UNIVERSITATEA DIN CRAIOVA (*Sigla echipei*)
Facultatea de Automatica, Calculatoare si Electronica
Specializarea Automatica si Informatica Aplicata
Grupa AIA 2.3A

PROIECT la disciplina BAZE DE DATE

"STATIUNE DE TURISM, ODIHNA SI TRATAMENT"

Echipa (*PDG TURISM*),
Membri echipei
Penoiu Dennis-Gabriel

Coordonator,
Prof. Dr. Ing. Viorel Stoian

Craiova, 2021

CUPRINS

Capitolul I.
Introducere

1.1. Definiţii

Datele reprezintă informaţii fixate pe un anumit suport fizic în vederea utilizării şi prelucrării într-un anumit scop.

Baza de date (data base) este o colecţie de date organizate care serveşte unui anumit scop (nu conţine date care nu sunt relevante). Faptul că sunt organizate înseamnă că sunt stocate, reprezentate şi accesate într-o manieră consistentă.

Dezvoltarea bazelor de date s-a bazat pe 2 cerinţe:

- persistenţa datelor (datele trebuie să fie valide pentru mai multe rulări),
- simplitatea stocării şi manipulării datelor.

1.2. Arhitectura unui sistem de baze de date

Sistemul bazelor de date are 4 nivele:

1.2.1. Nivelul conceptual

Este nivelul fundamental ce descrie într-un mod natural şi fără ambiguităţi sistemul ce urmează a fi modelat. La acest nivel se realizează schema conceptuală care reprezintă design-ul general al sistemului.

1.2.2. Nivelul extern

La acest nivel se realizează *schema externă* care este astfel realizată încât grupuri diferite de utilizatori să acceseze numai anumite subscheme ale schemei conceptuale globale (din motive de relevanţă şi securitate). Aceeaşi informaţie poate fi reprezentată în mod diferit (grafice, tabele) din motive de experienţă sau interes ale utilizatorilor.

1.2.3. Nivelul logic

Pentru o anumită aplicaţie dată schema conceptuală se converteşte într-o structură de nivel inferior (*schemă logică*) unde se alege un model logic adecvat de organizare a datelor (model relaţional, ierarhic, reţea etc.). Schema logică este reprezentată cu ajutorul unor structuri abstracte specifice modelului respectiv (ex.: tabele).

1.2.4. Nivelul intern

După ce a fost realizată schema logică aceasta se concretizează într-o *schemă internă* care este specifică sistemului de gestiune a bazelor de date ales (Oracle, Acces, DB2 etc.). Schema internă include toate detaliile despre stocarea fizică şi structurile de acces în sistemul respectiv (ex.: indecşi, clustere etc.). Chiar şi în cadrul aceluiaşi sistem de gestiune a bazelor de date utilizatori diferiţi pot construi scheme interne diferite.

1.3. Sisteme de gestiune a bazelor de date (SGBD)

1.3.1. Noţiuni despre SGBD

Un **SGBD** (**S**istem de **G**estiune a **B**azelor de **D**ate) sau **DBMS** (**D**ata**B**ase **M**anagement **S**ystem) este un sistem software care gestionează toate procesele dintr-o bază de date şi care permite utilizatorului să interacţioneze cu aceasta.
Principalele **funcţiuni** ale unui SGBD sunt:

- stocarea datelor,
- definirea structurilor de date,
- manipularea datelor,
- interogarea (extragerea şi prelucrarea) datelor,
- asigurarea securităţii datelor,
- asigurarea integrităţii datelor,
- accesul concurent la date cu păstrarea consistenţei acestora,
- asigurarea unui mecanism de recuperare a datelor,
- asigurarea unui mecanism de indexare care să permită accesul rapid la date.

1.4. Accesul concurent (simultan) la date

În cazul existenţei mai multor utilizatori, un SGBD trebuie să gestioneze accesul curent al acestora la date, menţinând în acelaşi timp integritatea bazei de date.

a) Când două sau mai multe persoane vor să vizualizeze aceleaşi date fără a le modifica însă, totul este în ordine şi nu trebuie luate măsuri suplimentare.
b) Când cel puţin o persoană doreşte să facă modificări asupra unor date care, în acelaşi timp, sunt vizualizate de alte persoane, atunci SGBD trebuie să realizeze şi să stocheze mai multe copii ale datelor şi să transfere toate modificările copiilor într-o singură versiune atunci când întreaga operaţiune este terminată.

c) În cazul când mai multe persoane încearcă să modifice aceleaşi date în acelaşi timp SGBD utilizează metoda blocării (locking). Utilizatorul care a efectuat primul modificarea datelor le blochează, ceilalţi utilizatori neputându-le modifica până ce operaţia efectuată de acesta nu este încheiată. În Oracle blocarea se poate face la nivel de tabel sau la nivel de rând. Cu cât unitatea de date este mai mică cu atât concurenţa este mai eficientă şi utilizatorii aşteaptă mai puţin. Oracle blochează în mod implicit orice rând asupra căruia se execută o operaţie de modificare.

1.5.Tipuri de utilizatori ai bazei de date

● **Administratorul BD (Data Base Administrator – DBA)**
- defineşte BD,
- asigură buna funcţionare a BD.

● **Programatorul (dezvoltatorul de aplicaţii)**
- creează programe pentru manipularea şi interogarea datelor din BD,
- se ocupă de accesul concurent (integritatea şi consistenţa datelor),
- urmăreşte performanţa, mentenanţa şi portabilitatea codului.

● **Utilizatorul final**
- interoghează şi manipulează datele fără să ţină cont de modul lor de organizare, de păstrarea integrităţii şi de accesul concurent.

Capitolul II
TEMA DE PROIECT

STATIUNE DE TURISM, ODIHNA SI TRATAMENT

In acest proiect voi prezenta o statiune turistica,iar obiectivul meu este prezentarea unui regim hotelier care are la baza toate cele enumerate mai jos.Statiunea Turistica se numeste Nisipurile de Aur din Bulgaria,iar hotelul este Melia Grand Hermitage.

