bazele de date relaționale au devenit populare, fiind urmate de bazele de date orientate pe obiecte, în anii 1990. Ulterior au apărut bazele de date NoSQL, ca rezultat al dezvoltării internetului și a necesității de procesare mai rapidă a datelor nestructurate. Astăzi, bazele de date în cloud și bazele de date autonome revoluționează modul în care datele sunt colectate, stocate, gestionate și utilizate.

Bazele de date și foile de calcul (precum Microsoft Excel) reprezintă modalități alternative de a stoca informațiile. Diferențele principale dintre acestea sunt:

* modul în care datele sunt stocate și gestionate
* accesul la date
* cantitatea de date care poate fi stocată

Foile de calcul au fost create inițial pentru un singur utilizator și caracteristicile acestora reflectă acest lucru. Sunt excelente pentru un singur utilizator sau pentru un număr mic de utilizatori, care nu au nevoie de o gestionare complexă a datelor. Bazele de date, pe de altă parte, sunt concepute pentru a stoca un volum mult mai mare de informații colectate și organizate – uneori chiar cantități uriașe. Bazele de date permit mai multor utilizatori să acceseze și să interogheze simultan și securizat datele, utilizând algoritmi și limbaje complexe.

Există multe tipuri de baze de date. Cea mai bună baze de date pentru o anumită organizație depinde de modul în care organizația intenționează să o utilizeze.

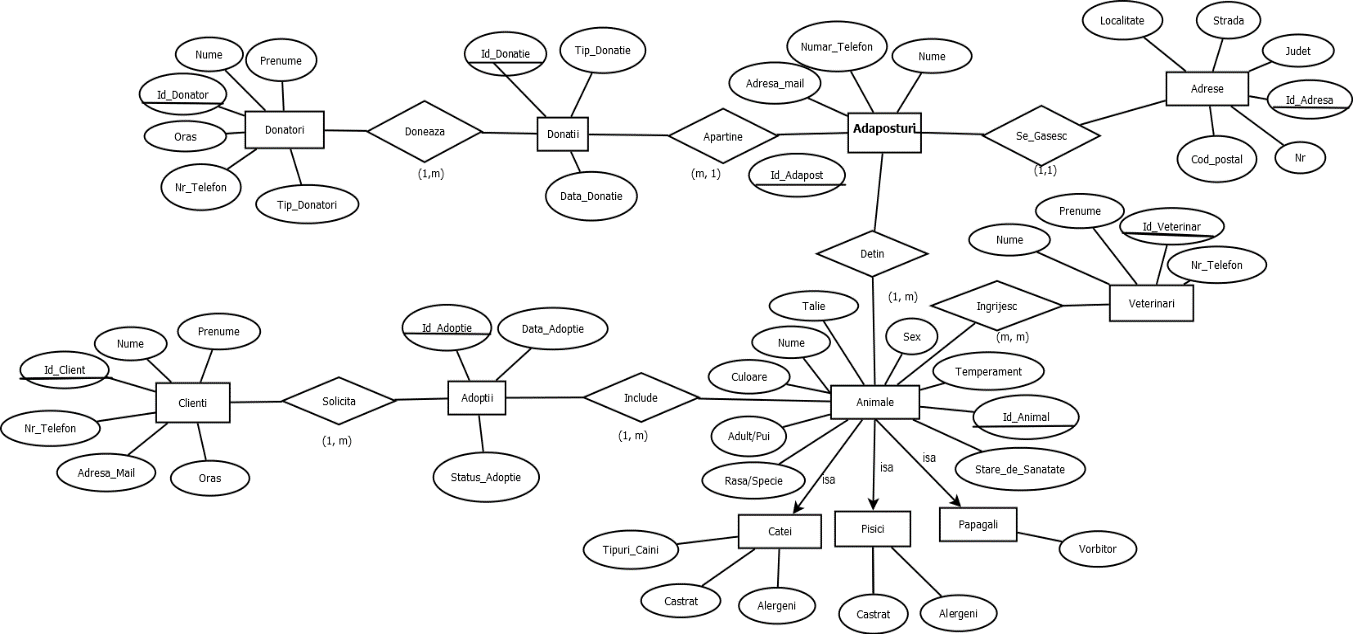
* **Baze de date relaționale.** Bazele de date relaționale au avut cea mai mare pondere în anii 1980. Elementele dintr-o bază de date relațională sunt organizate ca un set te tabele, cu rânduri și coloane. Tehnologia bazelor de date relaționale oferă cel mai eficient și flexibil mod de a accesa informațiile structurate.
* **Baze de date orientate pe obiecte** Informațiile dintr-o bază de date orientată pe obiecte sunt reprezentate sub formă de obiecte, așa cum se întâmplă în programarea orientată pe obiecte.
* **Baze de date distribuite** O bază de date distribuită constă în două sau mai multe fișiere aflate în site-uri diferite. Baza de date poate fi stocată pe mai multe computere aflate în aceeași locație fizică sau aflate în diferite rețele.
* **Depozite de date.** Un depozit central de date, reprezentând un tip de baze de date creat special pentru interogări și analize rapide.
* **Baze de date NoSQL.** O bază de date NoSQL, sau o bază de date non-relațională, permite stocarea și gestionarea datelor nestructurate sau semi-structurate (spre deosebire de o bază de date relațională, care definește modul în care trebuie compuse toate datele introduse). Bazele de date NoSQL au fost populare ca aplicații web, devenind din ce în ce mai uzuale și mai complexe.
* **Baze de date grafice.** O bază de date grafice stochează datele sub forma unor entități și a relațiilor dintre acestea.
* **Baze de date OLTP** O bază de date OLTP este o bază de date rapidă, pentru analize, destinată rulării unui număr mare de tranzacții efectuate de mai mulți utilizatori.

Acestea sunt numai câteva tipuri de baze de date dintre zecile de baze de date utilizate în prezent. Celelalte baze de date, mai puțin întâlnite, sunt personalizate pentru funcționalități științifice, financiare sau de alt tip. În plus față de diferitele tipuri de baze de date, schimbările în abordarea dezvoltării tehnologice și progresele substanțiale precum cloudul și automatizarea, propulsează bazele de date în direcții complet noi. Unele dintre cele mai recente baze de date includ

* **Baze de date open source** Un sistem de baze de date open source are codul sursă de tip open source, de exemplu bazele de date SQL sau NoSQL.
* **Baze de date în cloud.** O bază de date în cloud este o colecție de date structurată sau nestructurată, aflată pe o platformă de calcul în cloud privată, publică sau hibridă. Există două tipuri de modele de baze de date în cloud: bazele de date tradiționale și bazele de date ca serviciu (DBaaS). Cu DBaaS, activitățile administrative și de mentenanță sunt efectuare de furnizorul serviciului.
* **Baze de date multi-model** Bazele de date multi-model combină diferite tipuri de modele de baze de date într-un singur sistem back-end integrat. Acest lucru înseamnă că puteți acomoda diverse tipuri de date.
* **Baze de date pentru documente/JSON.** Concepute pentru stocarea, preluarea și managementul informațiilor orientate pe documente, bazele de date pentru documente reprezintă o modalitate modernă de stocare a datelor în format JSON, în loc de stocarea acestora pe rânduri și coloane.
* **Baze de date autonome.** Cel mai nou și mai revoluționar tip de baze de date, bazele de date autonome funcționează în cloud și utilizează machine learning pentru a automatiza reglarea, securizarea, backupul, actualizarea și alte activități obișnuite de management, efectuate de regulă de către administratorii bazelor de date.

II. ETAPELE DEZVOLTARII BAZEI DE DATE

* 1. Crearea Diagramei Entitate-Legatura



* 1. Structura Bazei de date
* Donatori Doneaza Donatii (1,m)
* Donatii Apartine Adaposturi (m,1)
* Adaposturi se Gasesc Adrese (1,1)
* Adaposturi detin Animale (1,m)
* Veterinari ingrijesc Animale (m,m)
* Animale include Adoptii (m,1)
* Clienti solicita adoptii (1,m)
* Adaposturi

In aceast tabel se stochaeaza informati utile (Id\_Adapost, Nume, Numar\_Telefon, adresa\_Mail,Responsabil\_Adoptii, Capacitate\_maxima) despre adposturile affiliate cu asociatia Animale Fericite. Multimea entitate Adaposturi are legatura de (1,1)cu multimea entitate Adrese care da adresa mai exacta a adapostului ales . Multimea entitate adrese este alcatuita din urmatoarele attribute :Id\_Adresa; Localitate; Strada; Judet; Cod\_Postal.