Cateva indicatii despre informatiile pe care trebuie sa le contina baza de date:

- informatii despre persoane(date de contact,nume,prenume,etc): angajati, turisti, bolnavi .....
- informatii despre functiile pe care le ocupa angajatii
- informatii despre salariile angajatiilor
- informatii despre gradul de confort al camerelor in functie de pret
- informatii despre adresa hotelului
- informatii despre posibilitatile de distractie, odihna, refacere
- informatii despre tipurile de tratamente si proceduri medicale
- informatii despre locatii unde se desfasoara diferite activitati (corpuri de cladiri, sali, terenuri ...)
- informatii despre alte locatii (biblioteci, sali de lectura, terase, restaurante ...)
- informatii despre contacte(numarul de telefon al hotelului,room service)
- informatii despre contacte urgente(urgenta si lift)
- informatii despre check in(sosire la ora 12:00)
- informatii despre check out(eliberare la ora 11:00)

Capitolul III.
Schema Conceptuala

3.1. Notiuni Teoretice

Proiectarea Bazelor de Date cuprinde 3 etape principale:
a)Realizarea schemei conceptuale a BD
b) Realizarea proiectului logic al BD (schemei logice a BD)
c)Realizarea proiectului fizic al BD (schemei fizice a BD)

3.1.1. Realizarea schemei conceptuale a unei BD (modelul entitate-legatura)

În prima fază,o echipă nominalizată colectează (achiziţionează) datele corespunzatoare din sistem, apoi urmează faza de organizare a acestora utilizându-se modelul entitate-legătură. Principalele concepte folosite în acest model sunt: entitatea, relaţia (legătura) şi atributul.

Entitatea este un obiect de interes din sistem pentru care trebuie să existe date înregistrate.

Observaţii:

- Fiecare entitate are o denumire unică în cadrul unui sistem.
- Entităţile sunt reprezentate prin substantive, dar nu orice substantiv folosit în descrierea sistemului este entitate, ci numai acelea care au o semnificaţie deosebită.
- Fiecare entitate trebuie să fie bine definită şi precizată pentru a se evita confuziile.

Relaţia (legătura) este o asociere (raport) nedirecţionată între 2 entităţi.

Observaţii:

- Relaţiile sunt reprezentate prin verbe, dar nu orice verb utilizat în descrierea sistemului este relaţie.
- Între 2 entităţi pot exista mai multe relaţii.
- Pot exista în cadrul unei scheme conceptuale mai multe relaţii cu acelaşi nume, dar cele care leagă aceleaşi entităţi trebuie să aibă nume diferite.

Cardinalitatea unei relaţii indică numarul de instanţe din fiecare entitate care poate participa la relaţie.

Există 3 tipuri de cardinalitate:

- "**mulţi-la-unu" (many-to-one, M:1)**.

Relaţia dintre entităţile A şi B este de tipul "mulţi-la-unu" dacă fiecarei instanţe din A i se poate asocia cel mult o singură instanţă din B şi fiecărei instanţe din B i se pot asocia mai multe instanţe din A.

- **"unu-la-unu" (one-to-one, 1:1)**.

Relaţia dintre entităţile A şi B este de tipul "unu-la-unu" dacă fiecărei instanţe din A i se poate asocia cel mult o singură instanţă din B şi fiecărei instanţe din B i se poate asocia cel mult o singură instanţă din A.

- **"mulţi-la-mulţi"** (**many-to-many, M:M)**.

Relaţia dintre entităţile A şi B este de tipul "mulţi-la-unu" dacă fiecărei instanţe din A i se pot asocia mai multe instanţe din B şi fiecărei instanţe din B i se pot asocia mai multe instanţe din A.

Valorile prezentate până acum (M:1, 1:1, M:M) reprezintă **cardinalitatea maximă** a unei relaţii. Pe de altă parte, o relaţie este caracterizată şi de o cardinalitate minimă ce indică obligativitatea participării entităţilor la relaţie. **Cardinalitatea minimă** a unei relaţii poate avea valorile: 0:0, 0:1, 1:1.

Dacă avem cardinalitatea minimă a unei relaţii egală cu 1 înseamnă că există o *participare totală* a entităţii la relaţie (participare obligatorie). Dacă avem cardinalitatea minimă egală cu 0 înseamnă că există o *participare parţială* a entităţii la relaţie.

Atributul este o caracteristică a unei entităţi sau a unei relaţii. Fiecare entitate are un anumit număr de atribute despre care sunt înregistrate date. Ex.: *nume, prenume, dată*. Fiecare atribut poate lua valori care furnizează informaţii despre entitatea respectivă. Ex.: *Bumbescu, Alina, 1.10.84*.
Şi relaţiile pot avea atribute. Ex.: "lucrează_în" → *data_angajării*.

Observaţii:

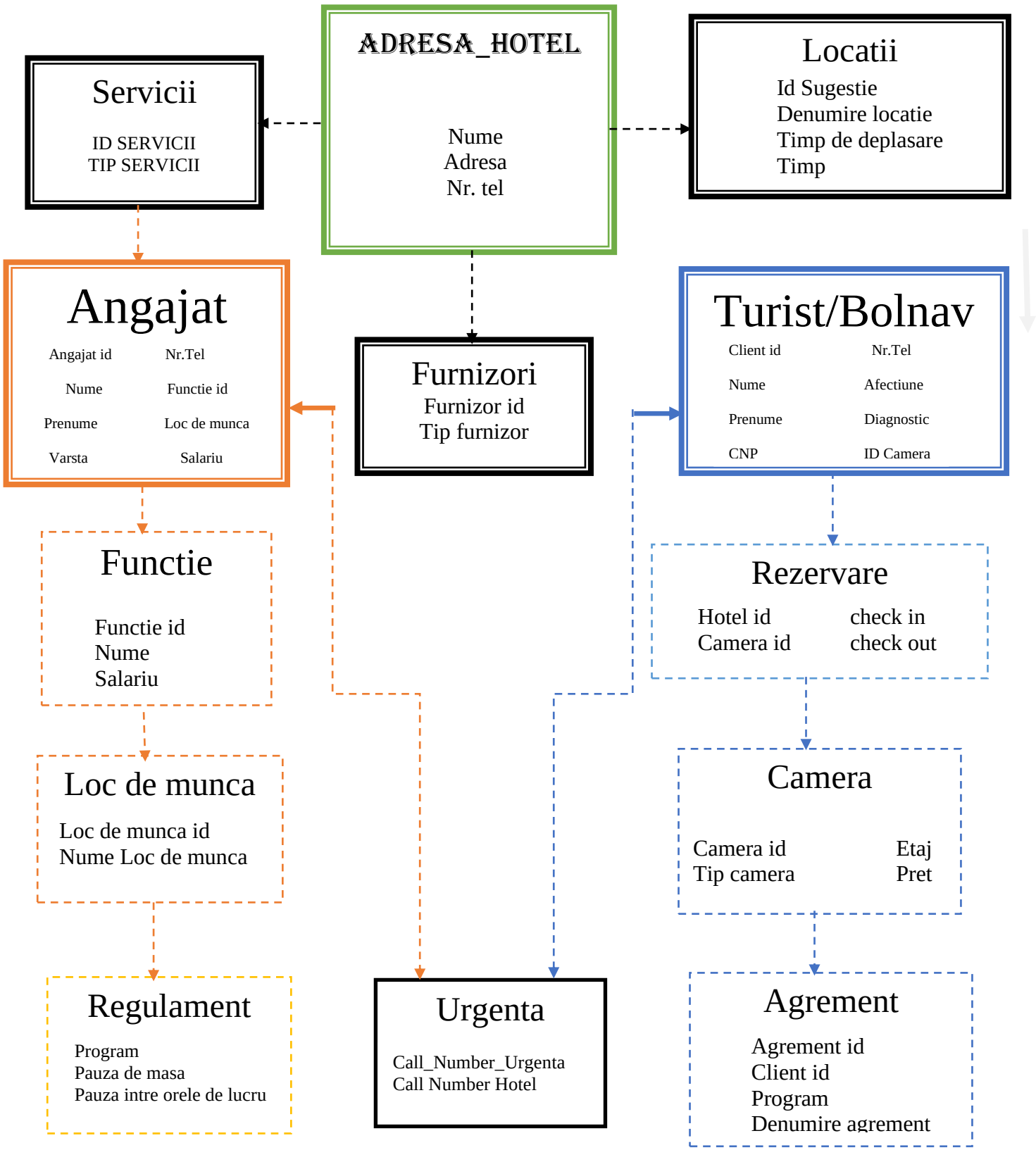
» Numele unui atribut este unic în cadrul unei entităţi sau al unei relaţii.