Legatura de (1,1) dintre Adaposturi si Adrese inseamna ca un adapost poate avea o singura adresa si o adresa poate fi selectata unui singur adapost.

Multimea entitate Adaposturi mai are o legatura de (1,m) cu multimea entitate Donatii care are urmatuarele attribute:Id\_Donatie; Tip\_Donatie; Data\_Donatie. Legatura de (1,m) dintre Adaposturi si Donatii inseamna ca un adapost poate primi mai multe donatii dar o donatie nu poate fii facuta la mai multe adaposturi.

Multimea entitate Donatii mai are o legatura (m,1) cu multimea entitate Donatori care are urmatoarele attribute: Id\_Donator; Nume; Prenume; Oras; Nr\_Telefon; Tip\_Donatori. Legatura (m,1) dintre Donatii si Donatori inseamna ca 1 donator poate face mai multe donatii dar o donatie poate fi facuta doar de un donator.

Multimea Entitate Adaposturi mai are o legatura de (1,m) cu multimea entitate Animale care are urmatoarele attribute (Id\_Animal; Nume; Talie; Culoare; Adult/Pui; Sex; Temperament; Stare\_de\_sanatate).Carela randul ei are trei legaturi “ISA”care sunt cazuri particulare de Animale :

* multimea entitate Catei (Tipuri\_Caini; Castrat; Alergeni);
* multimea entitate Pisici(Castrat; Alergeni);
* multimea entitate Papagali (Vorbitor).

Relatia (1,m) dintre Adaposturi si Animale inseamna ca un adapost poate detine mai multe animale dar un animal nu poate fi detinut de mai multe adaposturi.

Multimea entitate Animale are o relatie (m,m) cu multimea entitate Veterinari care are urmatoarele attribute(Id\_Veterinar, Nume, Prenume, Nr\_telefon)

Aceasta legatura insamna ca un animal poate fi ingrijit de mai multi veterinary iar un veterinary poate ingriji mai multe animale.

Multimea entitate Animale are o legatura (m,1) cu multimea entitate Adoptii care are urmatoarele attribute (Id\_Adoptie, Data\_Adoptie, Status\_Adoptie) cea ce insamna ca o adoptie poate include mai multe animale dar un animal nu poate fi adoptat de mai multe ori.

Multimea entitate Clienti cu urmatoarele attribute( Id\_Client, Nume, Prenume, Nr\_Telefon, Adresa\_mail, Oras) are legatura (1,m) cu multimea entitate Adoptii

Cea ce inseamna ca un client poate face mai multe adopti dar o adoptie nu poate fi facuta de mai multi client.

* 1. Implementarea Bazei De Date, query-urile folosite pentru crearea tabelelor:



CREATE TABLE `adaposturi` (

`Id\_Adapost` int(11) NOT NULL AUTO\_INCREMENT,

`Nume` varchar(255) NOT NULL,

`Numar\_Telefon` int(10) NOT NULL,

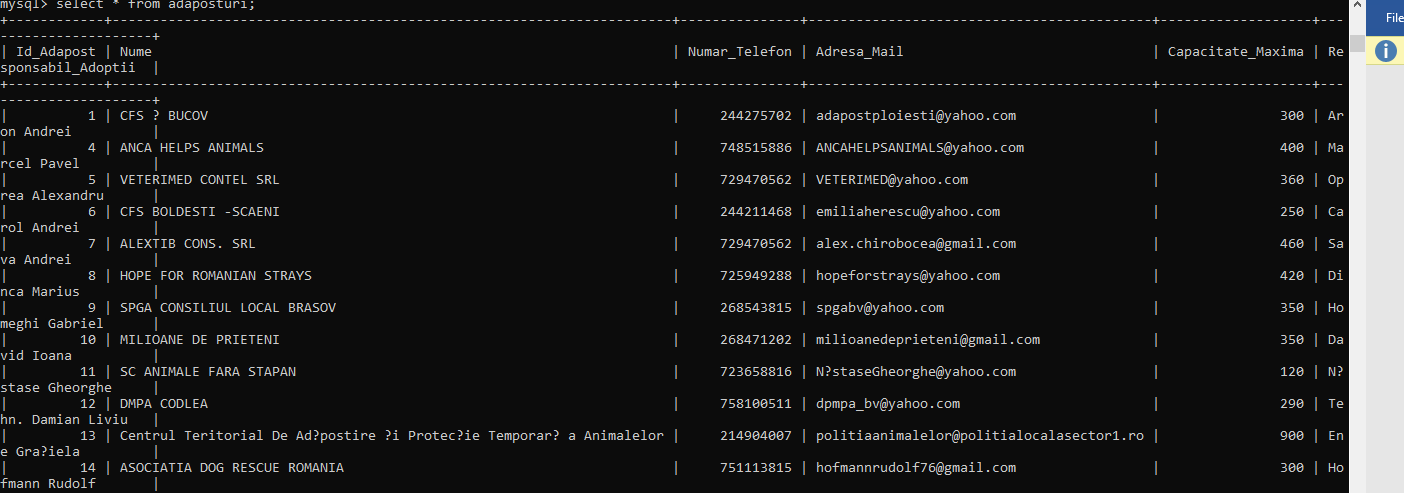
`Adresa\_Mail` varchar(100) NOT NULL,

`Capacitate\_Maxima` int(11) NOT NULL,

`Responsabil\_Adoptii` varchar(100) NOT NULL,

PRIMARY KEY (`Id\_Adapost`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `adoptii` (

`Id\_Adoptie` int(11) NOT NULL AUTO\_INCREMENT,

`Status\_Adoptie` enum('In Curs De Verificare','Aprobata','Refuzata') NOT NULL,

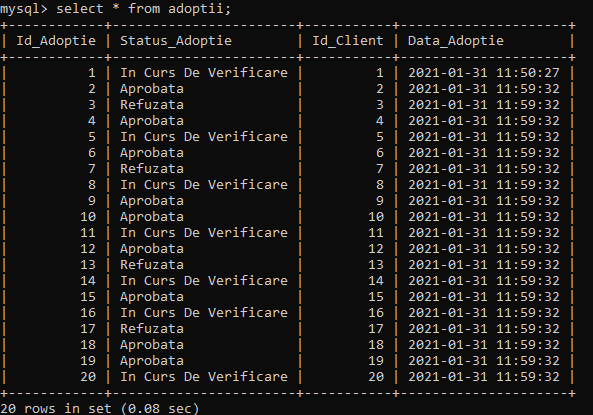
`Id\_Client` int(11) NOT NULL,

`Data\_Adoptie` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

PRIMARY KEY (`Id\_Adoptie`),

KEY `clienti` (`Id\_Client`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `adrese` (

`Id\_Adresa` int(11) NOT NULL AUTO\_INCREMENT,

`Nr\_Adresa` int(11) NOT NULL,

`Cod\_Postal` int(11) NOT NULL,

`Judet` varchar(30) NOT NULL,

`Strada` varchar(255) NOT NULL,

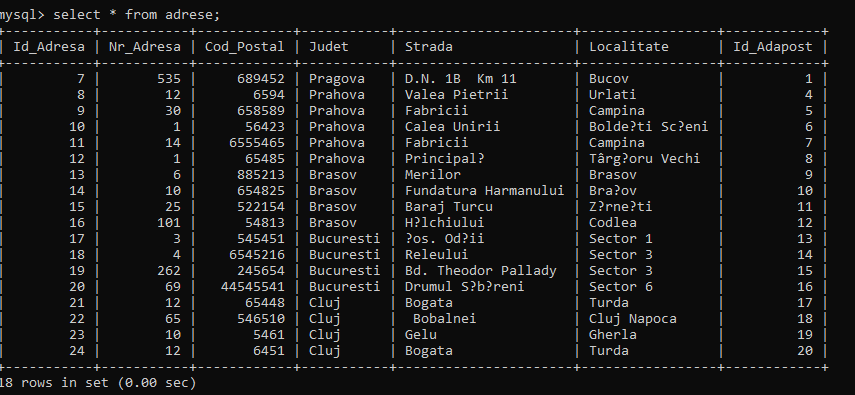
`Localitate` varchar(255) NOT NULL,

`Id\_Adapost` int(11) NOT NULL,

PRIMARY KEY (`Id\_Adresa`),

KEY `adaposturi` (`Id\_Adapost`)

) ENGINE=InnoDB AUTO\_INCREMENT=25 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `animale` (