» Atributele sunt întotdeauna substantive, dar nu orice substantiv este atribut.

» Pentru fiecare atribut este necesară o descriere, împreună cu domeniul de valori (întreg, şir de caractere, dată calendaristică etc.).

» Trebuie evitate *atributele indirecte*. Ex.: *numele_facultăţii* este un atribut indirect pentru tabelul STUDENT şi un atribut direct pentru tabelul FACULTATE.

Schema conceptuala
Hotel Melia Grand Hermitage



Capitolul IV.
Schema Logica

4.1. Notiuni Teoretice

4.1.1.Realizarea schemei logice a unei baze de date

Pentru realizarea schemei logice a unei baze de date se porneşte de la scheme conceptuală (modelul entitate – legătură) urmărindu-se conversia entităţilor şi a legăturilor în tabele relaţionale.

Regulile de conversie ale entităţilor, legăturilor şi atributelor sunt următoarele:

4.1.2. Transformarea entităţilor

Regulă generală: entităţile se transformă în tabele.
Subcazuri:

a) *Entităţile independente* devin tabele independente, adică tabele a căror cheie primară nu conţine chei străine.
b) *Entităţile dependente* devin tabele dependente (tabele detaliu) adică tabele a căror cheie primară conţine cheia străină ce face referinta la cheia primara a entitatii de care depinde entitatea in cauza.
c) *Subentitatile* devin subtabele, adica tabele a caror cheie primara este cheia straina pentru tabelul superentitate.

Avantajele supertabelelor -simplificarea programelor de manipulare a datelor.
Dezavantajele supertabelelor -probleme de integritate, apar valori de Null.
Avantajele subtabelelor -mai stabile, mai flexibile, ocupa spatiu mai mic, contin mai putine valori de Null.
Dezavantajele subtabelelor -se ingreuneaza manipularea datelor.

4.1.3. Transformarea relatiilor (legaturilor)

Regula generala: Relatiile (legaturile) se convertesc in chei straine.
Conventie: plasamentul cheii straine este simbolizat printr-o sageata. Atunci cand cheia straina este inclusa in cheia primara, varful sagetii este plin si este gol in caz contrar .
Cazuri:

a) *Relatiile 1:1* devin chei straine. Cheia straina este plasata in tabelul cu linii mai putine.

b) *Relatiile M:1* devin chei straine plasate in tabelul care se afla in partea de “multi” a relatiei.

Cazuri:

● Cheia straina nu poate avea valoarea Null, iar in cazul entitatilor dependente ea va face parte chiar din cheia primara a tabelului detaliu.

● Cheia straina poate avea valoarea Null si nu poate face parte din cheia primara.

c) O relatie M:M se transforma in 2 relatii M:1. In acest caz se construieste un tabel special numit *tabel asociativ* care are 2 chei straine care fac referinta la cheile primare ale celor 2 tabele aflate in relatia M:M. Cheia sa primara este formata din cele 2 chei straine plus (eventual) alte atribute suplimentare.

d) O relaţie de tip 3 se transforma intr-un numar de relatii de tip 2, egal cu numărul de tabele asociate. Aceste relatii (legaturi) se stabilesc intre un tabel asociativ si tabelele asociate. Tabelul asociativ are cate o cheie straina pentru fiecare tabel asociat, iar cheia sa primara este formata din toate aceste chei straine plus (eventual) alte atribute suplimentare.

4.1.4. Transformarea atributelor

Regula generala: Atributele se convertesc in coloane ale tabelelor provenite din entitati sau chiar in tabele.

Cazuri:

a) *Atributele simple* ale unei entitati devin coloane in tabelul provenit din acea entitate.

b) Toate componentele unui *atribut compus* devin coloane.

c) *Atributele repetitive (multivaloare)* ale unei entitati devin tabele dependente ce contin fiecare o cheie straina (care face referinta la cheia primara a entitatii) si atributul multivaloare. Cheia primara a unui astfel de nou tabel este formata din cheia straina plus alte coloane suplimentare.

d) *Atributele simple ale unei relatii 1:1 sau M:1* devin coloane ale tabelului care contine cheia straina.

e) *Atributele simple ale unei relatii M:M* vor deveni coloane ale tabelului asociativ.

f) *Atributele repetitive (multivaloare) ale unei relatii 1:1 sau 1:M* devin tabele dependente de tabelul care contine cheia straina.

Atributele repetitive ale unei relatii M:M devin tabele dependente de tabelul asociativ corespunzator relatiei. Cheia primara a acestor tabele dependente va fi formata din cheia straina respectiva plus una sau mai multe coloane suplimentare.

Capitolul V. Normalizarea BD

5.1. Notiuni Teoretice

In trecut normalizarea era utilizata pentru proiectarea unei BD. In prezent, proiectare unei BD se realizeaza pe baza celor prezentate anterior (schema conceptuala, schema logica), iar normalizarea intervine asupra tabelelor obtinute pe baza schemei logice eliminand unele probleme care pot apare in procesul de proiectare initial: redundanta in date, anomalii la actualizare.

Normalizarea reprezinta procesul de descompunere a unui tabel relational in mai multe tabele care satisfac anumite reguli si care stocheaza aceleasi date ca si tabelul initial astfel incat sa fie eliminate redundanta in date si anomaliile la actualizare.

a) **Caracterul reversibil al normalizarii**.

Prin caracter reversibil al normalizarii se intelege faptul ca descompunerea se face fara pierdere de informatie, adica tabelul initial poate fi reconstituit prin compunerea naturala, pe atribute comune, a tabelelor rezultate.