`Id\_Animal` int(11) NOT NULL AUTO\_INCREMENT,

`Nume` varchar(50) NOT NULL,

`Talie` enum('Mica','Medie','Mare','Urias') NOT NULL,

`Culoare` varchar(40) NOT NULL,

`Adult/Pui` enum('Adult','Pui') NOT NULL,

`Rasa/Specie` varchar(50) NOT NULL,

`Stare\_Sanatate` enum('Sanatos','Bolnav','Sub\_Tratament') NOT NULL,

`Temperament` enum('Prietenos','Sperios','Agresiv','Somnoros','Vulcanic','Energic') NOT NULL,

`Sex` enum('Masculin','Feminin') NOT NULL,

`Id\_Adapost` int(11) NOT NULL,

`Id\_Adoptie` int(11) NOT NULL,

PRIMARY KEY (`Id\_Animal`),

KEY `Adaposturi` (`Id\_Adapost`),

KEY `Adoptii` (`Id\_Adoptie`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `catei` (

`Castrat` enum('Castrat','Necastrat') NOT NULL,

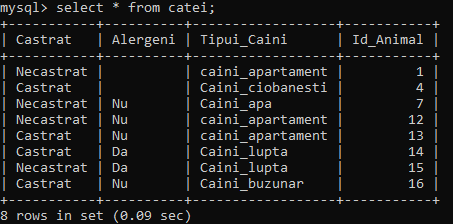
`Alergeni` enum('Da','Nu') NOT NULL,

`Tipui\_Caini` enum('Caini\_lupta','caini\_apartament','Caini\_buzunar','Caini\_paza','Caini\_vanatoare','Caini\_apa','Caini\_ciobanesti') NOT NULL,

`Id\_Animal` int(11) NOT NULL,

KEY `animale` (`Id\_Animal`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `clienti` (

`Id\_Client` int(11) NOT NULL AUTO\_INCREMENT,

`Nume` varchar(100) NOT NULL,

`Prenume` varchar(100) NOT NULL,

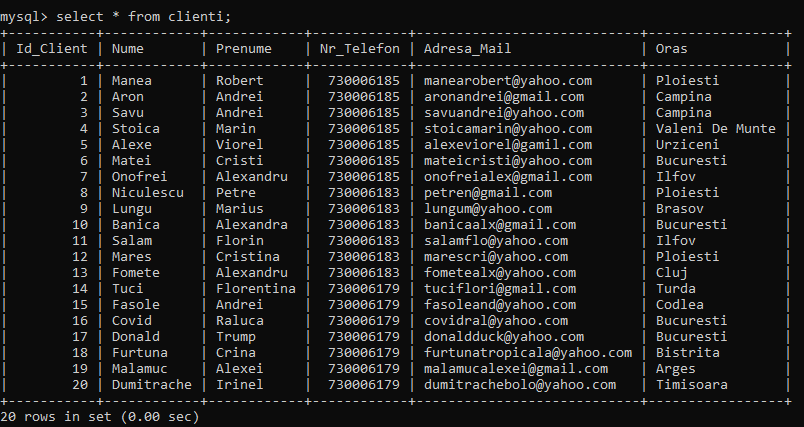
`Nr\_Telefon` int(10) NOT NULL,

`Adresa\_Mail` varchar(30) NOT NULL,

`Oras` varchar(30) NOT NULL,

PRIMARY KEY (`Id\_Client`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `donatii` (

`Id\_Donatie` int(11) NOT NULL AUTO\_INCREMENT,

`Tip\_Donatie` enum('Hrana','Bani','Animale') NOT NULL,

`Data\_Donatie` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

`Id\_Donator` int(11) NOT NULL,

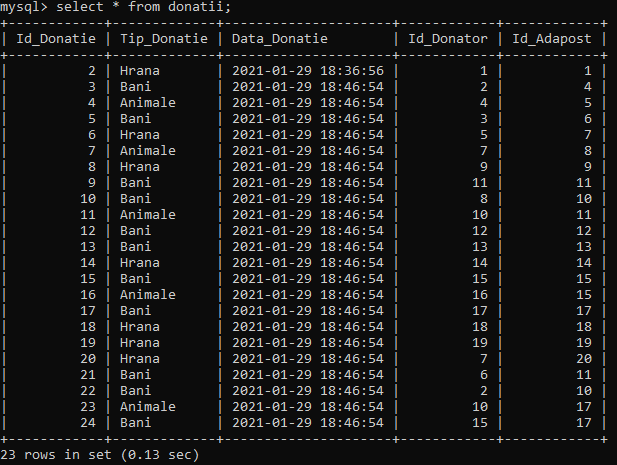
`Id\_Adapost` int(11) NOT NULL,

PRIMARY KEY (`Id\_Donatie`,`Data\_Donatie`),

KEY `Donatori` (`Id\_Donator`),

KEY `Adaposturi` (`Id\_Adapost`)

) ENGINE=InnoDB AUTO\_INCREMENT=25 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `donatori` (

`Id\_Donator` int(11) NOT NULL AUTO\_INCREMENT,

`Nume` varchar(100) NOT NULL,

`Prenume` varchar(100) NOT NULL,

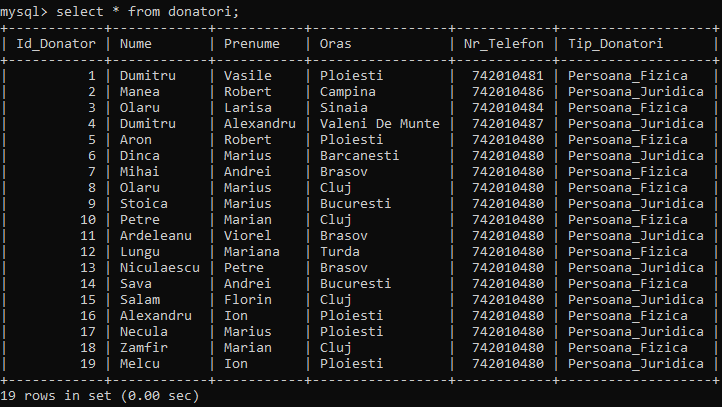
`Oras` varchar(50) NOT NULL,

`Nr\_Telefon` int(10) NOT NULL,

`Tip\_Donatori` enum('Persoana\_Fizica','Persoana\_Juridica') NOT NULL,

PRIMARY KEY (`Id\_Donator`)

) ENGINE=InnoDB AUTO\_INCREMENT=20 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `ingrijesc` (

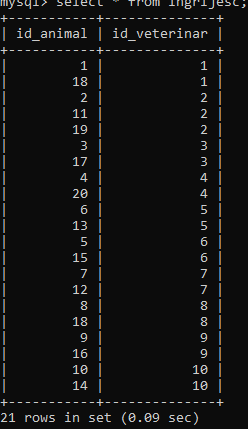
`id\_animal` int(11) NOT NULL,

`id\_veterinar` int(11) NOT NULL,

PRIMARY KEY (`id\_animal`,`id\_veterinar`),

KEY `id\_veterinar` (`id\_veterinar`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



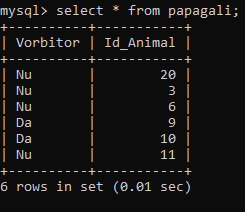
CREATE TABLE `papagali` (

`Vorbitor` enum('Da','Nu') NOT NULL,

`Id\_Animal` int(11) NOT NULL,

KEY `animale` (`Id\_Animal`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `pisici` (

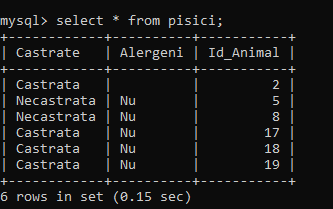
`Castrate` enum('Castrata','Necastrata') NOT NULL,