Pentru un tabel R care se descompune prin proiectie in mai multe tabele: R1, R2, … Rn, conditia de descompunere fara pierdere de informatie presupune ca in urma operatiei de compunere naturala a tabelelor R1, R2, … Rn sa se obtina tabelul R.

Regula Casey-Delobel (caz particular de descompunere fara pierdere de informatie):

Fie un tabel R(X, Y, Z) care se descompune prin proiectie in tabelele R1(X, Y) si R2(X, Z) unde prin X notam setul de coloane comune ale tabelelor R1 si R2, iar prin Y si Z, coloanele specifice lui R1, respectiv R2. Conditia de descompunere fara pierdere de informatie presupune ca tabelul R sa fie obtinut prin compunerea naturala a tabelelor R1 si R2.

In SQL:

```
SELECT   R1.X, R1.Y, R2.Z
FROM     R1, R2
WHERE    R1.X = R2.X
```

b) **Dependenta functionala**

Definitie: Fie R un tabel relational si X si Y doua submultimi de coloane ale lui R. Spunem ca *X determina functional pe Y* sau ca *Y depinde functional de X* daca nu exista doua randuri in tabelul R care sa aiba aceleasi valori pentru coloanele din X si sa aiba valori diferite pentru cel putin o coloana din Y.

Notatie uzuala: X → Y

unde X = determinant

Y = determinat

X → Y este triviala daca Y $\subseteq$ X.

Existenta dependentelor functionale pentru care determinantul nu este cheie a tabelului duce la aparitia redundantei in date si a anomaliilor la actualizare in lucrul cu BD.

c) **Dependenta functionala tranzitiva**

Fie R un tabel relational, X o submultime de coloane a lui R si A o coloana a lui R. Spunem ca *A este dependenta tranzitiv de X* daca exista o submultime de coloane Y care nu include coloana A si care nu determina functional pe X astfel incat X → Y si Y → A. Daca in aceasta definitie se doreste sa se evidentieze si Y atunci se spune ca *A depinde functional de X prin intermediul lui Y* si se scrie: X → Y → A.

d) **Descompunerea minimala**

Prin descompunerea minimala a unui tabel se intelege o descompunere astfel incat nici o coloana din tabelele rezultate nu poate fi eliminata fara a duce la pierderea de informatii si implicit la pierderea caracterului ireversibil al transformarii. Aceasta inseamna ca nici unul dintre tabelele rezultate nu poate fi continut unul in altul.

Este de dorit ca procesul de normalizare sa conserve dependentele functionale netranzitive dintre date (atat determinantul cat si determinatul sa se regaseasca intr-unul din tabelele rezultate prin descompunere). Dependentele functionale tranzitive nu trebuie conservate deoarece ele pot fi deduse din cele netranzitive.

Un tabel relational se poate afla in 6 situatii diferite numite forme normale:

- prima forma normala,
- a 2-a forma normala,
- a 3-a forma normala,

Prima forma normala (1NF - First Normal Form)

Un tabel relational este in 1NF daca fiecarei coloane ii corespunde o valoare indivizibila (atomica). Deci orice valoare nu poate fi compusa dintr-o multime de valori (atributele compuse) si nu sunt admise atributele repetitive.

A doua forma normala (2NF - Second Normal Form)

Un tabel relational R este in a doua forma normala (2NF) daca si numai daca:

- R este in 1NF
- Orice coloana care depinde partial de o cheie a lui R este inclusa in acea cheie.

Deci, 2NF nu permite dependentele functionale partiale fata de cheile tabelului, cu exceptia dependentelor triviale, de incluziune.

A treia forma normala (3NF - Third Normal Form)

Un tabel relational R este in a treia forma normala (3NF) daca si numai daca:

- R este in 2NF
- Pentru orice coloana A necontinuta in nici o cheie a lui R, daca exista un set de coloane X astfel incat X → A, atunci fie X contine o cheie a lui R, fie A este inclusa in X.

A doua conditie din definitie interzice dependentele functionale totale fata de alte coloane in afara celor care constituie chei ale tabelului. Prin urmare, un tabel este in 3NF daca orice coloana care nu este continuta intr-o cheie, depinde de cheie, de intreaga cheie si numai de cheie. Daca exista astfel de dependente functionale trebuie efectuate noi descompuneri ale tabelelor. Tinand cont de definirea notiunii de *dependenta functionala tranzitiva* (4.1.) se poate formula urmatoarea definitie:

Un tabel relational R este in a treia forma normala (3NF) daca si numai daca:

- R este in 2NF
- Orice coloana necontinuta in nici o cheie a lui R nu este dependenta tranzitiv de nici o cheie a lui R.

Capitolul VI. Denormalizarea BD

6.1.Notiuni Teoretice

6.1.1. Denormalizarea BD

Denormalizarea unei BD reprezinta procesul invers operatiei de normalizare si duce la cresterea redundantei datelor. Prin aceasta se doreste, in principal, cresterea performantei si simplificarea programelor de interogare a datelor.

Obs.:
- Denormalizarea se face numai dupa o normalizare corecta.
- Denormalizarea se face printr-o selectare strategica a structurilor care aduc avantaje semnificative
- Denormalizarea trebuie insotita de masuri suplimentare de asigurare a integritatii datelor.

a) Cresterea performantei

Un caz frecvent intalnit in interogarea BD este cazul unor operatii sau calcule foarte des utilizate.

Ex.: Fie tabelele care modeleaza tranzactiile unui magazin care vinde articole la comanda.:

VANZARI_2 (*cod_comanda*, *cod articol*, cantitate)
ARTICOL (cod_articol, nume_articol, cost_articol)
COMANDA_1 (cod_comanda, data, *cod_client*)
CLIENT (cod_client, nume_client, nr_telefon)

Sa presupunem ca majoritatea rapoartelor cerute de conducerea magazinului se refera la cantitatea totala vanduta intr-o luna pentru fiecare articol. In acest caz este util un tabel suplimentar:

ARTICOL_LUNA (cod_articol, luna, cantitatea_totala)

Utilizarea acestui tabel suplimentar ceeaza avantajul important al unei interogari usoare si rapide, dar si dezavantajul cresterii redundantei datelor fiind, in acelasi timp, periclitata integritatea datelor. Solutia gasita este atasarea la tabelele VANZARI_2 si COMANDA_1 de declansatoare (trigger-e) care se activeaza dupa fiecare modificare data de INSERT, UPDATE, DELETE. Un efect produs: incetinirea operatiilor de actualizare.

a) Simplificarea codului pentru manipularea datelor

Exemplu: Fie tabelul STOCURI utilizat de o firma pentru inregistrarea cantitatilor de materiale existente in fiecare din depozitele sale.

STOCURI (cod_depozit, cod_material, nume_material, cantitate)

Se cere aflarea tuturor depozitelor in care exista ciocolata.