`Alergeni` enum('Da','Nu') NOT NULL,

`Id\_Animal` int(11) NOT NULL,

KEY `animale` (`Id\_Animal`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



CREATE TABLE `veterinari` (

`Id\_Veterinar` int(11) NOT NULL AUTO\_INCREMENT,

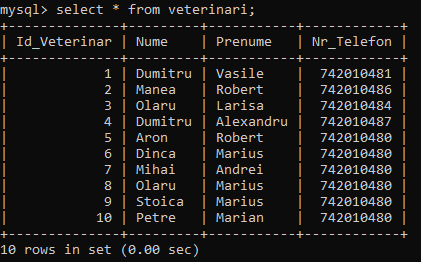
`Nume` varchar(100) NOT NULL,

`Prenume` varchar(100) NOT NULL,

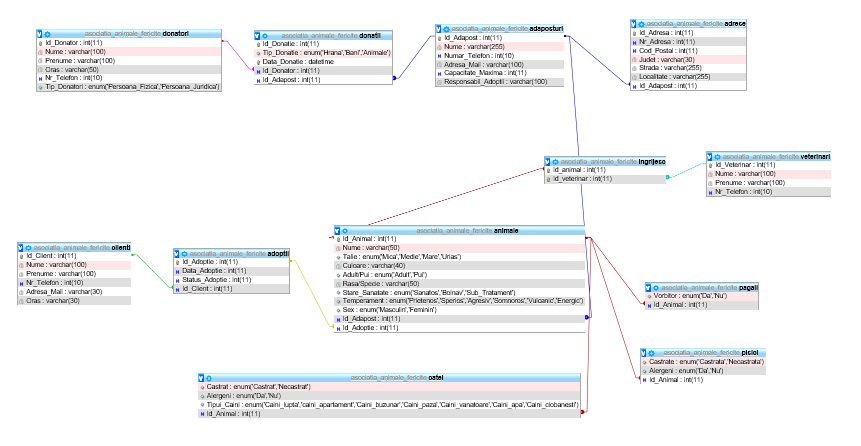
`Nr\_Telefon` int(10) NOT NULL,

PRIMARY KEY (`Id\_Veterinar`)

) ENGINE=InnoDB AUTO\_INCREMENT=20 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



* 1. Creare legaturilor in baza de date



* Animale include Adoptii (m,1)

ALTER TABLE `animale`

ADD CONSTRAINT `animale\_ibfk\_1` FOREIGN KEY (`Id\_Adoptie`) REFERENCES `adoptii` (`Id\_Adoptie`) ON DELETE CASCADE ON UPDATE CASCADE,

* Adaposturi detin Animale (1,m)

ADD CONSTRAINT `animale\_ibfk\_2` FOREIGN KEY (`Id\_Adapost`) REFERENCES `adaposturi` (`Id\_Adapost`) ON DELETE CASCADE ON UPDATE CASCADE;

* Donatori Doneaza Donatii (1,m)

ALTER TABLE `donatii`

ADD CONSTRAINT `donatii\_ibfk\_1` FOREIGN KEY (`Id\_Donator`) REFERENCES `donatori` (`Id\_Donator`) ON DELETE CASCADE ON UPDATE CASCADE,

* Donatii Apartine Adaposturi (m,1)

ADD CONSTRAINT `donatii\_ibfk\_2` FOREIGN KEY (`Id\_Adapost`) REFERENCES `adaposturi` (`Id\_Adapost`) ON DELETE CASCADE ON UPDATE CASCADE;

* Veterinari ingrijesc Animale (m,m)

ALTER TABLE `ingrijesc`

ADD CONSTRAINT `ingrijesc\_ibfk\_1` FOREIGN KEY (`id\_veterinar`) REFERENCES `veterinari` (`Id\_Veterinar`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `ingrijesc\_ibfk\_2` FOREIGN KEY (`id\_animal`) REFERENCES `animale` (`Id\_Animal`) ON DELETE CASCADE ON UPDATE CASCADE;

* Adaposturi se Gasesc Adrese (1,1)

ALTER TABLE `adrese`

ADD CONSTRAINT `adrese\_ibfk\_1` FOREIGN KEY (`Id\_Adapost`) REFERENCES `adaposturi` (`Id\_Adapost`) ON DELETE CASCADE ON UPDATE CASCADE; ALTER TABLE `adrese`

ADD CONSTRAINT `adrese\_ibfk\_1` FOREIGN KEY (`Id\_Adapost`) REFERENCES `adaposturi` (`Id\_Adapost`) ON DELETE CASCADE ON UPDATE CASCADE;

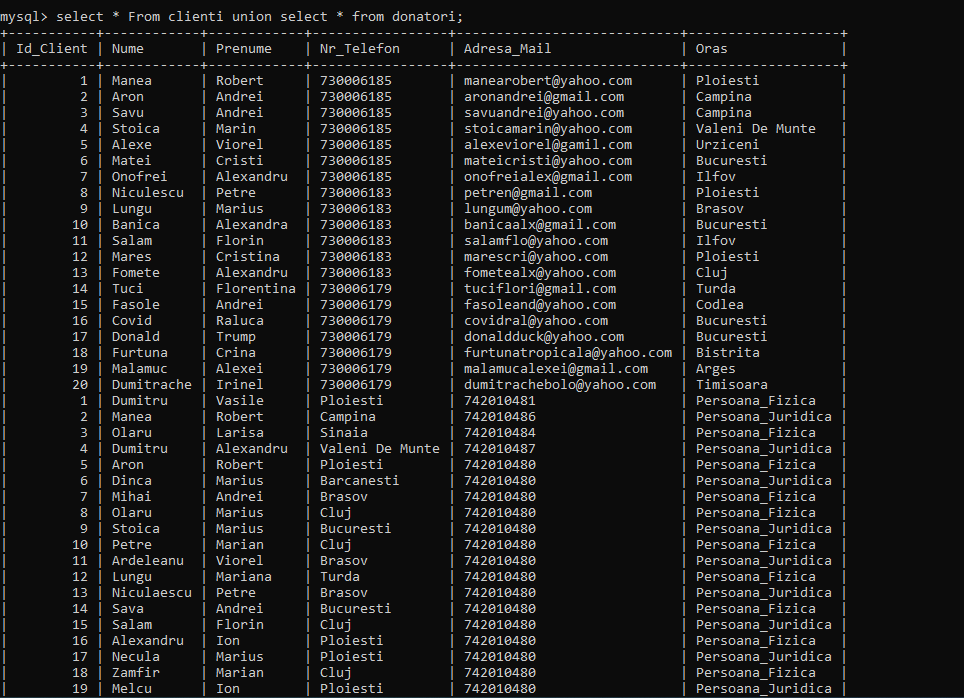
* Clienti solicita adoptii (1,m)

ALTER TABLE `adoptii`

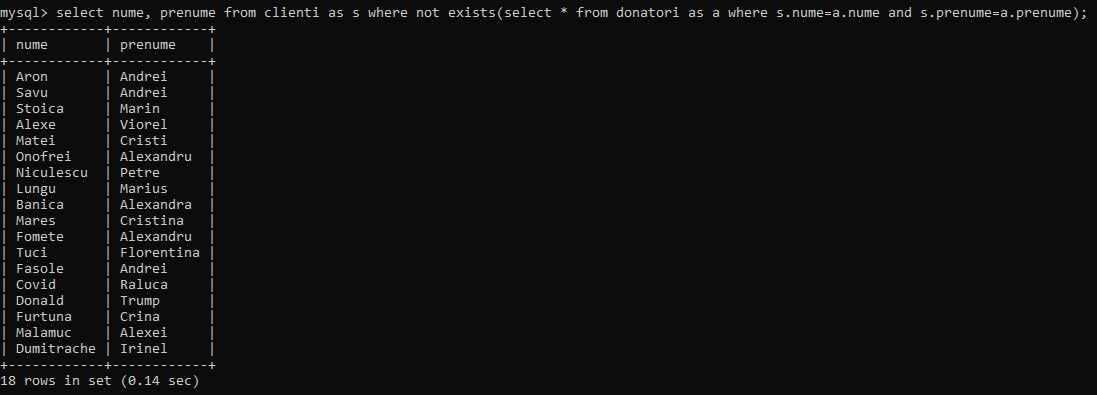
ADD CONSTRAINT `adoptii\_ibfk\_1` FOREIGN KEY (`Id\_Client`) REFERENCES `clienti` (`Id\_Client`) ON DELETE CASCADE ON UPDATE CASCADE;

1. Set de interogari care sa exemplifice urmatoarele operatiile ale algebrei relationale(Reuniune, Diferenta, Selectie, Proiectie)

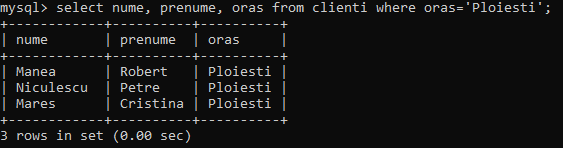
* Reuniunea



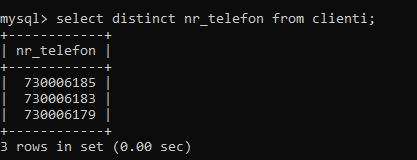
* Diferenta



* Selectie



* Proiectie



* + 1. Concluzii

Baza de date creata poate fi implementata cu usurinta pentru managementul unei asociati de animale cu mai multe adaposturi cu sedii aflate la adrese diferite. Pe langa administrarea adoptiilor, structura bazei de date permite stocarea informatiilor despre veterinarii responsabili cu sanatatea animalelor.

* + 1. BIBLIOGRAFIE

<https://www.oracle.com/ro/database/what-is-database/>