Dupa normalizare avem:

STOCURI_1 (cod_depozit, cod_material, cantitate)

MATERIAL (cod_material, nume_material)

Interogarea in SQL va fi:

- **varianta nenormalizata**:

```
SELECT cod_depozit, cantitate
FROM stocuri
WHERE nume_material = “ciocolata”;
```

- **varianta normalizata**:

```
SELECT cod_depozit, cantitate
FROM stocuri_1, material
WHERE stocuri_1.cod_material = material.cod_material
AND nume_material = “ciocolata”;
```

O solutie care rezolva problema conta in a crea vederi bazate pe tabele normalizate.

Ex:
```
CREATE VIEW stocuri AS
SELECT cod_depozit, stocuri_1.cod_material, nume_material, cantitate
FROM stocuri_1, material
WHERE stocuri_1.cod_material = material. cod_material;
```

6.2.Denormalizarea Bazei de Date

Deoarece în etapele elaborarii schemei conceptuale si logice s-au avut în vedere si caracteristicile de la etapa de denormalizare aceasta nu mai este necesară.

Capitolul VII. SGBD MySQL

7.1.Notiuni Teoretice

7.1.1. Sisteme de gestiune a bazelor de date (SGBD)

7.1.2.Noţiuni despre SGBD

Un **SGBD** (**S**istem de **G**estiune a **B**azelor de **D**ate) sau **DBMS** (**D**ata**B**ase **M**anagement **S**ystem) este un sistem software care gestionează toate procesele dintr-o bază de date şi care permite utilizatorului să interacţioneze cu aceasta.

Principalele **funcţiuni** ale unui SGBD sunt:

- stocarea datelor,
- definirea structurilor de date,
- manipularea datelor,
- interogarea (extragerea şi prelucrarea) datelor,
- asigurarea securităţii datelor,
- asigurarea integrităţii datelor,
- accesul concurent la date cu păstrarea consistenţei acestora,
- asigurarea unui mecanism de recuperare a datelor,
- asigurarea unui mecanism de indexare care să permită accesul rapid la date.

7.1.1.2.Modele de date (moduri de organizare a datelor)

Modelul de date reprezintă un tipar după care este organizată din punct de vedere logic baza de date. După modelul folosit există mai multe tipuri de SGBD.

- **SGBD ierarhic**

Modelul ierarhic stochează datele în structuri de tip arbore. Se consideră că între date există o relaţie de tip *părinte-copil*. Nivelul superior al arborelui (rădăcina) poate avea orice număr de descendenţi care şi ei, la rândul lor, au alţi descendenţi etc. În prezent, modelul ierarhic este depăşit şi nu se mai foloseşte aproape deloc.

- **SGBD reţea**

Datele sunt stocate sub formă de înregistrări cu legături multiple şi complexe între ele. Este o extindere a celui ierarhic. Aici un *copil* poate avea mai mulţi *părinţi* sau chiar niciunul. Caracteristicile principale ale SGBD reţea sunt:

- reprezentare date complexe
- extrem de puţin flexibil
- design extrem de complicat

În prezent este puţin folosit.

- **SGBD relaţional**

Reprezintă cea mai simplă structură pe care o poate avea o bază de date. Datele sunt organizate în tabele formate din înregistrări şi câmpuri. În acest caz bazele de date relaţionale sunt foarte flexibile şi uşor de mânuit. Cele mai populare baze de date relaţionale: Oracle, Acces, Informix şi Sybase. Altele : SQL server şi DB2.

- **SGBD orientat pe obiect**

Este tipul cel mai nou de SGBD fiind introdus conceptul de *obiect.* Integrează principiile programării orientate pe obiect (C^{++}, Actor etc.) cu cele ale bazelor de date. Gestionează *obiecte complexe* (date neconvenţionale) (texte, grafice, hărţi imagini sunete); *obiecte dinamice* (programe, simulări). Tehnologia este la început (Oracle 8 şi 9).

7.1.2. Sisteme de Gestiune a Bazelor de Date Relaţionale (SGBDR)

7.1.2.1. Noţiuni generale

Un SGBDR este un SGBD care utilizează modelul relaţional ca şi concepţie de organizare a datelor. În 1985 Codd a publicat un set de 13 reguli în raport cu care un SGBD poate fi considerat relaţional. În prezent niciun SGBD nu respectă întreg setul de reguli care are rolul de a stabili gradul în care unul sau altul dintre SGBD-uri este relaţional.

Regulile lui Codd

Rg. 1: Regula reprezentării logice a datelor

Într-o bază de date relaţională toate datele sunt reprezentate la nivel logic într-un singur mod, şi anume sub formă de valori atomice în tabele.

Valoarea stocată la intersecţia dintre un rând şi o coloană ale unui tabel trebuie să fie atomică, adică să nu se mai poată descompune din punct de vedere logic.

Regula este de bază. Când este încălcată crează probleme de integritate a datelor, demonstrează o proiectare deficientă a BD, iar SGBD-ul îşi pierde calitatea de relaţional.

Rg. 2: Regula accesului la date

Toate datele individuale din tabele trebuie să fie accesibile prin furnizarea numelui tabelului, numelui coloanei şi valorii cheii primare.

Modelul relaţional presupune inexistenţa rândurilor identice, iar fiecare rând poate fi identificat prin valoarea cheii primare.

Rg. 3: Regula reprezentării valorilor necunoscute

Un sistem relaţional trebuie să permită declararea şi manipularea sistematică a valorilor Null cu semnificaţia unor valori necunoscute sau inaplicabile.

Un SGBDR trebuie să facă diferenţa între valoarea numerică 0 şi Null sau între şirul de caractere “spaţiu” şi valoarea Null. Valoarea Null trebuie să reprezinte absenţa informaţiei respective şi are un rol important în implementarea restricţiilor de integritate structurală (integritatea entităţii şi integritatea referirii).

Rg. 4: Regula dicţionarului de date

Descrierea bazei de date (dicţionarul de date) trebuie să fie reprezentată la nivel logic tot sub formă de tabele, astfel încât asupra acesteia să se poată aplica aceleaşi operaţii ca şi asupra datelor propriu-zise.

Dicţionarul de date trebuie să fie organizat la nivel logic şi accesat la fel ca orice tabel din baza de date. Constă din tabele şi tabele virtuale (vederi) care pot fi interogate la fel ca oricare alte tabele sau vederi, folosind comanda SELECT.

Rg. 5: Regula limbajului de acces

Într-un sistem relaţional trebuie să existe cel puţin un limbaj de accesare a datelor, care să asigure următoarele operaţii: definirea tabelelor de bază şi a tabelelor virtuale (vederilor), manipularea şi interogarea datelor (atât interactiv cât şi prin program), definirea restricţiilor de integritate, autorizarea accesului la date, delimitarea tranzacţiilor.

Limbajul SQL permite:

- definirea tabelelor (comenzile CREATE TABLE, ALTER TABLE, DROP TABLE);
- definirea vederilor (comenzile CREATE VIEW, ALTER VIEW, DROP VIEW);
- manipularea datelor (comenzile INSERT, UPDATE, DELETE);
- interogarea datelor (comanda SELECT);
- definirea restricţiilor de integritate (clauza CONSTRAINT folosită la definirea tabelelor)
- autorizarea accesului la date (printr-un set de privilegii de sistem şi la nivel de obiect);
- delimitarea tranzacţiilor (operaţiile COMMIT şi ROLLBACK).

Rg. 6: Regula de actualizare a tabelelor virtuale (vederilor)

Un SGBD trebuie să poată determina dacă o vedere poate fi actualizată sau nu.

Rg. 7: Regula manipulării datelor

Un sistem relaţional trebuie să ofere posibilitatea procesării tabelelor (de bază sau virtuale) nu numai în operaţiile de interogare a datelor cât şi în cele de inserare, actualizare şi ştergere.

Rg. 8: Regula independenţei fizice a datelor

Programele de aplicaţie nu trebuie să depindă de modul de stocare şi accesare fizică a datelor.

Un SGBDR trebuie să separe complet aspectele de ordin fizic ale datelor (modul de stocare şi modul de acces la date) de cele de ordin logic.

Rg. 9: Regula independenţei logice a datelor

Programele de aplicaţie nu trebuie să fie afectate de nici o restructurare logică a tabelelor bazei de date care conservă datele.

Orice modificare efectuată asupra unui tabel care conservă datele din acesta (de ex., dacă un tabel trebuie divizat în 2 părţi din motive de creştere a performanţei) nu trebuie să afecteze funcţionarea programelor de aplicaţie.

Rg. 10: Regula independenţei datelor din punctul de vedere al integrităţii

Regulile de integritate a bazei de date trebuie să fie definite în limbajul utilizat de sistem pentru definirea datelor şi nu în cadrul aplicaţiilor individuale: în plus, aceste reguli de integritate trebuie stocate în dicţionarul de date.

Această regulă se referă la faptul că restricţiile de integritate trebuie impuse la definirea tabelelor bazei de date şi nu în cadrul aplicaţiilor care folosesc aceste tabele

Rg. 11: Regula independenţei datelor din punctul de vedere al distribuirii

Programele de aplicaţie nu trebuie să fie afectate de distribuirea pe mai multe calculatoare a bazei de date.

Rg. 12: Regula privind prelucrarea datelor de către un limbaj de nivel inferior

Orice limbaj nerelaţional folosit pentru accesarea datelor trebuie să respecte aceleaşi condiţii de integritate ca şi limbajul relaţional de acces.

Rg. 0: Regula de bază

Un SGBD Relaţional trebuie să fie capabil să gestioneze BD exclusiv pe baza caracteristicilor sale relaţionale.

Această regulă are rolul de a rezuma concluziile desprinse din celelalte reguli. În esenţă, acesta înseamnă că sistemul trebuie să îndeplinească toate funcţiile prin manipulări în care unitatea de procesare să fie tabelul (mulţimi de rânduri), asupra căruia să se efectueze operaţiile specifice modelului relaţional.

Nici unul dintre SGBD-urile actuale nu satisface în totalitate toate cele 13 reguli ale lui Codd. De aceea, in practică, pentru a putea fi considerat relaţional, un SGBD trebuie să îndeplinească un set minimal de cerinţe.

Un SGDB se numeşte ***minimal relaţional*** dacă satisface următoarele condiţii:

- Toate datele din cadrul bazei de date sunt reprezentate prin valori în tabele.
- Nu există pointeri observabili de către utilizator între tabele.
- Sistemul asigură operatorii relaţionali de proiecţie, selecţie şi compunere naturală, fără limitări impuse de considerente interne.

Un SGDB se numeşte ***complet relaţional*** dacă este minimal relaţional şi satisface în plus următoarele condiţii:

- Sistemul asigură toate operaţiile de bază ale algebrei relaţionale, fără limitări impuse de considerente interne.
- Sistemul asigură restricţiile de integritate de bază ale modelului relaţional (integritatea entităţii şi integritatea referenţială).

Un SGDB definit prin regulile lui Codd este un SGDB relaţional ideal.

7.2.Aplicatii

Pentru exemplificarea noțiunilor teoretice s-a creat o baza de date in MySQL, iar în cele ce urmează o să vă prezentăm o parte din etapele și interogările parcurse.

```
CREATE TABLE ADRESA_HOTEL(
NUME VARCHAR2(30),
ADRESA VARCHAR2(100),
NR_TEL NUMBER);

CREATE TABLE RECEPTIONER(
ANGAJAT_ID INTEGER,
NUME VARCHAR2(30),
PRENUME VARCHAR2(30),
```

```
TELEFON NUMBER,
VARSTA NUMBER,
FUNCTIA VARCHAR2(30));

CREATE TABLE ANGAJAT(
ANGAJAT_ID INTEGER,
NUME VARCHAR2(30),
PRENUME VARCHAR2(30),
TELEFON NUMBER,
VARSTA NUMBER,
FUNCTIA_ID INTEGER,
LOC_DE_MUNCA_ID INTEGER,
SALARIU NUMBER);

CREATE TABLE FUNCTIE(
FUNCTIE_ID INTEGER,
NUME VARCHAR2(30),
SALARIU NUMBER);

CREATE TABLE LOC_DE_MUNCA(
LOC_DE_MUNCA_ID INTEGER,
NUME_LOC_DE_MUNCA VARCHAR2(30));

CREATE TABLE REGULAMENT.ANGAJATI(
PROGRAM NUMBER,
PAUZA.DE.MASA NUMBER,
PAUZA_INTRE_ORELE_DE_MUNCA NUMBER);

CREATE TABLE TURIST(
CLIENT_ID INTEGER,
NUME VARCHAR2(30),
PRENUME VARCHAR2(30),
TELEFON NUMBER(10),
CNP NUMBER(14),
ID_CAMERA NUMBER);

CREATE TABLE BOLNAV(
BOLNAV_ID INTEGER,
```

```
NUME VARCHAR2(30),
PRENUME VARCHAR2(30),
TELEFON NUMBER(10),
CNP NUMBER(14),
AFECTIUNE VARCHAR(30),
DIAGNOSTIC_ID NUMBER,
ID_CAMERA NUMBER);

CREATE TABLE REZERVARE(
HOTEL_ID INTEGER,
CAMERA_ID NUMBER,
CHECK_IN DATE,
CHECK_OUT DATE);

CREATE TABLE CAMERA(
CAMERA_ID NUMBER,
TIP_CAMERA VARCHAR2(30),
ETAJ NUMBER,
PRET NUMBER,
MONEDA VARCHAR2(10));

CREATE TABLE AGREMENT(
AGREMENT_ID NUMBER,
CLIENT_ID INTEGER,
PROGRAM NUMBER,
DENUMIRE_AGREMENT VARCHAR2(30));

CREATE TABLE LOCATII(
ID_SUGESTIE INTEGER,
DENUMIRE_LOCATIE VARCHAR2(30),
TIMP_DE_DEPLASARE NUMBER,
TIMP VARCHAR2(10));

CREATE TABLE SERVICII(
ID_SERVICII INTEGER,
TIP_SERVICII VARCHAR2(30));

CREATE TABLE FURNIZOR(
```

```
FURNIZOR_ID INTEGER,
TIP_FURNIZOR VARCHAR2(30));

CREATE TABLE URGENTA(
CALL_NUMBER_URGENTA NUMBER(10),
CALL_NUMBER_HOTEL NUMBER(10));

insert into ADRESA_HOTEL(NUME,ADRESA,NR_TEL)
VALUES('HOTEL MELIA GRAND HERMITAGE','Este situat în partea
central? a sta?iunii Nisipurile de Aur',0251764537896);

insert into
RECEPTIONER(ANGAJAT_ID,NUME,PRENUME,TELEFON,VARSTA,FUN
CTIE) VALUES (1,'Caprescu','Gheorghe',0775856802,45,'Receptioner');

insert into ANGAJAT(ANGAJAT_ID, NUME, PRENUME,TELEFON,
VARSTA,FUNCTIA_ID,LOC_DE_MUNCA_ID,SALARIU) values (2, 'Puiu',
'Vasile', 0749938559, 05 , 102 , 2500);
insert into ANGAJAT(ANGAJAT_ID, NUME, PRENUME,TELEFON,
VARSTA,FUNCTIA_ID,LOC_DE_MUNCA_ID,SALARIU) values (3, 'Panait',
'Alexandru', 0722948557, 10 , 212 , 1500);
insert into ANGAJAT(ANGAJAT_ID, NUME, PRENUME,TELEFON,
VARSTA,FUNCTIA_ID,LOC_DE_MUNCA_ID,SALARIU) values (4, 'Matei',
'Ion', 0748348554, 15 , 214 , 1500);
insert into ANGAJAT(ANGAJAT_ID, NUME, PRENUME,TELEFON,
VARSTA,FUNCTIA_ID,LOC_DE_MUNCA_ID,SALARIU) values (5, 'Pirvu',
'Cosmin', 0722834539, 25 , 332 , 2500);
insert into ANGAJAT(ANGAJAT_ID, NUME, PRENUME,TELEFON,
VARSTA,FUNCTIA_ID,LOC_DE_MUNCA_ID,SALARIU) values (6, 'Vlad',
'Arnold', 0766977550, 07 , 52 , 2300);
insert into ANGAJAT(ANGAJAT_ID, NUME, PRENUME,TELEFON,
VARSTA,FUNCTIA_ID,LOC_DE_MUNCA_ID,SALARIU) values (7, 'Ionescu',
'John', 0786929879, 23 , 122 , 3500);

insert into FUNCTIE(FUNCTIE_ID,NUME,SALARIU) values (05,'Puiu
Vasile',2500);
insert into FUNCTIE(FUNCTIE_ID,NUME,SALARIU) values (10,'Panait
Alexandru',2000);
```

```
	insert into FUNCTIE(FUNCTIE_ID,NUME,SALARIU) values (15,'Matei
Ion',2500);
	insert into FUNCTIE(FUNCTIE_ID,NUME,SALARIU) values (25,'Pirvu
Cosmin',2500);
	insert into FUNCTIE(FUNCTIE_ID,NUME,SALARIU) values (07,'Vlad
Arnold',2300);
	insert into FUNCTIE(FUNCTIE_ID,NUME,SALARIU) values
(23,'Ionescu John',3500);

	insert into LOC_DE_MUNCA(LOC_DE_MUNCA_ID ,
NUME_LOC_DE_MUNCA) VALUES (102,'Ajutor receptioner');
	insert into LOC_DE_MUNCA(LOC_DE_MUNCA_ID ,
NUME_LOC_DE_MUNCA) VALUES (212,'Camerist');
	insert into LOC_DE_MUNCA(LOC_DE_MUNCA_ID ,
NUME_LOC_DE_MUNCA) VALUES (214,'Camerist');
	insert into LOC_DE_MUNCA(LOC_DE_MUNCA_ID ,
NUME_LOC_DE_MUNCA) VALUES (122,'Manager');
	insert into LOC_DE_MUNCA(LOC_DE_MUNCA_ID ,
NUME_LOC_DE_MUNCA) VALUES (332,'inginer tehnic');
	insert into LOC_DE_MUNCA(LOC_DE_MUNCA_ID ,
NUME_LOC_DE_MUNCA) VALUES (52,'secretar');

	insert into
REGULAMENT.ANGAJATI(PROGRAM,PAUZA.DE.MASA,PAUZA_INTRE_
ORELE_DE_MUNCA)VALUES( 1 , 30 , 10);

	insert into
TURIST(CLIENT_ID,NUME,PRENUME,TELEFON,CNP,ID_CAMERA)VALU
ES(01,'Penoiu','Dennis',0782645324,5000814292521,12);
	insert into
TURIST(CLIENT_ID,NUME,PRENUME,TELEFON,CNP,ID_CAMERA)VALU
ES(02,'Mitea','Bogdan',0722535329,2646814292521,23);
	insert into
TURIST(CLIENT_ID,NUME,PRENUME,TELEFON,CNP,ID_CAMERA)VALU
ES(03,'Mitea','Mihnea',0722335364,5003544692522,31);
	insert into
TURIST(CLIENT_ID,NUME,PRENUME,TELEFON,CNP,ID_CAMERA)VALU
ES(04,'Aron','John',0774829046,3574830272456,100);
```

```
	insert into
TURIST(CLIENT_ID,NUME,PRENUME,TELEFON,CNP,ID_CAMERA)VALU
ES(05,'Pasare','Catalin',0725445555,5000826292521,56);
	insert into
TURIST(CLIENT_ID,NUME,PRENUME,TELEFON,CNP,ID_CAMERA)VALU
ES(06,'Halep','Simona',0766443322,5000854294520,79);

	insert into BOLNAV(BOLNAV_ID,NUME,PRENUME,TELEFON ,CNP
,AFECTIUNE ,DIAGNOSTIC_ID
,ID_CAMERA)values(01,'Sharapova','Maria',0722765432,2364739603897,'Sc
olioza',174,36);
	insert into BOLNAV(BOLNAV_ID,NUME,PRENUME,TELEFON ,CNP
,AFECTIUNE ,DIAGNOSTIC_ID
,ID_CAMERA)values(02,'Basarab','Cristian',0722955850,2876509876123,'Di
abetul zaharat insulino-dependent',132,03);
	insert into BOLNAV(BOLNAV_ID,NUME,PRENUME,TELEFON ,CNP
,AFECTIUNE ,DIAGNOSTIC_ID
,ID_CAMERA)values(03,'Dimitrie','Cantemir',0766998801,2000313191817,'Ci
foza',173,15);
	insert into BOLNAV(BOLNAV_ID,NUME,PRENUME,TELEFON ,CNP
,AFECTIUNE ,DIAGNOSTIC_ID
,ID_CAMERA)values(04,'Magelan','Gigel',0785043974,3000654323130,'Tum
ori maligne ale stomacului',96,27);
	insert into BOLNAV(BOLNAV_ID,NUME,PRENUME,TELEFON ,CNP
,AFECTIUNE ,DIAGNOSTIC_ID
,ID_CAMERA)values(05,'Purcelan','Vasile',0766442345,4000234989796,'Car
diomiopatia',476,08);
	insert into BOLNAV(BOLNAV_ID,NUME,PRENUME,TELEFON ,CNP
,AFECTIUNE ,DIAGNOSTIC_ID
,ID_CAMERA)values(06,'Tarie','Virgil',0765478937,230048393123,'Pancreatit
a acuta',587,35);

	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
12,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
23,'12-03-2020','14-03-2020');
```

```
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
31,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
100,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
56,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
79,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
36,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
03,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
15,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
27,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
08,'12-03-2020','14-03-2020');
	insert into
REZERVARE(HOTEL_ID,CAMERA_ID,CHECK_IN,CHECK_OUT)values(01,
35,'12-03-2020','14-03-2020');

	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(12,'Dubla',1,400,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(23,'Single',2,300,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(31,'twin',3,400,'EURO');
```

```
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(100,'Dubla',10,400,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(56,'Dubla',5,400,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(79,'Single',7,300,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(36,'Dubla',3,400,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(03,'Tripla',0,500,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(15,'Dubla',1,400,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(27,'Single',2,300,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(08,'Dubla',0,400,'EURO');
	insert into CAMERA(CAMERA_ID, TIP_CAMERA ,ETAJ
,PRET,MONEDA)values(35,'Dubla',3,400,'EURO');

	insert into
AGREMENT(AGREMENT_ID,CLIENT_ID,PROGRAM,DENUMIRE_AGREM
ENT)VALUES(01,12,08,'MIC DEJUN');
	insert into
AGREMENT(AGREMENT_ID,CLIENT_ID,PROGRAM,DENUMIRE_AGREM
ENT)VALUES(02,23,10,'SPORT');
	insert into
AGREMENT(AGREMENT_ID,CLIENT_ID,PROGRAM,DENUMIRE_AGREM
ENT)VALUES(03,31,12,'PRANZ');
	insert into
AGREMENT(AGREMENT_ID,CLIENT_ID,PROGRAM,DENUMIRE_AGREM
ENT)VALUES(04,100,14,'ACTIVITATI RECREATIVE');
	insert into
AGREMENT(AGREMENT_ID,CLIENT_ID,PROGRAM,DENUMIRE_AGREM
ENT)VALUES(05,56,18,'CINA');
	insert into
AGREMENT(AGREMENT_ID,CLIENT_ID,PROGRAM,DENUMIRE_AGREM
ENT)VALUES(06,79,20,'CLUB NOCTURN KARAOKE,DANS,ETC');
```

```
	insert into
AGREMENT(AGREMENT_ID,CLIENT_ID,PROGRAM,DENUMIRE_AGREM
ENT)VALUES(07,36,00,'NOAPTE BUNA!');

	insert into LOCATII(ID_SUGESTIE ,DENUMIRE_LOCATIE
,TIMP_DE_DEPLASARE,TIMP)VALUES(01,'PLAJA GOLDEN
SANDS',10,'MINUTE');
	insert into LOCATII(ID_SUGESTIE ,DENUMIRE_LOCATIE
,TIMP_DE_DEPLASARE,TIMP)VALUES(02,'Magazine
comerciale',5,'MINUTE');
	insert into LOCATII(ID_SUGESTIE ,DENUMIRE_LOCATIE
,TIMP_DE_DEPLASARE,TIMP)VALUES(03,'Centrul statiunii',10,'MINUTE');
	insert into LOCATII(ID_SUGESTIE ,DENUMIRE_LOCATIE
,TIMP_DE_DEPLASARE,TIMP)VALUES(01,'Restaurant',5,'MINUTE');
	insert into LOCATII(ID_SUGESTIE ,DENUMIRE_LOCATIE
,TIMP_DE_DEPLASARE,TIMP)VALUES(01,'Sala de sport',5,'MINUTE');

	insert into
SERVICII(ID_SERVICII,TIP_SERVICII)VALUES(01,'HOTELIERE');
	insert into SERVICII(ID_SERVICII,TIP_SERVICII)VALUES(02,'ROOM
SERVICE');
	insert into
SERVICII(ID_SERVICII,TIP_SERVICII)VALUES(03,'URGENTE');

	INSERT INTO
FURNIZOR(FURNIZOR_ID,TIP_FURNIZOR)VALUES(127,'SUPERMARKET'
);
	INSERT INTO
FURNIZOR(FURNIZOR_ID,TIP_FURNIZOR)VALUES(234,'ANGRO-URI');
	INSERT INTO
FURNIZOR(FURNIZOR_ID,TIP_FURNIZOR)VALUES(01,'PERSOANA
FIZICA');

	INSERT INTO
URGENTA(CALL_NUMBER_URGENTA,CALL_NUMBER_HOTEL)VALUES(
911,0251764537896);
```

Capitolul VIII *Concluzii*

Considerăm că în acest proiect am parcurs toți pașii teoretici și practici ai creării unei baze de date plecând de la un scenariu și formându-ne atât ideile cât și acțiunile în conformitate cu cerințele impuse. Totodată s-au realizat și principalele obiective:

- ✓ Noțiuni generale
- ✓ Prezentarea aplicației
- ✓ Proiectarea structurii bazei de date
- ✓ Definirea relațiilor
- ✓ Operații cu tabele
- ✓ Operații cu datele tabelelor
- ✓ Crearea relațiilor între tabele
- ✓ Sortarea, filtrarea și indexarea datelor

BIBLIOGRAFIE

- VIOREL STOIAN, Baze de Date, Note de curs, 2019
- MANCAŞ, C., Modelul relaţional al datelor. Editura Ovidius University Press, 2005.
- POPESCU, I., Modelarea bazelor de date. Editura Tehnicã, Bucureşti 2001.
- https://ro.wikipedia.org/wiki/Baz%C4%83_de_date
- http://www.piatafinanciara.ro/studiu-mercury-research-asupra-pietei-de-turism-din-romania/